WILEY EMERGING TECHNOLOGY SERIES

Blockchain Technology

Concepts and Applications

Kumar Saurabh
Ashutosh Saxena

Foreword by:

Dr. V. K. Saraswat
Member, NITI Aayog, Government of India

Dr. Bimal Kumar Roy
Head, R. C. Bose Centre for Cryptology & Security, ISI

C P Gurnani
MD and CEO, Tech Mahindra

Interview Questions and Answers

Instructive Videos

Project Work

WILEY

WILEY EMERGING TECHNOLOGY SERIES

Blockchain Technology

Concepts and Applications

Kumar Saurabh
Ashutosh Saxena

Foreword by:

Dr. V. K. Saraswat
Member, NITI Aayog, Government of India

Dr. Bimal Kumar Roy
Head, R. C. Bose Centre for Cryptology & Security, ISI

C P Gurnani
MD and CEO, Tech Mahindra

Interview Questions and Answers

Instructive Videos

Project Work

WILEY

Praise for
Blockchain Technology: Concepts and Applications

This book is a great read on blockchain. Having been involved in conceiving practical blockchain solution for telecom sector, I loved the way this book takes the readers step by step in helping discover blockchain components through a live platform. That is what makes the book a "practical read." The book covers not only the technical aspects of blockchain but also discusses basic building blocks such as hash functions, cryptography and consensus. The chapters are organized well and in logical sequence, but each chapter is an independent read for someone who wants to skip to topic of interest. Advanced concepts such on Ethereum, Bitcoin, Smart Contracts and Mining are simply explained for anyone trying to understand the basics. Not only that, for several techno-business readers, the authors have presented several real-world usages in financial services, insurance, banking industries where blockchain can be used effectively to counter the challenges faced by these industries.

— **Prashant Shukla,** Managing Director, Udemy for Government, India and South Asia

An evolving technology, blockchain is rapidly changing the feel of the technology landscape and is now the focus of increased attention. The authors of this book have laid good foundation on what blockchain is and introduced a set of fundamental ideas such as migrating to blockchain technologies, types of blockchain and working in decentralized ecosystem to enable the reader to grasp the essentials of the concept. The different blockchain concepts discussed in this book include, but are not restricted to, the blockchain concepts but it talks technology as a whole and it integrates with vertical use cases as well as artificial intelligence, cloud computing, machine learning and robotic process automation technologies. Each chapter is adequately maintained by a comprehensive list of questions, fact alert and quick tips to ensure that the reader gets the advantage of an in-depth understanding of the blockchain arena.

— **K. R. Sanjiv,** Chief Technology Officer, Wipro Ltd.

This book is highly recommended for those looking to learn the building blocks of Blockchain Technology and apply those to create use cases in areas like smart contracts and domain applications. A well-written treatise with applications in mind.

— **Prasad Joshi,** Senior Vice President, Infosys Ltd.

Blockchain Technology

Concepts and Applications

WILEY EMERGING TECHNOLOGY SERIES

Blockchain Technology

Concepts and Applications

Kumar Saurabh
Chief Architect,
IBM

Ashutosh Saxena
Professor, Department of Computer Science,
CR Rao Advanced Institute of Mathematics, Statistics and Computer Science,
Hyderabad

WILEY

Blockchain Technology

Concepts and Applications

Cover Image: © iStock.com/monsitj

Limits of Liability: While the publisher and the author have used their best efforts in preparing this book, Wiley and the author make no representation or warranties with respect to the accuracy or completeness of the contents of this book, and specifically disclaim any implied warranties of merchantability or fitness for any particular purpose. There are no warranties which extend beyond the descriptions contained in this paragraph. No warranty may be created or extended by sales representatives or written sales materials. The accuracy and completeness of the information provided herein and the opinions stated herein are not guaranteed or warranted to produce any particular results, and the advice and strategies contained herein may not be suitable for every individual. Neither Wiley India nor the author shall be liable for any loss of profit or any other commercial damages, including but not limited to special, incidental, consequential, or other damages.

Disclaimer: The contents of this book have been checked for accuracy. Since deviations cannot be precluded entirely, Wiley or its author cannot guarantee full agreement. As the book is intended for educational purpose, Wiley or its author shall not be responsible for any errors, omissions or damages arising out of the use of the information contained in the book. This publication is designed to provide accurate and authoritative information with regard to the subject matter covered. It is sold on the understanding that the Publisher is not engaged in rendering professional services.

Trademarks: All brand names and product names used in this book are trademarks, registered trademarks, or trade names of their respective holders. Wiley is not associated with any product or vendor mentioned in this book.

Other Wiley Editorial Offices:
John Wiley & Sons, Inc. 111 River Street, Hoboken, NJ 07030, USA
Wiley-VCH Verlag GmbH, Pappellaee 3, D-69469 Weinheim, Germany

John Wiley & Sons Australia Ltd, 42 McDougall Street, Milton, Queensland 4064, Australia
John Wiley & Sons (Asia) Pte Ltd, 1 Fusionopolis Walk #07-01 Solaris, South Tower, Singapore 138628
John Wiley & Sons Canada Ltd, 22 Worcester Road, Etobicoke, Ontario, Canada, M9W 1L1

First Edition: 2020
ISBN: 978-81-265-5766-0
eISBN: 978-81-265-9012-4 (ebk)
www.wileyindia.com

To my loving wife Anju and adorable daughters Shivika and Ashna.

— **Kumar Saurabh**

To the memory of my father, Brijendra Kumar Saxena (1938–2018)

— **Ashutosh Saxena**

डॉ. वी.के. सारस्वत
Dr. V.K. Saraswat
सदस्य
Member
Tele : 23096566, 23096567
Fax : 23096603
E-mail : vk.saraswat@gov.in

सत्यमेव जयते

भारत सरकार
नीति आयोग, संसद मार्ग
नई दिल्ली–110 001
Government of India
NATIONAL INSTITUTION FOR TRANSFORMING INDIA
NITI Aayog, Parliament Street
New Delhi-110 001

FOREWORD

Blockchain technology uses cryptography to link a list of records, (called blocks). It had humble beginnings in the early 1900s to times-stamp digital documents using cryptographic hash to prevent back-dating and tampering with them. More famously, in 2009, this technology was adopted in a big way by Bitcoin.

Two leading cryptographic experts, Dr. Ashutosh Saxena and Dr. Kumar Saurabh have authored a book *Blockchain Technology: Concepts and applications* and the same is being published by wiley India. Dr. Saxena, presently working in CRRAO-AIMSCS, is an academician and industry expert with over two decades of experience, has patented over two dozen technologies and has over 90 international publications to his credit. Dr. Saurabh, Chief Architect at IBM, is the author of more than two dozen research papers. His research interest is in the areas of Enterprise and Cloud Computing.

'Blockchain' has since emerged as a potentially transformative force in multiple aspects of government and private sector operations. Though the technology is still in a nascent stage of its developments and adoption as it continues to evolve, it is important for stakeholders such as policy makers, regulators, industry, and citizens to understand the functional definition of the entire suite of blockchain technologies along with technical and regulatory issues and other implementation prerequisites. This book on

Blockchain Technology: Concepts and Applications aims to address these needs. It presents the evolution of Blockchain with the amalgamation of various disciplines like: cryptographic science, economic game theory, software engineering, distributive computing and network and security. The book discusses how Blockchain technology enables the formation of currencies that are decentralized, self-operational digital smart contracts and intellectual digital assets that can be governed over the Internet while touching upon peer to peer networking.

As blockchain develops trust via cryptographic process, authors have presented all the important building blocks of cryptography and consensus algorithms required for operationalizing the blockchain. Readers will appreciate the contents on how Ethereal facilitate for building decentralized applications and extending the features of blockchain in this book along with details about Bitcoins.

The book comprises 11 chapters and deals with the blockchain technology concepts in its entirety. blockchain combined with IoT is critically poised to immense add value to society in various walks of life and the authors have successfully presented various usage of blockchain in the real world and applied side of the blockchain over cloud computing. I am confident that Authors' effort in this book will be useful not only to the instructors but also to the developers and architectures of blockchain for solving there real world problems.

Dr. V. K. Saraswat,
Former Director General of DRDO,
Member, NITI Aayog, Government of India

INDIAN STATISTICAL INSTITUTE

R.C. BOSE CENTRE FOR CRYPTOLOGY & SECURITY

DR. BIMAL KUMAR ROY
Ex, Director, Indian Statistical Institute
Professor, Applied Statistics Unit
Head R.C. Bose Centre For Cryptology & Security

203, Barrackpore Trunk Road,
Kolkata 700108, India.
Telephone: (91) (33) 2575-2809
Fax: (91) (33) 2578-5260
e-mail : bimal@isical.ac.in

FOREWORD

For the Book "Blockchain Technology: Concepts and Applications"

Blockchain Originally a Growing list of blocks which are linked by using cryptography, allows data to be stored globally on thousands of servers. It has emerged as an incorruptible digital ledger of economic transactions and transformative power in many government and private sector operations. Several instructions across globe had recognized its potential, and highlighting the benefits of its application in reducing costs of operation without compromising the efficiencies. While the technical underpinnings of the technology can be intimidating to a large section of policy and decision makers - simply and functionally, blockchain can enable ease of collaboration for enterprises and the ease of living for our citizens by bringing in transparency across government and private sector interfaces. This book on *Blockchain Technology: concepts and Applications* have come at the right time.

The book start with evolution of Blockchain and have presented how this technology enables the formations of currencies that are decentralized, self-operational digital smart contracts and intellectual digital assets that can be governed over the Internet while touching upon peer to peer networking. Cryptography is the backbone of the blockchain and authors have concisely, without any ambiguity, covered them pretty well apart from the several consensus approaches. Prominent topics like Blockchain, Bitcoin, Smart Contracts are been covered in the book and readers will appreciate the contents while reading the contents as they very lucidly explained.

It is worth to mentions that technology will consider useful only when it sees the light of the day and brings value to the society. I must congratulate the authors that they are successful enough in presenting various usage of blockchain in the society by addressing real world problems in financial and insurance sectors.

I am hopeful that this book will be worthwhile not only to the Instructors but also to the developers and architectures of blockchain.

Bimal Roy

March 5, 2020. **Prof. Bimal Kumar Roy,**
Head, R. C. Bose Centre for Cryptology and Security,
Professor, Applied Statistics Unit,
Indian Statistical Institute

FOREWORD

It gives me immense pleasure to write a foreword for this book; it's on a topic very close to my heart – Blockchain.

Blockchain may be the buzzword today, but is it really understood correctly by all? There are different views and debates around it and in this book, Dr. Saurabh gives a fresh perspective, highlighting the vitals, concepts and use cases of Blockchain.

As we all know, blockchain is a digital, decentralized, distributed public ledger database where blocks are linked cryptographically, and transactions digitally signed and managed using consensus model. Blockchain, based on its exclusive trust-based mechanism, is able to integrate industry and technology and today, with the backing of government policies, technological advancements, and market penetration, blockchain is endorsing the amalgamation of the real economy.

This book is a must read for students, academicians, developers, system architects, solution architects, technical managers and business professionals because it gives specific solutions to real-world problems. It facilitates the user to understand the features and benefits of blockchain through distinct examples and defines a new way to manage IT resources with mutual trust, immutability, decentralization and auditability.

I congratulate Dr. Kumar Saurabh and Dr. Ashutosh Saxena for this book and wish them all the very best for the upcoming future.

C P Gurnani
Managing Director and Chief Executive Officer
Tech Mahindra

PREFACE

The notion of blockchain has developed so ubiquitous in the mainstream that most professionals are foreshadowing it as the subsequent foremost disruptive transformational technology. Blockchains are substantial novel way for technical developments, empowering protected transactions without the necessity for a centralization and authority. How blockchain has developed a significant way for the in-depth integration of vertical industry and technology is presented in Chapter 1. Blockchain develops trust via cryptographic process by permitting diverse groups to securely exchange information without the use of a middleman or central agency. The evolution of blockchain with the amalgamation of various disciplines such as cryptographic science, economic game theory, software engineering, distributive computing and network and security are also discussed in this chapter.

Chapter 2 presents a decentralized system useful for blockchain technology that exists in a peer-to-peer network. Blockchain technology permits the formation of currencies that are decentralized, self-operational digital smart contracts and intellectual digital assets that can be governed over the Internet. Blockchain also empowers the advancement of new management systems with more self-governing or shared decision pursuers, and decentralized enterprises that can function over a network of computing ecosystem without any manual interference. A blockchain's synchronizing power is not merely restricted to simplifying the execution of machines. It also permits for the action and links of diverse smart contracts that are discussed in detail in Chapter 7, and that interrelate and network with each other in a distributed and decentralized fashion. Several smart contracts can be collected and organized to create decentralized enterprise that function conferring to explicit instructions and measures well-defined by code-based smart contracts and code, thus altering philosophy that enterprises and objects are nothing more than an assembly of contracts and associations in real terms.

In Chapter 3, we present one of the most important building blocks of blockchain known as hash. Hashing usually means compressing, that is, the output of a function is shorter than the input to the function. This property is extensively used in blockchain; however, it should be secure. In order to have a secure hash function, it should possess some properties and a generally accepted secure hash function should be one-way and collision resistant. All that is required in the secure hash function for implementing a blockchain is discussed in Chapter 3. This chapter categorically discusses the hash function SHA3, which is derived from the original algorithm known as Keccak. After covering various core concepts on hash function, we also discuss Merkle tree. Hash trees allow efficient and secure verification of the contents of large data structures and play an important role in blockchain.

Chapter 4 presents different consensus approaches and is not exclusive to blockchain. In fact, this is a classical problem in distributed computer systems. The consensus in the blockchain is a way to ensure that nodes on the network verify the transactions and agree with their order and existence on the immutable ledger. In a typical case of cryptocurrency, the process is critical to prevent double spending and therefore the consensus mechanism plays an important role. Fifteen consensuses algorithms have been discussed in this chapter.

Blockchain components with Ethereum as a platform are discussed Chapter 5. Readers will appreciate how Ethereum with chronological development of its platform, facilitate the building decentralized apps and extend the features of blockchain. This chapter discusses its client's working, key generation, address system, wallets, transactions, language, browsers, tools, and usage of IDE.

Cryptography is the core of blockchain technology. Therefore, Chapter 6 covers the topic of cryptography to have complete understanding on blockchain, including both symmetric and asymmetric types. It also covers four primary cryptography primitives – confidentiality or privacy, authentication, non-repudiation, and integrity. We are hopeful that this should be sufficient to understand the underling cryptography principles in blockchain.

Much before the proliferation of the Internet, several attempts were made by many people to have digital cash technologies in place. But it was Bitcoin that got huge popularity as a digital and global currency system that allows people to send or receive money across the Internet without being linked to a real identity. Bitcoins are portable, durable, divisible, recognizable, fungible, scarce, and difficult to counterfeit. Chapter 8 discusses in detail about Bitcoin block structure, Bitcoin address, Bitcoin transactions, Bitcoin network, Bitcoin wallets, Bitcoin payments, and Bitcoin clients.

Decentralized applications (DApps) are a blockchain aided website, while a smart contract permits to link nodes of a blockchain using code based on the agreement. In Chapter 9, we differentiate between DApps and smart contracts. We also discuss Whisper, which is a DApp message communication for messaging in a peer-to-peer network of DApps. Additionally, we discuss mining hardware–software, with single mining and pooled mining. This chapter also presents the basics of Swarm and blockchain forks, including hard forks and soft forks.

No technology will see the light of the day unless it brings value to society. Chapter 10 presents various uses of blockchain in the real world. These include two prominent sectors – finance and insurance – apart from various others such as business transformation. We extensively cover use cases related to health insurance, asset insurance, marine insurance, home insurance, and auto insurance. Towards the end, we present a use case related to treatment tracing to distinguish overdose and overtreatment using blockchain. Blockchain technology can generate boundless prospects in accessing underutilized computing resources and equipment, and deliver the infrastructure to provision a completely new environment. Chapter 11 discusses allied topics of blockchain and cloud computing, blockchain and artificial intelligence, blockchain and IoT, blockchain and machine learning, and blockchain and robotic process automation.

We have provided an annexure (Annexure B) that discusses several technologies such Ethereum, MetaMask, and Remix. This facilitates readers to immediately start using and experience the global blockchain by writing programs of their own. This also contains step-by-step procedure to install MetaMask extension in the browser followed by instructions to create and deploy a contract using Remix, for which important commands are briefly presented in it.

We hope that this book will be useful not only to beginners in this subject but also to developers and architects of blockchain to help them solve real-world problems.

Instructor Resources

The following resources are available for instructors on request. To register, log onto https://www.wileyindia.com/Instructor_Manuals/Register/login.php

1. Chapter-wise Solution Manuals.
2. Chapter-wise PowerPoint Presentations (PPTs).

The presentation decks (one for each chapter) can be taken to class directly or can be customized as per your requirements.

Icons Used in This Book

Technical Stuff	This addresses granular technical practices.
Flash Quiz	This is an out-of-the-box question that will test your level of understanding.
Fact Alert	This points out an important truth for readers to take note.
Quick Tip	This represents ideas that are practical advice that you can apply in the described context.
Activity	This gives essential activities to carry out for better comprehension of the topic.

ABOUT THE AUTHORS

Kumar Saurabh (Ph.D, PMP®, TOGAF Certified) has several years of industry experience with companies like IBM, CenturyLink, and Tech Mahindra. Currently, he is Chief Architect at IBM. He was Principal Architect and Head – Digital ITS (Innovation, Transformation and Solution) at CenturyLink, and Principal Architect and Head – Strategic Cloud Solutions group at Tech Mahindra. He possesses Master of Science (M.Sc), Master of Technology (M.Tech), and Doctor of Philosophy (Ph.D) degrees. He has been Panelist, Chair, Keynote Speaker, Editorial Member, and Tutorial Presenter at various forums throughout the globe. He has authored multiple research papers published in various international and national journals and proceedings. He has authored several books published by Wiley, including *Unix Programming – The First Drive' Discovering Enterprise Architecture, and Cloud Computing: Architecting Next – Gen Transformation Paradigms Fourth Edition.*

Ashutosh Saxena is an industry expert and academician with over two decades of experience, 90+ international publications with over 2000 citations, 31 filled patents out of which 27 are now US granted patents, and author of book *PKI: Concept, Design and Deployment*. His research interest is in the areas of information security and privacy. He has been a member of the review board for many international journals, conferences, and committees.

He began his career as a lecturer and computer engineer in the university and IUC-DAE facilities at Indore. Also worked as a faculty member (Associate Professor) at the Institute for Development and Research in Banking Technology (established by Reserve Bank of India), Hyderabad, for eight years, supervised TWO PhD.

He joined Infosys in June 2006 and worked as Principal Research Scientist and AVP at Infosys Labs, Hyderabad. He is a five-time winner of the Infosys Excellence Award. In 2014, he received the award for maximum publishing in CSI-Communication from Computer Society of India. He served as Adjunct Faculty at NIT Warangal. He is currently Professor (CS) at CRRAO-AIMSCS, University of Hyderabad.

SPECIAL ACKNOWLEDGEMENTS

We are grateful to Saji Thoppil, Fellow, Chief Technologist, Wipro Ltd., for authoring the annexure "Ethereum Essentials: MetaMask and Remix".

We are grateful to Dr. D. N. Reddy, Director CRRAO-AIMSCS, for all his guidance and support.

CONTENTS

CHAPTER 1

Basics of Blockchain

LEARNING OBJECTIVES

After reading this chapter, you will be able to:

- Understand the concept, fundamentals, characteristics, and definition of blockchain.
- Analyze consensus in trust-building exercise.
- Describe public, private, and hybrid blockchains.
- Identify distributed ledger technologies and their architecture.
- Describe the trusted-service utility.

1.1 Introduction

Blockchain, based on its exclusive trust-based mechanism, has developed a significant way for the in-depth integration of vertical industry and technology. With the backing of strategic policy, technological advancements, and market penetration, blockchain mechanism is endorsing the amalgamation of the real economy, which is extending its help for superior development. Consequently, it is of boundless implication for the world to discover the following:

1. New sharing model of economy.
2. Develop digital economy for business, engineering, and ecology.
3. Transform the value of public service and governance model.

We can view blockchain as an integration of the following innovations:

1. Peer-to-peer networks.
2. Cryptography.
3. Smart contracts.
4. Consensus mechanisms.

Figure 1.1 shows blockchain as an integrated network.

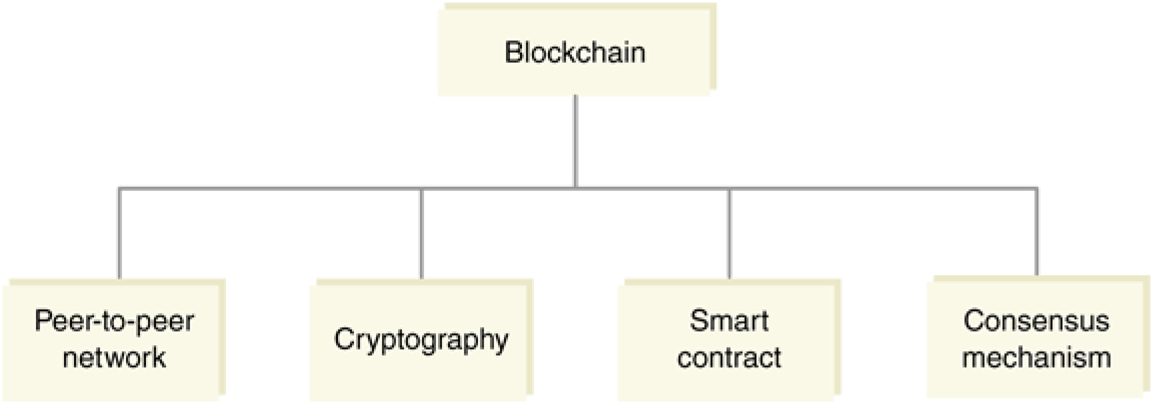


Figure 1.1 Blockchain constituents.

Using the aforementioned innovations, blockchain delivers a trusted network for data information and knowledge transactions in the untrusted channels. At a current juncture, usage of blockchain has been taking pace and transforming, and crucial technological advancements such as cross-chain, identity and privacy protection, and security administration are fetching greater research interests. Although blockchain technology is still at the phase of societal experimentation and exploration, there is no agreement on the notion, architecture, standards, technical features, development direction, governance, administration, and regulation of blockchain.

The notion of blockchain has developed so ubiquitous in the mainstream that several are foreshadowing it as the subsequent foremost disruptive transformational technology. There have been contrasts of its value to that of the World Wide Web and the Internet technologies. Although at its principal, blockchain is just a technique of securely archiving and sharing information, it is the possible practices of blockchain technology that develops it so vesting: distributing asset communications between unlike mediators with undisputable transparency – without working on a central controlling governing body.

Fact Alert

Nowadays, WWW is considered as synonym to Internet and has reached over X billion people on earth, without any owner.

Blockchain develops trust via cryptographic process by permitting diverse groups to securely exchange information without the use of a middleman or central agency. Numerous market segments are attracted for this transformation and new undertakings are competing for supremacy in these domains with an enthusiasm not observed since the Internet boom. Notwithstanding noteworthy interest, there is a dearth of theoretical literature which confers the practical technology supporting blockchains as well as the possible business applications of blockchain. This chapter aims to address this breach and is also envisioned to aid as a conduit to blockchain. It delivers a technological introduction to launch the key concepts and sightsees industry use cases and inclinations. We will see beneath the covering, recognize key features of blockchain technology, and shape an official definition.

1.2 Concept of Blockchain

Conceptually, a blockchain is a distributed, decentralized, and public digital ledger. It is mutually sustained by various parties, such as by means of cryptography to safeguard the security of transactions, identity access, to realize data storage reliability, tamper-proof data, and protection of denial. Blockchain is also referred as distributed ledger technology (DLT).

Flash Quiz

Do you know what is size of the digital economy?

A characteristic blockchain archives data in block units, where every block comprises the cryptographic hash of the previous block in the blockchain for connecting the two head-to-head blocks. The connections of blocks are known as *chains*. Blockchain, as a novel computing archetype and sharing model in an untrusted competitive ecosystem, is transforming the application development situations and operating guidelines of many verticals with its exceptional trust-building methodologies. It is one of the necessary technologies for structuring and developing a new trust structure and evolving digital economy in the coming era. Figure 1.2 shows a characteristic blockchain.

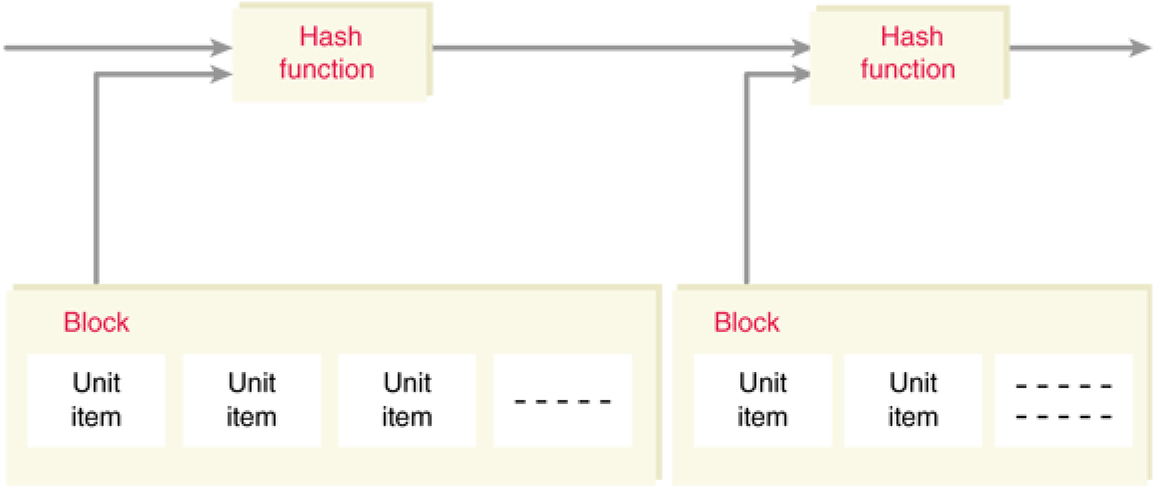


Figure 1.2 A characteristic blockchain.

In a characteristic blockchain system, every party distributes information and scopes consensus in agreement with guidelines decided in advance. To avoid the consensus information from being interfered with, the ecosystem archives data in units of blocks that develop a cryptography-based chain of data structure in sequential order. The record nodes are nominated by the consensus method to know the information (data) of the latest block and other nodes contribute in the authentication, storage and upkeep of the latest data block. As soon as the data is established, it is tough to modify/delete, and only the lawful and authenticated query operation can be made.

Based on the situation, whether the system has a node entry method/control, "blockchains" can be divided into two parts: permissioned and permissionless. The join and exit operations of the nodes in the permissioned blockchain need the authorization/agreement of the blockchain. Based on the circumstances, whether the units with authorization rights are centralized or not, permissioned blockchains can be segmented into the following:

1. Consortium blockchain.
2. Private blockchain.

Public blockchain is also referred to as permissionless blockchain. This is completely open, in which nodes can perform join operation and exit operation at any point of time.

1.3 History

We can imagine the evolution of blockchain with the amalgamation of various disciplines, such as:

1. Cryptographic science.
2. Economic game theory.
3. Software engineering.
4. Distributive computing.
5. Network and security.

Blockchain is the intersection of the aforementioned fields that deliver the equilibrium for a scalable and stable software ecosystem, a foundation for security of digital transactions and units, and supporting worldwide decentralized peers' network along with the financial encouragements for these peers to be respectable performers in the network. Practical blockchain use cases encompassing these multidisciplinary arenas are frequently conversed under the canopy term of cryptocurrency/economy aligned with the development, usage and transmission of wealth using computing node networks, and cryptography to improve wealth of parties in exiting and upcoming digital economies.

Blockchain is a fundamental paradigm of Bitcoin. Consequently, nowadays no conversation of blockchain is possible without talking on the cryptocurrency Bitcoin. In 21 October 2008, Satoshi Nakamoto published a revolutionary art of work in a cryptography arena. He sketched a method to solve the double-spend situation – a problem that overwhelmed earlier cryptocurrencies. Nevertheless not stating blockchain clearly, he defined its data structure as a hashed chain of timestamps. Every timestamp comprises the earlier timestamp in hash, which ultimately makes a chain with individual additional timestamp supporting the ones trailing it. Although this method was polished for Bitcoin, the notion was placed out: chain comprises blocks, every block is cryptographically associated to the earlier one by means of a hash digest. We signify that a blockchain is slightly more than an order of records, each linked and hashed to the preceding block.

Fact Alert

The real identity of Satoshi Nakamoto is still not known.

Still, this method was not enough to eradicate the double spend problem. In a double spend situation, the attacker challenges to generate a race condition in which they spend the same digital assets two times before either is authenticated. For a blockchain to counter this, Bitcoin required a mechanism for its network to develop consensus. Nakamoto presented a proof-of-work method, which will be discussed in detail in subsequent sections, in which agents would recurrently hash the block with a nonce (random number) till a value less than a stated target is attained. Once determined, the block would then encompass the existing blockchain. Over peer-to-peer chain is decentralized, and any agent can contribute in the consensus model such as proof-of-work to authenticate transactions.

Quick Tip

In similar context to the proof-of-work, there are several "proof of *", where * could be stake, importance, activity, elapsed time, burn etc. which we will be discussing in Chapter 4.

The cryptocurrency Bitcoin required a data structure into which it might record the series of transactions, validate them, and finally secure the entry. Blockchain mechanism delivered this with a group of forever extending blocks, where each and every block characterizes a set of transactions, which are linked cryptographically to a previous block through a hash digest. Subsequently, the blockchain is characterized in a free file which can be distributed, and also there is no master file concept. The related information and status is distributed among all the nodes. The network of Bitcoin trusts this decentralized distribution network based on the consensus model (i.e. the proof-of-work) to organize the blocks that are appended to the chain and update information to all the node copies.

Bitcoin blockchain purposes and works as a database to archive transactions. It uses a sequence of inputs/outputs approaching double entry accounting. Remarkably, Bitcoin balances are not preserved, as it just maintains inputs and outputs. It is important to note that no minting of coins exists and nothing is serialized for consumption purpose, as one can imagine. Just by traversing the blockchain, one can quickly calculate the available balance. Any trial to corrupt the blockchain will push to spend more coins and recalculation will be required. It becomes a difficult problem for computing, and the decentralized nature of blockchain safeguards other nodes to have genuine tamper-proof copies.

An immutable and secure blockchain rely on the following technological advancements:

1. A link based on the cryptography among records that becomes increasingly more problematic the longer the chain.

2. The data distribution to all contributing network nodes on the decentralized network in which it is predicted that honest nodes surpasses all the attacks.

Numerous electronic cash systems were developed prior to Bitcoin, but nothing attained extensive use. By accepting blockchain technology, Bitcoin realized persuasive abilities that encouraged its use. The usage of a blockchain empowered Bitcoin is to be executed in a distributed style so that no single user managed the currency, and there exists no single point of failure. Its main benefit was to empower direct digital financial transactions among users without the necessity for a third party. It also permitted the mint of new currency in a reasonable fashion to those nodes termed as miners, who maintained the blockchain and permitted small transaction charges for using the system. The compensation of mining nodes empowers distributed management of the system without the necessity to establish a ecosystems of nodes those preserving the system.

Also, the blockchain permitted node users are anonymous and their accounts are not known. All their transactions can be observed publicly to manage authenticity. This has efficiently empowered Bitcoin to propose anonymity since accounts can be formed without any credentials or agreement process. At last, the distributed management of the blockchain developed a system with comprehensive transparency, which endorsed trust in usage.

1.4 Definition of Blockchain

There are numerous diverse blockchain definitions presented, which ranges from applications specification to technical extensively. For a cryptocurrency organization, they can define blockchain as a decentralized, distributed public digital ledger that comprises the past of every transaction. For application-centered definition, this classification does not interpret for the detail that blockchains can be reclaimed for other cryptocurrencies and vertical applications self-reliantly. The dictionary widens the definition slightly, defining blockchain as an electronic ledger in which transactions completed in cryptocurrency are recorded publicly and chronologically. The earlier definitions also signify the importance of a blockchain in the form of a digital ledger, and much of the literature would settle with this. Although this area is evolving swiftly and ledger practice is just a characteristic of the blockchain but not its kernel. This characteristic only refers to blockchain applications that emphasize on dealing the value exchange in the instance of digital assets.

There exists a wider meaning definition where a blockchain is termed as a data structure, which permits digital classification and tracking of transactions and distributes this information across a network of nodes, forming in a sense a trusted distributed network. The DLT presented by blockchain delivers a translucent and secure method for tracking the rights and transfer of digital assets. The important difference here is that a distributed system divides work among parties (nodes) in an ideal way, while a blockchain should preserve a full node of the network in each party node and impose its rules self-sufficiently. In a computing network where nodes function on local information to achieve objectives rather than the outcome of a centrally managing effect, this decentralization safeguards only one node required to remain working for the network functionality.

Obviously, there is a requirement for a strong and succinct definition of blockchain. On the basis of the hypothetical foundations of blockchain technology discussed in the preceding sections, the following definition of blockchain can be given:

> *Blockchain is a digital, decentralized, distributed public ledger database where blocks are linked cryptographically, and transactions are digitally signed and managed using consensus model.*

Based on this definition, we depict the essential elements for blockchain technology as a network-based peer-to-peer database administered by a set of instructions. Also, blockchains characterize a change away from old style trust agents and march towards transparency. As a technical structure, it allows applications from a comprehensive wrapping of vertical domains to enjoy the benefit of distribution, tracing, and inspecting (auditing) virtual digital assets.

A blockchain works like a digital ledger formed to capture transactions directed among various nodes in a network. It is an Internet-based peer-to-peer distributed ledger which comprises all the transactions since its conception.

All the parties, whether they are users and business individuals or groups, use the shared database that are nodes linked to the blockchain and each block maintains the same copy. Each entry into a chain is treated as a transaction that signifies an exchange of value between parties, which depicts digital ownerships and rights. Different categories of blockchains are being evolved, established, and tested. However, most of the blockchains work on this general mechanism and method.

When one node desires to send value to the other node, all the nodes in the network start communicating with each other using an established control to authenticate that the new transaction is lawful. This control is referred to as a consensus algorithm. If a transaction has been acknowledged by the network, all the ledger copies are reorganized with the new set of information. Generally, numerous transactions are grouped into a block that is appended to the ledger. All the party nodes can include time-stamped transactions, but no node can delete or change the entries after they have been authenticated and acknowledged by the network. If a node changed a preceding block, it would not work with the whole network and would be omitted from the blockchain. An appropriately operative blockchain is thus immutable without having a central governing body.

1.5 Fundamentals of Blockchain

The capability of blockchain to permanently archive transactions and negotiate the authority of an attesting governing body to distribute consensus development is based on the amalgamation of different mechanisms offered in simplified form in the following order:

1. The digital signature is the first step, where a transaction such as transfer of a cryptocurrency or a document is registered and is digitally signed once it is generated. This transaction is traversed into the network and distributed through the nodes required. The network nodes validate transactions and it is entered into the blockchain.
2. During this process, the transactions are archived in blocks that are changed in a homogenous format using hash function. At first, each statement is encoded in hash values and then it is hierarchically compressed. This tree level compression of each statement is denoted as a hash or Merkle tree, which helps to represent the block of

statements. The statements coding is harmless against tampering trials. Meanwhile, modifying a single statement would also modify the block's hash value, and the Merkle tree will not be consistent.

Quick Tip

Digital signature is different from electronic signature. A digital signature is mainly used to secure documents and is authorized by certification authorities while an electronic signature is often associated with a contract.

3. Blocks are linked using chains with the existing blocks, and consequently, a chain is created. If we want to add a new element as a block in a Bitcoin, use a cryptographic puzzle to solve where a string gives same (similar) hash value when the new block is encoded, which needs to be implemented. The likeness (similarity) of both the block values is described in the hash value by character numbers. The similarity complexity degree can be varied.

Fact Alert

Merkel tree is the name given in honor of the author, Ralph C. Merkle. He also devised Merkle's Puzzles, a scheme for communication over an insecure channel.

Hash function by its characteristic is not reversible. Currently, there exists no productive mechanism for originating the string that is to be predicted for the given hash value. Because of this, one needs to try various strings, which involves proper or a quite large computing volume. The node in the blockchain looks for this string and if found, the new block gets added to the chain into the last of the validated block. By calculating the hash value of other blocks, the correctness of any network node can be checked.

Consequently, linking correct blocks is realized. To achieve perseverance across the network, these chains are now shared across large number of nodes, that is, with all the nodes comprising same information. If new blocks are developed in any nodes as an addition to the current blockchain, a consensus about the modification can be touched through the network. The cryptography-based puzzle resolves of this consensus discovery. If a node solves a puzzle, the solution is validated and acknowledged by all the nodes engaged. The blocks in the waiting list for consensus are prearranged in a list of successors, where blocks of concurrently formed links are also involved in order to integrate them once again into the one public blockchain.

A blockchain comprised of the individual blocks can consequently be governed in a network of nodes. The consensus discovery regulates which block is accepted as a subsequent component in the blockchain. Formerly, the cryptographic puzzle was utilized to develop new blocks in the mining process as part of proof-of-work. The puzzle's difficulty level can be attuned for the sake of diverse privacy and security needs.

Flash Quiz

What is data mining?

Another method of consensus discovery may share certificates in a network node system. Consensus is touched when the widely held stakeholders touches the same outcome that is considered as proof-of-stake. Otherwise, nodes may be treated as miners for consensus discovery and work like the umpires, or choices are made just like the lottery. In addition to this, other opportunities as well as groupings of the stated types of consensus discoveries are also likely.

Blockchains can be simply defined as the distributed databases that are systematized by the participant node in the network. In comparison to centralization approaches, blockchains are much less vulnerable to faults, tampering and, specifically, to avoid Byzantine faults. However, these systems also carry various problems sideways. Data redundancy is very critical and discussed at length at various platforms. As the same copies of data are retained across the network, it requires large storage spaces. Besides, Even though blockchain technology is still in its initial phase, it has experienced several transformations in the current scenarios, particularly with respect to its usage in a closed industrial background. Because of the diverse purposes, there is significant variance among private and public blockchains.

1.5.1 Public Blockchains

Public blockchain comprises public node systems where anyone can copy information with the open access. This is not identical to the programmed reading and writing on a blockchain node. It is achieved via hypothetical full nodes, which executes the authentication-independent requirements of a user. Examples of public systems blockchain are:

1. Ethereum.

2. First generation Bitcoin blockchain.

1.5.2 Private Blockchains

Private blockchains define systems that are individually accessible to a closed association, such as an enterprise. The public feature is to be compared with the query of the authentication privileges. Public blockchains are mostly considered as permissionless. In the scenario of private blockchains, node entree privileges are usually governed or constrained to an association. In all the major scenarios, these blockchains are approval derived blockchain consortia. Hyperledger can be considered as an example of private blockchain.

1.6 Characteristics of Blockchain

If blockchain is compared to the traditional distributed database data structure, it replicates the following characteristics:

1. We will first talk about double-entry with respect to distributed computing. In old-style information management system, each auditor records discretely. There are several diverse ledgers at each settlement. The blockchain disrupts this double-entry accounting practice and develops a distributed ledge book. The entire network node shares the information where party nodes working in the accounting practice avert data altering and safeguard the reliability of data using synchronous consensus controls evading the complex multiple-party resolution process.

2. Traditional databases works on four classical operations – insertion, deletion, selection, and update. With respect to network, the ledger gives up deletion and update operations. It can only do manipulation operations, which includes insertion and selection, within the data structure of block and its linked list. Its equivalent timestamps combine the receipt, thus creating trusted data sets that are linking and difficult to alter.

Flash Quiz

Write an SQL statement for insertion, deletion, selection, and update with a criterion of your choice.

3. Another characteristic difference is from independent central governance to multi-node multi-party decentralized governance. For each object, the traditional database is an independent central governance-based maintenance information system. It either comes in a distributed architecture or in a centralized architecture, which looks for a high degree of governance over records of data. The blockchain presents a distributed ledger (i.e., shared account bookkeeping database), which is a distributed decentralized data structure that is mutually managed by the multiple node with high availability and without single point of failure. Data updating and syncing are not restricted to one entity or operation and are additionally required to be validated by various other node parties for reaching consensus to resolve and adopt as to which information can be updated.

4. In a traditional model, both financial and business process flows are different. The physical contract is signed mutually by the business stakeholders. Once it is validated after the manual review, one notifies to finance team to complete the payment process. Evolution of the smart contracts that are based on the rules and governing principles is independently executed on the basis of the piece of code. It is a collaborative style of updating onto the nodes that include both financial and business process flow. Plugged-in smart contract joins together information on business and financial capital flow using algorithms and code.

In spite of being originally linked to Bitcoin, blockchain mechanism can be utilized self-reliantly in a variety of use cases and target markets, ranging from BFSI, utility, manufacturing, and healthcare industry. A blockchain can be deployed in virtually any sector in which digital virtual/physical assets are governed and transactions exist. It delivers a secure chain of protection for both digital and virtual/physical assets using its efficient features that simplify transactions through trust, tamper-proof consensus, authentication, and smart contracts. These features of blockchains are briefly explored in the following section.

1.6.1 Smart Contracts with Respect to Transaction

A transaction is a sharing of assets (digital/physical) that is governed under the guidelines and access rights of object service. Such guidelines are frequently made available using scripting such as “Forth” and are utilized for cutting-edge transactions such as escrow and multi-party signature.

A smart contract is a set of logical guidelines that is coded using script and can be integrated with the blockchain to manage transaction. The contract is accomplished freely and is utilized to manage transaction. Therefore, the contracts work as smart code agents. Once integrated, it turns into an independent agent that is always tamper-proof. The contract characteristic of a smart contract is not just limited to the application or vertical specific script. It can also be utilized to write script for the rules and clauses of a contract into the transaction workflow.

1.7 Consensus in Trust-Building Exercise

Blockchain’s decentralization is a fundamentally strong point, as a replica of the database file is possessed by all actors. To safeguard the reliability of each replica, a consensus mechanism in the form of code is required. The consensus mechanism permits the consortia to confirm that each newly appended block is authentic. It also avoids attackers from conceding and forking the blockchain Nakamoto that recommends using proof-of-work mechanism, where a difficult puzzle powered by the cryptography must be resolved by the miners. Miners employ computing assets and are paid for their hard work using several enticements. There are other consensus models, such as:

1. Proof-of-stake.
2. Proof-of-capacity.
3. Proof-of-burn.
4. Proof-of-elapsed-time.

These models are proposed in the research to resolve some of the problems of the original proof-of-work model by trying to maintain equilibrium equality and resource usage.

Blockchain is considered to eradicate the requirement for any one object to provide a way to transactions. It launches a trust mechanism built on a consensus formed by the group of nodes, where network authenticates transactions and approves their addition in the block of chains. There are no agents and the idea of belief develops implied as each node in the blockchain is confirmed by the neighboring node which embraces multiple replicas of the blockchain. By eliminating trust of the middlemen from transactions, blockchain has the capacity to transform many major businesses.

Old-style transaction mechanisms trust on the centralized body to act in governing the ecosystem. Trust is approved to the central body with the anticipation that it will endure to be truthful while authenticating and validating transactions. The illustrations of records exist in with the governing body. If the central body is negotiated, either deliberately or accidentally, the panelist can create widespread chaos in the system. The blockchain model eradicates the central body by placing distributing replicas of the records to all the participating nodes in it. Each participating node preserves their own occurrence of the blockchain. The transfers are modified by creating new blocks, and authentication based on the instructions of the consensus mechanism is invited. Once authenticated, the block is appended to every node chain. The method is possibly harmless with respect to the old model, and the intermediary agent is not needed, appealing a problem to the status of the blockchain.

A blockchain gets strength based on the mathematics using cryptography to create self-governing trust for every transaction and using expensive computing resources-based consensus models to substitute the central governing bodies. Pools of current transactions are sequenced into a block. Cryptographically, the block is linked to a chain of blocks and is confirmed through a consensus model that includes substantial computing resources referred to as mining. Now, it is understood that blockchain is an open entry-based file that is copied on multiple nodes of the network, but none of the node controls the transaction as a central body. As every hash block is inserted into the chain, blocks become immutable and work as an ultimate record of historical transactions. An object cannot modify the chain without modifying all the blocks that shadow it, a work that is computation-wise very difficult, expensive and not worthy. This safeguards the blockchain and launches independent-trust model of a central governing body.

1.8 | Public, Private, and Hybrid Blockchains

Public, private, and hybrid blockchains were already discussed in brief at the beginning of the chapter. Blockchains can be divided into these three categories based on their applications.

1.8.1 Public Blockchain Characteristics

The characteristics of the public blockchain include the following:

1. No single owner.

2. Visible by anyone.

3. Consensus process is open to all to party.

4. Full decentralized.

An example of public blockchain is Bitcoin.

1.8.2 Private Blockchain Characteristics

The characteristics of the private blockchain include the following:

1. Permissioned blockchains.
2. Use privileges to govern.
3. Read from and write operations in the blockchain.
4. Generally, consensus and mining mechanism are not required as centralized single node can have the rights and controls to block creation.

1.8.3 Hybrid Blockchain Characteristics

Hybrid blockchain is also termed as consortium blockchain. The characteristics of the hybrid blockchain include the following:

1. Have public characteristics limited to a privileged cluster.
2. Consensus model is governed by known nodes, called privileged nodes.
3. Agreement for the set of guidelines and instructions agreed to by all parties.
4. Replicas of the blockchain are shared only among the permitted member nodes.
5. Network is partially decentralized.

While a public blockchain allocates itself in a decentralized network between nodes fashion, this is not inevitably accurate with respect to the private blockchain. Private blockchains are essentially utilized by organizations to log asset transactions within a restricted and known user base. Hybrid blockchains can be envisaged as minor public blockchains, it is decentralized among a restricted known participant base.

1.9 Distributed Ledger Technologies

Distributed ledger technologies (DLT) are predecessors to blockchain and it is good to understand that DLT will help us understand the concept. It can be considered that DLT implementation is an instance of blockchain.

A ledger or account book or register by meaning is a procedure by which information is saved for all the transactions of an enterprise. Since the beginning of human evolution, ledgers have been utilized for caring economic dealings to maintain payment transactions against goods and services, holdings, inventory tracking, contracts, and agreements. A centralized ledger is administered by a single object that is trusted with suitable governance of controls and access rights. A distributed ledger functions in the form of network where users accept, authenticate, and maintain the set of transactions. The information is copied with numerous users and there exists multiple databases.

Every node or computing resource on the DLT network has to develop its individual objective, and then the nodes nominate the correct form (data structure) of the record of transactions. With an accepted and authenticated consensus, the ledger is restructured with the transaction. All the nodes or computing resource within the network preserve their own replica of the ledger. There is no central owner or administrator of the distributed ledger. The data is archived and distributed among each participant on the large network regardless of their position or establishment. Any modification to the record is instantly recorded in all replicas of the distributed ledger. This can happen in any format whether digital, financial, physical, virtual, or legal. The safety and accuracy of the distributed ledger are preserved through a cryptographical method.

1.10 DLT Decentralized Applications and Databases

Decentralized applications and databases existed since decades. Well-known examples are BitTorrent, IPFS, and eMule. General characteristics include the capability to scale as required, append nodes, data load balancing, and capacity to authenticate content which is copied between several nodes.

1.10.1 Private versus Public DLT

DLT network possess individual practice that administrates the guidelines for participation, authenticates the record of transactions, and manages the ledger. This sharing can be public or private. Public DLT allows entry by any party, user, or node whereas private DLT entry is based on permission control.

1.10.2 Public Distributed Ledger Technologies

Public DLT can be imagined as Bitcoin and Ethereum. The rewards of a public chain are that an object or group of objects cannot take govern of the ledger and insert deceitful transactions. It is included in the public ledgers to check the current and historical transactions without using any third-party mediators.

Flash Quiz
Who owns the credit of Monero and zCash?

There is one more category of public ledgers. Although they are public, they possess private nature as they can be audited. The nodes participating in the transactions can see the content of these transactions.

Public DLTs utilize highly distributed consensus controls such as proof-of-work or proof-of-stake. It is based on the controls that require rewards (coins) and tokens to protect the network for developing the incentives for participating mining nodes.

1.10.3 Private Distributed Ledger Technologies

Private DLTs deliver exclusive attributes such as the totally safe from non-authenticated node and possess all private transaction information, and it can be only accessed by the ledger nodes. It is important to note that generally number of nodes that are very less in private DLT, so they prove to be a little bit more vulnerable toward attacks.

Private DLT inclines toward the low-intensity distribution consensus algorithms, such as Practical Byzantine Fault Tolerance (PBFT), Raft, and Paxos.

There are various limitations in these distribution consensus algorithms as a restricted number of nodes participate in the network. It is noteworthy that the transactions in private ledgers are usually higher as per node in comparison to the public DLT because of a smaller number of participating nodes working in the network.

1.11 Architecture of Blockchain

Blockchain systems look very intricate architecturally; although it can simply be understood by investigating individual element technology. Blockchains employ familiar computer science and financial topics, such as:

1. Linked lists.
2. Distributed computing (networking).
3. Cryptographic primitives.
4. Hashing.
5. Digital signatures.
6. Public/private keys.
7. Ledgers.

1.11.1 Hashes

A vital element of blockchain technology is the practice of hash functions coming from cryptography for various operations such as content hashing within a block. Hashing is a calculation mechanism of scheming a comparatively exclusive fixed-size digest (also termed as message digest) with respect to an input of any size. It can be of any format such as text, image, or even a file. If there is a smallest change (single bit also) in the input–output message digest, the resultant output digest will be completely different.

Hash algorithms are architecturally designed to operate in a one-way direction. Computing wise, it is not practicable to treasure any input that results to any predicted or forecasted output. If a predicted or forecasted output is anticipated, numerous inputs must be executed through the hash function till an input is mapped that creates the wanted result, but it will be resource extensive. Hash algorithms are also architected in a way that it should be practically impossible to guess some inputs that can give desired output, which computation-wise is very difficult. It is impossible to be collision resistant (known as second pre-image resistant), it is computationally infeasible to find two or more inputs that produce the same output.

Hashing mechanism utilized in the blockchains employs some famous hash algorithms such as:

Fact Alert
A hash function by the name MD5, which is 128 bits long, is considered to be unsafe.

1. Secure Hash Algorithm (SHA) comprising output size of 256 bits (SHA-256). (This is the most famous algorithm.)
2. Federal Information Processing Standard (FIPS).
3. Elliptic Curve Digital Signature Algorithm (ECDSA).

It is important to understand that there exist a large set of input values. However, there exist only a finite number of message digest. It is possible to have a collision when an input message produces the same message digest. This is represented by

$$\text{hash}(a) = \text{hash}(b)$$

where a and b are inputs that produce the same message digest. It is extremely doubtful for any inputs, a and b, to hash the same digest to both situations and validate function with respect to blockchain system and compute at the same timestamp and collide with each other. This is the reason why SHA-256 is a collision-resistant algorithm. We will learn more about SHA-256 in subsequent chapters.

1.12 Transactions

A transaction is a record that transfers assets such as currency and inventory units from one party to another. An equivalence is like a bank passbook with all the records where a transaction occurs whenever money is deposited or withdrawn from an account. A viable transaction happens when these data fields are contained in any operation.

1.12.1 Amount

Amount is the total quantity of the assets transferred over a transaction.

1.12.2 Inputs

Input is the list (inventory) of assets that will participate in a transaction. It is important to understand that all the input assets are identified by unique values and do not match other assets. Also, operations such as insertion and deletion of assets are not possible. It is possible to create new assets by splitting or combining the existing inventory-based assets.

1.12.3 Outputs

Outputs are the final parties that will receive the hashed processed assets when the transaction occurs. Output requires the value to be transacted to the new node, the key (ID) of the new node, and a group of rules that ensures the new node must receive the processed transaction-based new value.

1.12.4 Hash ID

Hash ID is a key or unique identity for any transaction.

1.12.5 Asymmetric-Key-Public-Private Key Cryptography

Asymmetric-key-public-private key cryptography uses a pair key, public key, and private key. It is connected to each other based on mathematics. Here, the public key will be visible to everybody without compromising on security, but the private key remains secret for maintaining cryptographic shield. While there is a link among the keys, the private key cannot be known based on the information of the public key.

It utilizes the diverse keys of the key pair for precise functions, reliant on which service is to be delivered. For the service such as digital sign, signing private key is used whereas the equivalent public key is used for signature verification.

Quick Tip

Asymmetric-key–public–private-key cryptography is also referred as asymmetric-key cryptography or public–private key cryptography.

1.12.5.1 Blockchain Systems Utilization for Asymmetric Key

This part we will discuss about the system based requirements for the usage of asymmetric keys. It comprises of the following:

1. Digital signs utilize private keys.
2. To know addresses, public keys are used.
3. Verification of digital signs is done through public keys.

1.12.6 Addresses

Addresses are short alphanumeric strings. An address is derived from the public key utilizing hash function. It comprises some additional data in order to detect vulnerabilities. It is used to transfer the digital assets. Blockchain uses an address like endpoints – to and from. Addresses are not secret and are also shorter with respect to public keys. Generation of address requires the following operations:

1. Use public key.
2. Hash it.
3. Conversion of hash to text.

Users can produce any number of key pairs, either private or public. Consequently, addresses as required to permit for a changing degree of virtual secrecy. Addresses work as the user fronting "ID" on a blockchain for the individual user, and most of the time an address will be changed into a QR code to use it in an easier way.

When a blockchain shares virtual assets, it organizes them by fixing them to an address. To employ that digital asset, a user essentially demonstrates ownership of the address's matching private key. By signing digitally, a private key-based transaction can be validated using the public key.

Quick Tip

Wallets need to be safeguarded in the same way we do our physical wallets.

1.12.7 Private Key Storage

Blockchain users do not archive their private keys physically. In actuality, the wallet software securely archives it. The software wallet can store private keys, public keys, and corresponding addresses.

Generally, a private key is created using a locked random function, in the sense that rebuilding it is tough, if not impossible. If a user misses a private key, then any digital asset linked with that key is gone. If a private key is taken, the attacker will have full entree to all digital assets governed using that private key. The private keys security is so vital that several users use distinct secure hardware to archive it.

1.12.8 Ledgers

Ledger is referred to as a group of transactions. Through ages, written ledgers have been utilized to preserve transaction to track any goods and services. Recently, ledgers have been archived digitally, frequently in the large databases possessed and functioned exclusively by trusted and centralized third-party body for user community. Ledgers can have the following problems:

1. It may be missing or ruined; a user must believe that the owner is appropriately maintained and replicated in the system.
2. Transactions may not be legal; a user necessarily needs to believe that the owner is authenticating each acknowledged transaction.
3. A list of transaction is not complete, but the user necessarily needs to believe that the owner is counting all the legal transactions that have been acknowledged.
4. If the data is tampered, a user needs to believe that the owner is not tampering the historical transactions.

Actually, it is important for any ledger specially centralized to replicate data, authenticate transactions, contain all the legal transactions, and not to tamper past transactions.

A ledger executed using a blockchain can alleviate these problems using a consensus control in a distributed fashion. One feature of this is that the blockchain ledger will be replicated and shared between each node within the network.

Whenever new entries or transactions occur to a node, it will alert all the nodes of the network and inform that transaction is to be executed.

In this situation, this transaction is pending and not comprised in a ledger block.

Ultimately, a node will append new transactions inside a block and conclude the network essential consensus mechanism. This new block will be shared among all the network nodes and all ledgers will be reorganized by updating the new transactions.

1.12.9 Blocks

Users submit entrant transactions to the ledger by sharing these transactions to some of the nodes contributing to the blockchain. Submitted transactions are spread to the other nodes in the system. The shared transactions maintain wait queue or poll of transaction till they are appended to the blockchain by a specific node termed as mining node.

Mining nodes are the subsection of nodes that preserve the blockchain by broadcasting new blocks. Transactions are appended to the chain when a mining node broadcasts a block.

Flash Quiz

Do you know what is radio broadcasting?

In simple words, a block is a collection of authenticated transactions. Authenticity is safeguarded by inspecting that the fund provider in each transaction has signed the transaction cryptographically. This confirms that the fund provider for a transaction had an entree to the private key which could authenticate over the funds available. The additional mining nodes will inspect the legitimacy of all the transactions in a broadcasted block and will not admit a block if it comprises any inacceptable transactions.

After formation, every block is hashed thus generating a message digest that signifies the block. The modification of even a single bit in the block would wholly change the value of the hash. The block's message digest is utilized to assist in guarding the block from alteration. Meanwhile, all the nodes will have a replica of the hash of block and can then validate to make a certain block that has not been altered.

A block is the complex data structure. The data fields comprising a block typically consist of the following:

1. Block height or block number.
2. Block hash value.
3. Size.
4. Prior block hash value.
5. Merkle tree root hash.
6. Timestamp.
7. Block's: list of transactions.
8. A unique nonce value. It is a number used by the mining node to resolve the hash puzzle that provides privileges to broadcast the block.

Merkle tree is a data structure that is used in the header of the block (it is not required to store hash of each transaction in the block header). A Merkle tree associates the hash values of data until there is a root, which is termed as *root hash* of Merkle tree. The root is a well-organized control utilized to give summary of the transactions occurring in a block and to authenticate the occurrence of a transaction within. This data structure confirms that the data shared in a distributed network is legal. Meanwhile, any modification to the original data would be perceived and can be rejected. Figure 1.3 shows the structure of a Merkle, which is described as follows:

1. The lowest row signifies the summarized data. This is the transaction data in blockchain.

2. The second to bottom row shows that the data being hashed.

3. The second row's hashed data is then joined and hashed for the third to bottom row.

4. Root hash is shown on the top. It is joined and hashed to the value of hash AB and hash CD. In this way, all the prior values are joined and hashes are developed. The top row shows that the root hash, which combines and hash AB and hash CD (i.e., H_{AB} and H_{CD}, respectively). The root hash is a hash of all the previous combinations and hashes made.

Figure 1.3 expresses the connection between a Merkle tree and a block. The tree's lowest row comprises transactions 0, 1, 2, and 3. The root is archived inside the header.

The Merkle tree root hash will be further discussed in subsequent chapters.

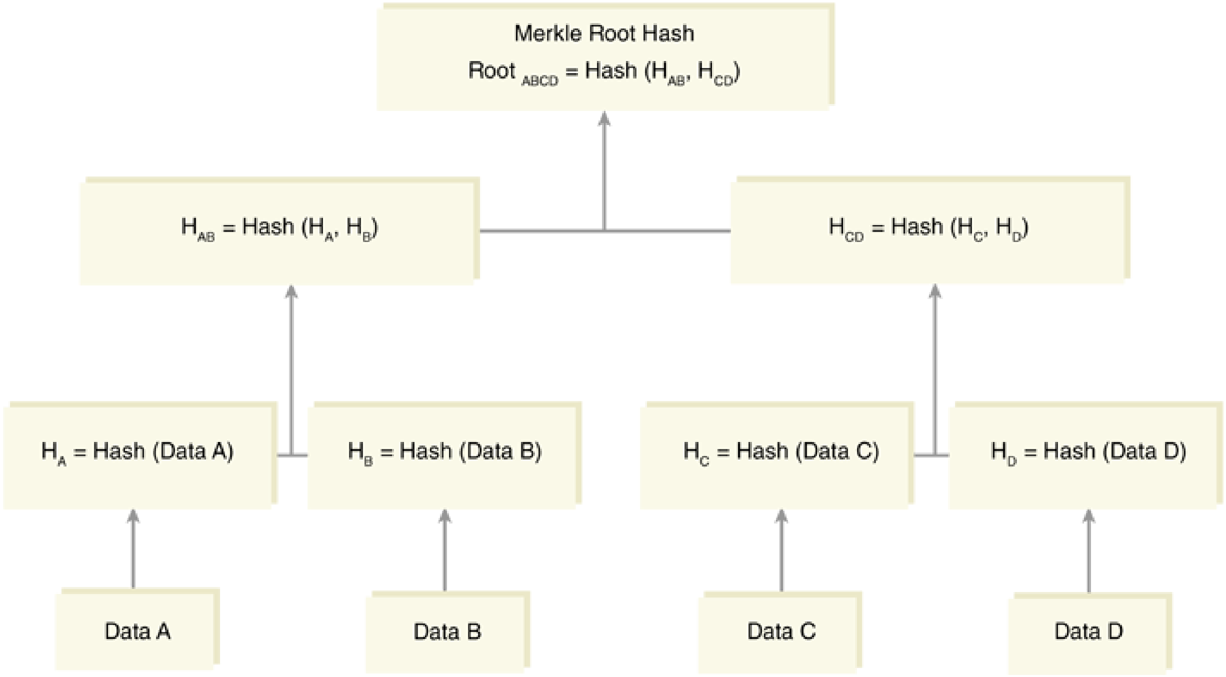


Figure 1.3 Connection among Merkle tree and block.

The whole header is hashed. The header value of the hash value is archived in the block and at the same time, in the next block. This supports to provide the transaction's immutability. Subsequently, the root hash will not tally if any alteration is executed to the transactions.

1.13 Chaining Blocks

Chaining is done between the blocks when every block encompasses the hash value of the prior header block, thus creating the blockchain (Figure 1.4). If an earlier broadcasted block were modified, it would have a new hash with a different value. In this situation, it would change all the succeeding blocks to possess diverse hashes since they comprise the hash of the prior block. This allows chaining to simply notice and discard any alterations to earlier broadcasted blocks.

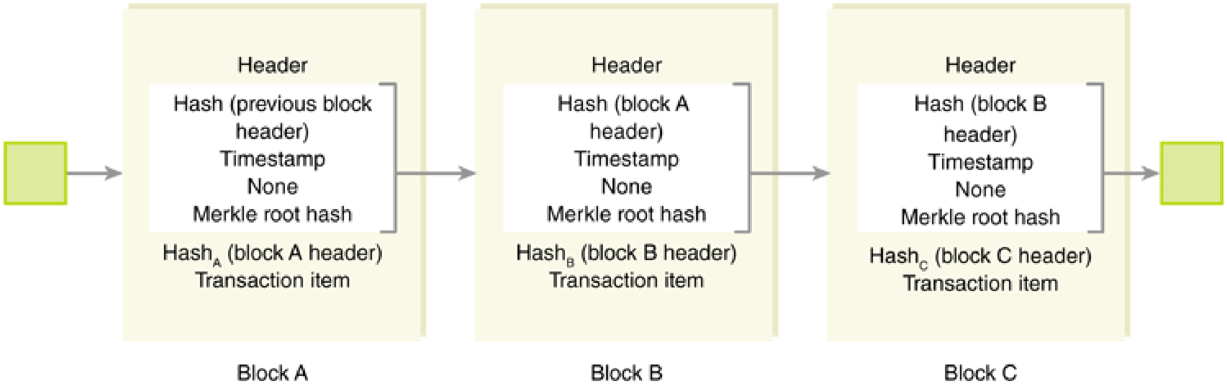


Figure 1.4 Chaining blocks.

1.14 Value Proposition of Blockchain Technology

The ultimate application (Figure 1.5) for blockchain is to provide holistic transparency and authenticate the precision of data of any transaction in the digital shared network ecosystem. The possible use cases of such technology are almost infinite. In addition to the digital currencies like Bitcoin, the large number of other probable blockchain use cases and services have been proposed and deliberated by industry experts and enterprises.

Blockchains can also be classified on the basis of their characteristics. These are as follows:

1. Development platform.

2. Smart contract accelerator.

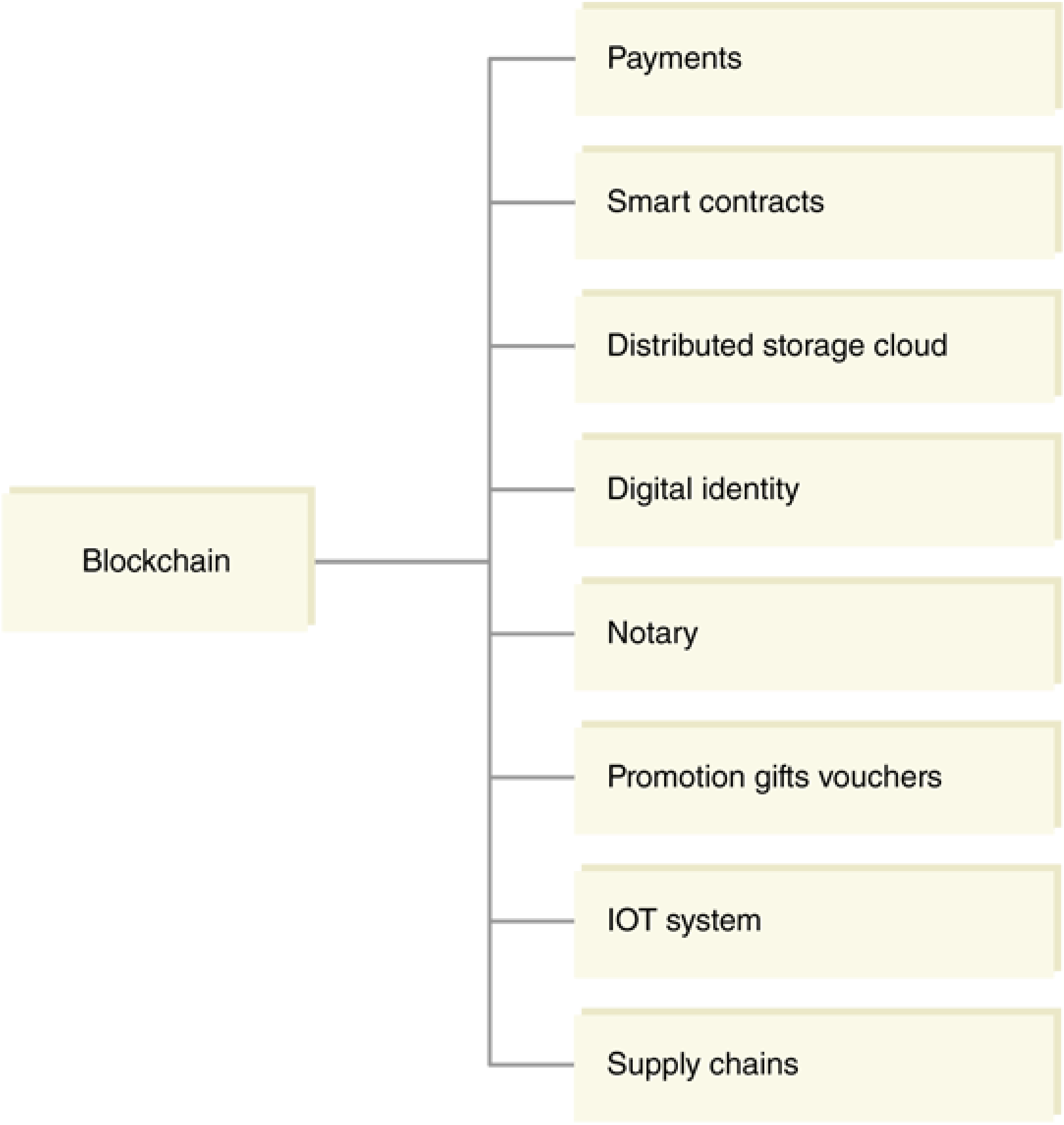


Figure 1.5 Blockchain business applications.

3. Marketplace.
4. Trusted-service utility.

1.14.1 Development Platform

Currently, the growth of blockchain use cases and services necessitates focused skills and the condition developer toolkits at the early stages. Currently, some of the developed blockchain platforms provided by public cloud providers over their cloud offerings deliver a cheap ecosystem for the developers to quickly prototype and examine blockchains before installing the production environments. Other illustrations in this area comprise technology platforms that permit protected distribution of data among the node's industrial networks over blockchain immutable ledgers and controls that provide blockchain empowered authentication of the transactions. These blockchain platforms provide the foundation code-based trust, possession and individuality, and enable the process and maintenance of enterprise blockchain use cases and services.

1.14.2 Smart Contract Accelerator

Smart contracts deliver a software code base edge to the blockchains. Smart contract accelerator is well-defined as a utility to achieve suitable functions to generate, govern, or enhance the digital assets value. When a smart contract is activated, it executes value created on virtual assets. The entire execution is seized in code and archived on the blockchain. This code is implemented when a prearranged rule exists. Actions are frequently accomplished and governed by the central establishments and bodies are alleviated to the blockchain in its place, without any intermediation among the transactions. This can be used at various junctures, such as:

1. Escrow of the bond, deed in the digital format.
2. Digital notarization for legal work.
3. Multi-party transactions.
4. Timestamping.

Currently start-ups are working with financial institutions and bodies to execute blockchains as a new operating model for the financial ecosystems.

1.14.3 Marketplace

Any healthy ecology needs a market for producing value. In the crypto financial marketplace, blockchain delivers a payment structure using cryptocurrencies and an ownership proof data structure using tracking of digital asset. This has permitted node to node-based marketplaces without any central authority-based intermediation delivering reachable, handy, and free independent decentralized trade system. Blockchain ecosystem can be utilized to match buyers and sellers, thus permitting them to perform operations using smart contracts. Blockchain mechanism is also being flaunted as a provider for future online manpower marketplaces in the background of the human economy that trusts on self-governing contract workforces and suppliers for temporary activities, and the distributed economy where customers increasingly become professionals. In such examples, blockchain ecosystem can safeguard that the service providers are not forced by any centralized system; henceforth permitting them to provide service offerings and payment exchanges and service operations that can be executed in a transparent atmosphere.

1.14.4 Trusted-Service Utility

In its character as a trusted-service utility, blockchain technology includes holistic features by enabling highly specific use cases for any objective imaginable. This widespread use of blockchains allow all the types of requests through a grouping of coded assets, ownership, trust, identity, money, and contracts is occasionally referred to as next version blockchain 2.0. At the front-end services, trusted service utilities developed on blockchain by means of smart contracts can deliver decentralized shared, protected services and assets to end users. At the back-end, several utilities exist in public blockchains which cannot be closed or delimited. Also, many enterprises now offer application programming interfaces permitting developers to develop applications using blockchain technologies and platforms.

In general, these existing and intended applications for blockchain mechanism are composed to produce a global disintermediated ecosystem, thus creating a far trusted value-based atmosphere that can help to march to a healthy and safe new financial opening in the public and private segments.

SUMMARY

Blockchains are a substantial novel way for technical developments, empowering protected transactions without the necessity for a centralization and authority. In early 2009, with Bitcoin using blockchain technology, it number of blockchain-based cryptography-based currencies have been growing. Perhaps more prominently, new use cases beyond the empire of currencies have been developing upon the foundations of blockchain technology.

The first use cases were cryptocurrencies with the shared delivery of a global ledger comprising all transactions in the distributed fashion. These transactions are protected with hashes generated using cryptography, and transactions are digitally signed and validated by means of private/public pairs of key. The history of transactions is concise with Merkle trees, to capably and strongly maintain the information pertaining to chain of events in a method, in which even the slight effort to modify or alter or tamper a historical transaction will also entail a re-accounting of all the successive transactions.

The application of blockchains is still in its nascent phases, but it has developed extensively comprehensive cryptographic rules and established methods. It is also expected that blockchains will be an added instrument that can be utilized to resolve fresh sets of difficulties. Economic establishments are expected to be the industries most affected by blockchains. They may require familiarizing or even wholly modify their methods to emphasize value exchange-based digital shared decentralized platform rather than just a storage of the values.

Digitization of the virtual assets is now also possible using blockchains. Enterprises that require preserving a record that may be public, such as property land title, marriage certificate, or birth certificates, should understand how their troubles might be tackled by blockchain technologies. Blockchains also have robust strength for archiving and maintaining the records of supply chain use cases. A blockchain maintains each stage in a product's lifecycle, from when it was formed in a workshop to when it was transported and afterward finally delivered to a warehouse, and lastly to when a customer bought it. It may be possible that even new businesses, such as digital solicitors and attorneys who can demonstrate a person had entree to an explicit portion of information by maintaining the hash value of it into the corresponding blockchain. There are various potential applications and prospects for blockchain technologies.

A blockchain trusts on current network, cryptographically, and accounting/ledger technologies but utilize them in a novel style. It will be significant that enterprises are intelligent to observe at the skills and both the rewards and drawbacks of using them. Once a blockchain is realized and extensively accepted, it becomes very problematic to remove it without dividing. Once something is documented/archived in a blockchain, it is typically there persistently, if it is even an error. For some enterprises, these are necessary characteristics, but for others, these might be contract rollers stopping the acceptance of blockchain.

Blockchain technologies have the authority to interrupt many businesses. To evade missed prospects and unwanted shocks, enterprises should initiate exploring as to how blockchain can help in their problem scenarios.

SHORT ANSWER QUESTIONS

1. What is a cryptocurrency?
2. What do you understand by timestamping?
3. How many types of blockchains do you know? Name them.
4. What is a permissionless blockchain?
5. How you can create a digital signature?
6. Define hash function.

7. What is a Merkle tree?
8. What is distributed ledger technology?
9. What is “block” in blockchain?
10. How you define consciences and why it is important in blockchain?

LONG ANSWER QUESTIONS

1. Explain with the help of a diagram the different types of blockchains.
2. What is proof-of-work? Why does it play an important role in blockchain?

CHAPTER 2

Decentralized System

Learning Objectives

After reading this chapter, you will be able to:

- Understand about the distributed decentralized databases.
- Analyze the concepts related to decentralized enterprises.
- Describe decentralization and disintermediation.
- Understand about the decentralized enterprise regulation.

2.1 Introduction

Today our communication ecosystem, such as the Internet, is decentralized and has helped create a new avenue and era. In the same manner, nowadays, a novel technology – blockchain – has the latent to distribute and disintermediate the mode in which we archive data and maintain information, possibly working on to a diminished role for one of the utmost significant controlling personas in our culture, that is, the middleman.

Blockchain technology permits the formation of currencies that are decentralized, self-operational digital smart contracts and intellectual digital assets that can be governed over Internet. It also empowers advancement of new management systems with more self-governing or shared decision pursuers, and decentralized enterprises that can function over a network of computing ecosystem without any manual interference. These use cases have guided many to equate blockchain against Internet, with associated forecasts that this technology will swing the equilibrium of power gone from centralized bodies in the arena of communication systems, industries, and even government or law.

We are now positioned at the perimeter of an innovative digital disruption. Internet is opening a new chapter of decentralization. After two decades of research, there have been extensive developments in cryptography and computer networks working in decentralized fashion. Consequently, in the advent of a thoughtful new technology referred to as blockchain, which has the possibility to primarily change the way in which society functions. Blockchain is an encrypted, shared distributed database which helps as an irretrievable and tamperproof public storage of information. It aids, probably for the first time in history, disparate individuals to reach a consensus on the existence of a specific transaction or occurrence without the necessity for a third-party controlling body.

Technical Stuff

Distributed database is a database in which not all storage devices are attached to a common processor and may be stored in multiple computers, located in the same or dispersed physical locations.

Blockchain technology has the budding power to diminish the character of one of the most significant financial and controlling actors in our culture, that is, the middleman. By permitting people to handover an exclusive piece of digital assets or data to an individual in a harmless, protected, and incontrovertible way, the technology can produce digital currencies that are:

1. Not supported by any central authority.
2. Self-administering digital smart contracts whose implementation does not require any manual interference.
3. Decentralized markets that object to function unrestricted from the reach of guidelines.
4. Decentralized network platforms that will be gradually tough to control.
5. Internet empowered possessions that can be governed, just like virtual digital assets termed as smart property.

Blockchain has the power to grow in a new arena considered by decentralized payment systems, virtual assets, decentralized power, and even decentralized law ecosystems. It permits cooperative enterprises and community-based establishments to become more lenient, transparent, and endorse greater contribution, possibly renovating how organization governance and self-governing organizations function. The technology might influence capital markets by allowing average citizens to execute monetary bonds/holdings by means of a few lines of code.

Quick Tip

A payment system is any system used to settle financial transactions through the transfer of monetary value.

Outside of these prospects, blockchain has the likelihood to essentially modify the manner in which an individual shapes their businesses. The technology can be utilized to produce new IT-based organizations known to as decentralized enterprise and decentralized-independent enterprise. These enterprises can reorganize specific characteristics of old-style corporate governance utilizing software utilities, empowering bodies to gain the paybacks of formal corporate building blocks, while sustaining the elasticity and scale of casual online sets at the same time. These enterprises can also be functioned freely, without any manual interference and involvement. They can possess, share, or exchange assets and communicate with other persons or machines, asking new questions around the outdated philosophies of legal character, distinct body, and accountability.

2.2 Distributed Decentralized Databases

Since blockchain are decentralized and encrypted data stores, they have started to influence how we connect and distribute the data online. Not only are they transforming the method of governing the Internet, but they are also progressively observing it as a method to enable machine-to-machine (M2M) interaction of Internet-powered assets. It is good to recognize that blockchain can be used for goods and services, with very less-to-negligible charges and no centralized governance.

Blockchain also permits the sharing of information in a method that is one-way decentralized and also secure at the same time. Data can be broadcasted in encrypted encoded set-up and disseminated across various nodes, developing it virtually so that it is not achievable for any single object to edit. Initial instances comprise unidentified dispersed storage ecosystems that utilize blockchain technology and other node-to-node technology to inspire an individual to use any additional volume existing on their drives.

With respect to the viewpoint of end users, these robust platforms can be observed as an identical to widespread centralized cloud management platforms. Although at a technical level, these function totally inverse. In these schemes, nodes are recognized with a digital currency for archiving other individual data, in which node in exchange can utilize the same rewards to recompense for archival of their own data on other nodes. Because of this reward system, individuals who utilize these facilities are stimulated to lease out their own storage. Consequently, they can grant admission to the pooled storage drive of the network. Architecturally, the decentralized, encoded, and encrypted characteristics of these platforms develop apparently tamperproof, without centralized body we are legitimately intelligent to get the file content on the network or can stop any transaction.

Not only limiting to governing data, the developers are also discovering the possibilities of blockchain to permit distinct individuals to strongly reach the consensus over Internet-enabled computing resources and mobile devices. Consensus nodes could validate that their own decisions were calculated, and because of encryption any blockchain-based consensus building system would be resilient to hacking. Consensus votes and substitution/delegation competitions would not be extended and It will not require trusting on the unreliable pieces of paper. These could be securely remunerated on the computing resources.

Flash Quiz

Can you deploy intranet based DNS in your local area network?

In blockchain, data store decentralization is additionally observed as a technical change for Internet-based domain name system (DNS) domain name registry. Presently, DNS such as xyz.com are accomplished using the Internet Corporation for Assigned Names and Numbers (ICANN), which is responsible for sustaining as to how individuals access sites. Blockchain applications pursue to topple this sequence by creating a distributed DNS that can archive blockchain link lists of domain names on a distributed database. Hence, removing the regulations to go through large organizations (government, ISPs to route the Internet traffics). In this manner, we can have tamperproof and secure DNS system using digital currency transaction that is worth several dollars.

Technical Stuff

The domain name system (DNS) is a hierarchical and decentralized naming system for computers, services, or other resources connected to the Internet or a private network.

A blockchain's capability to govern data from a diverse source that is not trusted may create an introductory device for conventional implementation of Internet of Things (IoT). IoT comprises of billions of linked Internet-empowered devices, among which all cannot be trusted, while some can even be vulnerable for the network. All these devices require a centralized location point that can simplify isolated, protected, and without trust M2M management.

In this situation, blockchain presents a well-designed resolution. Devices and other physical assets can be recorded onto a blockchain and converted into a smart property, utilizing smart contracts, physical assets to be managed over Internet, and even governed by other technologies. A blockchain can archive the association among Internet-powered machines at a specific moment, and smart contracts can assign equivalent rights and responsibilities of linked devices. Other than this, diverse associations and identities could be encrypted into the blockchain with respect to the specific cryptographically initiated properties such as lock keys and phones to safeguard, and only specific individuals have entree to the asset's characteristics at any given point of time.

Fact Alert

The concept of a network of smart devices (IoT) was discussed as early as 1982, with a modified Coke vending machine at Carnegie Mellon University becoming the first Internet-connected appliance, which was able to report its inventory and whether newly loaded drinks were cold or not.

2.3 Decentralized Enterprise

The synchronizing power of a blockchain is not merely restricted to simplifying the execution of machines. It also permits the action and links of diverse smart contracts that interrelate and network with each other in a distributed and decentralized fashion. Several smart contracts can be collected and organized to create decentralized enterprise that function conferring to explicit instructions and measures well-defined by code-based smart contracts and code. Thus, altering philosophy that enterprises and objects are nothing more than an assembly of contracts and associations in real terms.

By means of decentralized enterprises based on blockchain, individuals and machines can organize functionalities using a collection of code-based smart contracts, without the prerequisite to integrate into old-style corporate objects. Management and transparency can be realized by tracking and storing transactions into a blockchain, dropping operative costs, while delivering a more crystal clear and auditable traces of every transaction-based decision making. Corporate governance representations can be simulated by distributing policy-making power to various parties by means of multiple signature mechanism, which averts the implementation of an act until several bodies give their respective consent to a transaction.

In contrast to old-style enterprises, where administration is focused at the top executive level, the decision-making progression of a decentralized enterprise can be coded into software source code. Stakeholders can contribute in policymaking through decentralized consensus model, distributing bodies through the enterprises without the necessity for any trusted centralized body.

By simplifying management and trust, a blockchain permits new methods of collective deed that have the power to sidestep present governance fiascos. In practice, it can therefore resolve many of the shared problems linked to transparency and corruption intrinsic in the policymaking of several enterprises. Big hierarchical enterprises are both deficient and incompetent. Their limitations are, for the greatest part, owing to extreme centralization, proxy decision-making, controlling apprehension, and occasionally even corruption. Using blockchain, these inadequacies could disappear. Communications and enterprises can be designed and defined by smart contract, and individuals or machines can communicate without the necessity of trusting other bodies. Trust does not time-out with the enterprise, but somewhat within the safety and auditable trace of the primary code, whose processes can be analyzed by millions of nodes. In that wisdom, decentralized enterprises can be treated as open-sourced enterprises.

With the passage of time, Internet-powered devices become more independent and self-directed. These devices can also utilize decentralized enterprises and the blockchain to organize their communications with the external world. Therefore, we could observe the appearance of decentralized autonomous independent and self-directed enterprises that come into contractual associations with persons or other machines to generate a complex network of independent and self-directed agents cooperating with each other, conferring to a set of pre-defined, and self-empowering regulations.

Flash Quiz

Do you know how the connected car concept works?

Decentralized enterprise is an explicit kind of enterprise, which are independent and self-directed and no longer want nor notice their parent nodes. They are self-reliant and can collect monetary values such as digital currencies/physical assets. Decentralized enterprises can also take fee from the users for providing them the services and assets, in order to reward other party (node) for the properties (such as computing resources) they require. If they accept adequate assets to function on their own, they can consequently survive self-sufficiently of any third party. If a decentralized enterprise is independent, no one, not even parent node can regulate it after it has been organized and implemented on the blockchain.

Therefore, a malicious decentralized enterprise could be analogous to a biotic virus or an irrepressible power of nature.

2.4 Decentralization

One of the stimulating features of blockchain technology is that it is completely decentralized, in comparison to be recoded or stored in one central juncture. This eradicates the requirement for controlling central establishments and instead fingers governance back to the specific user.

2.4.1 Blockchain Decentralization

The decentralized characteristic of blockchain depicts that it does not trust on a control that is centralized. In the absence of a central body, the system works reasonably and significantly more protected. The technique in which data is stored onto a blockchain exemplifies its most radical excellence – its worth of decentralization. Rather than trusting on a central authority to safely transact with other individuals, blockchain uses innovative consensus controls among nodes network to authenticate transactions and store data in a method that is immutable. As blockchain is a book of recorded information, it is tremendously significant that the data being recorded is truthful and precise.

2.4.2 Consensus Control

A consensus control is a collection of guidelines that defines how interaction and broadcast of data between digital devices, such as nodes, executes. Consensus is realized when adequate node is in agreement about what is accurate and what should be stored onto a blockchain. Consequently, consensus controls are the maintenance guidelines that permit digital devices that are dispersed across the globe to truthfully come to a contract, letting a blockchain network to operate without being malicious.

As system does not trust on a central body, the dues that are generally accumulated by these enterprises are no longer an issue. Thus, transacting on the blockchain can be observed to be inexpensive, as the only charges done by the nodes participated are the insignificant fees utilized to recompense the miner or coiners that execute a node on the network.

Moreover, the information stored on the blockchain can be convinced to be accurate as it is mostly impossible to alter owing to there being numerous replicas that involve a complex consensus to be amended also, in blend with the character played by hash functions in the network.

The information is developed even more protected by the circumstance that there is no dependence on a central storage system, dropping the risk of it being missing or ruined or tampered. Hacking one storage point would affect no damage of data since the data is archived on several devices around the globe. With respect to this, Bitcoin is the most weather-beaten and robust platform, having endured volleys of tried hacks, all of which have been ineffective.

Those who contribute in the blockchain have the choice to subsidize their computer storage and copy a replica of the blockchain to safe the data that is stored on it.

We will now learn the differences between centralized, decentralized, and distributed systems.

2.4.2.1 Centralized System

Centralized systems are based on the client–server architecture. Here, multiple nodes are connected to a centralized server. It is the most general type of ecosystem that exists in any enterprise where the request is transacted by the client and it is processed by the server as a response (Figure 2.1).

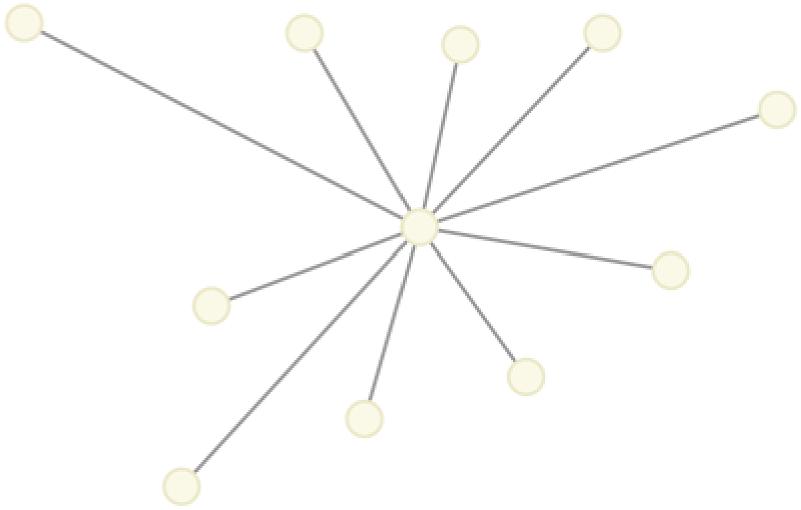


Figure 2.1 Centralized system.

2.4.2.1.1 Characteristics of Centralized Systems

Following are the characteristics and essentials of centralized systems.

1. **Global clock:** The system comprises central system-based node, a master server, and many nodes work as a client or slave. All the nodes are synced on the basis of the global clock that exist in the central server.
2. **Single central unit:** This consists of the central unit that assists/synchronizes all the other system nodes.
3. **Failure prone:** If the central system fails, it becomes the reason of the entire system failure. This is obvious as no other node will work as they will not get any response for their sent and receive requests and responses.
4. **Scaling:** Horizontal scaling will oppose the single unit based server feature of this system. Therefore, the central system proposes on vertical scaling.
5. **Components and architecture:** Components of the centralized system comprise of nodes that can be any device such as computer, mobile, etc., (it can work as client/server). It also comprises

communication objects, such as cables and Internet components. If we talk about architecture, the central node serves all the other nodes of the system.

2.4.2.1.2 Limitations of Centralized Systems

There are various limitations while implementing the centralized systems. Some of these are listed below.

1. Centralized systems bear the limit of vertical scaling. After a certain point, if we scale the hardware/software of the central system, the performance of the central node will not be affected in the way it should be.
2. Blockages can be observed when traffic increases on the central server. As the number of ports is limited in the system, it gives finite options for all the clients to relate to the central system. It originated problems such as denial of service (DOS) or distributed denial of service (DDOS).

2.4.2.1.3 Advantages of Centralized Systems

Following are the benefits of centralized systems.

1. Central systems are easy to secure. Service is very easy between the client and the server.
2. These systems come with a personal touch. Here, the client node can be customized on the basis of the client expectations.
3. This system looks for dedicated computing resources such as CPU, memory, and storage.
4. This system is cost effective for small deployments up to a certain level, and consumes less funds to establish the ecosystems.
5. System updates are quick in the central systems. It can work as a drive where nodes can be updated as a single unit of an update drive.

6. Decommissions is an easy process in a centralized system, as it is easy to detach the nodes from the overall network since these nodes are independent.

2.4.2.1.4 Disadvantages of Centralized Systems

There are various problems while using the centralized systems and it is dependent on the use case. The following are some disadvantages of such systems.

1. Central systems are network dependent. All the nodes will stop working if the network connectivity is lost and the server will fail at connectivity end.

2. It may be possible that the system fails abruptly. This exit is not elegant and smooth.

3. Backup of the central system is very critical because if it does not exist, we will lose the entire data from the server node.

4. Maintenance of the central system is very complex as this should always be available. It is unproductive and unethical to make a server down for maintenance. So, updates are done on the live system that is risky, expensive, and complex.

2.4.2.1.5 Applications of Centralized Systems

So far, we have discussed the advantages and disadvantages of centralized systems. We will now look at some of its applications.

1. It is easy to perform application development level jobs as it is easy to set up the systems that can send and receive request from the client–server environment.

2. It is opted for synced data analysis jobs, as the date is stored at a central location.

3. It is the best candidate to be used for the use cases such as personal computing, databases, game applications, development, and test.

2.4.2.2 Decentralized Systems

Decentralized systems are now receiving a lot of acceptance, principally because of the huge publicity of cryptocurrency – Bitcoin. Nowadays, several enterprises desire to treasure the application of decentralized systems.

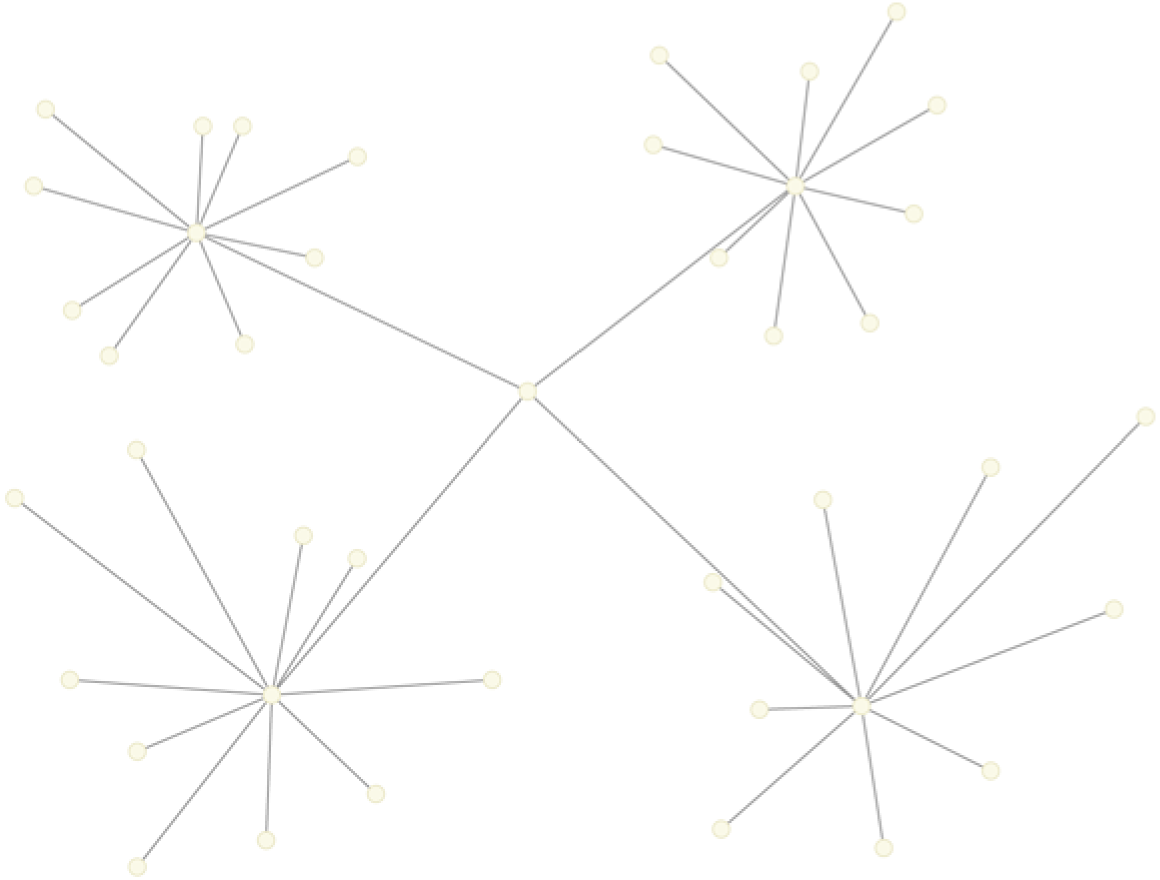


Figure 2.2 Decentralized system.

Every node of decentralized systems is a decision-maker and serves as the server node. The resultant outcome of the system is the combined total summation of the responses of participating nodes of the systems. It is important to understand that there is no single unit that receives/responds to the request (Figure 2.2).

2.4.2.2.1 Characteristics of Decentralized System

The below text gives the insights for decentralized system and talks about the characteristics of it.

1. There is no global clock such as that of the central system. Every node is autonomous and owns different clocks and there is no synchronization among them.
2. It comprises various central processing units (we can say) as each node is empowered to process the data and work as a central unit.
3. If one of the nodes fails, system does not fail; it only effects that part of the system.
4. Unlike the central systems, decentralized systems can scale vertically. Here, the node can add computing resources as per performance requirements.
5. Components of the centralized system are similar to that of the central system and comprise nodes that can include any device such as computer, mobile, etc., (can work as client/server). Architecturally, all the nodes follow peer-to-peer (P2P) architecture. There is no super user over all the nodes. Here, at the consensus building time, any node can work like the master node and cast its vote for decision-making, which will not lead to any type of authority or sovereignty.

2.4.2.2.2 Limitations of Decentralized Systems

There are various limitations while implementing decentralized systems.

1. At the enterprise level, it may be complex to form the consensus as all the nodes are decision-makers and can behave in their own way.

2. It is not vouched for small-scale deployments as it will not be feasible to maintain because of cost benefit analysis.

3. There is supervisory node that exists so that the governance cannot be regulated for a single node; rather it is achieved at a system level.

2.4.2.2.3 Advantages of Decentralized Systems

The following are benefits of decentralized systems.

1. There is no traffic issue as the load can be balanced across nodes. Therefore, blockages are very less with respect to performance.

2. This is considered as a highly available system as one node can take charge of others in case of failure at different levels such as node, network, etc.

3. Nodes are independent, and they look for self-governing mechanisms and achieve autonomous nature and hence provide better overall control.

2.4.2.2.4 Disadvantages of Decentralized Systems

There are various problems while using decentralized systems and it is dependent on the use case. Following are some disadvantages of such systems.

1. Because of its autonomous nature, a decentralized system is complex and difficult to realize huge transactions. There is no standard set of instructions to control others.

2. Automatically, it is difficult to observe in the global network where node is failed. It is rather achieved by pinging each node. Checks are done for its availability. Multiple checks require dividing the work among the nodes on the basis of the expected output as to what the node generates.

3. Node search is difficult for knowing which node responded to corresponding request. In a decentralized system, it is difficult to know which node is actually serving the system for decision-making/processing transaction/voting for any request.

2.4.2.2.5 Applications of Decentralized Systems

Following are some of the applications of decentralized systems.

1. It can be used for maintaining nodes that work in the private mode.
2. The most common application is cryptocurrency for which it is getting all the limelight. Public address is visible, but these addresses are changeable, and therefore, very tough to trace.
3. It can be used as a distributed database where all the node storage is used to store data and are split over the network.
4. There are various vertical based use cases that exist. Some of these have been discussed in Chapter 1.

2.4.2.3 Distributed Systems

It is considered as a network of nodes that collectively works as a single system. The nodes can be positioned in a nearby location to each other and linked physically in a local network node. Blockchain network nodes are geographically distributed nodes. Search engines use distributed systems where hundreds of nodes act together which crawl the web and produce appropriate outcomes (Figure 2.3).

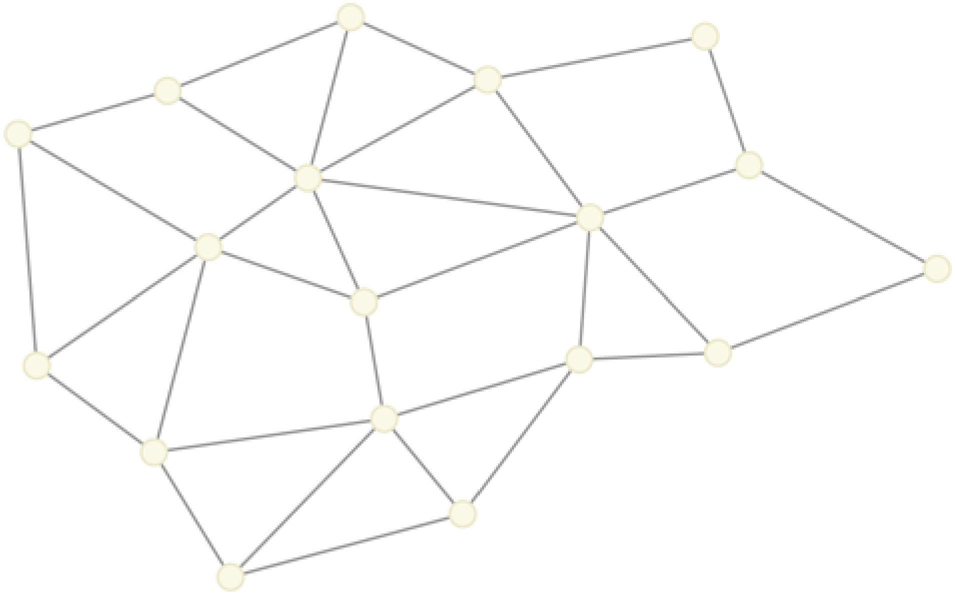


Figure 2.3 Distributed system.

2.4.2.3.1 Characteristics of Distributed Systems

Following are the characteristics and essentials of distributed systems.

1. Concurrency works in distributed systems and consensus controls are required to have agreement on transaction values and log commands.
2. Decentralization nodes own their own clock, but no centralized global clock exists in the network system.
3. It provides high availability. Failure of the network node does not affect the whole system. The whole network works of one of the nodes fails.
4. It can span and scale both horizontally and vertically.
5. Components of the distributed system are like the central system and decentralized system, which comprise nodes that can be any device such as computer, mobile, etc. (which can work as client/server). Architecturally, all the nodes follow P2P architecture. Here, different parts of the applications are executed on different nodes of the system.

2.4.2.3.2 Limitations of Distributed Systems

There are various limitations while implementing distributed systems.

1. The distributed system is very difficult to design and implement as algorithms are difficult to develop. There is no global clock, so for a temporary period, there is no sequencing of commands and logs. All the nodes possess varied latencies that impact the design in a big way. Node increment in the network results in the increased complexity.
2. It is difficult to design and debug algorithms for the system. These algorithms are difficult because of the absence of a common clock. Thus, no temporal ordering of commands/logs can take place. Nodes can have different latencies which have to be kept in mind while designing such algorithms. The complexity increases with increase in the number of nodes.
3. The geographical view of the node system is very difficult to visualize. Hence, collective decision-making process is really difficult in the distributed system.

2.4.2.3.3 Advantages of Distributed Systems

A distributed system requires low latency in comparison to centralized system because of high level of geographical-based nodes spread. Therefore, it takes less time to get the response of the request.

2.4.2.3.4 Disadvantages of Distributed Systems

Consensus mechanism is difficult in the distributed system because it looks for logging into the events at the absolute time when it happens, and it is not possible.

2.4.2.3.5 Applications of Distributed Systems

We can use distributed systems to achieve cluster computing where various computer nodes are joined in a group to attain the common objective. It works as a single computer system.

We can use it to deploy grid-computing ecosystem also. Here, all the computing resources such as memory, CPU, etc., are pooled together and shared in a manner so as to get the power of a supercomputer. It supports service-oriented architecture, multitier applications, search engines, etc.

2.5 Disintermediation

Disintermediation is the process of diminishing the practice of mediators or interventions or agents between manufacturers and customers. It helps to achieve transactions between two parties by removing the requirement mediators or middlemen or third party. To understand better, we can consider this example. Using disintermediation, we can directly invest in the stock market without using any financial intermediary institution such as a bank. Disintermediation is becoming the transformation disruption. It is hitting hard on the entire ecosystem, transforming the business model, and the entire financial system that is attached with rewards-based incentive system powered by mediation.

The following illustrative examples can help us understand this concept better.

1. Various social networking sites connect two parties such as product (users) and consumers (market offerings) without any mediators.
2. Many public transportation network company link the user with service products while not owning the single assets. It uses unique algorithm to support the disintermediation-based ecosystem.
3. The largest retails companies in the world, whether online or brick/mortar-based, never own warehouses, they help connect customers with manufacturers using transaction-based disintermediation model.
4. There are many hospitality portals that have changed the entire tourism and travel industry with the same model. It has removed all

the travel operators for hotel booking, ticketing, and other travel-related works using marketplace powered by the Internet.

2.5.1 Disintermediation in Blockchain

Blockchain intends to shoot disruption such as disintermediation by pushing it in business models to new levels. Blockchain looks forward to introducing disintermediation across the financial institutions working on the transactions.

Blockchain helps in transaction of goods and services using a web portal supported by various partner stakeholders. It permits transacting currency, assets, status, access rights, or all the things that create value chain across ecosystem. This indicates the use of digital assets and creation of a legitimacy, identity, and trust system in the neighboring economy.

Blockchain establishes the system governed by rules and regulations across the network, which is assumed to be a system that does not rely on trust, yet it is trusted by their users. It is achieved by ownership of nodes that supports transaction of goods and services.

Blockchain produces a network that can archive and authenticate ledger accesses, without using centralized platforms. Using this, no single node gets the privilege of single point of control.

2.5.2 Blockchain Disintermediation

Blockchain provides an outstanding system that can drastically transform verticals such as banking, financial services and insurance (BFSI), supply chains, and other transaction-based sectors. It provides huge prospects for novelty and growth, while diminishing cost and risk prone events and transactions. Blockchain can topple sectors such as finance, logistics, insurance, real estate, and healthcare, which are essentially driven by the mediators.

Disintermediation, which is based on blockchain intercession in any sector, offers venture attraction for many blockchain-based applications and use cases. Although the characteristics of blockchain drives application far ahead than just the revolution, it also has an incredible power to progress and augment productivity to present business developments, methods, processes, and transactions.

Blockchain has been considered as the paramount revolution since the beginning of the Internet. Several predict that this technology has the capability to transform the business processes handled today throughout the globe. Internet helps in transferring the information in digital fashion using decentralized system, and blockchain offers the transfer of assets by distributed, decentralized, shared, and disintermediated records of transaction database.

Blockchain mechanism is recognized to offer ledger keeping for Bitcoin cryptocurrency, but other applications can go beyond our imaginations, as blockchains can archive a series of records counting transactions such as payment, sales histories, purchase transaction, inventory control, logistics control, corporate accounts, and retail transaction.

It can also archive other forms of information that is not based on the transaction such as deed titles, agreements, affidavits, disclosure, minutes of meeting, industry-specific logs, audit logs, and legal assets.

These potentials put diverse blockchain applications into the reticle of race, law implementation, and compliance with audits. The race instructions must be well thought out at the initial phase as they influence the guidelines overriding a blockchain database, the archives, and the marketplaces it pursues to facilitate.

2.5.2.1 Blockchain Race Conditions

Mostly all blockchains assure to improve productivity and are prone to race conditions. The factors that propel this competition are decentralization and transparency.

To understand race-based competition, we need to emphasize that all the data is available to all the nodes using P2P network, irrespective of permissioned or permissionless open or closed network.

At the same time, as the data that is seized are the entries that may be unidentified, encoded, and encrypted reliant on the guidelines applying to blockchain, the data nevertheless offer in-depth vision into participating nodes for the financial transactions at specific timestamp without the expected vagueness about the reliability of the information.

Private blockchains restrict and they will be only accessible by those nodes that are entitled to and have permission to participate in the mining or transaction task. The data itself may not be accessible to all, but all the nodes will have entree to timestamps of the access and to the ID (using digital certificates) of those who made any transaction to it.

To be really effective and attain its holistic profitable possibilities, blockchains will frequently be organized up in P2P networks together with genuine or possible race condition (competition) nodes (contestant nodes).

Race condition (competition) regulation pursues to forbid limitations of race condition between contestant nodes. It may arise as an outcome of some arrangement of contract or agreement among contestant nodes or via one-sided behavior by a leading market monopoly. Valuation of blockchain will probably come under the first group. The matters confronted by race condition (contestants) nodes organizing or contributing in blockchains will mostly come under the following groups:

Fact Alert

The term "race condition" was first mentioned in 1954 by David A. Huffman in his doctoral thesis "The synthesis of sequential switching circuits".

1. Association/membership guidelines.
2. Data exchange.
3. Standardization.
4. Joint venture and merger acquisition-based regulations.

The preliminary fact for race condition (competition) rule investigation is that every enterprise essentially regulates autonomously the rule which it proposes to accept on the market and the environments it proposes to bid to its consumers. The rather overgeneralized race condition (competition) rules impede any straight or unintended interaction between contestants. This makes some circumstances of race condition to not meet the standard. Race conditions of the market in question have reasonable explanations for such interactions and a conclusive outcome is not so competitive.

2.6 Decentralized Enterprise Regulation

Though it may appear that decentralized enterprises and smart contracts could change various purposes of rules and governments, the conventional placement of blockchain utility is improbable to eradicate the character of these centralized establishments. It may somewhat swing the equilibrium among law and planning, necessitating another controlling instrument to effectively govern the culture.

As time passes, the extensive arrangement of smart assets such as contracts, property, and cryptographically stimulated properties will increase a sequence of significant problems to the present law ecosystems. So far, the blockchain is and will essentially endure a governable technology. Though governments primarily have a tough time understanding how to control a worldwide and decentralized network, they will ultimately be made to consider. As far as there are centralized obstruction points, governance can be realized via the secondary directive of numerous mediators and online administrators that truly run the network system – a duty that has been significantly enabled by rising absorption and centralism of Internet services in current years.

A similar condition will be expected to occur in the background of blockchain technology. In this world that is managed by decentralized information and enterprises, controlling intermediate mediators will persist. If vulnerable, agencies and administrative bodies could implement a sequence of recorded legislator actions to control the developing online network and to hold regulator over the blockchain environment. Foremost, the Internet service providers could be stressed to stop encoded data transient over their network, stopping Internet service providers from communicating any traffic workload to and from the autonomous independent decentralized enterprise. Another guideline could be agreed to necessitate that business or individual run online mediators such as search engines can resolutely evade cataloguing any blockchain utilities to shove this technology to unreliable grey marketplaces. Also, centralized establishments could challenge to freeze the development of illegal blockchain-based enterprises by looking to impeach software designers or operators of blockchain-based enterprises. In addition to this, push could be functional to hardware original equipment manufacturer (OEM) such as instructing that these enterprises persistently interrupt their products to stop the use of specific encryption methods or to deploy actions to trace.

The consequence would be an uncivilized misuse of government authority, and many of these methods would expect to freeze the monetary advantages that permissionless blockchain technology intent. It would signify a withdrawal from existing efforts to help the independent exchange of data, thoughts, and trade on Internet, which may eventually raise substantial legitimate fundamental rights. By pushing measures on software developers, the administration agencies would consequently command the code that developers write.

Likewise, rules that necessitate making the wrecked hardware to stop encryption would disturb central human rights by compelling people to interconnect in a method that confines their capability to guard their individual secrecy.

In reality, the application of blockchain technology could ultimately land into the identical trick as the actual Internet. The Internet was initially observed as a foundation of discrete freedom and liberation. Nowadays, while it is indisputable that it has heightened the face of free dialog, it has also developed a means for shadowing and surveillance. Legal agencies leverage Internet technology to govern volume-based scrutiny. Internet publicists trace consumers traversing websites to well target commercials. Search engine archives each click on various websites over the Internet, just revealing how it leverages user information with imprecise declarations in its confidentiality sections.

Deprived of the suitable lawful protections, it is reasonable that the growth of blockchain technology could follow an analogous track, marching to amplified investigation. Despite the prospects for the progress of global systems, the agencies could certainly utilize the technology to work out a substantial degree of governance over the individual's communications and online networking. As financial operations and social communications grow in a networked system, the mechanism could gradually be used to control an individual's nature, to guarantee that they are dependable with the law or with the predetermined responsibilities that they have arrived into. The blockchain could be utilized, for example, to achieve access rights-based identity, and trace or just monitor numerous online actions. Every transmission, consensus-based casted vote, buying action can be logged on the blockchain, generating a lasting record that will possibly thrust the restrictions of confidentiality law. In addition to this, controllers might necessitate that the online service provider within the blockchain environment to trash to contract or say no to perform tasks with anonymous parties that have not fulfilled specified requirements, discouraging the autonomous behavior of the blockchain, and involving it into an influential instrument of scrutiny and govern.

Outside the developing dynamic behavior amid regulations (law) and design architecture, blockchain-based utilities could also increase thought-provoking and fresh lawful queries regarding the governance of decentralized enterprises. In contrast to old style online tools, which are eventually kept on a centralized server at a specific location, decentralized enterprises are arranged straight on the blockchain. They do not exist at any agreed physical site, but somewhat function irrespective of any nationwide limitations or authorities. While decentralized enterprises, which are aggressively governed by online consumers, could be managed much like an LLP enterprise, the equivalent cannot be termed as decentralized enterprise. Contrasting to current enterprises, decentralized enterprises are not possessed nor governed by some sole business or state body, nor any distinct individual. So far, they can cooperate with the citizens in a mechanism that may grow to precise privileges and responsibilities. Decentralized enterprises can consequently have a noteworthy consequence on some bodies and may become the basis of certain offences or misconducts.

SUMMARY

A decentralized system depends on several service nodes. As an outcome, blockchain technology exists in on a peer-to-peer network. It actually cannot exist with a single node or point-based link. In its place, it needs a group of other nodes to link in, so a precise transaction can be completed in the network.

It is a continuously mounting chain of sequence of blocks holding information. It holds timestamp, and a connection to the prior block. The best thing about blocks is that they are constructed and cannot be altered after they have been documented and archived. It means that once the data is recorded, it lasts forever. It cannot be modified post development.

By design, it ensures that blockchains are protected. Since the data is timestamped, it is also recorded in a manner that we must concurrently access each single node to hack the system.

Since centralized systems have a single central system of information recording, it is tremendously vulnerable to hacking. In decentralized blockchain technology, there is an efficient mechanism to eradicate this weakness. Consequently, recording data on a P2P network system is optimized with respect to security.

Currently, both the networks (centralized/decentralized) occur in widespread platforms. Centralized systems definitely have their period and positions, particularly for monitoring all the transactions on a network. Decentralized, distributed P2P blockchain are dignified to become the principal entity to touch large enterprises or even great utility for single individual. But it is a certainty that everyone requires blockchain.

SHORT ANSWER QUESTIONS

1. What is a payment system?
2. What are the pros and cons of centralized systems?
3. What do you understand by disintermediation in blockchain?
4. How is race condition used in blockchain?
5. What are the pros and cons of decentralized systems?

LONG ANSWER QUESTIONS

1. Explain how blockchain supports digital currency that does not require central authority.
2. Enumerate five characteristics of centralized systems.
3. Enumerate five characteristics of decentralized systems.

CHAPTER 3

Hash Functions

LEARNING OBJECTIVES

After reading this chapter, you will be able to:

- Understand about hashing.
- Analyze the concepts related to message authentication code of hashing.

- Know more about Secure Hash Algorithms and Secure Hash Algorithm Version 3.
- Understand about the distributed hash tables, hashing, and data structures.
- Describe the hashing in blockchain mining.

3.1 Introduction

Hash tables are one of the oldest and most widely used data structures in computer science, tracing their origins back to January of 1953 when H P Luhn wrote an internal IBM memorandum creating a new line of history in one page. Hashing applications are abundant in literature, for example, computer compilers use hash tables to keep track of declared variables in source codes in the form of symbol table. One common use of hash functions is in programs that play games and are also used in online spelling checkers as well. Designing of secure and efficient hash functions is important for many practical applications including blockchain technology.

Fact Alert

Hans Peter Luhn was a researcher in the field of computer science, library and information science for IBM, and creator of the Luhn algorithm, key words in context (KWIC) indexing, and selective dissemination of information (SDI). His inventions have found applications in diverse areas such as computer science, textile industry, linguistics, and information science.

3.2 Hashing

Hashing usually means compressing, that is, the output of a function is shorter than the input to the function. Many times, the hash of a function takes an input of arbitrary length and produces fixed length string of 128, 160, 256, or 512 bits. Hash functions are frequently used in many parts of cryptography, and there are many hash functions with varying security properties. In the context to blockchain technology, the transactions are taken as an input and run through a hashing algorithm (e.g., Bitcoin uses SHA-256) that gives an output of a fixed length. Table 3.1 gives the hash value of different text.

One can observe that in the case of SHA-160 bits, no matter how small or big the input, the output will always have a fixed 160-bits length. This becomes critical when someone is dealing with a huge amount of data and transactions. Thus, instead of conveying the input data that could be big in size, one can just convey and use the hash and keep track. Let us now see the various properties of hash functions and how they get used and implemented in the blockchain.

Table 3.1 **Hash value of different text***

Input	Hash Value with 160 bits (SHA-1)
Hello!!	0X2493aee21afffb24a33a22cb5543c9dde8620911
Hello. How are you?	0X9a835fb38e3ee482eca5314f80ce214fa024a3d6
Hash functions are frequently used in many parts of cryptography, and there are many hash functions with varying security properties. In the context to blockchain technology, the transactions are taken as an input and run through a hashing algorithm (e.g., Bitcoin uses SHA-256) that gives an output of a fixed length.	0X c5f70ad7ccaff85aee62c616cf2ea545494b6daf

* Authors have used library from openssl.org for this table.

Activity

You may download the library from openssl.org and try to explore it. This will provide many cryptographic and related algorithm implementations.

3.2.1 Characteristics

In order to have a secure hash function, it should possess some properties. Generally, the accepted secure hash function should be one-way and collision resistant. There are additional properties that may be necessary when a hash function is being used in blockchain technology. These are discussed in the following subsections.

3.2.1.1 Deterministic

Deterministic means that for a given input value, it must always generate the same hash value no matter how many times parsing is done with a particular input through a hash function. This is critical because if it produces different hashes every single time, it will be impossible to keep track of the input. Deterministic algorithms are one of the most practical and by far the most studied and familiar type of algorithm, as they can be efficiently run on real machines.

3.2.1.2 Efficient in Computation

How efficiently the hash function produces hash values for the input within a set of data determines its efficiency. While analyzing the hash function, it is generally assumed that they have a complexity of O(1). The hash function should be capable of returning the hash of an input quickly with the linear traversal through a string or byte array of data that is to be hashed is very quick. However, the fact that hash functions are generally used on a primary finite set of inputs that are supposed to be limited in size or blocks of data implies that the whole operation should be fast and deterministic.

3.2.1.3 Collision Resistance

A hash function is said to be collision resistant only if it is hard to find two different inputs such that the hash value produced by it is the same output. This implies that it is practically impossible to find two different strings A and B (A ≠ B) such that H(A) = H(B). We may explain this with an interesting concept *The Birthday Paradox*; it is a paradox because the mathematical truth of the situation contradicts intuition. Basically, it needs minimum 366 people in a room to be 100% sure that at least two of them share the same birthday. However, one can mathematically prove that 70 persons are required in the room for having a chance of 99.9% for the same birthday, and it is an even chance (50%) when there are only 23 people in the room. This is the "paradox" that is counter-intuitive, where only 23 random people are required to share a birthday. In mathematics, it is said to be a problem as the calculation of the probability is quite complex. However, an algorithmic method of quickly calculating the probability of the birthday problem is frequently used in cryptography to break hashed codes, which is typically known as birthday attack. The same logic that drives matching birthdays also drives the probability that one can find collisions with a hash function. In other words, given a uniform hashing function that outputs a value between 1 and 365 for any input, the probability that two hashes would collide in a set of 23 values is also 50%.

Activity

Collect birth dates (not year) of around 30 friends and see how many fall on the same day. Use this information to explain the *birthday paradox*.

3.2.1.4 Preimage Resistance

A preimage resistance conveys that the given H(A) is infeasible to determine A, where A is the input and H(A) is the output of the hash function. However, if the total number of input vectors are limited and small in number, then there exist a possibility to know that the value of A (e.g., a six-face dice having a unique number on each surface). Finding the hash value of each surface is deterministic and unique. Therefore, while rolling a dice and the output is the hash of the number that comes up from the dice one can easily determine the original number on the dice. It is very simple; all that one needs to do is to find the hashes of all numbers from 1 to 6 and compare with the precomputed hash values on 1 to 6. Since hash functions are deterministic, the hash of a particular input will always be the same. Thus, we simply need to compare the hashes and find the original input. However, this only works when the given amount of data is very less. What happens when a huge amount of data set is provided? In this scenario, the brute-force method is not practically viable because of the increase in the space points. But one may notice that the best case scenario would be on the first try itself. On the other hand, this would only happen to the luckiest person in the world, as the odds of this happening are in astronomical numbers. Let us see for a 128-bits hash function, what would be the worst case scenario to obtain the answer after 2^128 – 1 times. Basically, it means that finding the correct answer at the end of all the data. For an average scenario, it is somewhere in the middle; this means basically after 2^128/2 = 2^127 times. To put in proper perspective, 2^127 = 1.7 × 10^38, which is again a huge number. Thus, while it is possible to break preimage resistance via brute force method, the number of input vectors is small in number but impossible for a large set of data.

3.2.1.5 Avalanche Affect

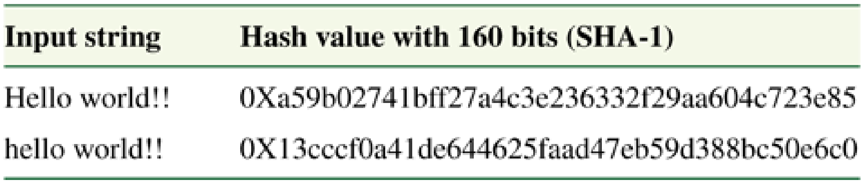
A small change in the input drastically changes the output of the hash function. In other words, two similar but not identical inputs should produce radically different outputs when fed through a hash function. Let us consider the following example of two strings in SHA-1 and see with one character change (just only the case change), there is a drastic change in output. Table 3.2 gives the hash value of two strings in SHA-1.

Technical Stuff

Avalanche effect, in physics, is a sudden increase in the flow of an electrical current through a non-conducting or semiconducting solid when a sufficiently strong electrical force is applied. The ability of most non-metallic solids to carry an ordinary electrical current is limited by the scarcity of electrons free to move in the presence of an externally applied electric field. A sufficiently strong electrical force can break free a large number of electrons from atoms that form the structure of the solid so that a large current can flow through the material. This avalanche effect is responsible for the phenomenon of breakdown in insulators and semiconductors, where it is called *Zener effect*.

A hash function is said to be a good qualifier if it demonstrates the avalanche affect such that in a single flip of a bit in the input causes a change of, on an average, half the bits of the output. This is an important property of hashing that leads to one of the greatest qualities of the blockchain – its immutability. Immutability is discussed in detail in previous chapters.

Table 3.2 **Hash value of two strings in SHA-1***



Input string	Hash value with 160 bits (SHA-1)
Hello world!!	0Xa59b02741bff27a4c3e236332f29aa604c723e85
hello world!!	0X13cccf0a41de644625faad47eb59d388bc50e6c0

* Authors have used library from openssl.org for this table.

3.2.2 Security Requirements

After understanding the properties of hash functions, we may now look into its security aspects. Typically, the security can be seen from three different angles: preimage resistance, second preimage resistance, and collision resistance.

Before discussing these angles, we need to understand the mathematical meaning of the word “hard” in the context of cryptography. The word “hard” in most intuitive form means that it is almost impossible for any adversary to break the system with the finite and limited amount of resources. The formal way to explain “hard” is based on the computational complexity theory. The problem P is considered to be “hard” if there exists a formal security reduction from a problem that is widely considered unsolvable in polynomial time, such as discrete logarithm problem, finding modular square roots (sometimes also known as quadratic residue), subset sum problem, or integer factorization problem. It should be noted that the non-existence of a polynomial time algorithm does not automatically ensure that the system is secure.

Flash Quiz

Adam Spencer is a comedian, mathematician, and the author of *The Number Games*. He wrote down the numbers 1 to 19 out of numerical order, with the last five numbers missing. For example, 8, 18, 11, 15, 5, 4, 14, 9, 19, 1, 7, 17, 6, 16, ?, ?, ?, ?, ?

To solve this riddle, you need to figure out the order of the first 14 numbers. Try to solve this within five minutes if possible.

Difficulty of a problem also depends on its size, for example, one of the most popular public key cryptography algorithms, RSA, relies on the difficulty of integer factorization. However, it is considered secure only with the keys that are at least 2048-bit long, the lower key size may be vulnerable to break given the current trends in computational technology. Therefore the Hash function should be resistance to the know attacks as mentioned below.

1. **Preimage resistance:** Given a hash value H from a hash function, it should be hard to find the original string S such that $H = \text{hash}(S)$. The hash functions that do not possess or lack this characteristic are vulnerable to preimage attacks.

2. **Second preimage resistance:** Given an input string S1, it would be practically hard to find another string S2 (other than S1) such that hash(S1) = hash(S2). In cryptographic community, this property is referred to as weak collision resistance. Hash functions that lack this property are vulnerable to second preimage attacks.

3. **Collision resistance:** It should be hard to find two different strings S1 and S2 such that hash(S1) = hash(S2). Such a pair is known as a (cryptographic) hash collision. It is also sometimes referred to as strong collision resistance. It requires a hash value at least twice as long as what is required for preimage resistance, otherwise collisions may be found by a birthday attack. This has already been explained in Section 3.2.1.

Technical Stuff

When a pdf file is protected by a password, it stores them as tagged contents in the files itself with multiple rounds of hashes on it.

3.2.3 Attacks

In cryptography terms, the attack is defined as a mechanism to circumvent the security of a cryptographic function by finding a weakness in the algorithm, implementation code, and protocol. It is often known as *cryptanalysis*. In simple terms, it is an information security threat that involves an attempt to obtain, alter, destroy, remove, implant, or reveal information without authorized access.

Technical Stuff

A collision avoidance system is also known as precrash system, forward collision warning system, or collision mitigation system. It is an automobile safety system designed to prevent or reduce the severity of a collision. It uses radar (all-weather) and sometimes laser (LIDAR) and camera (employing image recognition) to detect an imminent crash. Also, GPS sensors can detect fixed dangers such as approaching stop signs through a location database.

In the simplest manner, an attack on hash function is an attempt to find two different input strings (S1 and S2) of a hash function that produces the same hash result (H). As hash functions can take any variable input length and have a predefined output length, there is inevitably going to be the possibility of two different inputs that produce the same output hash. As conveyed earlier, collision in hash function occurs when two separate inputs produce the same hash output. This can be exploited by an automated application that compares two hashes together (such as password hashes and data file integrity). However, the odds of a collision are extremely low, especially for functions with a large output size such as lengthy document. However, as available computational power increases, the ability to attack hash functions becomes more feasible. A collision attack on a hash function tries to find two input strings (S1 and S2) that produce the same hash value. The attacker does not have control over the content of the message, but they are randomly chosen by the algorithm. In this case, H(S1) is equal to H(S2). Theoretically, no hash function is collision free, but it usually takes extremely long to find a collision.

Even though a hash function has never been broken, a successful attack against a weakened variant may undermine the user's confidence and lead to its abandonment. One can observe that in the past, weaknesses have been found in several notable hash functions, including SHA-0, RIPEMD, MD5, and SHA-1, but stronger functions such as SHA-256 appear to be safe for now and is being used by Bitcoin.

Some examples of cryptographic hash functions, which were attacked, are as follows:

1. **MD5:** It produces 128-bit hash value. Collision resistance was broken after ~2^21 hashes.
2. **SHA-1:** It produces 160-bit hash value. Collision resistance was broken after ~2^61 hashes.

3. **Keccak-256:** It produces 256-bit hash and is currently used by Ethereum.

Flash Quiz

Use the OpenSSL syntax dgst -sha1 -hex file.txt

In the next section, we will be discussing about these hash functions, though they are broken now. This is to impress upon readers how design principles were evolved to construct them and how further advancements are done in this area.

3.3 Message Authentication Code

A message authentication code (MAC) is a cryptographic checksum on data along with a session key to detect any modifications in the original data. Typically, MAC requires two inputs: data and a secret key known only to the originator of the data. However, the originator may share the secret to the intended recipient(s) also. This allows the recipient of the data to verify the integrity of the message.

There are four types of MACs: unconditionally secure, hash function-based, stream cipher-based, and block cipher-based. The most common approach to create a MAC is to use block ciphers such as Data Encryption Standard (DES). However, hash-based MACs (HMACs) that use a secret key in conjunction with a cryptographic hash function to produce a hash value are more widely used.

Technical Stuff

```
OpenSSL syntax
enc -des -e -in file.txt –a –out
outfile.txt
enc -des -d -in outfile.txt –a –out
nwfile.txt
```

3.3.1 MD5 Design

The MD5 message-digest algorithm takes as input a message of arbitrary length and produces as output 128-bit “fingerprint” or “message digest” of the input. The MD5 algorithm is intended for digital signature applications, where a large file must be “compressed” in a secure manner before being encrypted with a private key in a public-key cryptosystem.

The MD5 algorithm is designed for 32-bit machines focusing on the speed. In addition, the MD5 algorithm does not require any large substitution tables; it can be coded quite compactly. The MD5 algorithm is an extension to its predecessor MD4 message-digest algorithm. It is slightly slower than MD4 but more “conservative” in design. MD5 is slightly compromised by giving a little in speed for a much required security. This is for incorporating suggestions made by various reviewers, and contains additional optimizations.

Let us look at the terminology and notation used for explaining MD5, before discussing its design. A “word” is a 32-bit quantity and a “byte” is an 8-bit quantity. A sequence of bytes can be interpreted as a sequence of 32-bit words, where each consecutive group of 4 bytes is interpreted as a word with the low-order (least significant) byte given first.

Let *x_i* denote “*x* sub *i*”. If the subscript is an expression, we surround it in braces, as in *x*_{*i* + 1}. Similarly, we use ^ for superscripts (exponentiation), so that *x*^*i* denotes *x* to *i*th power. Let the symbol “+” denote addition of words (i.e., modulo-2^32 addition). Let X <<< *s* denote the 32-bit value obtained by circularly shifting (rotating) X left by *s* bit positions. Let not(X) denote the bit-wise complement of X, and let X v Y denote the bit-wise OR of X and Y. Let X xor Y denote the bit-wise XOR of X and Y, and let X • Y denote the bit-wise AND of X and Y.

3.3.1.1 MD5 Algorithm Description

Let us start with *b*-bit message as input and that we wish to find its message digest. Here, *b* is an arbitrary non-negative integer; *b* may be zero, it need not be multiple of eight, and it may be arbitrarily large. We imagine that the bits of the message written down is as follows:

m_0 m_1… m_{*b* – 1}

The following steps are performed to compute the message digest of *b*-bit message.

Step 1: Append Padding Bits – Initially, the message is "padded" (extended) so that its length (in bits) is congruent to 448, modulo 512. That is, the message is extended so that it is just 64 bits short of being a multiple of 512 bits. In order to have uniformity, padding is always performed, even if the length of the message is already congruent to 448, modulo 512. Padding is performed as follows: a single "1" bit is appended to the message, and then "0" bits are appended so that the length in bits of the padded message becomes congruent to 448, modulo 512. Overall, at least 1 bit and at the most 512 bits are appended.

Quick Tip

PKCS#5 considered to the most used padding in modern cryptographic operations.

Step 2: Append length – A 64-bit representation of *b*, before the padding bits were added, is appended to the result of the previous step. In an unlikely event when *b* is greater than 2^64, then only the low-order 64 bits of *b* are used. These bits are appended as two 32-bit words and appended low-order word first in accordance with the previous conventions. At this point, the resulting message (after padding with bits and with *b*) has a length that is an exact multiple of 512 bits. Equivalently, this message has a length that is an exact multiple of 16 (32 bit) words. Let $M[0\ldots N-1]$ denote the words of the resulting message, where N is a multiple of 16.

Step 3: Initialize MD buffer – A four-word buffer (A, B, C, D) is used to compute the message digest. Here, each of A, B, C, D is 32-bit register. These registers are initialized to the following values in hexadecimal, low-order bytes first:

word A: 01 23 45 67
word B: 89 ab cd ef
word C: fe dc ba 98
word D: 76 54 32 10

Step 4: Process message in 16-word blocks – We first define four auxiliary functions that each take as input three 32-bit words and produce as output one 32-bit word.

F(X, Y, Z) = X • Y v not(X) • Z
G(X, Y, Z) = X • Z v Y • not(Z)
H(X, Y, Z) = X xor Y xor Z
I(X, Y, Z) = Y xor (X v not(Z))

In each bit position F acts as a conditional: if X then Y else Z. The function F could have been defined using “+” instead of “v” since X•Y and not(X) Z will never have 1s in the same bit position. It is interesting to note that if the bits of X, Y, and Z are independent and unbiased, then each bit of F(X, Y, Z) will be independent and unbiased.

The functions G, H, and I are similar to the function F. However, they act in “bit-wise parallel” to produce their output from the bits of X, Y, and Z in such a manner that if the corresponding bits of X, Y, and Z are independent and unbiased, then each bit of G(X, Y, Z), H(X, Y, Z), and I(X, Y, Z) will be independent and unbiased. Note that the function H is the bit-wise “xor” or “parity” function of its inputs.

This step uses a 64-element table T[1… 64] constructed from the sine function. Let T[*i*] denote the *i*th element of the table, which is equal to the integer part of 4,294,967,296 times abs(sin(*i*)), where *i* is in radians. The elements of the table can be constructed a priori and kept ready for consumption.

Do the following:

```
/* Process each 16-word block. */
For $i$ = 0 to $N$/16 – 1 do
  /* Copy block $i$ into X. */
  For $j$ = 0 to 15 do
    Set X[$j$] to M[$i$*16+$j$].
  end /* of loop on $j$ */

/* Save A as AA, B as BB, C as CC, and D as DD. */
AA = A
BB = B
CC = C
DD = D

/* Round 1. */
/* Let [abcd $k$ $s$ $i$] denote the operation
```

```
   a = b + ((a + F(b, c, d) + X[k] + T[i]) <<< s). */
/* Do the following 16 operations. */
[ABCD 0 7 1] [DABC 1 12 2] [CDAB 2 17 3] [BCDA 3 22 4]
[ABCD 4 7 5] [DABC 5 12 6] [CDAB 6 17 7] [BCDA 7 22 8]
[ABCD 8 7 9] [DABC 9 12 10] [CDAB 10 17 11] [BCDA 11 22 12]
[ABCD 12 7 13] [DABC 13 12 14] [CDAB 14 17 15] [BCDA 15 22 16]

/* Round 2. */
/* Let [abcd k s i] denote the operation
   a = b + ((a + G(b, c, d) + X[k] + T[i]) <<< s). */
/* Do the following 16 operations. */
[ABCD 1 5 17] [DABC 6 9 18] [CDAB 11 14 19] [BCDA 0 20 20]
[ABCD 5 5 21] [DABC 10 9 22] [CDAB 15 14 23] [BCDA 4 20 24]
[ABCD 9 5 25] [DABC 14 9 26] [CDAB 3 14 27] [BCDA 8 20 28]
[ABCD 13 5 29] [DABC 2 9 30] [CDAB 7 14 31] [BCDA 12 20 32]

/* Round 3. */
/* Let [abcd k s t] denote the operation
   a = b + ((a + H(b, c, d) + X[k] + T[i]) <<< s). */
/* Do the following 16 operations. */
[ABCD 5 4 33] [DABC 8 11 34] [CDAB 11 16 35] [BCDA 14 23 36]
[ABCD 1 4 37] [DABC 4 11 38] [CDAB 7 16 39] [BCDA 10 23 40]
[ABCD 13 4 41] [DABC 0 11 42] [CDAB 3 16 43] [BCDA 6 23 44]
[ABCD 9 4 45] [DABC 12 11 46] [CDAB 15 16 47] [BCDA 2 23 48]

/* Round 4. */
/* Let [abcd k s t] denote the operation
   a = b + ((a + I(b, c, d) + X[k] + T[i]) <<< s). */
/* Do the following 16 operations. */
[ABCD 0 6 49] [DABC 7 10 50] [CDAB 14 15 51] [BCDA 5 21 52]
[ABCD 12 6 53] [DABC 3 10 54] [CDAB 10 15 55] [BCDA 1 21 56]
[ABCD 8 6 57] [DABC 15 10 58] [CDAB 6 15 59] [BCDA 13 21 60]
```

```
    [ABCD 4 6 61] [DABC 11 10 62] [CDAB 2 15 63] [BCDA 9 21 64]
    /* Then perform the following additions. (That is increment each of the four registers by the value it had before this block was started.) */
    A = A + AA
    B = B + BB
    C = C + CC
    D = D + DD
    end /* of loop on i */
```

Step 5: Output – The message digest produced as output is A, B, C, D. That is, we begin with the low-order byte of A, and end with the high-order byte of D.

The MD5 message-digest algorithm is simple to implement, and provides a "fingerprint" or message digest of a message of arbitrary length. It is conjectured that the difficulty of coming with two messages having the same message digest is typically of the order of 2^64 operations, and that the difficulty of coming with any message having a given message digest is the order of 2^128 operations.

3.3.2 Vulnerabilities

MD5 was published in 1992 as an informational request for comments (RFC). Since that time, MD5 has been extensively studied and new cryptographic attacks have been discovered. Message digest algorithms are designed to provide collision, preimage, and second preimage resistance. In addition, the message digest algorithms are used with a shared secret value for message authentication in HMAC. With advancements in the computation speed, MD5 is no longer acceptable where collision resistance is required such as digital signatures, and new protocol designs should not be explored. Few alternatives to HMAC-MD5 include HMAC-SHA-256 (Nystrom 2005; Song *et al*. 2006) when AES is more readily available than a hash function.

Fact Alert

Request for comments (RFC), in information and communications technology, is a type of text document from the technology community. An RFC document may come from many bodies including from the Internet Engineering Task Force (IETF), Internet Research Task Force (IRTF), Internet Architecture Board (IAB), or from independent authors. The RFC system is supported by the Internet Society (ISOC).

Pseudo-collisions for the compress function of MD5 were first described in 1993 (den Boer 1993). In 1996, Dobbertin demonstrated a collision pair for the MD5 compression function with a chosen initial value (Dobbertin 1995). The first article that demonstrated two-collision pairs for MD5 was published in 2004 (Wang 2004). The detailed attack techniques for MD5 were published at EUROCRYPT 2005 (Wang 2005). Since then, a lot of research results have been published to improve collision attacks on MD5. The attacks presented in Klima (2006) can find MD5 collision in about 1 minute on a standard notebook PC (Intel Pentium, 1.6 GHz). Stevens in his article claims that it takes 10 seconds or less on 2.6 GHz Pentium4 to find collisions (Stevens 2007). One may also apply the collision attack on MD5 to password-based challenge-and-response authentication protocols such as the Authenticated Post Office Protocol (APOP) option in POP (Myers 1996) used in the post office authentication as presented in Leurent (2007). These aforementioned results have been provided sufficient reason to eliminate MD5 usage in the applications where collision resistance is required such as the digital signatures.

3.4 Secure Hash Algorithms (SHA-1)

The hash function, finalized in 1995, is known as SHA-1. It is a simple algorithm, which converts a message of any length < 2^64 bits as input to 160-bit output as message digest. The message digest can then be input to another utility, for example, a signature algorithm. This function came out as FIPS-180 (FIPS stands for Federal Information Processing Standard) from the US National Institute of Standards that specified SHA-1. The SHA-1 is known to be secure as it is computationally infeasible to find a message that corresponds to a given message digest, or to find two different messages that produce the same message digest. Any change to a message in transit will, with very high probability, result in a different message digest, and the signature will fail to verify. SHA-1 forms a part of several widely used security applications and protocols, including TLS and SSL, PGP, SSH, S/MIME, and IPsec. In general, the applications that use MD5 can also use SHA-1, as both MD5 and SHA-1 are descended of MD4.

Fact Alert

In computing, the Post Office Protocol (POP) is an application-layer Internet standard protocol used by clients to retrieve e-mail from a mail server. POP version 3 (POP3) is the version in common use.

3.4.1 SHA-1 Design

In order to understand the design of SHA-1, we will first learn the terminology related to bit strings and integers that will be used.

A hex digit is an element of the set {0, 1, …, 9, A, …, F}. A hex digit is the representation of a 4-bit string. For example, 6 = 0110, B = 1011. A word equals 32-bit string that may be represented as a sequence of 8 hex digits. To convert a word to 8 hex digits, each 4-bit string is converted to its hex equivalent as described earlier. For example, 1011 0001 0000 0011 1111 1110 0010 0111 = B103FE27.

Technical Stuff

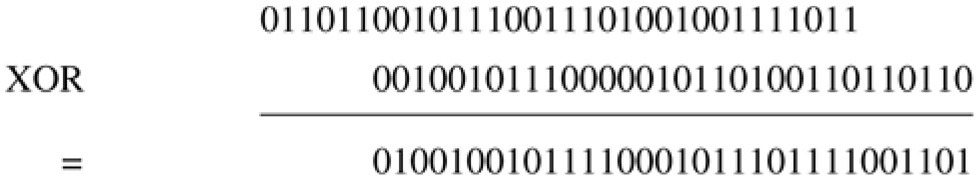
Transport Layer Security (TLS) and its now deprecated predecessor, Secure Sockets Layer (SSL), are cryptographic protocols designed to provide communications security over a computer network. Several versions of the protocols find widespread use in applications such as web browsing, email, instant messaging, and voice over IP (VoIP). Websites can use TLS to secure all communications between their servers and web browsers.

An integer between 0 and 2^32 – 1 inclusive may be represented as a word. The least significant 4 bits of the integer are represented by the rightmost hex digit of the word representation. For example, the integer 291 = 2^8 + 2^5 + 2^1 + 2^0 = 256 + 32 + 2 + 1 is represented by the hex word, 00000123. If z is an integer, 0 ≤ z < 2^64, then z = (2^32)x + y, where 0 ≤ x < 2^32 and 0 ≤ y < 2^32. Since x and y can be represented as words X and Y, respectively, and z can be represented as a pair of words (X, Y).

Block = 512-bit string. A block (e.g., B) may be represented as a sequence of 16 words. Furthermore, the following logical operators will be applied to the words:

X AND Y = bit-wise logical "and" of X and Y.
X OR Y = bit-wise logical "inclusive-or" of X and Y.
X XOR Y = bit-wise logical "exclusive-or" of X and Y.
NOT X = bit-wise logical "complement" of X.

For example:



```
                 01101100101110011101001001111011
XOR                       00100101110000010110100110110110
                 ----------------------------------------
  =                       01001001011110001011101111001101
```

The operation X + Y is defined as follows: Words X and Y represent integers *x* and *y*, where 0 ≤ *x* < 2^32 and 0 ≤ *y* < 2^32. For positive integers *n* and *m*, let *n* mod *m* be the remainder upon dividing *n* by *m*.

Compute *z* = (*x* + *y*) mod 2^32.

Then, 0 ≤ *z* < 2^32. Convert *z* to a word, Z, and define Z = X + Y.

The circular left shift operation is S^*n*(X), where X is a word and *n* is an integer with 0 ≤ *n* < 32, and is defined by S^*n*(X) = (X << *n*) OR (X >> 32 – *n*).

In aforementioned, X << *n* is obtained as follows: Discard the leftmost *n* bits of X and then pad the result with *n* zeroes on the right and the result will still be 32 bits. X >> *n* is obtained by discarding the rightmost *n* bits of X and then padding the result with *n* zeroes on the left. Thus, S^*n*(X) is equivalent.

Suppose a message has length l < 2^64. Before it is input to SHA-1, the message is padded on the right as follows:

1 is appended. For example, if the original message is 01010000, this is padded to 010100001.

Technical Stuff

Pretty Good Privacy (PGP) is an encryption program that provides cryptographic privacy and authentication for data communication. PGP is used for signing, encrypting, and decrypting texts, e-mails, files, directories, and whole disk partitions and to increase the security of e-mail communications. Phil Zimmermann developed PGP in 1991.

0s are appended. The number of 0s will depend upon the original length of the message. The last 64 bits of the last 512-bit block are reserved for the length l of the original message. For example, suppose the original message is bit string

01100001 01100010 01100011 01100100 01100101

After 1 is padded this gives:

01100001 01100010 01100011 01100100 01100101 1

Since, $l = 40$, the number of bits in the above is 41 and 407. 0s are appended, now making the total 448. This gives (in hex):

61626364 65800000 00000000
00000000

00000000 00000000 00000000
00000000

```
   00000000 00000000 00000000
00000000
   00000000 00000000
```

Obtain the two-word representation of *l*, which is the number of bits in the original message. If *l* < 2^32, then the first word is all zeroes. Append these two words to the padded message.

If we consider the original message above, then *l* = 40 (note that *l* is computed before any padding). The two-word representation of 40 is hex 00000000 00000028. Hence, the final padded message is hex:

```
   61626364 65800000 00000000
00000000
   00000000 00000000 00000000
00000000
   00000000 00000000 00000000
00000000
   00000000 00000000 00000000
00000028
```

The padded message will contain 16 * *n* words for *n* > 0. The padded message is regarded as a sequence of *n* blocks M(1), M(2), first characters (or bits) of the message.

Technical Stuff

Secure Shell or Secure Socket Shell (SSH) is a network protocol that gives users, particularly system administrators, a secure way to access a computer over an unsecured network.

3.4.1.1 Functions and Constants Used in SHA-1

A sequence of logical functions $f(0), f(1), \ldots, f(79)$ is used in SHA-1. Each $f(t)$, $0 \le t \le 79$, operates on three 32-bit words B, C, and D, and produces a 32-bit word as an output. $f(t;$ B, C, D) for words B, C, D is defined as follows:

$$f(t; B, C, D) = (B \text{ AND } C) \text{ OR } ((\text{NOT } B) \text{ AND } D) \qquad (0 \le t \le 19)$$
$$f(t; B, C, D) = B \text{ XOR } C \text{ XOR } D \qquad (20 \le t \le 39)$$
$$f(t; B, C, D) = (B \text{ AND } C) \text{ OR } (B \text{ AND } D) \text{ OR } (C \text{ AND } D) \qquad (40 \le t \le 59)$$
$$f(t; B, C, D) = B \text{ XOR } C \text{ XOR } D \qquad (60 \le t \le 79)$$

A sequence of constant words K(0), K(1), …, K(79) is used in SHA-1. In hex, these are given by:

$$K(t) = 5A827999 \quad (0 \le t \le 19)$$
$$K(t) = 6ED9EBA1 \quad (20 \le t \le 39)$$
$$K(t) = 8F1BBCDC \quad (40 \le t \le 59)$$
$$K(t) = CA62C1D6 \quad (60 \le t \le 79)$$

Let us now see how to compute the message digest. The message digest is computed using the message padded as described earlier. The computation is described using two buffers, each consisting of five 32-bit words, and a sequence of eighty 32-bit words. The words of the first five-word buffer are labeled as A, B, C, D, and E. The words of the second five-word buffer are labeled as H0, H1, H2, H3, and H4. The words of the eighty-word sequence are labeled as W(0), W(1), …, W(79). A single word buffer TEMP is also employed. To generate the message digest, the sixteen-word blocks M(1), M(2), …, M(n) defined earlier are processed in order. The processing of each M(i) involves 80 steps.

Before processing any blocks, H's are initialized (in hex) as follows:

H0 = 67452301

H1 = EFCDAB89

H2 = 98BADCFE

H3 = 10325476

H4 = C3D2E1F0

Now, M(1), M(2), …, M(*n*) are processed. To process M(*i*), we proceed as follows:

Divide M(*i*) into 16 words – W(0), W(1), …, W(15), where W(0) is the leftmost word.

For, *t* = 16 to 79 let W(*t*) = S^1(W(*t* – 3) XOR W(*t* – 8) XOR W(*t* – 14) XOR W(*t* – 16)).

Let, A = H0, B = H1, C = H2, D = H3, E = H4.

For, *t* = 0 to 79 do TEMP = S^5(A) + *f*(*t*; B, C, D) + E + W(*t*) + K(*t*);

E = D; D = C; C = S^30(B); B = A; A = TEMP

Let, H0 = H0 + A, H1 = H1 + B, H2 = H2 + C, H3 = H3 + D, H4 = H4 + E. After processing M(*n*), the message digest is 160-bit string represented by the five words: H0, H1, H2, H3, and H4.

3.4.2 Vulnerabilities

Because of the block and iterative structure of the algorithms and the absence of additional final steps, all the SHA functions (except SHA-3, as of writing this book) are vulnerable to length extension and partial message collision attacks (Ferguson 2010).

In the early 2005, Rijmen and Oswald published an article on the attack on a reduced version of SHA-1, where for 53 of 80 rounds, they could find collisions with a computational effort of fewer than 2^80 operations (Rijmen and Oswald 2005).

In February 2005, an attack by Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu was announced (Wang *et al*. 2005). The attacks can find collisions in the full version of SHA-1, requiring fewer than 2^69 operations. We should note that a brute-force search would require 2^80 operations. The researchers wrote, “In particular, our analysis is built upon the original differential attack on SHA-0, the near collision attack on SHA-0, the multi-block collision techniques, as well as the message modification techniques used in the collision search attack on MD5. Breaking SHA-1 would not be possible without these powerful analytical techniques.” The authors have presented a collision for 58-round SHA-1, found with 2^33 hash operations. The research article was with full attack description and was published in August 2005 at the CRYPTO conference. In an interview, Yin states that, “Roughly, we exploit the following two weaknesses: One is that the file preprocessing step is not complicated enough; another is that certain math operations in the first 20 rounds have unexpected security problems.” (Lemos 2005).

On 17 August 2005, an improvement on SHA-1 attack was announced on behalf of Xiaoyun Wang, Andrew Yao, and Frances Yao at the CRYPTO 2005 Rump Session, lowering the complexity required for finding a collision in SHA-1 to 2^63. (Bruce Schneier Blog 2005). On 18 December 2007, the details of this result were explained and verified by Martin Cochran (2007).

Christophe de Cannière and Christian Rechberger further improved the attack on SHA-1 in the article “Finding SHA-1 characteristics: General results and applications” (De Cannière *et al*. 2006), which received the Best Paper Award at ASIACRYPT 2006. A two-block collision for 64-round SHA-1 was presented; it was found using unoptimized methods with 2^35 compression function evaluations. Since this attack requires the equivalent of about 2^35 evaluations, it is considered to be a significant theoretical break (TUGRAZ 2006). Their attack was extended further to 73 rounds (of 80) by Grechnikov (2010). In order to find an actual collision in the full 80 rounds of the hash function, tremendous amount of computer time is required. To that end, a collision search for SHA-1 using the distributed computing platform BOINC began on 8 August 2007, organized by the Graz University of Technology. The effort was abandoned on 12 May 2009 because of the lack of progress.

One attack against SHA-1 was given by Marc Stevens with an estimated cost of USD 2.77M to break a single hash value by renting CPU power from the cloud servers (Stevens 2012; Bruce Schneier Blog 2012). Stevens developed this attack in a project known as HashClash (Stevens 2010) implementing a differential path attack. On 8 November 2010, he claimed that he had a fully working near-collision attack against full SHA-1 working with an estimated complexity equivalent to 2^57.5 SHA-1 compressions. He estimated that this attack could be extended to a full collision with a complexity of around 2^61.

On 8 October 2015, Marc Stevens, Pierre Karpman, and Thomas Peyrin (Stevens 2012) published a free start collision attack on SHA-1's compression function that requires only 2^57 SHA-1 evaluations. This does not directly translate into a collision on the full SHA-1 hash function (where an attacker is not able to freely choose the initial internal state), but undermines the security claims for SHA-1. In particular, it was the first time that an attack on full SHA-1 had been demonstrated; all earlier attacks were too expensive for their authors to carry them out. The method was based on their earlier work, as well as the auxiliary paths (or boomerangs) speed-up technique from the earlier research and using high performance/cost efficient GPU cards from NVIDIA. The collision was found on a 16-node cluster with a total of 64 graphics cards. The authors estimated that a similar collision could be found by buying USD 2000 of GPU time on EC2. They estimated that the cost of renting enough of EC2 CPU/GPU time to generate a full collision for SHA-1 at the time of their publication was between USD 75K and 120K, and noted that was well within the budget of the criminal organizations, not to mention the national intelligence agencies. As such, they recommended that SHA-1 be depreciated as quickly as possible.

On 23 February 2017, Centrum Wiskunde & Informatica (CWI) and Google announced the shattered attack (available at https://shattered.io), in which they generated two different PDF files with the same SHA-1 hash in roughly 2^63.1 SHA-1 evaluations. This attack is about 100,000 times faster than brute forcing SHA-1 collision with a birthday attack, which was estimated to take 2^80 SHA-1 evaluations. The attack required "the equivalent processing power of 6,500 years of single-CPU computations and 110 years of single-GPU computations".

3.5 Secure Hash Algorithm Version 3

Secure Hash Algorithm Version 3, popularly known as SHA-3, is derived from the original algorithm known as Keccak. It is a unidirectional function for generating digital prints of the selected length from an input data of any size. The SHA-3 family consists of four cryptographic hash functions, which are SHA-3-224, SHA-3-256, SHA-3-384, and SHA-3-512, along with two extendable output functions (XOFs), which are SHAKE-128 and SHAKE-256. This was developed by a group of researchers led by Yoan Dimen in 2008 and adopted in 2015 as the new FIPS 202 standard. The original Keccak algorithm has many configurable parameters such as data block size, algorithm state size, and number of rounds in the function to provide an optimal ratio of cryptographic stability and performance for application on the specific platform. However, the SHA-3 algorithm has several differences from the original Keccak algorithm, where slow modes are discarded and simplified algorithm of filling was introduced with an extended result. The Keccak algorithm is the research work of Guido Bertoni, Joan Daemen (co-designer of AES, the Rijndael cipher, with Vincent Rijmen), Michael Peeters, and Gilles Van Assche. This algorithm is based on some earlier hash function designs, such as PANAMA and RadioGatún. The PANAMA algorithm was designed by Daemen and Craig Clapp in 1998, whereas RadioGatún is a successor of PANAMA and was designed by Daemen, Peeters, and Van Assche in 2006. In 2006, NIST initiated hash function competition to create a new hash standard known as SHA-3. It may also be noted that SHA-3 was not meant to replace SHA-2, as no significant attack on SHA-2 has been demonstrated till the writing of this book. However, SHA-0, and SHA-1, NIST perceived a need for an alternative dissimilar cryptographic hash function because of the successful attacks on MD5.

3.5.1 Secure Hash Algorithm Version 3 Design

SHA-3 is a subset of the broader cryptographic primitive family Keccak. It is typically used on three features: Sponge construction, padding, and block permutation.

3.5.1.1 Sponge Construction

Sponge construction is the core of the hash function. In this, the data is "absorbed" into the sponge and subsequently, the result is "squeezed" out. In the absorbing phase, message blocks are XORed into a subset of the state, which is transformed using a permutation function. During the "squeeze" process, output blocks are read from the same subset of the state and alternated with the state transformation function. The size of the part of the state that is written and read is known as the "rate" and the size of the part that is untouched by input/output is known as the "capacity". The capacity determines the security of the scheme.

3.5.1.2 Padding

Just like in the previous hash function, in order to ensure that the message can be evenly divided into r-bit blocks, padding is required even in SHA-3. It uses the pattern 10 * –1 in its padding function, that is, 1 bit followed by 0 or more 0 bits up to a maximum of (n – 1) bits. The maximum of (n – 1) 0 bit occurs when the last message block is (n – 1) bits long. Then another block is added after the initial 1 bit, containing (n – 1) 0 bit before the final 1 bit. In a scenario when the length of the message is already divisible by n, two 1 bits will be added. In this case, another block is added to the message containing 1 bit, followed by a block of (n – 2) 0 bit and another 1 bit. This is required because a message with length divisible by r ending in something that looks like padding does not produce the same hash as the message when those bits are removed. The initial 1 bit is necessary as messages differing only in a few additional 0 bits at the end should not produce the same hash value.

3.5.1.3 Block Permutation

Block transformation is a permutation that uses XOR, AND, and NOT operations, and is designed for easy implementation in both software and hardware. It is defined for any power of two-word size, and in this design it uses 64-bit words. The basic block permutation function consists of 24 rounds each containing the following steps:

1. Computation of the parity.
2. Bit-wise rotation.
3. Permutation of 25 words.
4. Bit-wise combination along rows.
5. Exclusive-OR.

Let us discuss Keccak, which is the core of SHA-3. It uses a simple approach of “sponge”. The sponge construction is a framework for specifying functions on binary data with arbitrary output length. Its construction employs the following components:

1. An underlying function (*f*) on fixed length strings.
2. A parameter called rate (*r*).
3. A padding rule denoted by pad.

The function that the construction produces from these components is known as sponge function, SPONGE[*f*, pad, *r*]. A sponge function takes two inputs of the output string, SPONGE[*f*, pad, *r*](*N, d*): bit string *N* and bit length *d*. One can give an analogy to a sponge as an arbitrary number of input bits are “absorbed” into the state of the function, after which an arbitrary number of output bits are “squeezed” out of its state. The sponge function consists of a function that is applied on each round by implementing pseudorandom permutation state.

We adapted from Bertoni (2011) and depicted the sponge construction in Figure 3.1.

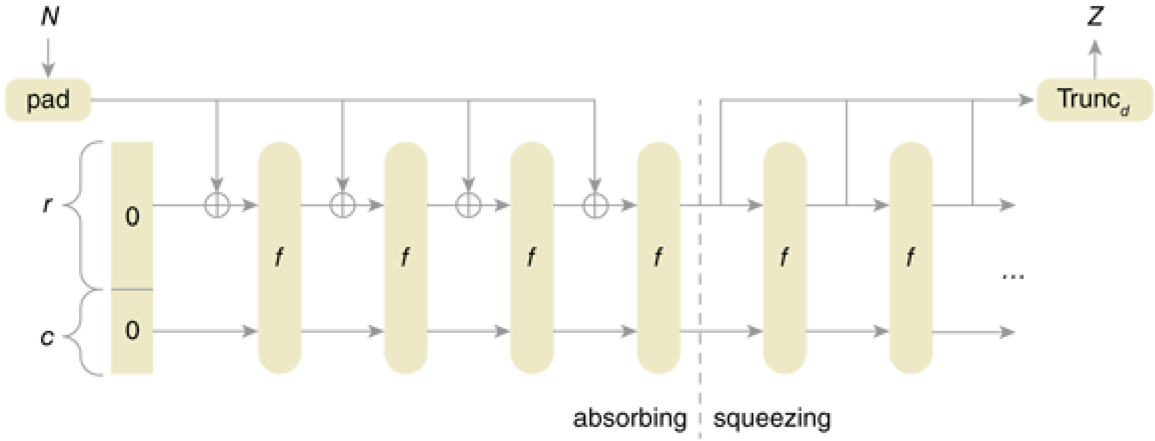


Figure 3.1 Sponge construction.

Algorithm for sponge: SPONGE[*f*, pad, *r*](*N*, *d*)
Input: string *N*, nonnegative integer *d*
Output: string *Z* such that len(*Z*) = *d*.

Steps:

1. Let *P* = *N* || pad(*r*, len(*N*)).
2. Let *n* = len(*P*)/*r*.
3. Let *c* = *b* – *r*.
4. Let *P*_0, …, P_*n* – 1 be the unique sequence of strings of lenagth *r* such that *P* = *P*_0 || … || *P*_*n* – 1.
5. Let $S = 0^b$.
6. For *i* from 0 to *n* – 1, let *S* = *f* (*S* ⊕ (*P*_*i* || 0^{c})).
7. Let *Z* be the empty string.

8. Let $Z = Z \| \text{Trunc\_}r(S)$.
9. If $d \leq |Z|$, then return Trunc_d (Z); else continue.
10. Let $S = f(S)$, and continue with Step 8.

We note that input d determines the number of bits that this algorithm returns, but it does not affect their values. In principle, the output can be regarded as an infinite string. However, in practice, the computation is halted after the desired number of output bits is produced.

To cache a string, one must first break it into pieces of a certain size. On each round, it should not be mixed with 1600-bit state, but only to its beginning of size r. Moreover, on each round, the next piece of the string is mixed only to a part of the state, while the pseudorandom permutation f handles the whole state and makes it dependent on the whole string. This is known as "absorption" stage, and now it is clear why it is called absorption. To get the actual hash, we continue to apply the permutation function f to the state, and at each stage copy only a piece of size r from it until we get the hash of the required length. This is known as sponge "squeezing".

More on SHA-3 standard can be obtained from Federal Information Processing Standards Publication (FIPS PUB 202), which is available in http://dx.doi.org/10.6028/NIST.FIPS.202.

3.6 Distributed Hash Tables

A distributed hash table (DHT) is a class of a decentralized distributed system that provides a lookup service similar to a hash table. It uses pairs (key, value) and store in a DHT. Any participating node can efficiently retrieve the value associated with a given key. Keys are unique identifiers that map to particular values. These can be anything, from addresses to documents to arbitrary data. Responsibility for maintaining the mapping from keys to values is distributed among the nodes in such a manner that a change in the set of participants causes a minimal amount of disruption in the setup. This enables DHT to scale up to extremely large numbers of nodes and handle continual node arrivals, departures, and failures. DHTs provide an infrastructure that can be used to build more complex services, such as cooperative web caching, distributed file systems, domain name services, instant messaging, multicast, and peer-to-peer file sharing.

The structure of DHT can be divided into several main components. Its foundation is an abstract keyspace, for example, it could be a set of 160-bit strings. The keyspace partitioning scheme splits ownership of this keyspace among various participating nodes. Then an overlay network connects all the nodes, thus allowing them to find the owner of any given key in the keyspace.

After these components are put in place, a typical use of the DHT for storage and retrieval can proceed. To understand it, let us consider the keyspace in the set of 160-bit strings. To index a file with given filename and data in DHT, the SHA-1 hash of filename is generated, producing 160-bit key *k*, and a message put(*k*, data) is sent to any node participating in DHT. Then the message is forwarded from node to node through the overlay network until it reaches the single node responsible for key *k* as specified by the keyspace partitioning. This node then stores the key and data. Subsequently, any other client can retrieve the contents of the file by hashing filename locally to produce *k* and asking any DHT node to find the data associated with *k* with a message get(*k*). The message will then be routed through the overlay to the node responsible for key *k*, which will reply with the stored data.

A basic keyspace partitioning and overlay network components are described in the following paragraphs, with the goal of capturing the principal ideas of the most common DHTs.

Most DHTs use some variant of consistent or rendezvous hashing to map keys to nodes. From a historic perspective, these two algorithms (i.e., consistent hashing and rendezvous hashing) appear to have been devised independently and almost simultaneously to address the DHT problem. Both the algorithms possess essential property, that is, removal or addition of one node changes only the set of keys owned by the nodes with adjacent IDs, and leaves all the other nodes unaffected. This is in contrast to a traditional hash table in which an addition or removal of one bucket nearly causes the entire keyspace to be remapped with the new values. Since any change in ownership typically corresponds to bandwidth-intensive movement of the objects stored in the DHT from one node to another, minimizing such reorganization is required to efficiently support high rates of node arrival and departure.

3.6.1 Consistent Hashing

Consistent hashing is based on mapping each object to a point on a circle. The system maps each available machine or other storage bucket to many pseudorandomly distributed points on the same circle. To find where an object should be placed, the system finds the location of that object's key on the circle, and then walks around the circle until falling into the first bucket it encounters (i.e., the first available bucket with a higher angle). The result is that each bucket contains all the resources located between each one of its points and the previous points belonging to the other buckets. In a scenario where a bucket becomes unavailable (e.g., because of the computer unreachability), the points it maps to will be removed. Requests for resources that would have mapped to each of these points will now map to the next highest points. As every bucket is associated with many pseudorandomly distributed points, the resources that were held by one such bucket will now map to many different buckets. In case of departure, the items that mapped to the departed bucket must be redistributed among the remaining buckets. However, values mapping to other buckets will still be valid need not to be moved. A similar process occurs when a bucket is added, and then new bucket point is added. Any resources between these and the points corresponding to the next smaller angles map to the new bucket. Furthermore, these resources will no longer be associated with the previous buckets and any value previously stored in it will not be available. The portion of the keys associated with each bucket can be altered by altering the number of angles that bucket maps.

3.6.2 Rendezvous or Highest Random Weight Hashing

Rendezvous hashing is more general than consistent hashing, which becomes a special case for $k = 1$. It is an algorithm that allows clients to achieve distributed agreement on a set of k options of a possible set of n options. Given its simplicity and generality, it has been applied in a wide variety of applications, including mobile caching, router design, secure key establishment, and it also solves the DHT problem.

A typical application is when clients need to agree on which servers the objects are to be assigned. How can a set of clients, given an object O, agree on where in a set of n sites (say, servers) to place O? Each client is to select a site independently, but all the clients must pick the same site. This is non-trivial if we add a minimal disruption constraint, and it requires that only objects mapping to a removed site may be reassigned to other sites.

The core idea is to provide each site Sj a score (or some weight) for each object Oi, and assign the object in question to the highest scoring site by first agreeing on a hash function h(). For object Oi, the site Sj is defined to have weight wi,j = hash(Oi, Sj). The highest random weight (HRW) algorithm assigns Oi to the site Sm whose weight wi,m is the highest. As hash function h() is agreed upon, all the clients can independently compute the weights wi,1, wi,2, …, wi,n and pick the highest. If the goal is distributed k-agreement, the clients can independently pick the sites with k having highest hash values. In a scenario when site S is added or removed, only the objects mapping to S are remapped to different sites, satisfying the minimal disruption constraint. The HRW re-assignment can be computed independently by any client, as it depends only on the identifiers for the set of sites $S1$, $S2$, …, Sn and the object being assigned. HRW algorithm is also flexible to accommodate different capacities among sites. If site Sk has triple the capacity of the other sites, it can be simply represented as Sk thrice in the list, for example, as Sk,1 Sk,2 and Sk,3.

3.6.3 Comparison of Consistent and Rendezvous Hashing

Typically, consistent hashing operates by mapping sites uniformly to points on a unit circle by randomly using tokens. Objects are mapped to the unit circle. They are placed in the site whose token is the first encountered while traveling clockwise from the object's location. When a site is removed, the objects that are owned by the site are transferred to the site owning the next token encountered moving clockwise. However, this is feasible when each site is mapped to a large number of tokens. This will uniformly reassign objects among the remaining sites.

In contrast to consistent hashing, rendezvous hashing (or HRW) is simpler both conceptually and practically. It is considered as a uniform hash function as it distributes the objects uniformly over all the sites. HRW requires no precomputing or storage of tokens, unlike consistent hashing where tokens are precomputed.

3.7 Hashing and Data Structures

After covering various core concepts on hash function, we will now present data structure, which is a specialized way of storing data. There are two data structure properties that are critical to understand the working of a blockchain: pointers and linked lists.

In any programming language, variables are used to store data. For example, int $x = 20$ implies that there is a variable x that stores integer data type and is currently holding the value 20. Pointers are variables that store the address of another variable. That is, instead of storing values of the variable, it will store addresses of that or other variables. This is why they are known as pointers, as they literally point towards the location of other variables.

Linked list is one of the most important items in data structures. A simple linked list is shown in Figure 3.2.

A linked list is typically a sequence of blocks, where each block contains data that is linked to the next block via a pointer. The pointer variable contains the address of the next node in the list and is linked via the connection by the pointer. The last node has a null pointer and it has no value. The following should be noted:

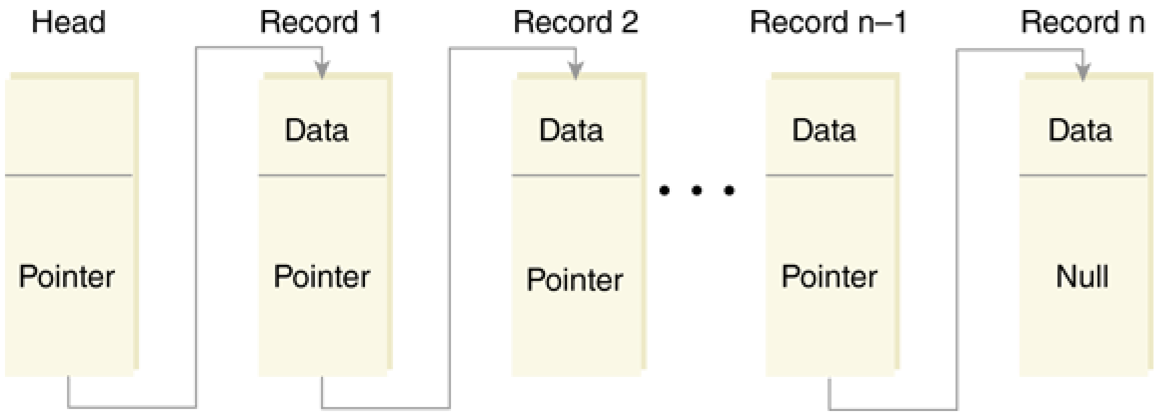


Figure 3.2 Example of a simple linked list.

1. The pointer inside each block contains the address of the next block.
2. The first block is also known as *genesis block*. Its pointer lies out of the system.

The linked list forms the structure of blockchain, and a simplified Bitcoin blockchain is shown in Figure 3.3.

In simple terms, a blockchain is a linked list that contains data (block) and a hash pointer pointing to (chaining) its previous block, thus creating the chain. As explained earlier, a hash pointer is similar to a pointer, but instead of just containing the address of the previous block, it also contains the hash of the data inside the previous block. Even though it is a small addition, it makes the blockchain more reliable and immutable.

Let us consider a scenario where an attacker attacks the *N*th block and tries to manipulate the data. We know from Section 3.2.1 that a slight change in data will change the hash value drastically. This implies that any slight changes made to block *N* will change the hash value that is stored in the previous block, and in turn will change the data and hash value of the previous block. This will result in changes in the previous to previous block and so on. This will completely change the chain, which is impossible and thereby provides the blockchain its immutability property.

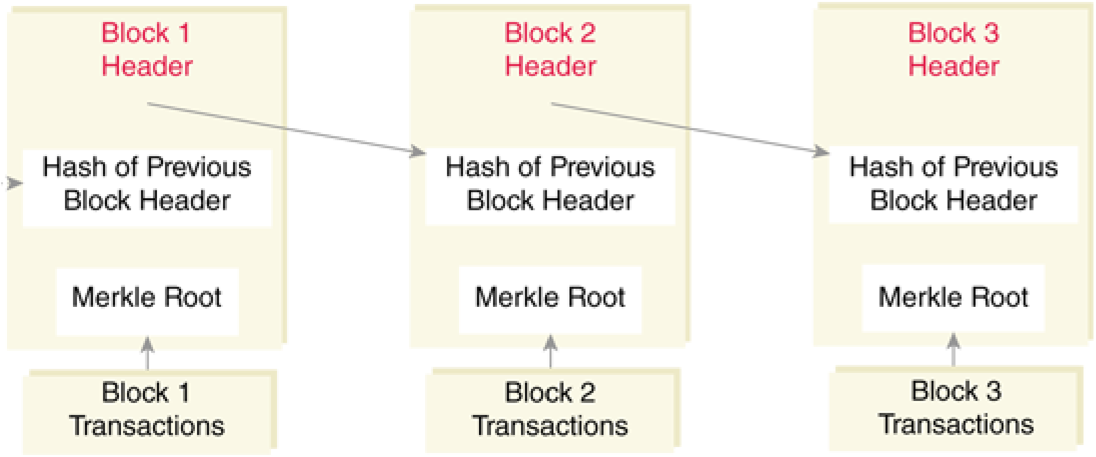


Figure 3.3 Simplified Bitcoin blockchain.

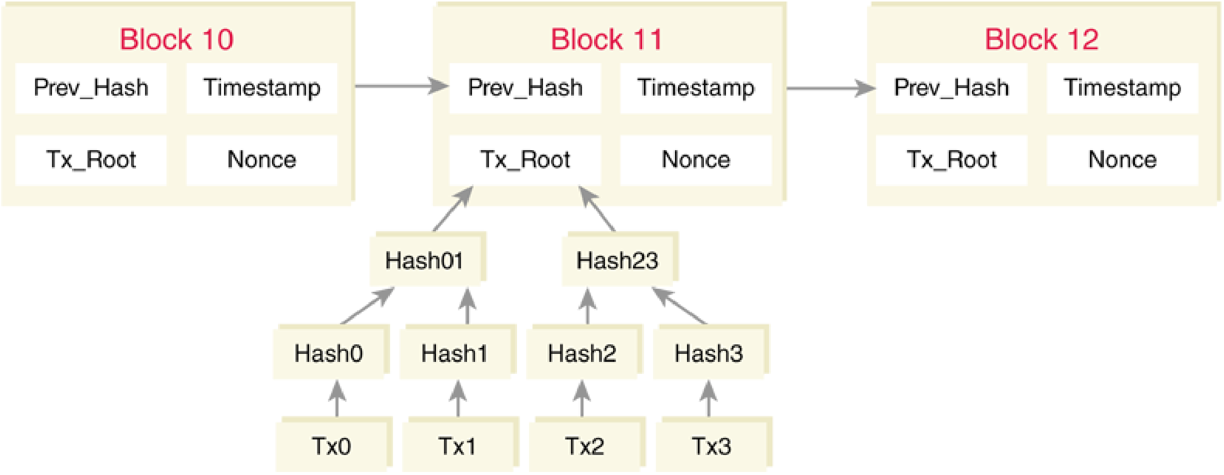


Figure 3.4 The blockheader.

Thus, we observe that blockchain consists of series of various blocks that are used to store information related to data that occurs on a blockchain network. A block contains a unique header, and each such block is identified by its block header hash. Figure 3.4 shows the blockheader.

A block header typically contains the following items:

1. **Version:** The block version number indicates which set of block validation rules should be followed.
2. **Time:** The block time (timestamp) is a Unix epoch time when the miner started hashing the header (i.e., the time according to the miner). This must be greater than the median time of the previous 11 blocks.
3. **Target hash:** A target hash is a number that a hashed block header must be less than or equal to in order for a new block to be awarded. The target hash is used in determining the difficulty of the input, and can be adjusted in order to ensure that blocks are processed efficiently.
4. **Hash of the previous block:** As explained earlier, hash function converts an input of letters and numbers into a cryptic output of fixed length. Here, the input to it is the previous block contents.
5. **Nonce:** A number used only once is added to a hashed block that when rehashed meets the difficulty level restrictions. The nonce is the number that blockchain miners are solving for.
6. **Hash of the Merkle root:** Merkle trees are binary hash trees that serve to encode blockchain data more efficiently and securely. The hash of Merkle trees is useful as it allows users to verify a specific transaction without downloading the whole blockchain.

To illustrate this, the header of the 80-byte long string is given below. It comprises of 4-byte long Bitcoin version number, 32-byte previous block hash, 32-byte long Merkle root, 4-byte long timestamp of the block, 4-byte long difficulty target for the block, and 4-byte long nonce used by miners.

02000000 .... Block version: 2

b6ff0b1b1680a2862a30ca44d346d9e8910d334beb48ca0c0000000000000000 … Hash of previous block's header

9d10aa52ee949386ca9385695f04ede270dda20810decd12bc9b048aaab31471 … Merkle root

24d95a54 .... Unix time: 1415239972
30c31b18 .... Target: 0x1bc330 * 256**(0 × 18 – 3)
fe9f0864 .... Nonce

3.7.1 Merkle Tree

Merkle tree is defined as a tree in which every leaf node is labeled with the hash value of a data block, and every non-leaf node is labeled with the cryptographic hash value of the labels of its child nodes. These hash trees allow efficient and secure verification of the contents of large data structures. Let us look into more details on this tree.

Figure 3.5 depicts how a Merkle tree looks like, where each non-leaf node is the hash of the values of their child nodes. The leaf nodes are the nodes in the lowest layer of the tree. Therefore, with respect to Figure 3.5, the leaf nodes will be L1, L2, L3, and L4 (Figure 3.6).

Child nodes: The nodes below another node's layer, which are feeding into it, are its child nodes. With respect to the diagram, the nodes labeled "Hash 0-0" and "Hash 0-1" are the child nodes of the node labeled "Hash 0".

Root node: The single node on the highest layer labeled "Top Hash" is the root node (Figure 3.7).

Each block can have transactions in the range of thousands. Thus, it will be inefficient to store all data inside each block in sequence, while at the same time finding any particular transaction proves to be an extremely cumbersome and time-consuming process. However, using a Merkle tree will greatly reduce the time required to find if a particular transaction belongs to a block or not.

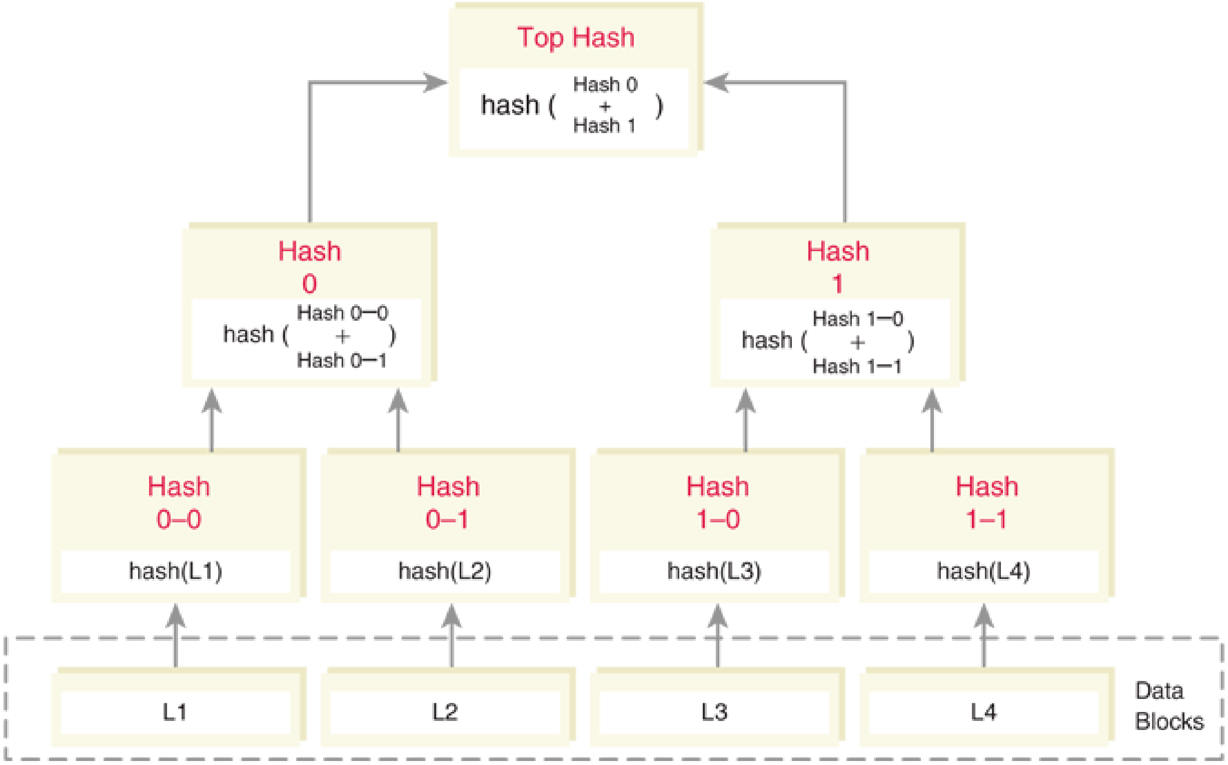


Figure 3.5 Binary hash tree.
Source: https://en.wikipedia.org/wiki/Merkle_tree

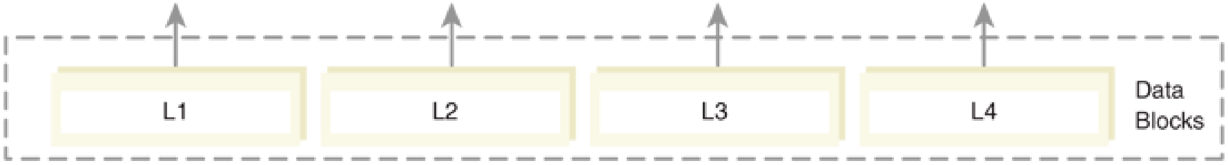


Figure 3.6 Leaf nodes.

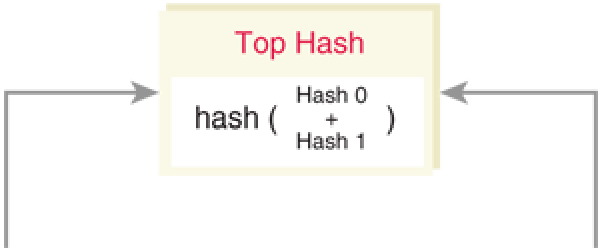


Figure 3.7 "Top Hash" root node.

Let us consider the example shown in Figure 3.8.

Now, we need to find whether a particular data belongs to a block or not (Figure 3.9).

The process of the conventional approach is quite cumbersome, as it looks at each individual hash and checks whether it belongs to the data or not. However, it can simply be tracked by following the trail of hashes leading up to the data. Doing this significantly reduces the time taken (Figure 3.10).

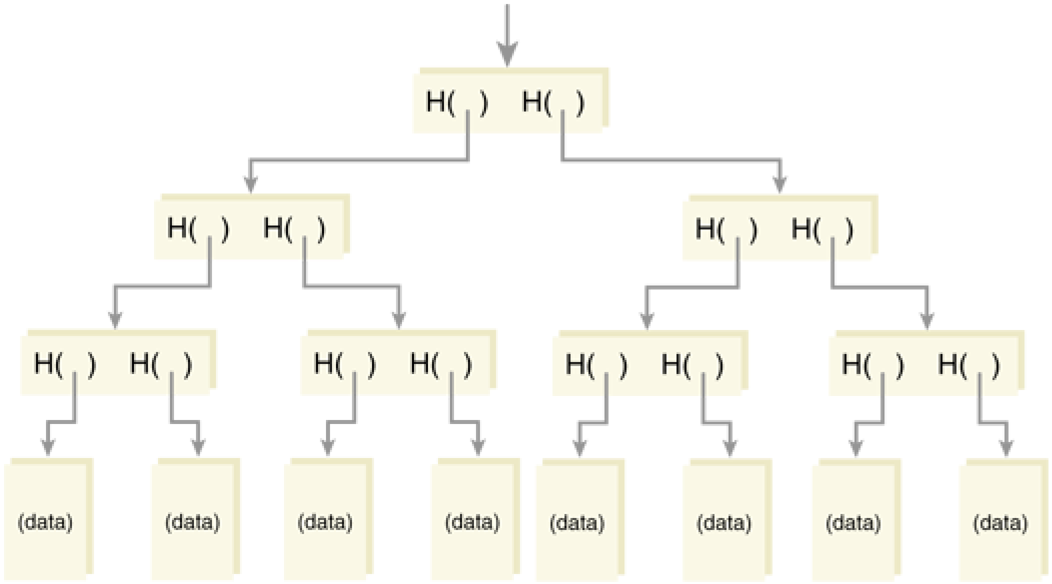


Figure 3.8 Example of a Merkle tree.

Source: https://www.coursera.org/lecture/cryptographic-hash-integrity-protection/hash-tree-merkle-tree-zc2WU

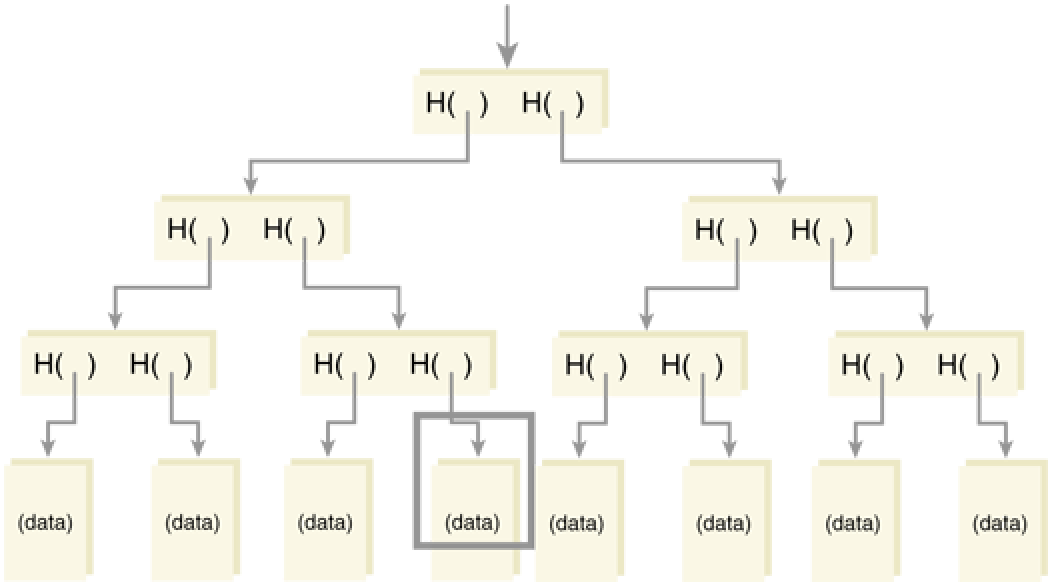


Figure 3.9 Merkle tree – level 2.

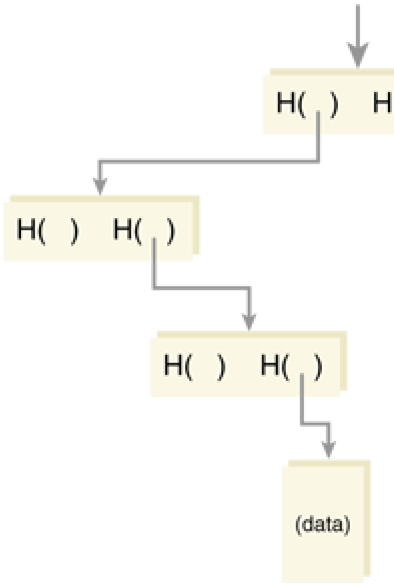


Figure 3.10 Merkle tree – level 3.

Hash functions play an important role in the blockchain and we will also discuss its real use in mining.

3.8 Hashing in Blockchain Mining

Mining basically means searching for a new block to be added in the blockchain. Miners across the globe are constantly working and insuring that the chain keeps growing. Earlier, it was easy for people to mine by just using their laptops, but over time, people started forming mining pools. In these pools, they increase their computer powers by clubbing several computing devices and mine more efficiently. However, this might lead to a problem as there is a cap for each cryptocurrency, for example, for Bitcoin, it is just 21 million. If the miners are allowed to move ahead with significant high rate, they will dig all the Bitcoins in existence very soon. Furthermore, there needs to be a specific time limit in between the creation of each block, for example, in Bitcoin, the time limit in between block creation is 10 minutes. If the blocks were allowed to be created at a faster rate, it would result in more collisions and orphaned blocks. At a higher rate, the generation of hash value will be high and that will inevitably cause more collisions. Also, a lot of miners are over mining and they will come with new blocks simultaneously. This will result in more blocks not qualified to be part of the main blockchain, thus resulting as orphan blocks.

Thus, in order to restrict block creation, a specific difficulty level is set for miners. Setting higher difficulty makes mining much harder to solve and hence more time-consuming, for example, in Bitcoin, the difficulty target is a 64-character string (using SHA-256 output) that begins with a bunch of zeroes. The number of zeroes increases as the difficulty level increases and it changes after every 2016th block. Thus, hashing has truly been fundamental and significant in the creation of the blockchain technology.

SUMMARY

In this chapter we have learned about hash functions. They are frequently used in many parts of cryptography. There are many hash functions, with varying security properties. The accepted secure hash function should be one-way and collision resistant. There are additional properties that may be necessary when a hash function is being used in blockchain technology which are presented in this chapter. We have also seen the detail construction of frequently used hash functions, along with the vulnerabilities. The chapter also explains Merkel tree and its usage in blockchain.

SHORT ANSWER QUESTIONS

1. What do you understand by deterministic function?
2. Why should hash function be collision resistant?
3. How would you define avalanche effect in cryptography? Why is it important?
4. What do you understand by mathematical hard problem in cryptography?
5. Compare the technical properties of MD5 and SHA1.
6. What basic arithmetical and logical functions are used in SHA1?
7. What is sponge function in SHA3?
8. What is distributed hash table? Where are they used?
9. Differentiate between consistent and rendezvous hashing.
10. What is a Merkle tree?

LONG ANSWER QUESTIONS

1. Consider that H( ) is a collision-resistant hash function that maps a message of arbitrary bit length into an *N*-bit hash value. Prove that for x1 not equal to x2, H(x1) is not equal to H(x2). Explain your answer.
2. Take source code of one hash function of your choice from GitHub. Follow the instruction to compile and execute. After executing on your local machine, try to modify some of its building blocks and observe the change in output. Compare at least 10 different strings with original and modified versions.

CHAPTER 4

Consensus

Learning Objectives

After reading this chapter, you will be able to:

- Understand all about distributed decentralized databases.
- Analyze the concepts related to decentralized enterprises.
- Describe decentralization and disintermediation.
- Understand about decentralized enterprise regulation.

4.1 Introduction

We have already learned about the hash functions in the Chapter 3. It is one of the most important building blocks in blockchain. In this chapter, we will discuss some of the features that blockchain possesses to qualify as distributed, decentralized, and tamper-resistant database. This demands that it should have a trusted and formal way of tracking the current state of the system. As blockchain can include financial transactions and business agreements, it becomes extremely important that all the parties involved are in sync with each other with respect to the transaction and terms of agreement. This requires the nodes (or group) consensus which is one of the most important components to blockchain. There are several different methods to reach consensus, which will be discussed in this chapter.

4.2 Consensus Approach

Historically, consensus methods are being used to solve problems in medicine and health with an aim to define levels of agreement on controversial subjects. When consensus methods are properly employed, these can create structured environments in which experts are given the best available information, allowing their solutions to problems to be more justifiable and credible. There are several major methods, such as Delphi and Nominal Group, to arrive at consensus with varying parameters including choosing members for the consensus panels, specifying acceptable levels of agreement, usage of empirical data, and disseminating results. In the blockchain world, consensus mechanisms are essential as blockchain is decentralized with no middlemen and where trust has truly become decentralized with the trustless movement of value. The consensus in the blockchain is a way to ensure that the nodes on the network verify the transactions and agree with their order and existence on the immutable ledger. The consensus is not exclusive to blockchain; in fact, it is a classic problem in distributed computer systems. In a typical case of cryptocurrency, the process is critical to prevent double spending; and therefore, the consensus mechanism plays an important role.

Technical Stuff

In molecular biology and bioinformatics, the consensus sequence (or canonical sequence) is the calculated order of most frequent residues, either nucleotide or amino acid, found at each position in a sequence alignment.

For simple understanding of consensus, let us consider every block in a blockchain as being a sheet of paper by every node in that chain. With the given amount fixed area to write, we consider one transaction in every line. When that sheet of paper is completely written, all (or majority) the nodes as a group come together and compare different sheets (or blocks) and select the sheet or the version of the paper that the majority agrees with. The key point to understand is this consensus is from all of the participating nodes. Potentially speaking, there are tens of thousands of nodes and all of them repeat the same work. Then, these come together periodically and agree on whatever the majority select, which is then considered as the right version of the truth. This consensus gives blockchain a high level of trust and makes it a secure storage of records, thereby enabling all the nodes to validate the information to be appended to the blockchain by utilizing a consensus algorithm to ensure that they agree on a unique order in which entries are appended. In recent past, the consensus protocols for tolerating Byzantine faults have received renewed attention because they also address blockchain systems. One may note that most blockchain ecosystems have a hybrid of different consensus mechanisms with no single mandate to choose one over the other. The oldest and most widespread method is proof-of-work suggested by Nakamoto (Nakamoto 2008), in which a hard cryptographic puzzle must be solved by the miners. Miners are rewarded for their efforts using various incentives for their expenses on computing resources. By attempting to the balance fairness and resource expenditure, several other consensus approaches such as proof-of-stake, proof-of-burn, proof-of-elapsed-time, and proof-of-capacity have been proposed to overcome some of the weaknesses of the original proof-of-work model (Kiayias 2017; Zamfir 2015).

Quick Tip

There are over 2000 cryptocurrencies circulating in the world. Their exchange rates and other details are discussed at https://coinmarketcap.com/all/views/all/

Fact Alert

Double-spending is a problem in which the same digital currency can be spent more than once.

4.3 Consensus Algorithms

In computer science, a consensus algorithm is defined as a process to achieve agreement on a single data value among distributed systems. Consensus algorithms are designed to achieve reliability in a network involving multiple unreliable nodes and sometimes unavailable nodes. To accommodate this reality, consensus algorithms must be fault tolerant. For all practical purposes, they typically assume that only a portion of nodes will respond. However, in order to have majority consensus it requires a response from at least 51% of the participating nodes. When we talk about the blockchain, it is consensus algorithms that make the blockchain network so secure and decentralized. The consensus algorithm in blockchain does the following things:

1. Ensures that the next block in a blockchain is the one and only version of the truth.
2. Keeps powerful adversaries from derailing the system by successfully forking the chain.

The form of consensus relevant for blockchain is technically known as *atomic broadcast*, which is formally obtained as an extension of a reliable broadcast among the node. This atomic broadcast provides a total order on the messages delivered to all the correct nodes. It is characterized by two asynchronous events – *broadcast* and *deliver* – that may occur multiple times. Every node in the system may broadcast some message (or transaction) *m* by invoking the function `broadcast(m)`, and the broadcast protocol outputs *m* to the local application on the node through `deliver(m)` event function. Furthermore, this atomic broadcast ensures that each correct node delivers the same sequence of messages through the deliver events. Also, it provides validity and integrity of the messages delivered. The most important to implement atomic broadcast (consensus) in distributed systems is the family of protocols known as *Paxos* (L. Lamport 1998; L. Lamport 2001) and view stamped replication (Oki 1998; Liskov 2012). Although discovered independently, their core mechanisms exploit the same ideas in Lampson (2001) and Liskov (2010). These consensus algorithms have been implemented in dozens of mission critical systems and influence the core infrastructure of major cloud providers today; more about this can be studied in Chandra 2007. Many real-world systems including Google’s PageRank, load balancing, smart QuickTip.png>grids, clock synchronization, and drone control are supported in some sense by consensus algorithms. Today, there are several different kinds of consensus algorithms in the blockchain, each with different fundamental processes. Let us discuss how these work and are related to each other. Like in the real world, in a blockchain setup, there are several different methods that can be used to come to a consensus on a block. The different methods can be used to fit different situations, with the main difference being between consensus mechanisms in which they delegate and reward the verification of transactions. Scalability, privacy, security, and fault tolerance must be considered for consensus algorithms to be deployed in the system. Resistance to attacks, node ID management, consensus finality, correctness proofs, and assumptions about network synchrony are other parameters to consider while deciding the algorithm to be used. The algorithm must provide scalability with number of nodes. Proof-of-work can support thousands of nodes; however, it suffers on performance, that is, transactions per second (tps) and latency. Also, the algorithm must be efficient so that it consumes less power, or in general, consumption of any resource should be less.

4.3.1 Proof-of-Work

Satoshi Nakamoto, a pseudonymous author, published an article in which he introduced Bitcoin (Nakamoto 2008), which has become one of the most successful cryptocurrencies of modern times. Proof-of-work consensus is considered to be the key innovation of Bitcoin, which is fully decentralized.

Dwork and Naor (Dwork 1993) in 1993 were the first to present the idea of proof-of-work as a technique for combatting spam mail. They proposed that an email sender needs to compute the solution to a mathematical puzzle to prove that some computational work was performed. Later, in 1997, Back (Back 1997) independently proposed proof-of-work named *Hashcash* for the same cause (i.e., fighting spam). In this case, the computational puzzle is to find a hash value (SHA-1) of the header. The header includes the email recipient’s address and current date, such that the hash value contains at least 20 bits of leading zeros. As we have seen that the hashing algorithm is pre-image resistant, the challenge can be solved only by including random nonces in the header until the resulting hash value meets the leading zeros requirement. It requires a significant amount of computational work to achieve this. Thus, this computation effort for the valid hash value is considered to be proof-of-work. Consensus in Nakamoto Bitcoin paper is derived from Hashcash. In Bitcoin, Hashcash's SHA-1, hashing is replaced with two successive SHA-2 hashes such that valid hashes have a value below target integer value T. The difficulty of the challenge is therefore adjustable: decreasing T increases the number of guesses (and thus work or computation done) required to generate valid hash values. The nodes that generate hash values are known as *miners* and the process is referred to as *mining*. Miners calculate hash values of candidate blocks of transactions to be added to the blockchain. For this work, they are rewarded with new coins if they find a valid block. The value T is reset by the network for every 2016 blocks such that miners are successful in appending a block for every interblock interval of 10 minutes to the blockchain.

Technical Stuff

The SHA-2 family consists of several hash functions with digests (hash values) that are 224, 256, 384, or 512 bits.

Let us understand this with the help of the example shown in Figure 4.1. It contains several blocks of a blockchain. Every block contains a *Block number*, a *Nonce* (in the form of random number), the data related to transaction, the hash value of the immediate previous block (*Prev*), and the hash value of the current block (*Hash*). Let us assume that block 1 is the starting block of the chain, which implies that there is no block before block 1. Therefore, the previous hash value is set to all 0s. Given the block number, transaction data, and previous hash value, the objective or challenge is to find the new nonce such that the hash of all is less than the target value, that is, SHA-256 (*Block#; Nonce; Data; Prev*) ≤ *target integer value T*. Suppose, for block 1 the nonce is 42345 (Figure 4.1). The target integer value indicates the difficulty level of the proof-of-work and mandates that the 256-bit output of the hash function starts with a certain number of consecutive 0s (three in this example). Now, the same process is repeated for block 2 to obtain (guess) the nonce with the new block number and the transaction data. Also, *Prev* is updated with *Hash* of the immediate previous block. You may observe that a different nonce number is obtained and *Hash* begins with three consecutive 0s. As the target value decreases, the difficulty of finding the nonce increases because of the constrained condition of the leading three 0s.

Now, there can be a scenario where if two miners find two different blocks that build on the same previous block, they satisfy the criterion set for the new block. This is known as *forking* and is resolved using consensus rules. The consensus rule which is followed here is by accepting the “longest chain, which has the greatest proof-of-work effort invested in it” as the correct one. In reality, this is implemented as the chain with the most accumulated work. However, for an attacker to be successful, it must have sufficient computing power to be able to create a fork of the blockchain that has more accumulated work than the chain that is to be overridden. The threat model assumes an adversary that has the majority of the computing power in the network (sometimes referred to as 51% attack) can outpower the remaining nodes in generating a chain with the most accumulated work. Decker and Wattenhofer (Decker 2013), showed that because of the delays in the blocks propagation in Bitcoin network, increasing the block size and decreasing the interblock interval further increases the chance of forks. This occurs because the delayed miners may waste effort in attempting to mine on top of blocks that are no longer the latest ones. As a result, the network becomes more susceptible to 51% attacks from a miner that does not suffer from delays.

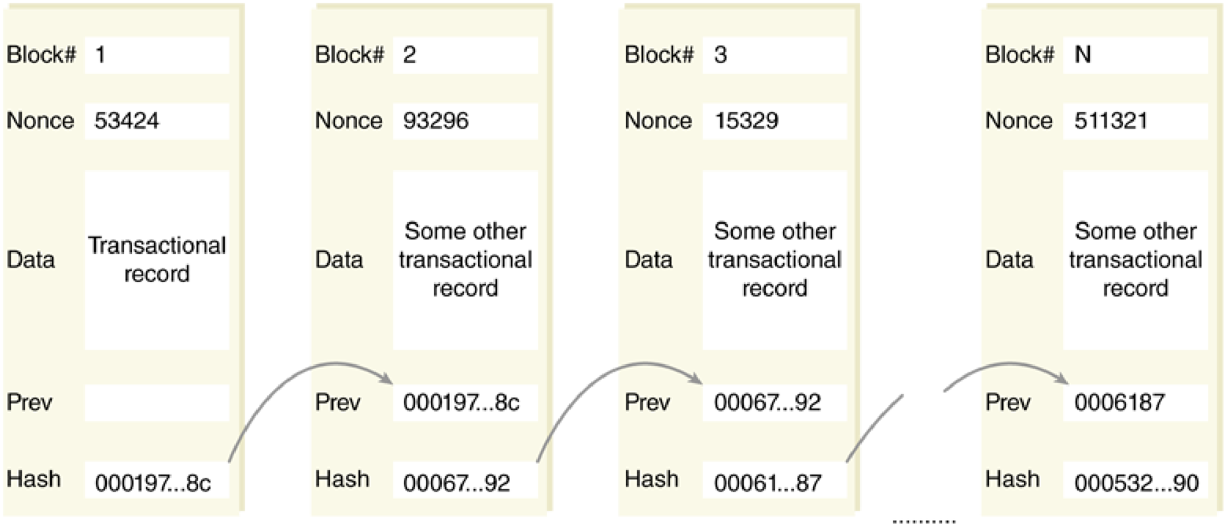


Figure 4.1 Example of chaining in blockchain with proof-of-work.

Despite countermeasures to avoid forks, they may still occur in proof-of-work blockchains. For more resilient and scalable system than Bitcoin blockchain, several new policies have been proposed to select the main chain in the forked blockchain. One such approach is GHOST (Sompolinsky 2013), which exploits blocks that are not in the main chain while achieving higher transaction rates without undermining Bitcoin security. This is achieved by replacing the linear blockchain by organizing blocks in a tree structure. The tree is shaped by the blocks that successful miners choose to expand. The chain selection algorithm chooses the heaviest path as main chain, where the block's weight depends on how density of its subtree. A common theme in enhancing proof-of-work consensus is to improve performance while maintaining the security threshold of the network and at the same time without requiring nodes (or the miners) to upgrade their network connections. Furthermore, to reduce the variance in rewards to the miners, they often aggregate resources and share rewards among themselves via pooled mining protocols. The mining pools undermine decentralization and are vulnerable to transaction censorship by a malicious pool manager. To mitigate such attacks, proof-of-work mechanism should be fair, that is, the number of valid blocks mined by a miner should be proportional to its computing power in the network. There are several techniques proposed to create decentralized mining pools such as SmartPool (Luu 2017), which implements a practical decentralized mining pool through an Ethereum smart contract. The smart contract typically is a replacement of the traditional pool manager.

Quick Tip
The GHOST protocol in Ethereum is Greedy Heaviest Observed Subtree and was introduced in 2013.

Power intensiveness is one of the biggest problems of proof-of-work. This motivates new class of consensus protocols based on proof-of-X that can replace wasteful computations with useful work derived from alternative commonly accessible resources. We will be discussing this in the following subsections.

4.3.2 Proof-of-Stake

This proof-of-stake mechanism is the most prevalent method of consensus in the blockchains used for cryptocurrencies, after proof-of-work. It works similar to proof-of-work but with a significant difference. It is on the basis of the idea of "one vote per unit of stake", as opposed to "one vote per unit of hash-power". That is, instead of mining power, the probability to create a block and receive the associated reward is proportional to the user's ownership stake in the system. In simple terms, it does not induce race among the nodes to solve the next block rather than competition between the nodes to solve the next block. The rationale behind this is that the node with the highest stakes in the system has the most interest to maintain a secure network. Also, if the security is not maintained, they will suffer the most in case the reputation (or price of the cryptocurrency) diminishes because of the attacks. The node selected to mine the next block is chosen on the basis of its proportional stake in the network. The selected node, instead of solving a complicated hash problem, will use a digital signature to prove its ownership over the stake. As a result, this method does not require high-computational power. Thus, in this case, the miners run a process that randomly selects one of them proportionally to the stake that each possesses according to the current blockchain ledger.

Technical Stuff

In computing, a denial of service (DoS) attack is a cyber-attack in which the perpetrator seeks to make a machine or network resource unavailable to its intended users by temporarily or indefinitely disrupting services of a host connected to the Internet.

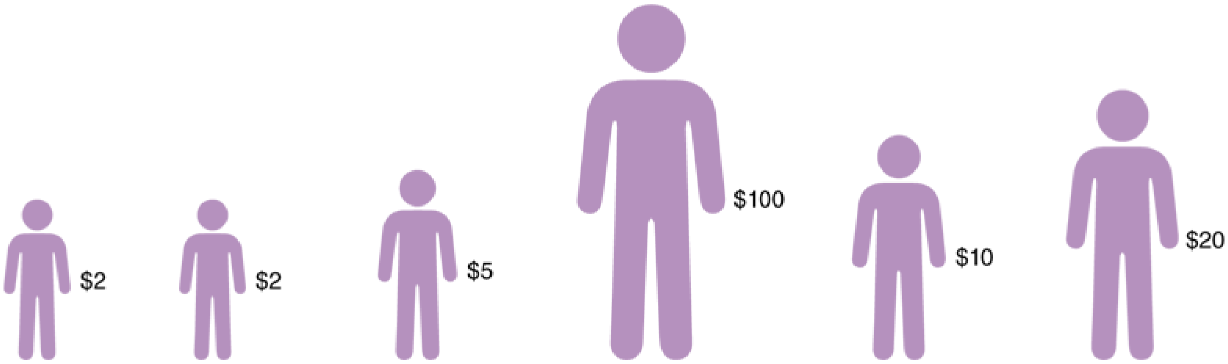


Figure 4.2 Proof-of-stake.

In proof-of-stake, the elect leader from among the stakeholders had a right to append a block to the blockchain. The election of leader may be in public and is visible to all the participants. Alternatively, when a private election of the leader is conducted among the participants, then it uses the private information to check if they have been selected as the leader, as shown in Figure 4.2. This can be verified by all other participants using public information. Private leader election is resilient to denial of service

(DoS) attacks because the candidates privately check if they are elected before revealing it publicly in their blocks. This makes the process too late for an attacker to materialize the DoS attack to them. There is a possibility that a malicious leader can censor transactions during its time. However, as leaders are often re-elected, a subsequent leader will add the censored transaction to the blockchain, although with some delay.

When this is applied to a cryptocurrency system, all the coins are available from the first day and no mining reward or coin creation exists and the nodes/miners are rewarded only with a transaction fee. Although this mechanism eliminates the computational requirements of proof-of-work, it creates new problems. This method is contingent upon nodes with the highest amount of stake making the blockchain centralized. Furthermore, there is another problem known as *nothing at stake*, which refers to the situation where a selected node has nothing to lose if it behaves maliciously. Therefore, without economic penalties for attackers, the chain can suffer nothing at stake attacks. In this attack, the stakers are incentivized to validate all the proposed forks to maximize their returns.

There is one derivative mechanism of proof-of-stake known as *delegated proof-of-stake* (DPoS). In contrast to proof-of-stake, which is direct democratic, this method is representative democratic. Here, all the stakeholders vote to choose some nodes as witnesses and delegates. Witnesses are responsible and rewarded for creating new blocks, whereas the delegates are responsible for maintaining the network. Delegates can also propose changes such as block sizes, transaction fees, or reward amount. *N* witnesses with the highest votes are chosen in each election round. The number *N* is selected by the consensus of at least 50% of the voting stakeholders who believe that there is enough decentralization. At the cost of making the blockchain more centralized, Bitshares, a cryptocurrency, uses DPoS that has shown significant improvement in the throughput and latency as compared to proof-of-stake.

There is yet another derivative mechanism of proof-of-stake known as *leased proof-of-stake* (LPoS). This works in the same way as proof-of-stake but with some improvements. It tries to address the centrality problem of proof-of-stake. It typically enables the nodes with low balances to participate in block verification by adding a leasing option to them. This leasing provision allows the wealth holders who have higher balances to lease their funds for some specific amount of time to nodes with low balances. The leased amount for that period will be in possession of the original wealth holders during the lease contract. However, it will increase the chance of solving a block for nodes with low balances. When these nodes solve a block, they will share the reward with the wealth holders, which is proportional to the leased fund. This decentralized approach makes blockchain more secure.

Proof-of-importance is a modified version of proof-of-stake. In this, a few additional factors are added to determine the next wining node. These factors include the number of transactions occurred to/from that node and its reputation. The reputation of the node is specified by a particular system-defined function. Therefore, productive network activity of nodes is considered in this method of consensus that is more efficient than only nodes' balances. The cryptocurrency *NEM* uses this method for consensus. It yields high throughout and exhibits comparatively low latency, and at the same time, it does not require large computational or network overheads.

4.3.3 Proof-of-Activity

Proof-of-activity is a hybrid method of consensus on the basis of proof-of-work and proof-of-stake. In this, the miners first try to solve a hash function in a competition to find the next block similar to proof-of-work. However, the solved block is different than have been seen in proof-of-work. It does not contain any transaction, but only the header and the miner's address. The transactions are then added to the block. According to the solved block's header, a group of validators is chosen to sign the new block in order to reach consensus, which is done by using proof-of-stake. This approach is safer against attacks but contains higher delay period.

4.3.4 Proof-of-Elapsed-Time

Proof-of-elapsed-time is also consensus mechanism used in blockchain. It prevents high-resource utilization and high-energy consumption while keeping the process more efficient by using a fair lottery system. It is based on spreading the chances of winning fairly across the largest possible number of nodes in the network. All the participating nodes in the network are required to wait for a randomly chosen time period. The first one to complete the designated waiting time wins the new block. In the blockchain network, each node generates a random wait time and goes to sleep for that specified duration. The one to wake up first (i.e., the one with the shortest wait time), wakes up and commits a new block to the blockchain. Then, it broadcasts the necessary information to the whole peer network and the same process repeats for the next block. Proof-of-elapsed-time mechanism needs to ensure the following important factor:

1. The participating nodes are genuine while selecting a time (i.e., they are not greedy to select shorter period in order to win), rather they indeed choose randomly.
2. The winner node has indeed completed the waiting time.

The proof-of-elapsed-time concept was invented by the famous chip-manufacturing giant Intel during 2016. It offers a readymade high-tech tool Intel Software Guard Extensions (SGX) to solve the computing problem of “random leader election”. This in-built mechanism allows applications to execute trusted code in a protected environment. It also ensures that the aforementioned two factors are fulfilled. Participants are requested to wait for a time, and the chip with the shortest wait time is elected as the leader. This elected leader can provide an attestation alongside the new block to convince other participants two things, which are as follows:

1. It is indeed the shortest wait time.
2. It did not broadcast the block until the wait time was over.

An alternative to this approach is known as *Resource-Efficient Mining* (REM) (Zhang 2017). In this, a trusted hardware is used. which proposes computing useful proof-of-work. Every instruction cycle for the useful proof-of-work can be considered as a lottery ticket. The participant whose cycle wins the lottery is authorized to mint a new block.

4.3.5 Proof-of-Burn

Proof-of-burn is an alternative consensus algorithm that targeted to address the energy consumption issue of proof-of-work. It is often known as proof-of-work without wasting energy. It works on the principle of allowing the miners to "burn" the virtual currency tokens, which refers to sending coins to an irretrievable address. In other words, the process of burning coins consists of sending these to a public verifiable address where they become inaccessible and useless. This grants them the right to write blocks in proportion to the coins they burnt. An analogy given by its inventor is that burnt coins are mining rigs. This provides the power to the miner to mine blocks by purchasing a virtual mining rig. More coins burnt by the miner provide a bigger virtual mining rig. In order to burn the coins, miners send them to a verifiably unspendable address that does not consume resources other than the burnt coins. Furthermore, this process ensures that the network remains active and agile. In a typical implementation, miners are allowed to burn the native currency or the currency of an alternative chain, for example, Bitcoin. In exchange, the miner receives a reward in the native currency or a token from blockchain.

To prevent any possibility of undue advantages for early adopters, the proof-of-burn intrinsically possesses a mechanism that promotes the periodic burning of coins to maintain mining power. This is achieved by the fact that the power of burnt coins "decays" or reduces partially each time a new block is mined. This is to say that the mining rigs become obsolete with the passage of time. Actually, it encourages regular activity by the miners, instead of the one-time early investment. In order to remain in the game with a competitive edge, miners also need to invest periodically in better equipment, similar to work done in physical mining rigs with advancement in technology.

Implementation that uses proof-of-burn can be customized. For example, Slimcoin, which is a virtual currency network, allows a miner to burn coins that not only gives him/her right to compete for the next block but also provides the chance to receive blocks for a longer period of time, ranging in multiple years. This currency network implementation combines three consensus algorithms – proof-of-work, proof-of-stake, and proof-of-burn. The process of obtaining the coins involves proof-of-work, the more coins one burns the more chances one has to mining rights that ensures proof-of-stake, and the whole ecosystem follows the proof-of-burn concept.

In order to prevent early adopters from benefitting too much or attacking the system, the power of burnt coins decays every time a block is mined. However, it does not decay significantly with respect to time. The mining rigs eventually become obsolete because better technology becomes available with time. Thus, in order to stay competitive, the miners will have to renew their equipment from time-to-time. The same is true for proof-of-burn, that is, if one wants to maintain the minting power, it must burn coins periodically. Proof-of-burn consensus protocol does not consume as much energy as proof-of-work.

4.3.6 Proof-of-Proof

Proof-of-proof is a relatively new concept as compared to other consensus protocols. Regardless the architecture of any blockchain, it typically allows to inherit same the proof-of-work security of Bitcoin blockchain. This is achieved in a decentralized, transparent, trustless, and permissionless manner. In this consensus protocol, a new type of miner is introduced, which performs periodic publications of one blockchain's current state to another. These publications of the state are then referred in the event of potential blockchain reorganization. Proof-of-proof requires that the existing blockchain has some means of creating new blocks such as proof-of-work, proof-of-stake, etc. It also aims to enable a security inheriting (SI) blockchain to inherit the complete proof-of-work security from the existing blockchain or security providing (SP) blockchain (sometimes also known as *parent blockchain*). This inheritance does not impose any non-trivial limitations on the blockchain and at the same time it does not require the permission of the existing blockchain. The non-mining users of the blockchain network should not have to interface with the parent blockchain network, nor should they be required to hold any of its native token.

The miners in proof-of-proof serve as the transactional bridges between an SI blockchain and an SP blockchain. These miners will take the recent blockchain state data from the SI blockchain and along with some identifier, publish it to the SP blockchain. This allows them to later receive compensation by creating an SP blockchain transaction with the SI blockchain state data. This is achieved by embedding an identifier in the SI blockchain, submitting the same to the SP blockchain network, and waiting for the transaction to be included in an SP blockchain block. The miner then constructs some form of proof of publication, and adds necessary identifiable information (of 16 bytes) for them to take credit for the publication. The miner then submits this proof to the SI blockchain as special proof-of-proof mining transaction. The mining transaction demonstrates that SI blockchain state data was included in the SP blockchain transaction. The simplest algorithm for this requires that the proof-of-proof miner submits adequate SP blockchain block headers (to build from the SI blockchain network's previously known) to the header of the block in which the proof-of-proof publication occurs. Peers on the SI blockchain validate a proof-of-proof mining transaction by checking the validity of the published SI blockchain state data and checking for its inclusion in the provided SP blockchain transaction. Checking the validity of SI blockchain state data only require looking back in the SI blockchain for the block corresponding to the published state data.

4.3.7 Proof-of-Capacity

In the proof-of-capacity approach, the participants vote on new blocks weighted by their capacity to allocate a non-trivial amount of disk space. Miners secure the network by using their computer storage instead of the more common energy-expensive method proof-of-work. In this computation, a process known as plotting is done only once and caches the results of the work done on the hard disk space. Then the mining only requires reading through the cache. It may be noted that the hard disk drive (HDD) is idle most of the time, and reading through the plot files only for a few seconds for each block is termed as *condensed proof-of-work*. During plotting, a file is created which contains pre-computed hashes that can be used to forge blocks in the blockchain. Incentivization mechanism can be similar to that of proof-of-work.

4.4 Byzantine[1] Agreement Methods

Before discussing the technical details about this topic, let us discuss the Byzantine Generals Problem. This problem describes a situation where spread out army units need to coordinate their attack but cannot trust each other and arrive at a consensus.

Let us consider a situation where multiple generals and their individual armies have surrounded a city to attack. By some means, the majority of the generals have to coordinate and come to a decision to either attack or retreat at the same time. In case they fail to do so, the situation will end in a major failure. The problem at hand is to come on a consensus between the Byzantine generals over the attack strategy with an assumption that some generals (or their messengers) may be traitors. These traitors may try to adopt treacherous actions to prevent loyal generals from reaching a consensus on how to conquer the city.

The main problem preventing consensus is low trust, which is due to the following reasons:

1. The generals are geographical far enough with no direct communication.

2. The generals cannot even see each other.

3. The generals must trust their messengers to deliver the messages.

4. One or more generals (or messengers) may be a traitor and could send false messages to other generals.

5. Worst of all, the enemy city may catch a messenger and send false messages back.

We will now discuss the Byzantine agreement methods. When we talk about cryptocurrencies, we can easily map trust-related issues with that of Byzantine generals. In the blockchain, we need to record and verify the block simultaneously in hundreds and thousands of computers across the globe and none of the computers can be trusted as a reliable source. The typical mapping of the Byzantine Generals Problem onto computer systems is that the computers are the generals and their digital communication system links are the messengers. The formal problem of obtaining Byzantine consensus was first conceived and formalized by Robert Shostak as an interactive consistency problem. It was under the NASA-sponsored Software Implemented Fault Tolerance (SIFT) project in the Computer Science Lab at SRI International. This project was the brainchild of John Wensley, and was based on the idea of using multiple general-purpose computers that would communicate through pairwise messaging in order to reach a consensus, even if some faulty computers were part of the communication. When the project began, it was not clear how many computers in total are required to guarantee that a conspiracy of n faulty computers could not *ruin* the efforts of the correctly operating ones to reach consensus. Shostak proved that a minimum of $3n + 1$ computers are needed to reach a consensus. He devised a two-round $3n + 1$ messaging protocol that would work in a trivial case for $n = 1$. Marshall Pease, his colleague, generalized the algorithm for any $n > 0$, and also proved that $3n + 1$ is both necessary and a sufficient number.

In context to blockchain, each general represents a network node, and these nodes need to reach a consensus on the current state of the block. This is to say that the majority of participants within a distributed network have to agree and execute the same action in order to avoid complete failure. We present several Byzantine agreement-based consensus methods in the following subsections.

Fact Alert

In 2007, a quantum protocol for Byzantine agreement was demonstrated experimentally by physicists S. Gaertner, M. Bourennane, C. Kurtsiefer, A. Cabello, and H. Weinfurter using a four-photon polarization-entangled state. This shows that the quantum implementation of classical Byzantine agreement protocols is indeed feasible.

4.4.1 Practical Byzantine Fault Tolerance

Practical Byzantine Fault Tolerance (pBFT) is the simplest method, which requires participation from all the nodes in the voting process. In order to reach a consensus and add the next block, more than two-thirds of all the nodes should agree upon that block. This approach can tolerate malicious behavior from up to one-third of all the nodes to perform normally. For example, with one malicious node in a system, there should be at least four nodes to reach a correct consensus, otherwise, the consensus is not achieved. As already mentioned, pBFT is the simplest of all, and the consensus is reached quicker and more economically as compared to proof-of-work. It does not require owning assets similar to proof-of-stake to take part in the consensus process. This method is well-suited for the private blockchains that are controlled by third-parties such as the Hyperledger. However, for permissionless, public blockchains, it is not the best choice because of its limited scalability and comparatively low tolerance towards malicious activities. One may note that the pBFT has high throughput, low latency, and low computational overhead. Therefore, this is more suitable where the computation resources are limited, such as Internet of Things.

4.4.2 Delegated Byzantine Fault Tolerance

Delegated Byzantine Fault Tolerance (dBFT) method follows the same rules as pBFT, except that it does not require the participation of all the nodes for adding a block to make it more scalable. In this approach, some nodes are selected according to some rules as delegates of other nodes, and these delegated nodes pursue the consensus protocol similar to pBFT. NEO, a cryptocurrency, uses this method of consensus in its blockchain setup.

4.4.3 Stellar Consensus Protocol

The Stellar Consensus Protocol (SCP) is a variant of pBFT, and it was proposed by Mazieres using federated Byzantine Fault Tolerance (FBFT) as the backbone. Nodes in FBFT belongs to intersecting groups (the federates) run a local consensus protocol among them. As this method is open to the public and decentralized, it allows everyone to participate in the consensus protocol. Similar to the web transactions, it has a very low latency. In order to reach a consensus, this is the first Byzantine agreement-based consensus method that provides users with the maximum freedom to choose among different combinations of other participants to trust. Using this approach, SCP reaches robustness through quorum slices. A set of nodes that participate in the consensus protocol forms a quorum. The quorum slice is its subset that helps a node in its agreement process. It is constructed by the individual trust decisions made by the nodes in the blockchain. The quorum slices connect the whole network together in a similar fashion as peering networks create Internet by binding together while the quorums are selected by the nodes involved in the transactions. There are two steps in SCP consensus – nomination protocol and ballot protocol.

1. **Nomination protocol:** It is executed where new values known as *candidate values* are proposed for the agreement. After receiving these values, each node in the quorum will vote for a single value among the candidate values. The process ends by unanimously choosing the values for that slot.
2. **Ballot protocol:** It is initiated after nomination protocol, and it involves a federated voting to either accept or reject the obtained values in the nomination protocol. Finally, the ballot for the current slot is finalized and aborted ballots are discarded. In situations that nodes cannot reach a consensus to abort or to commit a value, a higher valued ballot is initiated which can be considered as a new ballot protocol execution. SCP provides microfinance services on the blockchain platform.

4.4.4 Ripple Method

The Ripple method, similar to SCP, uses FBFT consensus approach that was proposed to reduce the latency of blockchain. This is a decentralized method, where each miner uses a trusted subset of nodes within the larger network to reach a consensus. It has two types of nodes in the network, which are as follows:

1. Server nodes that are responsible for the consensus protocol contains the Unique Node List (UNL).
2. Client nodes that only transfer funds. The nodes within an UNL are used for reaching consensus over new transactions.

The consensus is reached when 80% of the nodes within an UNL agree over a transaction. In any practical setup, this consensus protocol is executed every few seconds by all the nodes in order to reach consensus over new transactions, that is, Ripple. This method is mostly used for monetary purposes to enable transactions with no chargebacks and it can tolerate up to 20% faulty nodes in the UNL.

4.4.5 Tendermint

Tendermint is a permissioned consensus method, designed for hosting arbitrary application states. It belongs to the family of BFT consensus protocols. The nodes in Tendermint have different voting powers that are proportional to their stakes, which is contrary to pBFT where each node has equal voting power. Therefore, it can be considered as a hybrid consensus method based on proof-of-stake and pBFT. It can tolerate malicious activity from at most one-third of the total Byzantine voting power. Participants in this protocol are known as *validators*. These validators propose blocks of transactions and vote on them in turn. Pre-vote and pre-commit are the two steps in voting process. When more than two-third of the validators pre-commit for the same block and in the same round, then that block is added to the blockchain. In contrast to pBFT, in this method the validators are chosen based on proof-of-stake by locking their coins while punishing dishonest validators. It has high scalability, high throughput, and low latency, and is being used in several cryptocurrency.

4.4.6 Algorand

Algorand uses proof-of-stake consensus protocol built on Byzantine agreement and considered to be a novel public and permissionless blockchain implementation. Algorand has been proposed to address the limitations of decentralization, scalability, and security in the existing implementations of blockchain-based cryptocurrency. Like Bitcoin network, most of the available blockchain implementations are centralized to some degree in which the users with the highest hash power control the network. In contrast to this, in Algorand, each block is approved by a unique committee of users that are randomly selected. The random selection is done by the verifiable random function (VRF) in a private and non-interactive fashion by using the users' private key and public information from the blockchain. However, this selection is based on weights of the users assigned according to the amount of money in their account, which is something similar to proof-of-stake. After the committee members are selected, the committee reaches a consensus over the new block using a Byzantine agreement protocol.

There is a one main drawback of Algorand, which is the randomness used in producing VRF seeds, which can be prejudiced by an adversary. To overcome this shortcoming, a look-back mechanism is proposed to ensure strong synchrony and unbiased seeds. However, this makes the network vulnerable to a "nothing at stake" attack. The low latency (less than 1 min) of Algorand is desirable for a cryptocurrency with transaction finality of around 1 min. This feature is very positive in contrast to Bitcoin network where it takes about 10 min to grow a chain by one block and that it requires waiting for six blocks to ensure the transaction finality.

4.4.7 Paxos and Raft Consensus Protocols

Paxos and Raft are two well-known consensus protocols that have been around for roughly three decades. Paxos is a family of protocols that rely on a group of differing assumptions depending on the system while Raft is alternative consensus to Paxos, typically designed for better understand ability. In this chapter, comprehending both of these will be very helpful in cumulating the knowledge of how distributed consensus protocols work in cryptocurrencies such as proof-of-work and pBFT. Paxos was initially proposed in 1989 and is particularly elegant method of proving safety for fault-tolerant distributed consensus which is often viewed as challenging to understand due to its broad assumptions and complex behavior. However, in this chapter we attempt to present this in relatively simple form. Raft was developed as a simple alternative to Paxos and is equivalent to Paxos in performance and fault-tolerant guarantees.

4.4.7.1 What is Paxos?

In order to have distributed fault-tolerant consensus protocols involving live concurrence by the participants, Paxos was developed. The protocol ensures that a proposed value is eventually selected by the group of participants working towards consensus. In a nut shell there are three types of entities in Paxos consensus – proposers, acceptors, and learners. The goal of consensus is for a group of participants to come to an agreement on a single value per each round. A round of consensus begins when a proposer sends a proposed value to a group of acceptors. Acceptors may accept the proposed value by a given proposer. Once a certain threshold is achieved, then that value is approved by the participants in the network. In order for consensus to work correctly, the first condition of the protocol is: Acceptors must accept the first proposed value that they receive. This leads to the problem of several proposers sending proposed values that are accepted by acceptors, but all of them accept and no majority value is achieved since they are accepting the first proposed value. Paxos solves this by uniquely indexing each proposed value that an acceptor receives which allows them to accept more than one value in the proposal. A unique number defines each proposal, and the network selects a value once a specific proposed value is accepted by the majority of acceptors. This value is said to be as the chosen value. Multiple proposals can be chosen, but it is necessary to validate the safety property by guaranteeing that these proposals will all have the same value. As per Leslie Lamport's definition of the required second condition of Paxos that ensures safety: If a proposal with value v is chosen, then every higher-numbered proposal that is chosen has value v. In a scenario when the communication in the network is asynchronous, it is possible that certain acceptors have not received the chosen value and can be considered provided both conditions are not violated. Proposers employ certain limitations as messages to sets of acceptors along with the values. These are called *prepare requests* and contain two primary requests:

1. Promise never to accept a proposal less than n (n is the new proposal number).

2. Respond with the proposal with the highest number less than n that the acceptor has accepted.

Let us see its working now.

When the proposer receives the requested responses from a majority of the acceptors, it then issue a proposal with number *n* and value *v*, where *v* is the value of the highest-numbered proposal among the responses. However, it can be any value selected by the proposer if the responders reported no proposals. After receiving the acknowledgment of the acceptors the proposers subsequently send an accept request. The proposer then sends a commit message to the acceptors who can either ignore (without compromising safety) or indicate the success of the value commit. Once a certain threshold of acceptors has committed the value, then the protocol for that consensus round terminates and utters the value. The algorithm can accept values when a majority of nodes agree despite other nodes ignoring or denying a proposed value. As long as the proposal numbers are unique, the algorithm can select a value that guarantees safety. It is important to note that an acceptor only needs to remember the highest numbered proposal it has accepted. Conversely, a proposer can always abandon a proposal as long as it does not reissue a proposal with the same unique number.

Breaking down the roles of the proposer and acceptor in the protocol is as follows: Proposer submits proposal *n* to acceptors along with prepare request, wait for a majority to reply. If the majority of acceptor agrees to the reply, they will reply with the agreed value. If majority reject, then abandon the current proposal and restart the process. However, if the majority accepts the proposer subsequently sends a commit message with *n* and the value. If the majority of acceptors accept the commit message, protocol round is said to be completed.

While Acceptor, when receiving the proposal and he compare it to the highest numbered proposal already agreed to in recent past. If *n* is higher than accept the proposal, if *n* is lower than reject the proposal. Proposals can make multiple proposals but need to follow the algorithm for each proposal individually. Finally, the role of the learners is to discover that a majority of acceptors have accepted a proposal from the proposers. A distinguished learner is selected that propagates the chosen value to the other learners in the network. Variations of this process can be used where either all acceptors inform corresponding learners of their decisions or acceptors respond to a distinct set of learners who then propagate the message to the rest of the learners.

Formally, the Paxos algorithm distinguishes a leader (proposer) for each round that is required to make progress. Acceptors can acknowledge the leadership of a proposer which allows Paxos to be used to select a leader within a cluster of nodes. Paxos may stall if two proposers are competing for the leader position with no agreement on which one is the leader. However, it is unlikely that this state of non-termination will persist though.

4.4.7.2 What is Raft?

Raft was created as a more practical version of Paxos with the same fault-tolerance and performance guarantees. This algorithm employs a leader and follower model based on the assumption that a cluster of nodes has only one elected leader. The leader manages the log replication across the participating nodes and is replaced once it fails or disconnects. A leader is also elected when the algorithm begins. Leader plays a vital role in consensus and is distinguishable in specific algorithms. For instance in Bitcoin, using the proof-of-work, leader selection is achieved through the lottery-like mining process for each round. In pBFT, leader selection is performed through a round-robin style format. In this algorithm, the selection of the leader is done through a process initiated by a candidate node. If candidates do not receive communication during a phase known as the election timeout, then they vote for themselves after increasing their term-counter and broadcast it to the other nodes. Candidates become followers of other candidates who have a term number at least as large as theirs, and this ripple effect continues among the nodes until one candidate receives a majority of followers. The leader controls log replication among the nodes where it sends the client request commands to its followers. The request is said to be committed if a majority of followers confirm replication followed by all the followers to apply the commits to their local state machines. Leader candidates are required to have a more up-to-date log than follower logs. If a candidate's log is less up-to-date than a potential follower (a voter in this context), then the candidate is rejected.

To summarize, both Paxos and Raft are important consensus protocols that are core components of the larger distributed fault-tolerance ecosystem. The consensus protocols used in cryptocurrency networks, though not directly deployed but they derive many of their characteristic assumptions from the design of both Paxos and Raft protocols.

4.4.8 Lamport-Shostak-Pease Algorithm

We have already mentioned the NASA-sponsored SIFT project. Here, we will present in detail the algorithm which is sometimes also referred to as the Oral Message algorithm OM(m), $m > 0$, which solves the Byzantine agreement problem for $3m + 1$ or more processors in the presence of at most m faulty processors.

Let n denote the total number of processors (clearly, $n \geq 3m + 1$). The algorithm is recursively defined as follows for $m = 0$ and $m > 0$.

Algorithm when OM(0):

Step 1: The source processor sends its value to every processor.

Step 2: Each processor uses the value it receives from the source. (If it receives no value, then it uses a default value of 0).

Algorithm when OM(m), $m > 0$:

Step 1: The source processor sends its value to every processor.

Step 2: For each i, let v_i be the value processor i receives from the source. (If it receives no value, then it uses a default value of 0). Processor i acts as the new source and initiates Algorithm OM(m

– 1) wherein it sends the value v_i to each of the n – 2 other processors.

Step 3: For each i and each j ($\neq i$), let v_j be the value processor i received from processor j in Step 2 using algorithm OM(m – 1). (If it receives no value, then it uses a default value of 0). Processor i uses the value *majority* ($v_1, v_2, \ldots, v_{n-1}$).

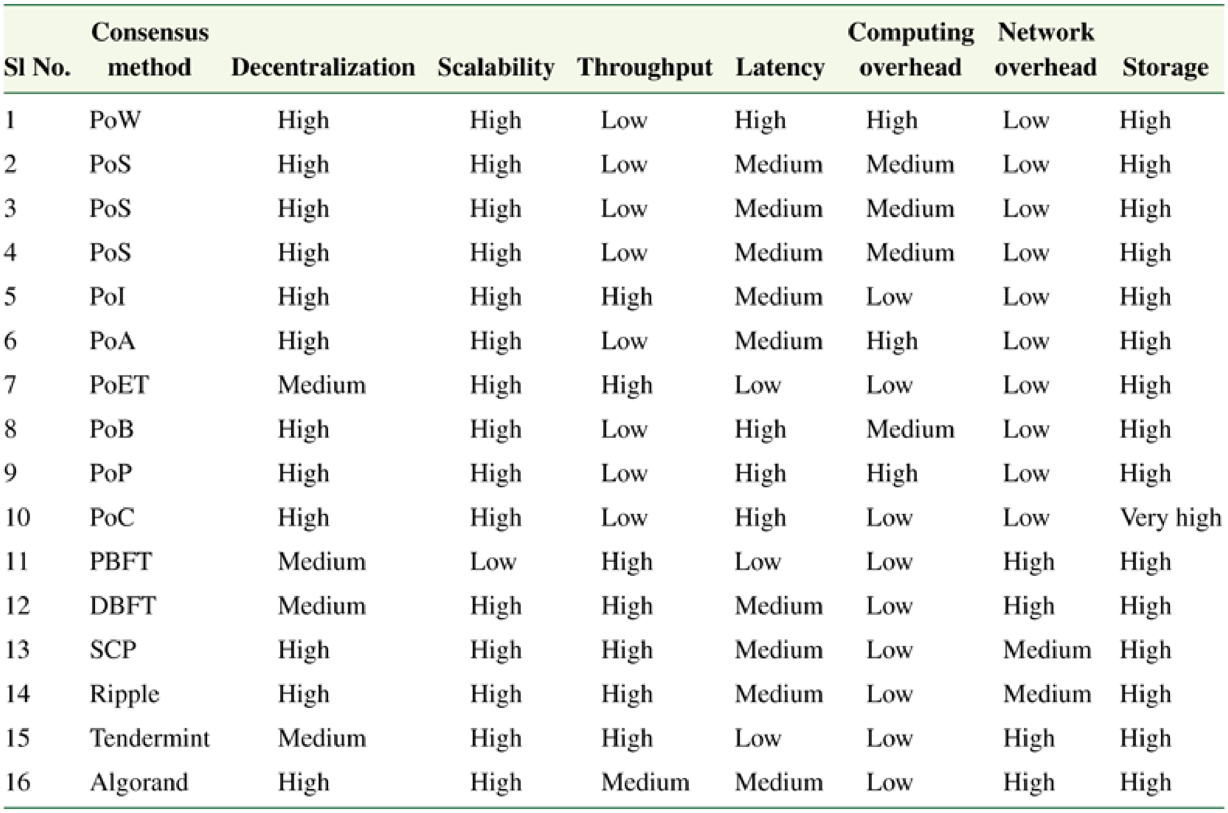
As we can see, this algorithm is evidently quite complex where the processors are successively divided into smaller and smaller groups. The Byzantine agreement is recursively achieved within each group of processors (Step 2 of "algorithm OM(m – 1)"). Step 3 is executed during the folding phase of the recursion, where a *majority* function is applied to select the majority value out of the values received in a round of message exchange (Step 2). The function *majority* ($v_1, v_2, \ldots, v_{n-1}$) computes a majority value of the values $v_1, v_2, \ldots, v_{n-1}$ if it exists (otherwise, it returns the default value 0).

The execution of the algorithm OM(m) invokes n – 1 separate executions of the algorithm OM(m – 1), each of which invokes n – 2 executions of the algorithm OM(m – 2), and so on. Therefore, there are (n – 1)(n – 2)(n – 3) … (n – m) with k separate executions of the algorithm OM(k), $k = n - 1, n - 2, n - 3, \ldots$

SUMMARY

Table 4.1 presents the overview of various consensus methods.

Table 4.1 **Overview of consensus methods**



Sl No.	Consensus method	Decentralization	Scalability	Throughput	Latency	Computing overhead	Network overhead	Storage
1	PoW	High	High	Low	High	High	Low	High
2	PoS	High	High	Low	Medium	Medium	Low	High
3	PoS	High	High	Low	Medium	Medium	Low	High
4	PoS	High	High	Low	Medium	Medium	Low	High
5	PoI	High	High	High	Medium	Low	Low	High
6	PoA	High	High	Low	Medium	High	Low	High
7	PoET	Medium	High	High	Low	Low	Low	High
8	PoB	High	High	Low	High	Medium	Low	High
9	PoP	High	High	Low	High	High	Low	High
10	PoC	High	High	Low	High	Low	Low	Very high
11	PBFT	Medium	Low	High	Low	Low	High	High
12	DBFT	Medium	High	High	Medium	Low	High	High
13	SCP	High	High	High	Medium	Low	Medium	High
14	Ripple	High	High	High	Medium	Low	Medium	High
15	Tendermint	Medium	High	High	Low	Low	High	High
16	Algorand	High	High	Medium	Medium	Low	High	High

SHORT ANSWER QUESTIONS

1. What is agreement protocol?
2. What is distributed system consensus?
3. How does distributed consensus work?
4. What is consensus mechanism?
5. Why do we need consensus?
6. How does consensus algorithm work?
7. What are consensus algorithms blockchain?
8. What is Byzantine Fault Tolerance consensus?
9. What is Byzantine agreement problem in distributed system?
10. What is the minimum number of nodes needed to reach consensus if there are two possible Byzantine fault?

LONG ANSWER QUESTIONS

1. Name and analyze five different types of consensus approaches on their scalability, latency, throughput, storage, and communication overheads.

2. Show how a solution to the consensus problem can be used to solve the interactive consistency problem.

[1] The eastern part of Europe controlled by the Roman Empire from approximately 330 AD to 1453 AD. Byzantine describes this as Byzantine Empire.

CHAPTER 5

Blockchain Components

Learning Objectives

After reading this chapter, you will be able to:

- Understand the history of blockchain components.
- Describe the Ethereum platform.
- Understand the Ethereum Virtual Machine (EVM), clients of EVM, Ethereum key pairs.
- Know about Ethereum addresses, its wallets, transactions, and languages.
- Describe Ethereum development tools.

5.1 Introduction

We know that a blockchain is a non-immutable decentralized ledger of digital transactions. The technology is architecture in a way to make it extremely difficult to change the rules that define the structure of a database or its content without consensus among the nodes in the chain that use it. A blockchain is typically categorized based on its permission model – public, private, and hybrid. However, the overall core components of these are almost similar. These components collaborate in performing secure communication among distrusted nodes by publishing a distributed log of the committed transactions, using a consensus mechanism.

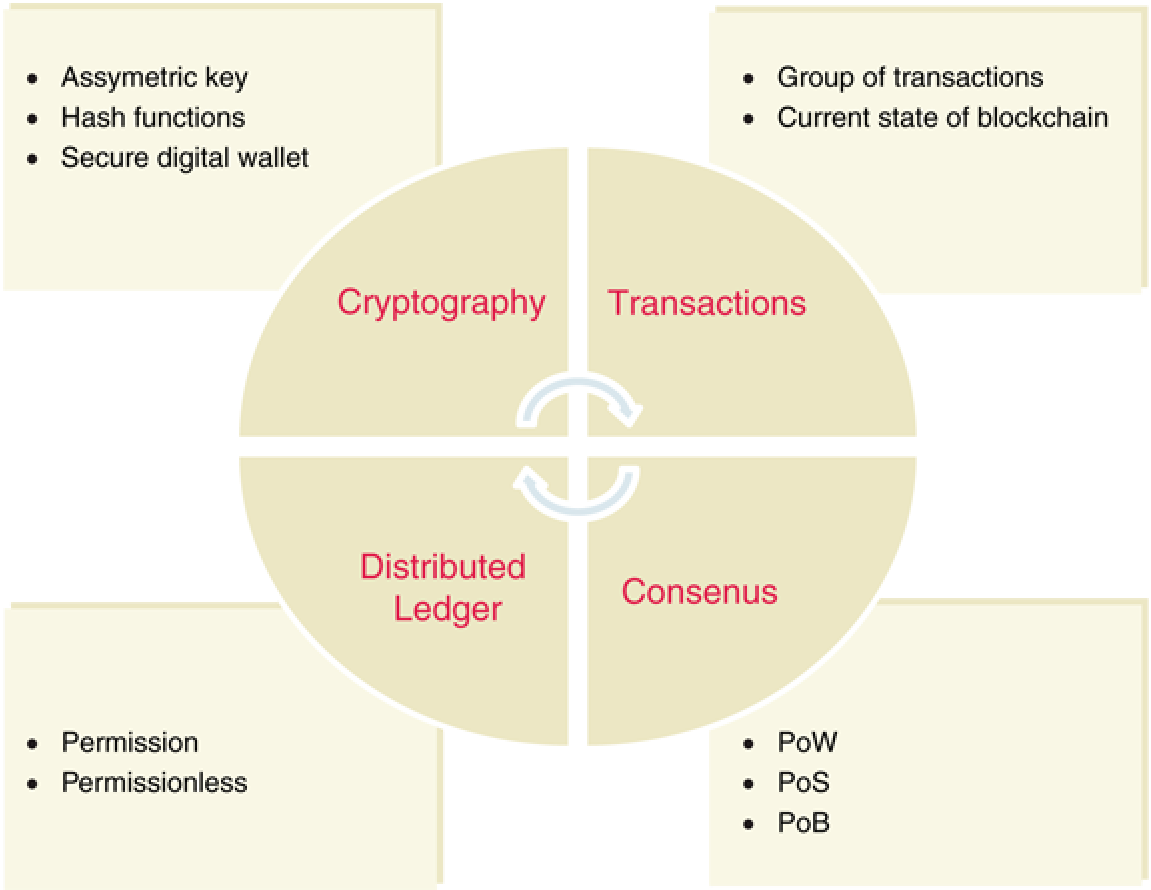


Figure 5.1 Components of blockchain.

Permissionless blockchains are decentralized ledger platforms open to anyone publishing blocks, without needing permission from any authority and often use open source software. Within a permissionless blockchain, any user can read and write to the ledger that leads malicious users to publish blocks in a manner that subverts the system. Thus, in order to prevent this, permissionless blockchain networks often utilize a multiparty agreement or “consensus” approach, as discussed in detail in Chapter 4.

Permissioned blockchain are analogous to a corporate intranet, often deployed for a closed group of organizations and individuals. These blockchain are ones where users publishing blocks must be authorized by some authority, be it centralized or decentralized. Since only authorized users are maintaining the blockchain, it is possible to restrict read access and restrict who can issue transactions. Thus, this allows anyone to read the blockchain or they may restrict read access to authorized individuals. This may also allow anyone to submit transactions to be included in the blockchain but may restrict access only to preauthorized individuals. Permissioned blockchain may be instantiated and maintained using open-source or closed-source software.

A hybrid or consortium blockchain can be considered as a partially private and permissioned blockchain. In this, there is not a single organization but a set of predetermined nodes are responsible for consensus and block validation. These nodes had power to decide who can be part of the blockchain and who had which type of rights. For block validation, a multi-signature scheme is typically used. A block is considered to be valid, only if it is signed by valid authorized nodes. This makes it is a partially centralized system, owing to the control by some selected validator nodes. It is the consortium who decided whether read or write permissions would be public or limited to the network participants.

For secure operation of the blockchain, it utilizes the capabilities of public key cryptography, which is discussed in detail in Chapter 6. All the users need to possess a digital wallet with the user's private key to perform any exchange using digital signatures generated using that private key.

Blockchain enables sharing and exchanging of information among peer-to-peer nodes. These exchanges take place by means of files containing information transferred from one node to another. It is initially generated by a source node and then broadcasted to the entire network for validation. The current state of blockchain is represented by these transactions, which are continuously generated by the nodes. After each transaction, the state of blockchain changes to a new state. With a huge number of transactions generated in unit time, it is very important to validate and verify the genuine ones and discard the fake. Therefore, all the nodes should agree on a common content updating protocol for the ledger, in order to maintain a consistent state. Without majority consent, blocks should not be accepted to be a part of the blockchain. This mechanism, by which blocks are created and added to the existing ledger for future use, is known as *consensus*. In short, the complete block formation process in the blockchain can be divided into the following phases:

1. Transaction generation and verification.
2. Consensus execution and block validation.

It should be noted that there should be no direct transactions between source and destination. Instead, all the transactions should be announced to the entire network for verification through broadcasting. To add the block to the blockchain, nodes put their resources at work and start the mining process to solve the cryptographic puzzle by finding proof-of-work. Upon solving the puzzle, the block is broadcasted to the entire network. The node that solves the puzzle first is rewarded with certain points as declared a priori.

So you see how many components are involved in setting and establishing a blockchain. This chapter discusses about how these components of blockchain are put into practical purpose for people to use. One of the prominent platforms, Ethereum, is discussed in this chapter.

5.2 Ethereum

We know that Internet is a network of networks consisting of several thousands of connected computers that are being identified in domain name system (DNS). Because of the openness of Internet technology, it is a place where individual or multinational corporate giants have a role to play. However, there are problems when something happens to one of the services from giants such as Google, Facebook, Apple, etc. Then one cannot communicate to friends, do searches, and so on until the issue is resolved. For example, in October 2016, the Dyn cyberattack affected major sites including Amazon, Twitter, Reddit, PayPal, Netflix, Airbnb, BBC, Fox News, HBO, etc., though the list is nod endless, but cover all major Internet domains.

Flash Quiz

Do you know where DNS is hosted on the Internet?

Now, imagine an Internet that has made everyone's computers. For example, imagine Facebook running on every computer so that when any of the computers is having a problem and has fallen down, there are other computers on the network that will still be working and Facebook services can still be available for the other users. In this scenario, the only way for all the services to put-off is when, literally, every computer on the network is shut down. On similar lines and in relatively not so technical terms, Ethereum is a platform like the Internet with availability of its service on all the computers that have Ethereum client running on it. But one may wonder how every computer can run Google. The answer is Ethereum. It is a decentralized blockchain platform for "building uninterruptible applications". Ethereum can be used for virtually any kind of transaction or agreement at a very lower cost as compared to the conventional alternatives, in a way that is decentralized, trustless, secure, safe and live, and censorship-resistant.

Decentralized technology uses peer-to-peer computer networks. The decentralized set-up are not subject to the whims of a central authority such as a government or server administrator (such as Google or Facebook). Blockchain networks are trustless environments as it does not mandate trusted third party to certify transactions. However, there is still a great deal of trust needed to work within a blockchain network in the form of cryptographic technologies utilized. The unbroken cryptography processes in the blockchain ensure that data stored on a blockchain is secure and tamper-proof. Censorship-resistance implies that everyone can transact with the blockchain network in the same terms and conditions, regardless of their personal identity or qualities.

Ethereum as a platform facilitates for building decentralized apps and extending the aforementioned features. These can be used for any activity that has an economic or governance aspect without restricting to any specific domain, such as:

1. Transparent governance for communities.
2. Secure communications, which involves authentication and messaging.
3. Secure backbone for e-commerce and Internet of Things.

Ethereum extend wide range of features that include Integrated Development Environment (IDE) for debugging, development, and deployment of applications. We will discuss all this in subsequent sections. However, we must first learn about the history of Ethereum and its development.

5.3 History

The name Ethereum was picked by Vitalik Buterin. After browsing Wikipedia articles about elements and science fiction, he found the name and noted, "I immediately realized that I liked it better than all of the other alternatives that I had seen; I suppose it was the fact that sounded nice and it had the word 'ether', referring to the hypothetical invisible medium that permeates the universe and allows light to travel."

Fact Alert

Bitcoin has a hard-coded rule that limits the total number of currency units that will ever exist to 21 million (or 20,999,999.9769 to be precise). This number will be reached approximately in the year 2140.

He described Ethereum in his white paper, when he was a programmer and was involved with *Bitcoin Magazine* in 2013. With the goal of building decentralized applications, he argued that Bitcoin needed a scripting language for application development. Failing to achieve consensus, he proposed to develop a new platform with a more general scripting language. In January 2014, he announced about Ethereum at the North American Bitcoin Conference in Miami. In 2014, with the help of a Swiss company, formal development of the Ethereum software project began as a non-profit foundation Stiftung Ethereum. Funds for the development was generated during July–August 2014 by an online public crowd sale, where the participants buying the Ethereum value token (ether) with another digital currency, Bitcoin. While there was early admiration for the technical innovations of Ethereum, doubts were raised about its security and scalability. In March 2017, various blockchain start-ups, research groups, and Fortune 500 companies announced the creation of the Enterprise Ethereum Alliance with 30 founding members and by July of the same year, there were over 150 members in the alliance.

Flash Quiz

Can you find the release date of Serenity?

As part of proof of concept, several codenamed prototypes of Ethereum platform were developed before the official launch of Olympic by the foundation. In order to identify bugs, the release of Olympic network provided users with a bug bounty of 25K ether for stress testing the limits of Ethereum blockchain. As mentioned in Table 5.1 chronological development of the Ethereum platform, the first version of Ethereum platform was named "Frontier" and its experimental release was on July 2015. The first stable release of Ethereum platform is considered to be "Homestead". It included the improvements to transaction processing, gas pricing, and security. The upgrades in the protocol are achieved by means of a soft fork of the open source code base. The "Metropolis Part 1: Byzantium" soft fork took effect on 16 October 2017, which provided more flexibility for smart contract developers and included changes to reduce the complexity of the Ethereum Virtual Machine (EVM). The "Metropolis Part 2: Constantinople" is a hard fork that occurred at block number 7,280,000 on 28 February 2019, due to the following reason.

Quick Tip

Changes to a blockchain network's protocol and data structures are called forks. There are soft forks and hard forks.

Table 5.1 **Chronological development of Ethereum platform**

Version	Code Name	Release Date
0	Olympic	May 2015
1	Frontier	30 July 2015
2	Homestead	14 March 2016
3	Metropolis (Byzantium)	16 October 2017
3.5	Metropolis (Constantinople)	28 February 2019
4	Serenity	To be announced

In 2016, a decentralized autonomous organization (DAO), which is a set of smart contracts developed on the platform, raised a record USD150 million in a crowd sale to fund the project. However, the DAO was exploited in June when USD50 million in ether denomination were taken by an unknown hacker (Popper 2016; Price 2016). The event sparked a debate in the community as to whether Ethereum should perform a contentious "hard fork" to reappropriate the affected funds. This ensued in a dispute, which resulted in splitting the network into two. Ethereum Classic continued on the original blockchain whereas Ethereum continued on the forked blockchain.

After the aforementioned hard fork, Ethereum was subsequently forked twice:

1. The first time was in the fourth quarter of 2016 to deal with other attacks.
2. The second time was at the end of November 2016 with increased protection against DDoS, which de-bloated the blockchain and thwarted further spam attacks by hackers.

5.4 Ethereum Virtual Machine

As mentioned earlier, Ethereum is a programmable platform for blockchain. It gives users a set of predefined operations and provides flexibility to users to create their own operations of any complexity of their choice. The platform serves for many different types of decentralized blockchain applications, including but not limited to cryptocurrencies. In terms of computer science, Ethereum is "Turing complete". At the core of the platform is the EVM, which is designed and developed in such a manner that it can execute code of arbitrary algorithmic complexity. It is the runtime environment for smart contracts in Ethereum and has a 256-bit register stack, designed to run the same code exactly as intended. Developers can create applications for execution on EVM using friendly programming languages modeled on existing languages such as JavaScript and Python. EVM have implemented several popular languages such as C++, C#, Elixir, Erlang, Go, Haskell, Java, JavaScript, Python, Ruby, and Rust.

Fact Alert

Do not confuse Ethereum Virtual Machine with Electronic Voting Machine.

Like any other blockchain, Ethereum platform also includes a peer-to-peer network protocol. The Ethereum blockchain database is maintained and updated timely by many nodes connected to the network almost concurrently. All the nodes of the Ethereum network run the EVM and execute the same instructions. For this reason, Ethereum is sometimes described as a “world computer”. This massive parallelization of computing across the entire Ethereum network is not for efficient computation, rather every Ethereum node runs the EVM in order to maintain consensus across the blockchain. As of 1 February 2018, there were 27,500 nodes in the main Ethereum network. The growing number of decentralized nodes and the consensus provided by them to the Ethereum network gives high level of fault tolerance and makes data stored on the blockchain forever immutable and censorship-resistant while ensuring zero downtime. The Ethereum platform itself is featureless, similar to programming language. It is up to entrepreneurs and developers to decide the purpose of its use. However, due to the nature of certain application types, they benefit more than others from Ethereum’s capabilities. For example, applications that automate direct interaction between peers or facilitate coordinated group action across a network are ideal. Ethereum’s impact may be more far reaching, interactions or exchanges of any complexity could be carried out automatically without any human intervention and reliably using code running on Ethereum. Any environment (even beyond financial applications) where trust, security, and permanence are of prime importance (e.g., asset-registries, voting, governance, and Internet of Things), could be massively influenced by the Ethereum platform.

5.5 Working of Ethereum

Ethereum's basic unit is the account that is in contrast to the Bitcoin blockchain, which is purely a list of transactions. The Ethereum blockchain tracks the state of every account, and all state transitions on Ethereum blockchain are transfers of value and information between accounts. There are two types of accounts:

1. Private keys controlled Externally Owned Account (EOA).
2. Contract code controlled Contract Account (CA).

The CA can only be "activated" by an EOA. The basic difference between these is that EOAs are controlled by human-user control, as they can control the private keys. On the other hand, CAs are administrated by their internal code. In case they are "controlled" by a human user, it is for a simple reason that they are programmed to be controlled by an EOA with a certain address. This is in turn controlled by one that holds the private keys associated with that EOA. *Smart contracts* refer to code in a CA. These are the computer programs that execute when a transaction is sent to that account. By deploying these computer programs to the blockchain, users can create new contracts. CAs are operative only when instructed by an EOA to do so. Thus, it is not possible for a CA to perform native operations such as random number generation or application programming interface (API) calls. However, they can do these things only if instructed by an EOA. This is because Ethereum mandates the nodes to agree on the outcome of computation, which requires a guarantee of strictly deterministic execution. This is similar to Bitcoin, where users pay small transaction fees to the network. By adopting this, the Ethereum blockchain is protected from frivolous or malicious computational tasks, such as distributed denial-of-service (DDoS) attacks or infinite loops. The sender of a transaction must not only pay for each step of the "program" they activated, but also for computation and memory storage. Ethereum's native value-token *ether* is used to pay the fees. The nodes that validate the network collect these transaction fees. These nodes are "miners" in

Ethereum network, which receive, propagate, verify, and execute transactions. The miners then group the transactions by including many updates to the "state" of accounts as "blocks" in the Ethereum blockchain. The miners then compete with one another for their block to be considered as the next one to be added to the blockchain. For each successful block the miner mines, they are rewarded with ether which is justifiable as an economic incentive for people to dedicate hardware and electricity to the Ethereum network. In order to successfully "mine" a block, miners in Ethereum network are tasked to solve a complex mathematical problem that is something similar to the Bitcoin network, which is typically known as proof-of-work. Any computational problem that requires more magnitude of resources to solve algorithmically than it takes to verify the solution is considered to be a good candidate for proof-of-work. In order to discourage centralization arising due to the use of specialized hardware (e.g., application-specific integrated circuit, or ASIC) in the Bitcoin network, Ethereum chose a combination of memory plus hard computational problem for proof-of-work. It has been observed that if the problem requires both memory as well as CPU, the ideal hardware is in fact a general computer. This empowers Ethereum's proof-of-work ASIC-resistant as it allows a more decentralized distribution than those blockchains where mining is dominated by specialized hardware, such as in Bitcoin.

Quick Tip

DDOS is a type of DOS attack where multiple compromised systems, which are often infected with a Trojan, are used to target a single system.

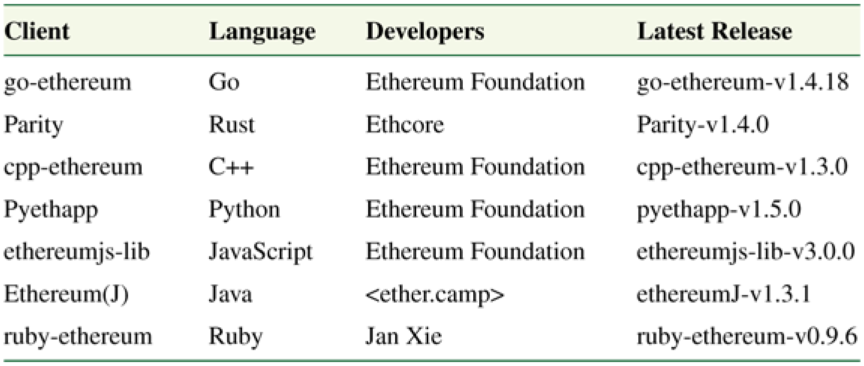
5.6 Ethereum Clients

Ethereum client is a software application that implements Ethereum specification and communicates over the peer-to-peer network with other Ethereum clients. From the initial days of the Ethereum project, client diversity was one of the key focuses. There have been multiple client implementations across a range of different operating systems, which keeps the door open for new innovation. These different Ethereum clients interoperate all the key features and comply with the reference specification and the standardized communications protocols. While these different clients are implemented by different teams and in different programming languages, they all “speak” the same protocol and follow the same rules, thus keeping us all honest while being in operation and interacting in the Ethereum network. However, there is no universal “Ethereum installer” to facilitate end users to directly install it on their environment. As Ethereum is an open source project, the source code for all the major clients is available under open source licenses that are free to download and can be used for any purpose. Ethereum’s formal specifications are documented on a paper that combines English and mathematical description. In addition, several Ethereum Improvement Proposals (EIPs) define the standard behavior of an Ethereum client in the Yellow Paper, which are periodically updated when there are major changes. This results in a number of independently developed yet interoperable software implementations of Ethereum clients. It is considered to be an excellent way of defending against attacks on the network. Meanwhile, other clients keep the network running almost unaffected.

Table 5.2 presents in a nutshell various Ethereum clients while instructions for installing and executing these clients are explained in detail in their respective sites. Therefore, they have not been discussed in this book. However, some familiarity with using the command-line interface on the operating system is necessary. It is worth installing these clients, whether you choose to run them as full nodes, as testnet nodes, or as clients to a local private blockchain. But one should ensure to have a computer with sufficient resources to run an Ethereum full node, which will require at least 80 GB of disk space to store a full copy of the Ethereum blockchain.

In order to run a full node on the Ethereum testnet, an additional of at least 15 GB is required. Downloading 80 GB of blockchain data will take a long time if not connected with fiber-based Internet connection. Syncing the Ethereum blockchain is very input/output (I/O) intensive work. Therefore, the best way is to have a solid-state drive (SSD). While syncing an Ethereum blockchain, the client will download and validate every block and every transaction from the start (i.e., from the genesis block). The sync will take a very long time and has high-resource requirements with respect to memory, and will take long time if not supported by high-speed Internet and fast storage.

Table 5.2 **Various Ethereum clients**



Client	Language	Developers	Latest Release
go-ethereum	Go	Ethereum Foundation	go-ethereum-v1.4.18
Parity	Rust	Ethcore	Parity-v1.4.0
cpp-ethereum	C++	Ethereum Foundation	cpp-ethereum-v1.3.0
Pyethapp	Python	Ethereum Foundation	pyethapp-v1.5.0
ethereumjs-lib	JavaScript	Ethereum Foundation	ethereumjs-lib-v3.0.0
Ethereum(J)	Java	<ether.camp>	ethereumJ-v1.3.1
ruby-ethereum	Ruby	Jan Xie	ruby-ethereum-v0.9.6

Minimum requirements suggested to sync a full copy of an Ethereum-based blockchain at the client side includes CPU with 2+ cores, at least 80 GB free storage space, 4 GB RAM minimum with an SSD, 8 GB+ if an HDD is being used, 8 MB/sec download Internet service. It is difficult to predict how fast the size of a blockchain will increase and when more disk space will be required, so it is always recommended to check the latest size of the blockchain before we start syncing. Fortunately, Ethereum client specifications include an option to perform a “fast” synchronization. Here, the process skips the full validation of transactions until it has synced to the tip of the blockchain, then subsequently resumes full validation.

Many Ethereum-based blockchains were victim to denial-of-service attacks at the end of 2016. Many of the affected blockchains will tend to sync slowly while a full sync is done. For example, a new client will then progress rapidly until it reaches block 2,283,397. This block was mined on 18 September 2016, and marks the beginning of the DoS attacks. From this block to block 2,700,031 (26 November 2016), the validation of transactions becomes extremely slow, memory and I/O intensive, resulting more than a minute for validation per block. As mentioned earlier, Ethereum implemented a series of upgrades using hard forks to address the underlying vulnerabilities that were exploited in the DoS attacks. These upgrades also removed nearly 20 million empty accounts created by the spam transactions, thus further cleaning the blockchain.

Technical Stuff

Spam refers to the use of electronic messaging systems to send out unrequested or unwanted messages in bulk.

5.7 Ethereum Key Pairs

In Chapter 6, we discussed about the symmetric and asymmetric keys. These are used in many places and one of the practical usages is in blockchain. As mentioned in Section 5.5, Ethereum has two different types of accounts – EOA and CA. Ownership of *ether* by EOAs is established through private keys and Ethereum addresses. The private keys are at the heart of all user interaction within the Ethereum platform. It uniquely determines a single Ethereum address known as an account address. The user's private keys are not used directly in the Ethereum system in any way, neither are they transmitted nor stored on the Ethereum platform. They remain private and never appear in messages that are passed to the network, nor are they stored blockchain. Only account addresses and digital signatures are transmitted and stored on the Ethereum system.

Digital signatures are used to access and control the funds. Digital signatures are created using the private key (refer to Chapter 6). Every Ethereum transaction requires a valid digital signature to be included in the blockchain. It is expected that a user keeps their private key safe. With this assumption, the digital signatures in Ethereum transactions prove the true owner of the funds because they prove ownership of the private key. However, anyone with a copy of a private key has control of the corresponding account and any *ether* it holds. For Ethereum users, these are stored (in encrypted form) in special files and managed by the Ethereum wallet software.

For conducting the payment operation during an Ethereum transaction, the intended recipient is represented by an Ethereum address. This Ethereum address is used in the same manner as the beneficiary account details of a bank transfer; and generated from the public key portion of a key pair. However, it may be noted that not all the Ethereum addresses represent public–private key pairs; they can also represent smart contracts.

In order to generate the keys, one needs to find a secure source of randomness as the private keys for Ethereum essentially involves picking a number between 1 and 2^{256}. There is no exact method that one should follow to pick that particular number; however, care must be taken that it is not predictable or deterministic. For producing 256 random bits, the Ethereum platform supports the usage of the underlying operating system's random number generator. Typically, the operating system's random number generator is initialized by a human source of randomness such as time difference between the two keys pressed on the keyboard, CPU temperature, wiggling the mouse around for a few seconds, pressing random keys on the keyboard, etc. A private key can be any non-zero large number slightly less than 2^{256}, which is a huge 78-digit number, roughly 1.158×10^{77} (i.e., decimal number of 77 digits). This number is almost comparable to the 10^{80} atoms contain in the visible universe. With a rough estimate, there will be almost enough private keys assigned to every atom in the universe with an Ethereum account. For the exact number, the first 38 digits are used to define the order of the elliptic curve in the Ethereum network. To create a private key, a random number of 256-bit is picked and checks are performed for the real randomness and valid range. In computer programming, this is achieved by feeding a larger string of random string into a 256-bit hash algorithm such as Keccak-256 or SHA-256. These hash functions will conveniently produce a 256-bit number. If the output is within the valid range, then we can be assured that a suitable private key is obtained, otherwise the same exercise is carried out again with another random number. It should be noted that the private key generation process is offline. It does not require any online communication with the Ethereum network, or with any third party. The only care to be adhered to is to pick a number that is truly random. In a scenario where the number is chosen by an individual, the chance that someone else will try it is too high, which can result in stealing of the *ether*. Using a pseudorandom rand function, available in most programming languages is even worse, as it is more obvious and easier to replicate. One should not worry that they need to remember the random number to operate the Ethereum network. The reality is that this private key is further stored in an encrypted form, so one can take the best possible approach for picking it (i.e., true randomness).

Now, let us talk about Ethereum public key. It is a point on an elliptic curve, meaning that it is a set of x and y coordinates that satisfies the elliptic curve equation. In simple terms, the public key is two numbers that are joined together. These numbers are produced from the private key by a calculation that can go only one way. This means that it is trivial to calculate a public key if the private key is available, but the converse is not feasible, that is, calculating the private key from the public key is not possible given the current state of computing power.

For calculating the public key, the private key is used along with the elliptic curve multiplication, which is practically irreversible: $K = k \times G$, where k is the private key, G is a known as the generator point, and × is the special elliptic curve “multiplication” operator resulting K as the public key.

A point to be noted is that the elliptic curve multiplication is not a normal multiplication. It only shares functional attributes with normal multiplication. For example, the reverse operation that would be the division for normal numbers, which is also known as "finding the discrete logarithm". That is, given K calculating k is as difficult as trying all the possible values of k is almost impossible. For instance, if a brute force search method is deployed, it will take more time than this universe will allow for.

Simply put, the arithmetic on the elliptic curve is different from "regular" integer arithmetic. The generator point (G) can be multiplied by an integer (k) to produce another point (K). However, there is no easy mechanism for division to perform, making things impossible to simply "divide" the public key K by the point G and calculate the private key k.

Sometimes, cryptographers call the Elliptic curve multiplication as a "one-way" function. As we have learned in Chapter 3, it is easy to perform one direction multiplication. However, it is impossible to do in the reverse direction, that is, the division to obtain the private key. The owner of the private key can easily create the shareable public key, knowing that no one can reverse it to calculate the private key. This mathematical approach forms the strong, unforgettable, and secure digital signatures platform that proves ownership of Ethereum funds and control of contracts.

5.7.1 Arithmetic for Generating a Public Key

Let us consider a randomly generated private key in the hexadecimal format (256 bits shown as 64 hexadecimal digits, each of 4 bits) preceded by 0x as convention:

0x15f8a2f43c8376ccb0871305060d7b27b0554d2cc72bccf41b2705608452f
3f8

Starting with this private key, which is in the form of a randomly generated number k, and multiplying it by a predetermined point G (the generator) on the curve to produce another point on the same curve, which is referred to the corresponding public key K:

$$K = k \times G$$

The generator point G is specified as part of the secp256k1 ECDSA standard derived from Certicom Research (http://www.secg.org/sec2-v2.pdf) and it is the same for all implementations of secp256k1. For all the Ethereum users, the generator point is always the same and therefore a private key k multiplied with G will always result in the same public key K. This provides a mechanism to generate an Ethereum address deriving from K that can be shared with anyone without disclosing the user's private key k.

With simple arithmetic logic, we are aware of the fact that multiplication is repetitive addition. On similar lines, we can also say that the multiplication of $k \times G$ is equivalent to repeated addition, that is, $G + G + G + G + \cdots + G + G + G$, k times. Therefore, in order to produce a public key K from a private key k, we add the generator point G to itself, k times. For example,

$$\begin{aligned} K = {} & \mathrm{15f8a2f43c8376ccb0871305060d7b27b0554d2cc72bccf41b27056} \\ & \mathrm{08452f3f8} \times G \end{aligned}$$

Now this seems to be difficult, but to ease the work there are different cryptographic library, which can help us calculate K using elliptic curve multiplication. The resulting public key K is defined as the point $K = (x, y)$ where:

$$\begin{aligned} x = {} & \mathrm{7b145ccef1033dea239875dd00dfb4fee6e3348b84985c92f103444} \\ & \mathrm{683bae06e} \end{aligned}$$

$$\begin{aligned} y = {} & \mathrm{d0b5c38e5e2b0c8529d7fa3f64d46daa1ece2d9ac14cab9477d042} \\ & \mathrm{c84c32cc83} \end{aligned}$$

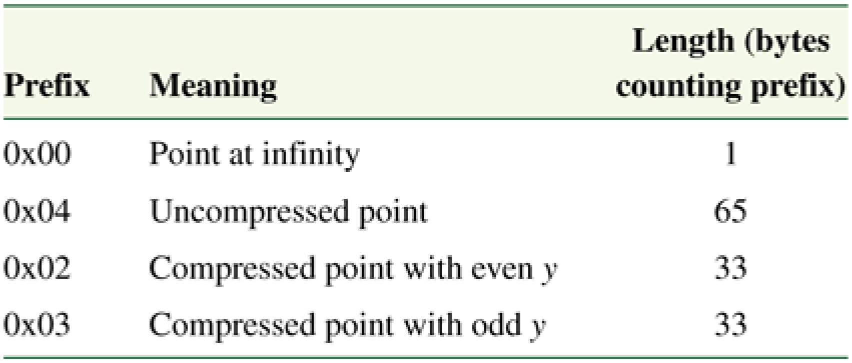
The Ethereum is adopted from a standard serialization format proposed by the industry consortium Standards for Efficient Cryptography Group

(SECG), and the public keys are represented as a serialization of 130 hexadecimal characters (65 bytes). The standard defines four possible prefixes that can be used to identify points on an elliptic curve, which are listed in serialized elliptic curve public key prefixes (Table 5.3).

The Ethereum decided to use only uncompressed public keys; and therefore the only prefix that is relevant is (hex) 04. The serialization concatenates the *x* and *y* coordinates of the public key:

04 + *x*-coordinate (32 bytes/64 hex) + *y*-coordinate (32 bytes/64 hex)

Table 5.3 **Serialized elliptic curve public key prefixes**



Prefix	Meaning	Length (bytes counting prefix)
0x00	Point at infinity	1
0x04	Uncompressed point	65
0x02	Compressed point with even *y*	33
0x03	Compressed point with odd *y*	33

The public key that was calculated earlier can be represented as:

```
047b145ccef1033dea239875dd00dfb4fee6e3348b84985c92f1034446
83bae06ed0b5c38e5e2b0c8529d7fa3f64d46daa1ece2d9ac14cab9477
d042c84c32cc83
```

There are a couple of implementations of the secp256k1 elliptic curve that are used in cryptocurrency-related projects, and we use OpenSSL frequently, as discussed in Chapter 3. This library offers an implementation of comprehensive set of cryptographic primitives, including a full implementation of secp256k1.

5.8 Ethereum Addresses

Addresses on an Ethereum platform are unique identifiers. They are derived from public keys or contracts using the hash function Keccak-256, as explained in Chapter 3. Let us continue from the previous example where we started with a randomly generated private key and used elliptic curve point multiplication to derive a public key. The only thing that one may note is that the public key is not formatted with the prefix (hex) 04 when the address is calculated. The Keccak-256 function is used to calculate the hash of the public key:

$$\begin{aligned}\text{Keccak256}(K) = {} & \text{f95bc342ed616b5ba5732269001d3f1ef827552ae1114} \\ & \text{027bd3ecf1f086ba02a}\end{aligned}$$

Now only the last 20 bytes (least significant bytes) are used for the Ethereum address:

001d3f1ef827552ae1114027bd3ecf1f086ba02a

The Ethereum addresses is constructed by prefixing 0x which indicates that they are hexadecimal-encoded, for example, 0x001d3f1ef827552ae1114027bd3ecf1f086ba02a

In contrast to Bitcoin addresses, which are encoded in the user interface of all the clients to include a built-in checksum to protect against mistyped addresses, Ethereum addresses are presented as a raw hexadecimal without any checksum. The rationale given behind this decision is that Ethereum addresses would eventually be hidden behind abstractions at the higher layers of the system and that checksums can be added at higher layers, if required. However, initially these higher layers were slow and this design choice led to a number of problems in the early days of Ethereum ecosystem. This includes the loss of funds because of the mistyped addresses and input validation errors.

Ethereum address encoding formally gave Inter-exchange Client Address Protocol (ICAP), which is partly compatible with the International Bank Account Number (IBAN) encoding that is mostly used for wire transfers. This offers a versatile, checksum, and interoperable encoding for

Ethereum addresses. These addresses can encode Ethereum addresses or common names registered with an Ethereum name registry.

Fact Alert

An international bank account number (IBAN) is a standard international numbering system developed to identify an overseas bank account. The number 34 alphanumeric characters starts with a two-digit country code, then two numbers, followed by up to thirty alphanumeric characters.

The US and Canada are two countries that do not use the IBAN system. However, they recognize the system and process payments according to the system.

Just as a ready reference, an IBAN consists of a string of up to 34 alphanumeric case-insensitive characters comprising a country code, checksum, and bank account identifier with respect to the specific country. On similar lines, ICAP uses the same structure with slight modification by introducing a non-standard country code, “XE,” that stands for “Ethereum” which is followed by a two-character checksum. There are three variations of an account identifier: Direct, Basic, and Indirect.

5.8.1 Direct

Direct is a big-endian base-36 integer comprised of up to 30 alphanumeric characters, representing 155 least significant bits of an Ethereum address. It only works for Ethereum addresses starting with one or more 0 bytes, as this encoding fits less than the full 160 bits of a general Ethereum address. It has an advantage as it is compatible with IBAN in terms of the field length and checksum. For example: XEJDHAMICDXSV5QXVJA7TJW47Q9CHWK60 (33 characters long).

5.8.2 Basic

Basic is similar to Direct encoding, except that it is 31 characters long. This allows it to encode any Ethereum address, and makes it incompatible with IBAN field validation. For example: XE2PCHDJBPLTBCJ03FE9O2NS0BPOJVQCU18 (35 characters long).

5.8.3 Indirect

In Indirect, the encoding of an identifier is done that resolves to an Ethereum address through a name registry provider. It uses 16 alphanumeric characters, comprising an asset identifier (e.g., ETH), a name service (e.g., XREG), and a nine-character human-readable name (e.g., SCOTTIGER). For example: XE##ETHXREGSCOTTIGER (20 characters long), where ## should be replaced by the two computed checksum characters.

5.9 Ethereum Wallets

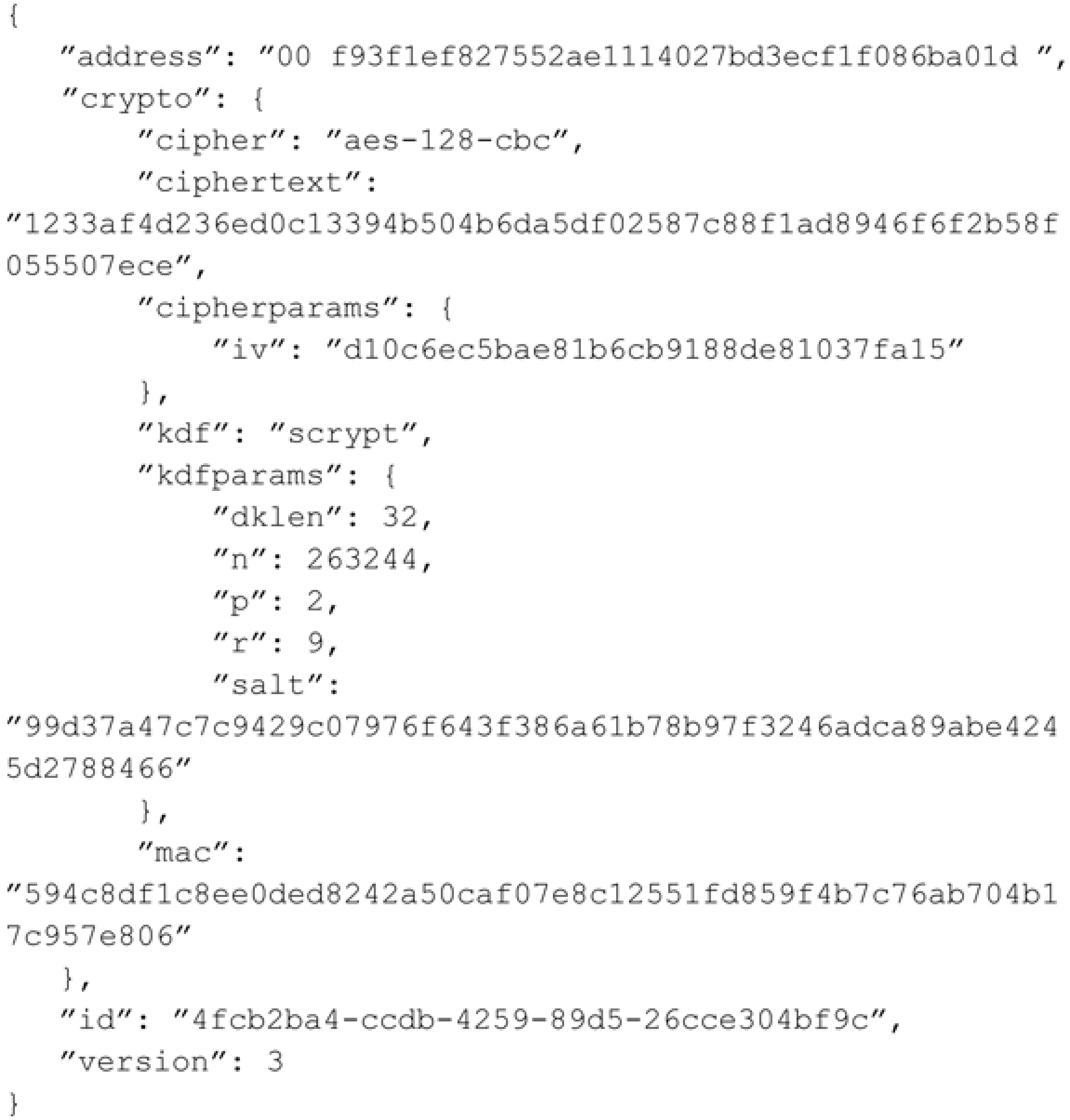
Ethereum wallet is a software application that serves as primary user interface to Ethereum. It controls access to the user's stored money, manage keys/addresses, key management component, tracking the balance, and creating and signing transactions. One key designing consideration for the wallets is to maintain a balance between convenience and privacy. The most convenient Ethereum wallet is one with a single private key and address that can be reused for everything. However, such a solution is terrible for privacy, as anyone can easily track and correlate all the transactions. It would be an ideal scenario for privacy to use a new key for every transaction, but becomes very hard to manage.

One common misconception about Ethereum wallets is that it contains *ether* or tokens. However, the fact is that it holds only keys while the *ether* or other tokens are recorded on the Ethereum blockchain. By signing transactions with the private key present in their wallets, users can control the tokens on the network. Wallets are like keychains containing pairs of private and public keys. There are two primary types of wallets: non-deterministic and deterministic.

1. **Non-deterministic wallet:** In this, each key is independently generated from a different random number and these are not related to each other. This type of wallet derived its name from the phrase "Just a Bunch of Keys", known as a JBOK wallet. It is considered as a good practice to avoid reuse of Ethereum address as part of maximizing the privacy while using Ethereum. However, using a new address for each transaction will be expensive while dealing a lot with the tokens. To follow this practice, a non-deterministic wallet will need to regularly increase its list of keys, which means to make regular backups. The use of non-deterministic wallets is discouraged for anything other than simple tests as they are too cumbersome to backup.

2. **Deterministic wallet:** In this, all the keys are derived from a single master key as the seed. If one has the original seed, then all the keys in this type of wallet can be generated again. A number of different key derivation methods are used in deterministic wallets, and the most commonly used method uses a tree-like structure. However, for protection the seeds are often encoded as a list of spoken words (mnemonic code) that can be used in the event of an emergency and suggested to note on paper and store it in a safe and secure place. As only the seed is needed to gain access to the entire wallet, the design of the security for the seed of utmost importance. On the contrary, being able to focus security efforts on a single piece of data can be seen as an advantage.

Ethereum clients use a keystore file that is a JSON-encoded file containing a single randomly generated private key, encrypted by a passphrase for extra security. A typical JSON file content may look like:



```
{
   ”address”: ”00 f93f1ef827552ae1114027bd3ecf1f086ba01d ”,
   ”crypto”: {
       ”cipher”: ”aes-128-cbc”,
       ”ciphertext”:
”1233af4d236ed0c13394b504b6da5df02587c88f1ad8946f6f2b58f
055507ece”,
       ”cipherparams”: {
           ”iv”: ”d10c6ec5bae81b6cb9188de81037fa15”
       },
       ”kdf”: ”scrypt”,
       ”kdfparams”: {
           ”dklen”: 32,
           ”n”: 263244,
           ”p”: 2,
           ”r”: 9,
           ”salt”:
”99d37a47c7c9429c07976f643f386a61b78b97f3246adca89abe424
5d2788466”
       },
       ”mac”:
”594c8df1c8ee0ded8242a50caf07e8c12551fd859f4b7c76ab704b1
7c957e806”
   },
   ”id”: ”4fcb2ba4-ccdb-4259-89d5-26cce304bf9c”,
   ”version”: 3
}
```

Key derivation function is a password-stretching algorithm which protects against brute force, dictionary, and rainbow table attacks. It is used for the keystore format. A passphrase is stretched by repeatedly hashing, and it is being used to encrypt the private key instead of the passphrase directly. The hashing function is repeated for 263,244 rounds, which can be seen in the keystore JSON as the parameter in the above pseudocode crypto.kdfparams.n. An attacker trying to brute force the passphrase would have to apply these many rounds of hashing for every attempted passphrase that will substantially reduce the attacks.

5.10 Ethereum Transactions

A transaction is qualified as a signed message originated by an account holder, transmitted by the Ethereum network, and recorded on to the Ethereum blockchain. In simple terms, a transaction is an activity that can trigger a change of state, or cause a contract to execute in the EVM. Being the global singleton state machine, Ethereum transactions are what make that state machine initiate the change in its state. Smart contracts cannot run on their own, rather it is a transaction that starts everything.

Let us see the structure of a transaction that is serialized and transmitted on the Ethereum network. After receiving a transaction, all the applications and clients will store it in-memory using its own internal data structure. A transaction contains serialized binary message data under the following different heads:

1. **Nonce:** It is used to prevent message replay. It is a sequence number that is issued by the originating EOA.
2. **gasPrice:** It is the price of gas (in *wei*) which the originator is willing to pay.
3. **gasLimit:** It is the maximum amount of gas the originator is willing to buy for a particular transaction.
4. **To (Recipient):** It is the destination Ethereum address.
5. **value:** It is the amount of *ether* to send to the destination.
6. **data:** It is the payload in binary form of variable-length.

7. **v,r,s:** These are the three components of an Elliptic Curve Digital Signature Algorithm (ECDSA) signature from the originating EOA.

Recursive Length Prefix (RLP) encoding scheme is used to serialize the transaction message that was created specifically for simple, byte perfect data serialization in the Ethereum. The numbers are encoded as big-endian integers. The lengths of these messages are in multiples of 8 bits. RLP does not contain any field delimiters or labels and it is the RLP’s length which is being prefixed and used to identify the length of each field. Anything beyond the defined length goes to the next field in the structure. While this is the actual transaction structure, most of the internal representations and user interface visualizations elaborate this with additional information derived from the transaction or from the blockchain.

Let us take the example where there is no “from” data in the address identifying the originator EOA. One can derive the address from the public key and the EOA’s public key can be derived from v,r,s components of the ECDSA signature. However, it can be observed a few times that in a transaction the “from” field is filled with data. In this scenario, the field data was added by the software used to visualize the transaction. Similarly, other metadata that is frequently added to the transaction by client software includes the block number and transaction ID. We should note that this data is derived from the transaction, and does not form part of the transaction message itself.

Let us consider the example of a transaction to send 5000 *wei* (1 *ether* = 10^{18} *wei*) of *ether* and leave a 0xc0de message that can be constructed as follows:

```
var rawTrnx = {
nonce: web3.toHex(0),
gasPrice: web3.toHex(20000000000),
gasLimit: web3.toHex(100000),
To: ’0x857422eEA2cB73B5d3e242bA5456b782919AFc65’,
value: web3.toHex(5000),
data: '0xc0de'
};
```

One important component in a transaction is transaction nonce, which is a scalar value equal to the number of transactions sent from the address. It is an attribute of the originating address; that is, it only has meaning in the context of sending address. However, the nonce is not stored explicitly as part of an account's state on the blockchain; rather it is calculated dynamically by counting the number of confirmed transactions that have originated from an address. The existence of a transaction-counting nonce is important in two scenarios:

1. Where the usability feature of transactions is included in the order of creation.
2. Where the vital feature is of transaction duplication protection.

Gas is the fuel of Ethereum as a separate virtual currency, with its own exchange rate against *ether*. Ethereum uses gas to control the amount of resources that a transaction can use. As the transaction will be processed on thousands of computers around the world, the open-ended computation model requires some form of metering in order to avoid denial-of-service attacks or inadvertently resource consuming transactions. It is separate from *ether* in order to protect the system from the volatility that might arise along with rapid changes in the value of *ether*. Also, it becomes easy to manage the important and sensitive ratios between the costs of the various resources that gas pays for computation, memory, and storage. The gasPrice field in a transaction allows the transaction originator to set the price they are willing to pay in exchange for gas, which is typically measured in *wei* per gas unit. In order to achieve faster confirmation of transactions, wallets can be used to adjust the gasPrice in transactions they originate. The higher the gasPrice, the faster the transaction is likely to be confirmed. Conversely, low-priority transactions can carry a reduced price, resulting in slower confirmation. For a fee-free transaction, the value that gasPrice can be set to zero, which is considered to be minimum as there is no negative gasPrice.

gasLimit provides the maximum number of units of gas that the originator of transaction is willing to buy in order to complete the transaction. For simple payments (i.e., transactions that transfer *ether* from one EOA to another EOA), the gas amount needed is fixed at 22,000 gas units. To calculate how much *ether* will cost, one needs to multiply 22,000 by the gasPrice they are willing to pay.

The “to” field is used for the recipient of a transaction; it contains a 20-byte Ethereum address that could be an EOA or a contract address. There is no further validation of this field performed by Ethereum, any 20-byte value is considered to be valid. If a 20-byte value corresponds to an address without a corresponding contract, or without a corresponding private key, the transaction is still considered as valid. Ethereum has no method of knowing whether an address was correctly derived from a public key or not, as the validation of the address happens at the user interface level. It will be simple burning of *ether* if the transaction is sent to the wrong address.

The last but main “payload” of a transaction is contained in two fields: value and data. Transactions can have four different valid combinations: only value, only data, both value and data, and neither value nor data.

1. **Only value:** A transaction with only value is a payment.
2. **Only data:** A transaction with only data is an invocation.
3. **Both value and data:** A transaction with both data and value is both an invocation and a payment.
4. **Neither value nor data:** A transaction with neither value nor data will still be valid but a waste of gas.

5.11 Ethereum Languages

The two primary languages that are used to write Ethereum smart contracts are Solidity and Serpent; both allow developing contracts and compiling to EVM byte code. Solidity is a high-level contract-oriented language with similarities to JavaScript and C languages while Serpent is similar to Python. Both of these are popular languages of Ethereum for writing EVM.

Programming with Solidity allows anyone to create a complementary currency with a simple token contract. There are four languages in the Ethereum protocol at the same level of abstraction, but the community has slowly converged on Solidity and Serpent based on their environment. The other two languages are Lisp-like language and Mutan, which were edged out slowly. Learning Solidity is easy and free due open source technology. It enables the movement of tokens of the value in any Ethereum-based system very easily. One can find the official Solidity documentation at http://solidity.readthedocs.io/en/develop/index.html. Another site that can also offer useful Solidity documentation is on the link http://solidity.eth.guide. The most common manner to test Solidity is by using the browser-based compiler that can be found at http://ethereum.github.io/browser-solidity. Using the node package manager, the SOLC compiler can be easily installed for locally compiling the Solidity code. Apart from that, open source text editors such as Sublime Text and Atom have very good support for Solidity.

In this chapter, we will restrict a detailed discussion on Ethereum languages as it is a very vast topic. For ready reference to the readers, the above-mentioned links will give them a wider, in-depth, and latest contents and development on the languages. However, this chapter will not be complete if we do not discuss the popular tools used by the communities for the development and usage of Ethereum platform. Therefore, the next section will briefly discuss some salient tools.

5.12 Ethereum Development Tools

Implementing smart contract support on Ethereum platform opens the blockchain for several possibilities. This enables developers to create any such application that is possible to run on the blockchain by developing smart contracts in Ethereum-specific languages such as Solidity, Serpent, etc. Apart from these languages, there are several tools that have been developed in recent years to make development less cumbersome for the community to use. This section covers a few prominent Ethereum development tools, which are classified into five categories: integrated development environment (IDE), test node, command line tool, code analyser, and browsers.

5.12.1 Integrated Development Environment

The productivity of the developer increases if there is an environment that provides various functionalities such as editing, compiling, and debugging at our fingertips, which is generally carried in an integrated development environment (IDE). A good IDE will boost productivity by reducing setup time, increasing the speed of development tasks, keeping developers up to date, and standardizing the development process. There are many IDEs present for various Ethereum development activities and features. In this chapter, we will restrict to only two: Remix and EthFiddle. It should be noted, however, that others have their own merits.

5.12.1.1 Remix

Remix, earlier known as Browser-Solidity, is a web-based IDE specifically aimed at Solidity. It compiles the code with up-to-date compiler versions. For test deployment and execution of smart contract, it provides a customized environment such as a JavaScript Virtual Machine. It also allows importing code from GitHub and Swarm. More on this is available at https://buildmedia.readthedocs.org/media/pdf/remix/stable/remix.pdf.

5.12.1.2 EthFiddle

Another IDE is EthFiddle, which is also browser-based Solidity IDE. It provides to compile test code and compiles the smart contract with error handling. It also provides the developer to share the fiddles with co-developers without much of effort. While Remix provides the flexibility of testing code on different networks and environments, EthFiddle shares the code on a presentation because of its easy embed features. Although it is not as feature rich as Remix and it is also slightly slow, it has a plus point on two features: embed and share. More on this is available at: https://ethfiddle.com.

5.12.2 Test Nodes

We know that everything on the blockchain is technically immutable. Even updates to smart contracts cannot register the same address and must be deployed at a new address as a new instance. This implies that smart contracts cannot be tested on the main blockchain network, as any change would be impossible to make once deployed on the main network. Therefore, one needs to have an environment where a prior testing can be conducted. Thus, for Ethereum developers, Ethereum development tools provide test network nodes to be used as local test nodes to test the interaction of the contracts. Some of the most popular local testing network nodes are Ganache and Pythereum.

5.12.2.1 Ganache

Ganache is one of the most widely used local test node by Ethereum developers. It is a personal blockchain for Ethereum developments, where contracts can be used to deploy additional development of the applications and execution of test cases can be done. It is available for Windows, Mac, and Linux, as these are desktop applications as well as command line tools. The silent features include Quick View status of all accounts, including their addresses, private keys, transactions, and balances. The logs of Ganache internal blockchain contain responses and other vital debugging information. It is easy to configure and customize the individual needs. In order to gain insight about what is happening under the hood, it facilitates to examine all blocks and transactions. More on this is available at https://github.com/trufflesuite/ganache.

5.12.2.2 Pythereum

Pythereum is a local test node tool written in Python. It is much more lightweight than Ganache; however, is not too rich in feature. It provides features to create a new test blockchain with a genesis block and to also create new test-state along with the passed genesis state. It can send a transaction using the given private key to the given address with the given value and data. More on this is available at https://github.com/ethereum/pyethereum.

5.12.3 Command Line Tool

Command line tool, also known as command line interface (CLI), is a type of human–computer interface that solely relies on textual request and response transaction process. It is quite common for users and developers of blockchain to redirect the output of one command as an input to another. The CLI can be used easily in the form of batch files or scripts for an automated testing or builds. The input, output, and error streams can be easily redirected using this feature in the three major command line-based Ethereum development tools: Truffle, Embark, and Dapple.

5.12.3.1 Truffle

Truffle happens to be the most popular of the three command line-based Ethereum development tools. With multiple utility at one place, it provides a development environment, testing framework, and asset pipeline for Ethereum. It aims to ease the life of an Ethereum developer. TruffleSuit comes with built-in smart contract compilation, linking, deployment, and binary management along with the automated contract testing with mocha and chai. It provides configurable build pipeline with support for custom build processes, scriptable deployment, and migrations framework, and with external script runner that executes scripts within a

Truffle environment and many such features. More on this is available at https://github.com/trufflesuite/truffle.

5.12.3.2 Embark

Using serverless HTML5 applications, Embark allows an easy development and deployment of decentralized applications (DApps). Currently, Embark integrates with EVM, decentralized storages (IPFS), and decentralized communication platforms (Whisper and Orbit). Swarm is also supported for deployment in it. With Embark, one can perform an automatic deployment of contracts. It also watches for changes, and if any update happened in a contract, it will automatically redeploy the contracts if required. More on this is available at https://github.com/embark-framework/embark.

5.12.3.3 Dapple

Dapple is a Solidity developer multitool, which is designed to manage the growing complexity of interconnected smart contract systems. It is deprecated in favor of a new tool known as DApp, which has been created by the same group of developers. DApp is a simple command line tool for smart contract development, with its core functionality that encompasses three main areas: package management, contract building, and deployment scripting. Because of the blockchain's universal singleton nature, these three concepts provide a flexible development feature, which is the smart contract ecosystem. The central data model for Dapple is the dappfile, which is normally reference in IPFS objects and Ethereum contract addresses. More on this is available at https://github.com/nexusdev/dapple.

5.12.4 Code Analysis Tools

The software required in blockchain is quite complex, and writing clean and secure code for a decentralized network is not an easy task. The most insidious and dangerous defects come from the interactions between the components of a complex system. These cannot be detected by traditional testing methods due to multitiered, multiple technology infrastructure. Line-by-line code review practices of reading to verify quality, performance, size, and productivity is neither cost effective nor reliable. Furthermore, there is a lot to worry about the storage and security, especially when a majority of the code handles money and any faulty rollbacks in the state could lead to major losses and confidence. In order to help developers write clean and secure code, few special code analyzers have been developed to avoid faculty and unsecure situations. Two such tools are Solium and Open-Zeppelin. Solium is a solidity code linter that allows writing robust and stylish smart contracts. It works like an interpreter, where it constantly checks the code for style and security issues and if required, fixes them too. More on this is available at https://github.com/duaraghav8/Ethlint.

OpenZeppelin is a Solidity framework for writing secure smart contracts. By using it, the developers can build distributed applications, protocols, and organizations using common contract security patterns in the Solidity language. The key feature of OpenZeppelin is that it seamlessly integrates with Truffle. More on this is available at https://github.com/OpenZeppelin/openzeppelin-contracts.

5.12.5 Browsers

Browsers for the Internet have made life wonderful for users. It is typically used to locate, retrieve, and display content on the World Wide Web, including web pages, images, video, and other files. Formally, in client/server architecture, the browser is the client that runs on a computer or mobile device, which contacts the Web server and submits the request/query. The web server in turn sends the information/response back to the browser, which displays the results to the user. On similar lines, Ethereum blockchain needs a browser which specially caters to its needs so that information regarding state, receipts, and transactions can be queried and/or viewed. Here, we present the two most popular browsers that are used by developers to analyze the interaction of their app on blockchain: Mist and MetaMask.

5.12.5.1 Mist

Mist is the end user interface for Ethereum, which is formerly known as Ethereum DApp Browser. This is used to store, send, and receive cryptocurrency funds, though it is different from the Ethereum wallet. It is a tool for browsing and using DApps and is specifically designed for non-technical users. It provides several functions, which are limited to send and receive transactions, store ether, create multi-signature wallets, deploy smart contracts, and view the state of the blockchain. More on this is available at https://github.com/ethereum/mist.

5.12.5.2 MetaMask

MetaMask is actually not a real "browser", rather it turns the Google Chrome browser into an Ethereum browser to fetch data from the blockchain and allows users to securely send or receive signed transactions. Installation of extension to the regular browser exports the Ethereum WEB3 API into every website's JavaScript, which enables DApps to read directly from the blockchain. MetaMask is easily installable on Chrome, Opera, and Firefox as a browser extension. More on this is available at https://github.com/MetaMask.

SUMMARY

In this chapter, we have discussed about various blockchain components which are put to practical purpose for people to use. We discussed about one of the prominent platforms Ethereum, which facilitates building decentralized apps with a set of predefined operations and provides flexibility to users to create their own operations of any complexity. These can be used for any activity that has an economic or governance aspect without restricting to any specific domain. Ethereum has a wide range of features, which includes Integrated Development Environment (IDE) for debugging, development, and deployment of applications.

Ethereum's basic unit is the account contrast to the Bitcoin blockchain which is purely a list of transactions. Ethereum blockchain tracks the state of every account, and all state transitions on the Ethereum blockchain are transfers of value and information between accounts. There have been multiple Ethereum client implementations across a range of different operating systems and keeps the door open for new innovation. These different Ethereum clients interoperate all key features and comply with the reference specification and the standardized communications protocols. We have discussed Ethereum Virtual Machine (EVM) and its functioning along with the arithmetic for generating a public key, Ethereum addresses, Ethereum wallets, and Ethereum transactions.

SHORT ANSWER QUESTIONS

1. What consensus algorithm does Ethereum use?
2. What is the purpose of EVM in Ethereum?
3. How is ether brought into the Ethereum system?
4. What is the value token for Ethereum?
5. What is the difference between Wei and Ether?
6. What is the average block time in Ethereum?
7. What is the average block size in Ethereum?
8. Does Ethereum support scripting? If so, what types of scripting does it support?
9. What extra properties does the "address" type have?
10. What happens when a transaction runs out of gas?

LONG ANSWER QUESTIONS

1. What is the most commonly used Ethereum development framework? What features does it have?
2. Explain the structure and functioning of Ethereum wallets.

CHAPTER 6

Cryptography

Learning Objectives

After reading this chapter, you will be able to:

- Define the term cryptography.
- Analyze the primitives of cryptography.
- Understand in detail about symmetric and asymmetric cryptographies.

6.1 Introduction

There are many distinguish authors who have defined cryptography in their manner and are well-recognized in the community. However, if we look into the past, it is the Renaissance period, covering the span between the 14th and 17th centuries, when the art of secret writing came into existence. Then, it was considered as a new science of codes and ciphers with contribution coming from the period's most brilliant inventions and most enduring legacies. Leon Battista Alberti is considered to be the author of one of the oldest cryptographic books. His short text on ciphers, *Opuscoli Morali*, was first published in 1568 collection. It is considered to be Europe's oldest extant treatise on ciphers, which earned him the title of "father of Western cryptology". However, Trithemius' work titled *Polygraphiae Libri Sex*, though posthumous, is considered as the first printed cryptographic book that was on polygraphic systems. Since then, cryptography has been evolved through numerous phases of evolution. Cryptography in the past was designed to perform encryption and decryption, which may take place by hand. On the contrary, the modern ciphers developed and used today are only possible due to the high-computational performance of modern machines. Not to forget the mechanical ciphers that were developed around World War II, and relied on sophisticated-gearing mechanisms to encipher text, namely, "Enigma Cipher" and "Lorenz Cipher". In parallel, the theoretical and practical aspects of cryptanalysis, or code breaking, were also well nurtured. It was Poland, around 1932, which accomplished the task to completely break into Enigma, passed the techniques and insights used to the French and British Allies just before the outbreak of the war in 1939. Subsequent improvement by the British at the Bletchley Park research station enables the decryption of Enigma cipher, which allowed the Allies to read important messages of German radio traffic and was an invaluable source of military intelligence throughout the war. Before the US entered the war, the US Army Signals Intelligence Service also did some similar activity by breaking into the most secure Japanese diplomatic cipher. The systematic approach in cryptography with proofs and definitions was visibly demonstrated in the work of Claude Shannon. Shannon, who is considered to be the father of information theory, is also said to be the father of the modern-era of cryptography. Modern cryptography algorithms typically use block ciphers, stream ciphers, and public key cryptosystems, etc., and are considered to be secure. As time passes, more and more people hang-up over the Internet for various purposes ranging from simple e-mail messages to sensitive corporate data exchange; security becomes an important component in the system. Cryptography is used to protect e-mail messages, credit card information, and corporate data. In order to have a complete understanding on blockchain, we will broadly discuss cryptography. We are hopeful that this should be sufficient to understand the underling cryptography principles in blockchain.

Fact Alert

Kerckhoffs's principle, also called Kerckhoffs's law, states that a cryptosystem should be secure even if everything about the system, except the key, is public knowledge. In the words of Claude Shannon, "The enemy knows the system".

6.2 Cryptography

In simple terms, cryptography is a mechanism of protecting information by transforming it (encrypting it) into an incomprehensible format known as *cipher text*. Only those who possess the mechanism (including the secret key) can decipher (or decrypt) the message into plain text. The reverse cryptography is known as *cryptanalysis*, which is used to break the encrypted messages. Modern public domain cryptographic techniques are virtually unbreakable and are all about constructing and analyzing protocols that overcome the influence of adversaries "A". For the sake of simplicity, let us assume that two different entities (sender "S" and receiver "R") are provided with a dedicated, untappable, impenetrable duct or pipe through which the sender can convey a message and the receiver will receive it. No one else can look inside the pipe or change what is there in it. This pipe provides the perfect medium, available only to the sender and receiver, as they are only two entities in the world. It can be considered as an "ideal" communication channel from a security point of view. However, in reality, there does not exist an ideal channel(s) connecting the (two or more) pairs of parties that might like to communicate with each other. In general, these parties are communicating over public network such as the Internet. The most fundamental goal of cryptography is to provide these parties with a means by implanting a mechanism between their communications channel such that their conversation does not fall on other unintended entities. Let us assume that this adversary has access to the network and is willing to compromise the security of the parties' communications in some manner. In practice, the cryptographers try to achieve security by adopting various combinations of protocols, while keeping in mind that these will be used in the public domain and they will be available to public scrutiny. A protocol is a collection of algorithms and programs (software) for each party (sender and receiver) involved. The sender's program performs the job of packaging the data for transmission, whereas the receiver's program completes the task of unpacking in order to recover the data. While performing the activities of packing/unpacking, the protocol may also facilitate verifying the credentials of the sender and integrity of the received data. This is to ensure that the data really originated from the sender, and was not sent by the adversary, or modified enroute from the sender to the receiver. During these processes, the protocol (or the set of programs) will use some cryptographic keys.

We will discuss in detail about cryptography primitives in the following section.

6.3 | Cryptography Primitives

In order to achieve security, there are four primary cryptography primitives that need to achieve the following:

1. **Confidentiality or privacy:** This is to ensure that no one can read the message except the intended receiver(s).

2. **Authentication:** This is the process to prove one's identity.

3. **Nonrepudiation:** This is to prove that the sender has really sent the message.

4. **Integrity:** This is to ensure that the received message has not been altered in any way from the original message.

Figure 6.1 shows the four cryptography primitives.

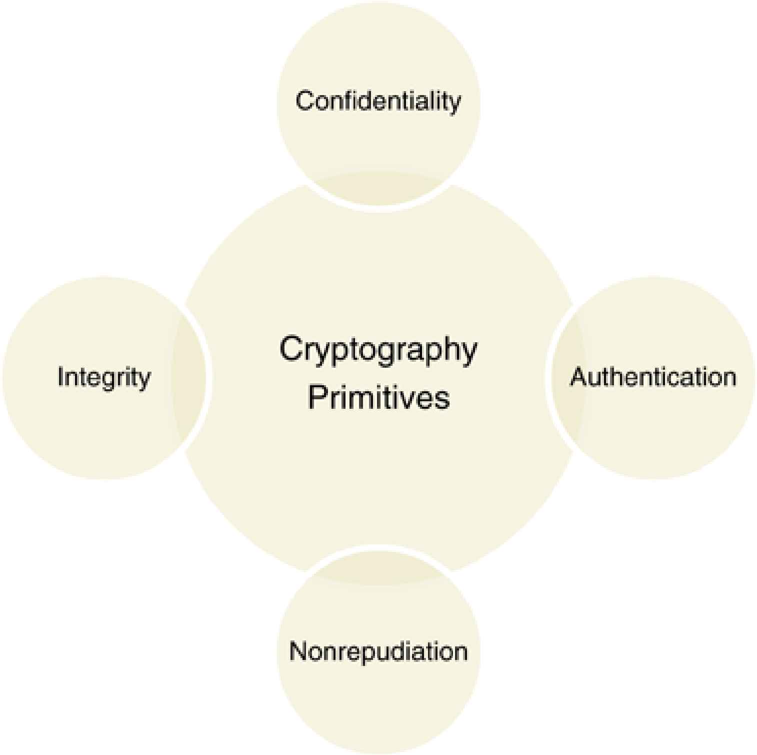


Figure 6.1 Cryptography primitives.

6.3.1 Confidentiality

Confidentiality is the protection of personal information and restricts access to the content of sensitive data to only those individuals who are authorized to view the data. Confidentiality provisions will prevent the disclosure of information to unauthorized individuals or processes. If it falls in the wrong hands, confidential information can be misused to commit illegal activities that can in turn result in costly lawsuits. Many countries have laws protecting the confidentiality of certain data or information, as compromising on privacy will result in a loss of productivity. Confidentiality is important when network communications are of sensitive nature, such as trade secrets, client information subject to privacy laws or policies, or business strategies. However, it is also significant for important data at rest. Technically, confidentiality of data is accomplished by using strong encryption algorithms which cannot be easily broken. Encryption algorithms will ensure that only the correct entity can read the information. In today's environment, encryption is widely used and can be found in almost every major protocol while using computing devices. However, other ways to ensure information confidentiality is by enforcing file permissions and access control list to restrict access to sensitive information. But, by and large it is achieved through either symmetric or asymmetric encryption. We will discuss this in Sections 6.4 and 6.5.

Quick Tip

In legal terms, confidentiality refers to a duty of an individual to refrain from sharing confidential information with others, except with the express consent of the other party.

6.3.2 Authentication

Authentication establishes the validity of the originator who had transmitted the message. The receiver of the data should be able to determine its origin. Traditionally, a personal identification number (PIN) or secret password is extended to the user for authentication; this PIN or password is used for accessing the particular system. Password-based authentication can be effective provided these are managed properly. It is observed that an authentication mechanism that typically relies only on passwords often fails as they do not provide adequate protection for computer systems for a number of reasons. In the scenario where users are given the liberty to choose their own passwords, the tendency is to choose ones that are easy to remember and unfortunately, easy to guess by an attacker. In contrast, passwords are difficult to guess if they are generated randomly by combining several characters, numbers, and special characters. Against the advice, users often write down passwords as they are difficult to remember. For applications where password-only authentication is not adequate, then a password is often used in combination with other authentication mechanisms. With advancement in technology, a system can offer one or more factors for authentication. In a single-factor authentication only one factor is used whereas when two factors are used, it is termed as a two-factor authentication. Now, one may think that there can be more than two factors. This is true; there can be three factors and these are termed as three-factor authentication. Usually, multi-factor authentication refers to the presence of more than one factor. Now, let us discuss a few of these authentications:

1. **Two-factor authentication (2FA):** When there are two factors used for authentication it is termed as two-factor authentication. For example, the first factor is a PIN or password and the second factor is something that we possess. Possession factor can include anything that users have in their possession such as OTP-generator application or smart card/phone.

2. **Three-factor authentication (3FA):** In addition to what we know and what we have, the third factor can be what we are such as iris, fingerprint, voice, or any other biometric feature.

3. **Four-factor authentication (4FA):** When there is use of four types of authentication, the aforementioned three and user location (i.e., the fourth factor), then it is termed as four-factor authentication. It is the recent paradigm that tends to utilize smartphones, as most of these have GPS features which enables reasonable guarantee on the confirmation of login location. Lower security measures might be the MAC address of the computer system at the login point. In scenarios where businesses and government agencies require extremely high security, the four-factor authentication is opted. By using such high-

level multi-factors, it becomes unlikely that an attacker can fake or pretend to be the real entity.

4. **Five-factor authentication (5FA):** Sometimes, time is considered to be the fifth factor for authentication. Take a simple example where we need to verify the presence of an employee against work schedules that can prevent some kind of user hijacking attacks. Another example is that of a person sitting in his house in the US, who cannot physically use his ATM card at home and within a time span of 30 min use it in Russia as well. Time can be considered as a distinct confirming factor that could make the five-factor authentication a possibility.

Quick Tip

Authentication means confirming your own identity, whereas authorization means being allowed access to the system.

The implementation of different factors involved will define the reliability of authentication in a system. Operation rules and options selected for authentication greatly affect the security of the entire system. It will make the system weaker if the rules and implementation procedures are lax. At the same time, care must be taken not to overburden users with difficult authentication routines, else they will tend to explore shorter and easier paths by subverting rules for easier logins, which will result in lower security.

6.3.3 Nonrepudiation

Repudiation is a rejection or denial of something as true. Nonrepudiation translates into an assurance that someone cannot deny the validity of something committed in the past. It is a legal concept and widely used today in information security as a service that provides proof of the origin of data.

Technical Stuff

An electronic signature, or e-signature, refers to data in electronic form, which is logically associated with other data in electronic form and which is used by the signatory to sign.

In simple terms, nonrepudiation makes it very difficult for the entity that has sent the message to deny. Let us consider the following example. In real-paper world, someone buys a smartphone for $500 and writes/signs a paper cheque for paying the bill. Later on, the person realizes that he/she cannot afford the product and claims that the cheque is a forgery. In the pen–paper world, the signature guarantees that only he/she has signed the cheque, and so his/her bank must honour the cheque. This is nonrepudiation as he/she cannot repudiate the cheque. Similarly, in the digital-computing world, when it comes to online transactions, it is crucial to ensure that a party to a contract or a communication cannot deny a commitment done through any transaction. Cryptographic digital signatures, combined with other technical and operational measures, can offer nonrepudiation services. In reality, pen-and-paper signatures sometimes are not too difficult to forge, but digital signatures can be (and are) very hard to break or compromise. However, nonrepudiation would be violated if associated cryptographic key were shared or lost or stolen and not immediately reported to the concerned authorities. The cryptographic digital signatures are used to introduce uniqueness and nondeniability to electronic transactions. For creating a digital signature, one needs to have a digital certificate; this digital certificate is digitally signed by a trusted third party, which is referred to as certificate authority. Thus, to achieve nonrepudiation, it requires creating some artefacts that may be used to resolve the dispute over claims of an entity or organization that denies being the originator of an action or communication.

Quick Tip

There are over 50 certificate authorities in the global SSL certification market.

6.3.4 Integrity

Information will have value if it is correct. In simple terms, integrity of information refers to protecting the information from being modified intentionally or unintentionally, that is, to check the unauthorized or accidental modification of data including data insertion, deletion, and modification. Any such activity can prove to be costly. For example, a person wants to send an online money transfer of $500. Unfortunately, the figure of 500 was changed by adding of one extra 0 to it. Now, the figure becomes $5000; this proves to be a very costly affair! In order to have data integrity in place, the system must be able to detect any data modification, that is, the sender and receiver of the data should be able to verify and prove that it has not been altered during transit.

The simplest mechanism to ensure integrity of data transmitted between devices (computers and terminals), is that when devices communicate over a noisy channel, systems will transmit an extra bit for each byte of data, which is known as *parity bit*. The value of extra bit is chosen to ensure that the number of 1s in the transmitted bits were odd for odd parity or even for even parity. If the parity is wrong, one can conclude that data had been altered and should be rejected. This mechanism is frequently used with modem connections. However, this is a relatively expensive form of integrity protection. The mechanism increases the size of the message by more than 12%, which is significant when the data size ranges in GBs. Furthermore, they may not detect multiple errors in the same byte.

Cyclic redundancy checks (CRCs) perform the same function for larger streams of data with less overhead. The sender, using a mathematical function, calculates CRC on the transmitted data and appends it in the transmitted data. The receiver calculates CRC from the data stream and matches it against CRC provided by the sender. If the two match, then the data has not changed accidentally or otherwise. Ethernet in network protocols is a good example of this technique. Parity bits and CRCs do protect against accidental modification of data. However, they fail to provide protection against an attacker. The attacker may modify data and try to derive new CRC for the modified data. This is possible if the digest is computed using simple mechanisms such as CRC. Similar to nonrepudiation, digital signature also plays a major role in ensuring data integrity.

In the following section, we will learn about cryptography protocol in detail. Cryptography is broadly classified into two categories: (1) symmetric-key systems, which use a single key that is common to both the sender and the recipient and (2) symmetric or public-key systems, which use two keys – a public key known to everyone and a private key that only the owner is aware of and is secret from rest of the world.

6.4 Symmetric Cryptography

In simple terms, using a single key for both encryption and decryption is known as *symmetric cryptography*. It is primarily used for achieving confidentiality. Substitution cipher is one of the earliest approaches to symmetric encryption. In order to understand this, let us consider the plain English text “HELLO WORLD”. We can consider this as a sequence of characters (denoted as “s”). However, the symbol can either be a character, a number, a blank, or a punctuation mark. While performing, encryption substitutes each symbol of “s” with another symbol *f*(s). The function *f* is the transform function, and it shifts each character in that sequence by a certain amount, say, 5. Thus, the encryption of this sequence will be as follows:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Shifting 5 units																									
V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U

This will transform “HELLO WORLD” to “CZGGJ RJMGY” and 5 will be considered as the key value.

Now, let us represent this in the mathematical form:

$$\text{Encryption} = (\text{Current position} + \text{Key}) \bmod 26$$

Here, mod is the modulo operation, which finds the remainder after division of one number; in this case, it is 26. For decryption, one can perform the inverse of the operation and will get the original text. However, this encryption algorithm is prone to simple brute force attack as there could be maximum of 26 possible substitution table. This algorithm is known as *Caesar cipher*, and history conveys that this cipher was used by Julius Caesar to communicate in secret.

Flash Quiz

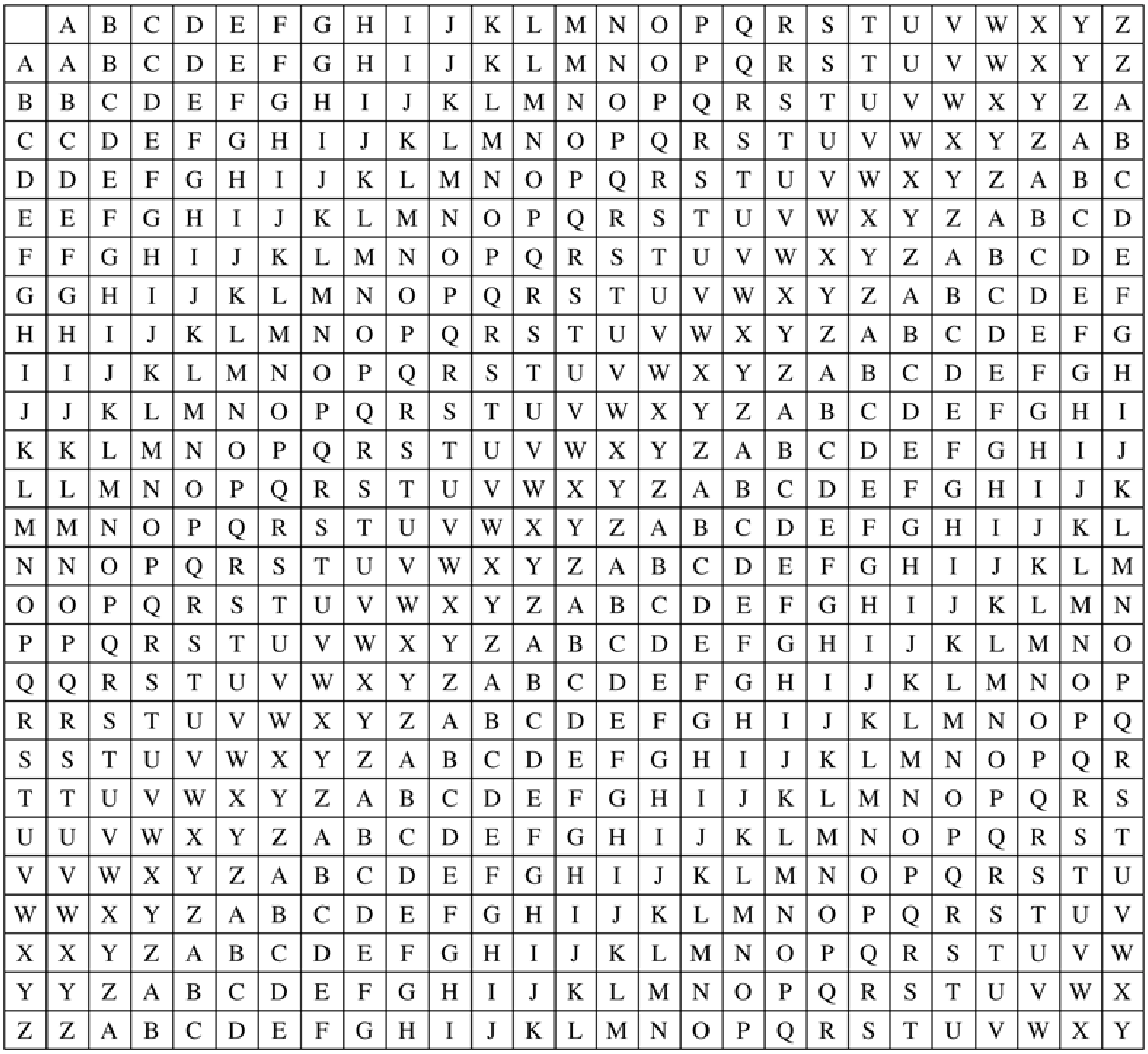
Can you write/search and execute the code for Vigenère cipher.

French cryptologist, Lewis Caroll, invented the Vigenère cipher in the mid-1500s. He proposed to use an entire word as the shift key, as opposed to the Caesar cipher’s single shift. To understand this better, let us consider that we want to encrypt the word “HYDERABAD” and use a shift key of SPARROW. In the first place, we need to repeat the shift key to line up with each of the letters in the sentence and then use Table 6.1.

For the first letter “H”, we select the row that starts with “H”. Since the corresponding shift key letter is “S”, we move to the column that has a header of “S”. The letter at the intersection of the “H” row and “S” column is “Z”. Thus, we encrypt “H” as “Z”. The letter at the intersection of “Y” row and “P” column is “N”, so we encrypt “Y” as “N”. We keep doing this until we complete the original word to get the encrypted word shown in Table 6.2.

Similar to encryption, we can also perform the decryption given the shift key word by using reverse substitution from Table 6.2. Now, we need to start by selecting the row for the first letter in the shift key, scan down that row until the first encrypted letter is found, and look up to see the header for that column which would be the decryption of the given letter. Let us consider the encrypted word “ZNAVIOXSS” and shift key “SPARROW”. We scan down that column (here “S”) until we find the first encrypted letter “Z”. Once we find “Z”, we look to see the header for that row, “H”. Thus, the decryption of “Z” is “H”.

Table 6.1 **Vigenère table**



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Table 6.2 **Computation of the encrypted word**

Original word	H	Y	D	E	R	A	B	A	D
Shift key	S	P	A	R	R	O	W	S	P
Encrypted word	Z	N	A	V	I	O	X	S	S

Table 6.3 **English letter frequency (based on a sample of 40,000 words)**

Letter	Count	Frequency
E	21912	12.02
T	16587	9.10
A	14810	8.12
O	14003	7.68
I	13318	7.31
N	12666	6.95
S	11450	6.28
R	10977	6.02
H	10795	5.92
D	7874	4.32
L	7253	3.98
U	5246	2.88
C	4943	2.71
M	4761	2.61
F	4200	2.30
Y	3853	2.11
W	3819	2.09
G	3693	2.03
P	3316	1.82
B	2715	1.49
V	2019	1.11
K	1257	0.69
X	315	0.17
Q	205	0.11
J	188	0.10
Z	128	0.07

Source: http://pi.math.cornell.edu/~mec/2003-2004/cryptography/subs/frequencies.html

We should note that this cipher maps the same letter to different characters, and therefore the Vigenère cipher is said to be a polyalphabetic cipher. Thus, the Vigenère cipher is relatively more difficult to crack than the Caesar cipher because of the use of an entire shift word. In case of any interception, the attacker has no idea what the shift key word was and he/she is left with brute force method to decrypt. This would require trying out all the possible shift words in the world, and sometimes made up words also, and this could take a lifetime to decipher. However, another approach to attack this cipher is known as *frequency attack*. For example, while analyzing a page in an English book, it was noted that the frequency of character "e" is the highest, as shown in Table 6.3.

Flash Quiz

Try to replace the shift key "SPARROW" with your "NAME" and compute the encrypted word.

People in the 1800s realized the use of frequency analysis to crack the cipher; for example, in a long message, a short word such as "THE" might get translated to the same three-encrypted letters multiple times, which reveals possible lengths for the shift key. In today's context where we have powerful computers, the Vigenère cipher is relatively easy to break. Thus in today's requirement, the design of a cipher should be such that it would be hard for an attacker to crack by a computer that can do trillions of calculations per second. We are going to learn about these ciphers in the following subsections.

6.4.1 Block Ciphers

A block cipher take in N bits of plain text and m bits of key to produce n bits of cipher text. These are known as block ciphers because these operate on blocks of n bits at a time. Messages that are longer than one block size need to be split into smaller message blocks. The basic principle is to create more confusion in the encrypted text from the plain text. This is being carried out by performing substitution with a Substitution-box (S-Box), which is a basic component of symmetric key algorithms. Actually, every system of symmetric encryption must have two characteristics: confusion and diffusion. These two characteristics were introduced by Claude Shannon (Shannon 1949).

Technical Stuff

In information theory, the Shannon–Hartley theorem tells the maximum rate at which information can be transmitted over a communications channel of a specified bandwidth in the presence of noise.

The property of confusion in block ciphers are typically used to obscure the relationship between the key and the cipher text. The design of S-boxes is carefully chosen such that they are well-qualified to resist cryptanalysis. In general, they take some number of input bits, m, and transforms them into some number of output bits, n: an $m \times n$. They can be implemented as a lookup (fixed) table with 2^m words of n bits each [e.g., the Data Encryption Standard (DES) algorithm]. However, in some ciphers the tables are generated dynamically from the key (e.g., the Blowfish and the Twofish encryption algorithms).

For a typical DES algorithm with a 6-bit input, the 4-bit output is found by selecting the row using the outer two bits (the first and last bits), and the column using the inner 4 bits. For example, for an input of bit string

“011011” which has outer bits “01”, ’0’ being the left side and ’1’ being the right side and inner bits are “1101” will have the output bits as “1001”. In DES, there are eight S-boxes that were the subject of intense study for many years, after its launch, to see if there is any backdoor vulnerability planted in the cipher by its designers and known only to them. After the public rediscovery of differential cryptanalysis (Coppersmith 1994), the S-Box design criteria were eventually published showing that they had been carefully tuned to increase the resistance against this specific attack. Subsequently, there has been a great deal of research into the design of good S-boxes.

Another important element in the block ciphers is permutation box (P-box). These are classified as compression and expansion depending on whether the number of output bits is less than, greater than, or equal to the number of input bits. It is a method of bit-shuffling used to permute or transpose bits across S-boxes inputs, retaining diffusion while transposing. In most of the cases it is a permutation of all the bits: P-box takes the outputs of all the S-boxes of one round, permutes the bits as per the set rule, and then feeds them into the S-boxes of the next round. P-box is said to be good if it possesses the property that the output bits of any S-box are distributed to as many S-box inputs as possible.

To conclude this section, we will touch upon the strength of a symmetric key cryptographic algorithm that is typically gauged by the size of the key it employs. The DES algorithm uses a 64-bit key, of which 8 bits are reserved leaving 56 bits to be used for encryption and decryption. Similarly, the Blowfish algorithm allows selecting a key length of 32 bits to 448 bits. Since then, Advanced Encryption Standard (AES) has become very popular as it became the encryption standard placed by the US government in 2002, replacing the DES. AES combines the speed of DES with the security level of Triple DES (3DES). Furthermore, it can use three different sizes of keys: 128-, 192-, and 256-bit.

6.4.2 Examples of Block Ciphers

In this section, we will be discussing few block encryption algorithms: DES, AES, Blowfish, and Twofish.

6.4.2.1 Data Encryption Standard

DES was invented in 1970. This symmetric key cipher was one of the predominant standards till 2000, but has fallen since then because of its low security and availability of high-end computation facilities. Its introduction initiated debates across the world about the role of standards in cryptography. However, the design of DES has given rise to many block ciphers and many ciphers present today are based on its design. The DES block cipher consists of 56 random bits, and another 8 bits for error detection that make DES unmalleable. In a scenario where there is a change in the cipher text on its way to its destination due to accidentally deleting a bit used for error detection, the receivers would then know that data had been modified. The algorithm cuts the data of 64-bit block into two 32-bit halves. These two halves are fed into the entire system in a crisscross manner, which is termed as *Feistel system*. It contains 16 layers and at each layer, one-half of the data goes through the Fiestel function. On the finishing layer, it is an exclusive-or operation (XOR) with the other half of the data. One may note that each layer has its own subkey that is derived by key scheduler from the main 56-bit key.

Quick Tip

Feistel cipher is a symmetric structure used in the construction of block ciphers, named after the German-born physicist and cryptographer Horst Feistel who did pioneering research while working for IBM.

However, the relatively small key size is an issue of debate since its inception. By 1999, researchers claimed to break DES in 1 day. To resolve the claims, multiple times execution of DES was suggested, which is known as *3DES*. In three times, the DES was executed in sequence of encryption–decryption – encryption and using 56 + 56 = 112 bits key. Differential cryptanalysis was very effective at breaking block ciphers and DES in particular. In this approach, the study of how changes in inputs can affect the output is analyzed. Another approach based on affine transformations, known as *linear cryptanalysis*, is also widely used to break the DES.

6.4.2.2 Advanced Encryption Standard

AES is similar to DES as it is also symmetric and uses blocks for encryption. It has become the international standard and is not only more secure than DES but it is also six times faster than 3DES. AES uses substitution–permutation instead of Fiestel approach and has a series of operations that either replaces input with output bits (substitution) or shuffles the bits (permutation). It operates on bytes rather than bits and the input of 128 bits is represented as 16 bytes, which are arranged in a 4 × 4 matrix. This matrix, known as the initial state, is modified as the algorithm progresses. AES also facilitates three different key sizes by selecting the appropriate number of rounds. It uses 10 rounds for 128-bit key length, 12 rounds for 192-bit key length, and 14 rounds for 256-bit key length. Similar to DES algorithm, each round in AES uses different subkey, which are 128-bits in length and are calculated from the original key. This implies that there are 2^{128} possible keys. If we represent this number in decimal notation, it is 340,000,000,000,000,000,000,000,000,000,000,000. For the fastest computer, it would take 500 trillion years to try every possible 128-bit key. The AES cipher does not reveal any information about the original text and therefore, the frequency analysis may not have any impact on it.

Fact Alert

On 2 October 2000, NIST announced that Rijndael had been selected as the proposed AES.

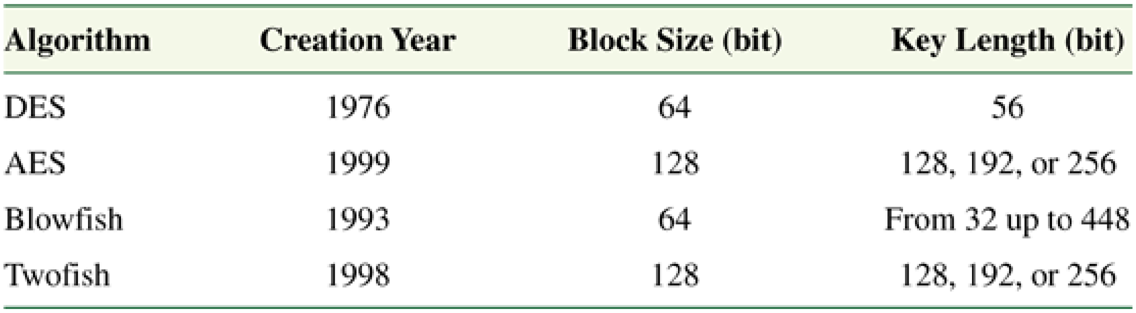
6.4.2.3 Blowfish

Another symmetric key encryption algorithm is Blowfish, which was created before AES and after DES. It uses a block size of 64 bits and can use any key lengths, ranging from 32 bit to 448 bits. Similar to DES, it also has a 16-round Fiestel cipher, but the S-boxes are key-dependent and are generated dynamically, unlike in DES where S-boxes are fixed. It has five subkey arrays, one being an 18-entry P-array and other four are 256-entry S-boxes. These arrays store the subkeys used in every round of this algorithm. One P-array subkey is used in each round while the S-boxes use 8-bit input and produce 32-bit output. This key scheduler algorithm takes the secret key and XORs with all the P entries in order. Followed by 64-bit, all the zero blocks are encrypted with this algorithm and the outcome of this replaces the first two entries in P. It is continued with new subkeys, and the result replaces the following two entries in P, and so on till all the entries are completed. The entire P array, with a total of 9 pairs of entries and each of the 4 S-boxes will be replaced in this initial setup process, which requires a total of 521 runs to generate all the subkeys for this encryption algorithm.

6.4.2.4 Twofish

The Twofish symmetric key encryption algorithm is the successor to Blowfish, and in many ways it is similar to it. It was one of the competing algorithms in the race of AES to be qualified for the global standard. On the parameter of execution, it is slower than AES when using 128-bit key sizes. However, it is faster when using 256-bit key sizes. Twofish and Blowfish have a higher space complexity, that is, both require more computer memory than AES, thus making them less attractive for commercial use. Furthermore, due to its increased key sizes, Twofish is considered to be better in security over Blowfish. This algorithm uses block sizes of 128 bits and key sizes of up to 256 bits. Like DES and Blowfish, this algorithm uses a Fiestel structure while encrypting the messages. It uses precomputed S-boxes, like in DES, splitting the key into two sections, using one to modify the encryption algorithm and the other for Fiestel structure (Table 6.4).

Table 6.4 **Block cipher with key length and block size**



Algorithm	Creation Year	Block Size (bit)	Key Length (bit)
DES	1976	64	56
AES	1999	128	128, 192, or 256
Blowfish	1993	64	From 32 up to 448
Twofish	1998	128	128, 192, or 256

We have briefly discussed a few block ciphers, and among these the AES cipher is considered to be secure but it may not be secure forever. Security researchers across the globe spend their time and energy to discover clever ways to break the cipher.

6.4.3 Stream Ciphers

A method of encrypting text when applied as a stream rather than a block is known as *stream cipher*, which is also often termed as *state cipher* due to the fact that each digit is dependent on the current state. They are inspired by one-time pad (OTP), also known as *Vernam cipher*. In this, a cryptographic key and algorithm are applied to each binary digit in a data input stream, that is, 1 bit of the plain text at a time. On the basis of an internal state, a stream cipher generates successive elements of the keystream. The current state is updated in the following two ways:

1. **Synchronous stream cipher:** Here, the state changes independently of the plaintext or cipher text messages
2. **Self-synchronous stream cipher:** Here, the update of the state is based on the previous cipher text digits.

In practice, a bit and the combining operation is an exclusive-or operation (XOR). In general, a pseudorandom keystream is generated from a random seed value using shift registers, which is used for encrypting plain text. In 1949, Claude E. Shannon proved that stream cipher can be secure provided the key stream is generated completely at random with at least the same length as plain text and cannot be used more than once. However, this makes the system cumbersome to implement in many practical applications, and as a result the OTP has not been widely used, except for more critical applications.

Stream ciphers are different from symmetric encryption block ciphers as the latter operate on a fixed size block of digits. Typically, stream ciphers execute on byte and at much higher speed than block. However, if used incorrectly, stream ciphers can be susceptible to serious security problems, for example, if the same starting state (seed) is used more than once, then it may compromise the message. In practice, a stream cipher uses smaller and more convenient key such as 128 bits in contrast to the length proposed by Shannon. Typically, it generates a pseudorandom keystream using this 128 bit key, which is combined with the plain text digits. As the keystream is now pseudorandom, the proof-of-security associated with the OTP by Shannon does not hold good. There may be possibility that the stream cipher may be completely insecure. However, they are a good fit whenever we need fast execution with low-resource consumption. For example, they are useful for encrypting wireless signals, which more naturally fit a streaming model than transmit data in larger and fixed-size blocks. We will be discussing the A5/1 stream cipher, which is used in GSM phones and the RC4 stream cipher, which is used in the security system for wireless networks.

6.4.4 Examples of Stream Ciphers

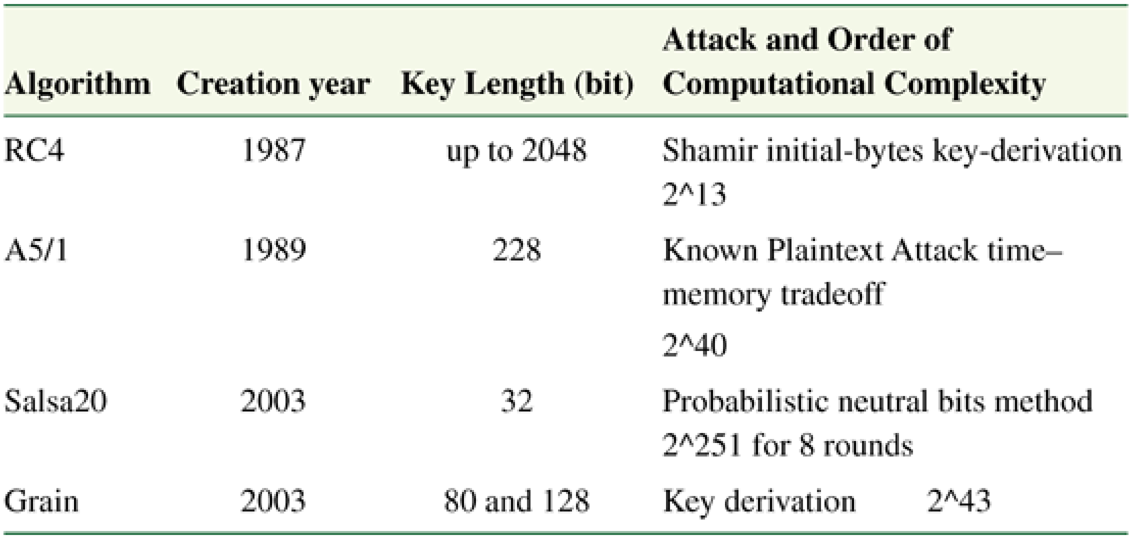
We will now discuss some of the widely used stream ciphers from the days of World War II till today, namely RC4, A5/1, Salsa, and Grain. Table 6.5 contains the information about these ciphers.

6.4.4.1 RC4

The RC4 encryption algorithm was developed by Ronald Rivest of RSA Security in 1987. As RC4 is a trademarked name, it is often referred to as

ARCFOUR or sometimes as ARC4. It is officially named after “Rivest Cipher 4”; however, the acronym RC is alternatively understood as “Ron’s Code” by many. It is a symmetric keystream cipher algorithm and requires successive exchanges of state entries based on the key sequence. It is one of the most popular ciphers and used by standards such as IEEE 802.11 within Wireless Encryption Protocol, using 40-bits key low-end security and 128-bit key for high-end security.

Table 6.5 **Stream ciphers with key length**



Algorithm	Creation year	Key Length (bit)	Attack and Order of Computational Complexity
RC4	1987	up to 2048	Shamir initial-bytes key-derivation 2^13
A5/1	1989	228	Known Plaintext Attack time–memory tradeoff 2^40
Salsa20	2003	32	Probabilistic neutral bits method 2^251 for 8 rounds
Grain	2003	80 and 128	Key derivation 2^43

The design of RC4 is relatively simple to be used in software solutions as it only manipulates single bytes at a time. It works in the following two phases:

1. **Key setup:** This generates keystream bytes, and is more complex than the encryption phase.
2. **Encryption:** The encryption of data is carried out by XOR byte-by-byte, one after the other, to keystream bytes. To generate the keystream, it uses secret internal state using permutation of all 256 possible bytes along with pseudorandom number generator.

The RC4 was created almost 30 years ago, and therefore some weaknesses are being researched by scholars. There is a possibility to find keystream byte values that are slightly more likely to occur than other combinations.

One of its weaknesses is the insufficient key schedule, which can lead to obtain some information about the secret key based on the first bytes of keystream. More on RC4 can be read in Sumartono *et al*. (2016).

6.4.4.2 A5/1

A5/1 is the strongest and most popular of the GSM A5 versions. It was developed in 1987 and is used in Europe and USA. The algorithm takes the symmetric session key and a frame counter, which generates 228 pseudorandom bits, termed as *keystream*, by using three linear feedback shift registers. The keystream is then XORed with a 228-bit segment of plain text resulting into 228 bits of cipher text. Its initial design was kept secret, but was eventually leaked in 1994. The algorithm was completely reverse engineered in 1999 by Marc Briceno from a GSM telephone. Later, Alex Birkyukov, Adi Shamir, and David Wagner (Birkyukov 2000) described in their paper that A5/1 can be decrypted in real time. Most of the attacks exploit the fact that the size of the key is small and that it can be shortened enough to be brute forced.

6.4.4.3 Salsa20

Salsa20 is a stream cipher developed by Daniel J. Bernstein in 2005, and it was submitted to eSTREAM project for widespread adoption. This project was organized by the EU ECRYPT network and was active from 2004 to 2008. This initiative was a result of the failure of all the stream ciphers submitted to the NESSIE project. The project was divided into separate phases, and the project goal was to find algorithms suitable for different application profiles with two sections: software- and hardware-centric stream ciphers.

Salsa20 was built on a pseudorandom function based on Add-Rotate-XOR operations. The core function maps 256-bit key, 64-bit nonce, and 64-bit counter to a 512-bit block of the keystream. Salsa20 offers speeds of around 4 to14 cycles per byte in software on modern ×86 processors and reasonable better performance on hardware. Bernstein did not patent and shared several public domain implementations that were optimized for common architectures; details are present in the public domain at https://www.ecrypt.eu.org/stream/papersdir/2006/007.pdf. Later, in 2008, Bernstein published another algorithm named ChaCha that is a modification of Salsa20. This algorithm uses a new round function, which increases diffusion and increases performance on some architectures.

6.4.4.4 Grain

Stream cipher Grain is also from the eSTREAM project, designed by Martin Hell, Thomas Johansson, Alexander Maximov, and Willi Meier. It was selected for the final eSTREAM portfolio. It was designed primarily for restricted hardware environments with 80-bit key and 64-bit initial vector. The cipher consists of 80 bits a non-linear feedback shift register (where bent function in 14 variables was taken) and same length of linear feedback shift register. This was suggested by the authors to provide high resilience and non-linearity. It was designed for parallel 16 rounds, which results in faster execution with maximum utilization of hardware. This version 1 of Grain supports a key size of 80 bits that is not feasible to exhaustively search with modern computers. However, recent research developments in time- memory-data trade-off conveys that it is possible to attack with complexity $O(2\text{^}(K/2))$ where K is the size of the key. Here, it is assumed that the attacker has a collection of $2\text{^}K/2$ plain texts encrypted under different keys, where the aim of the attack is to obtain one of these keys. In this scenario, 80-bit key size is not enough as it would have a complexity $O(2\text{^}40)$. As a consequence, the authors came out with new version named *Grain-128* that was designed to meet this requirement while preserving the advantages of its previous version. The new version Grain-128 supports key size of 128 bits and IV size of 96 bits and is still very small and easy to implement in hardware. More information on this is available at www.ecrypt.eu.org/stream/p3ciphers/grain/Grain128_p3.pdf.

6.5 Asymmetric Cryptography

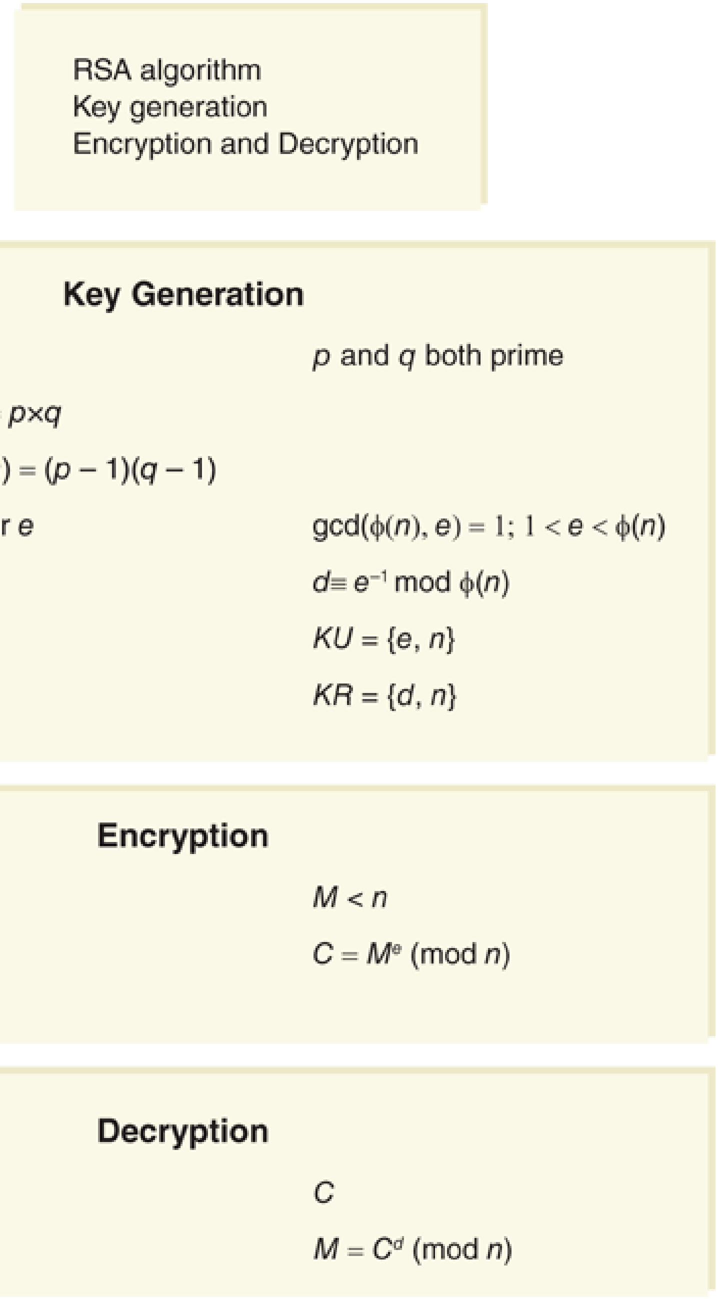
Asymmetric cryptography is one of the important areas of cryptography consisting of two parts: a public key and a private key. The public key is shared in public domain to anyone, trusted or untrusted, while the private key is kept secret with the owner. This is widely used for authentication and non-repudiation purposes. Messages can be signed with a private key, and then anyone with the corresponding public key is able to verify that the message was signed by someone who holds the corresponding private key.

Although encryption can also be carried out with asymmetric cryptography, it is different from the way symmetric encryption is performed. Public key is used to encrypt a message offering confidentiality, while decryption can be carried out only by the person in possession of the private key. However, the computation cost for this is much more than the symmetric cryptography, which we will see later on in this section. It is often preferred to use a hybrid approach, which clubs both symmetric and asymmetric cryptography and takes the respective merits from them.

It should be noted that one of the primary goals of asymmetric cryptography is to provide a mechanism for protecting (or encrypting) their communications lines to both the sending and receiving parties without meeting face-to-face in person. This raises the question of how the receiving party knows which public key should be used to verify the authentication or non-repudiation and decryption, if at all. Let us first see how encryption is accomplished; assume that receiving party's public key is Rpu, which the receiving party deliberately and purposefully notifies to the sending party in a public channel. This will facilitate to initiate the secure communication between them, despite the fact that the sending party and the receiving party have not met/know each other in advance. In fact, it is advisable that the receiving party discloses its public key globally so that whoever wishes to securely communicate with the receiving party can achieve it. It should be noted that even while making the public key open in the public domain, the private key needs to be kept strictly secret. This makes asymmetric cryptography secure. On receiving the encrypted message, the receiver uses the receiving party's private key (denoted as Rpr) to decrypt it, as the generation of these keys are done in a specific manner using sophisticated mathematical function, which is discussed later on in this section.

The basic idea is to create a number as a product of two very large prime numbers. On a basic premise that it will take an attacker a very long time to factor, this number into the two prime numbers makes the asymmetric cryptography harder to break, making the communication between the sending and receiving parties remain intact and secure. It should be noted that the same public key can be used by multiple, different sending parties to communicate with the single receiving party.

One important distinction between asymmetric cryptography and symmetric cryptography is that the former requires only half of the secrecy in terms of private key Rpr, while the latter needs a complete 100% secret key. Another difference is that there should be a different set keys for different users who want to send encrypt messages. Another difference is that while using symmetric cryptography, both the sender and the receiver need to communicate the secret key beforehand; this can happen if both parties meet face-to-face before the encrypted communication takes place. While asymmetric cryptography does not require this. One may note that in asymmetric cryptography, the roles of the sender and receiver are not interchangeable, which is in contrast with the symmetric cryptography where either party can use the same key to encrypt the message to send and receive. However, asymmetric cryptography does possess a significant disadvantage as it is very slow compared to symmetric cryptography. Furthermore, it takes enormous processing power and drains the machine power. There are many asymmetric cryptography algorithms typically based on the integer factorization problem (such as RSA algorithm shown in Figure 6.2) and discrete log problem (such as Diffie–Hellman algorithm), which are discussed here.



RSA algorithm
Key generation
Encryption and Decryption

Key Generation

Select p, q	p and q both prime
Calculate $n = p \times q$	
Calculate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\phi(n), e) = 1;\ 1 < e < \phi(n)$
Calculate d	$d \equiv e^{-1} \bmod \phi(n)$
Public key	$KU = \{e, n\}$
Private key	$KR = \{d, n\}$

Encryption

Plaintext	$M < n$
Ciphertext	$C = M^e \pmod n$

Decryption

Ciphertext	C
Plaintext	$M = C^d \pmod n$

Figure 6.2 The RSA algorithm.

6.5.1 Integer Factorization Problem

One can define integer factorization as a decomposition of a composite number into a product of smaller integers. If these integers are prime numbers, then these are known as *prime factorization*. For example, there is composite number 108, this can be prime factorized as 2 × 2 × 3 × 3 × 3.

Conversely, it is relatively easy to generate prime numbers and multiply them, for example, 19 × 41 = 779. Thus, we see that multiplying two numbers is mathematically easy, but this scales up for bigger numbers. In general, factoring smaller numbers is an easy task but while dealing with very large numbers, it can take even for the fastest computers, days, months, years, or even more to solve. Let us consider a large composite number of size 232 or more digits, then it becomes extremely difficult to find its prime factors. For example, the decimal representation of 768-bit RSA modulus is as follows:

1230186684530117755130494958384962720772853569595334792197322
4
5215172640050726365751874520219978646938995647494277406384592
5
1925573263034537315482685079170261221429134616704292143116022
2
1240479274737794080665351419597459856902143413.

Earlier, integer factorization was more of an academic interest. However, in recent past it had gained importance after introduction of the RSA public-key cryptosystem. When the integers are sufficiently large like the one mentioned, then there is no efficient factorization algorithm. Several researchers in 2009 used hundreds of machines for almost two years to factor 232-digit integer number (RSA-768 bits) using number field sieve method. They concluded that 1024-bits RSA modulus would take roughly thousand times as long (Kleinjung *et al*. 2010).

Flash Quiz

Explore the code for factoring a given integer to prime numbers.

However, there is no formal proof that an efficient algorithm does not exist to prove the same, and only the presumed difficulty is considered for this algorithm in cryptography. Several computer science and mathematical hypotheses have been explored and knitted to overcome the barrier, but so far no success is visible even in near future. One may note that not all the numbers of a given length are equally hard to factor. The hardest instances of these problems are semiprimes, which is the product of two prime numbers. When both these prime numbers are large, typically more than 2000-bits long and almost same size then even the fastest prime factorization algorithms on the fastest computers will take years to factor the product, thus making the search impractical. In simple terms, if the number of digits of the primes to be factored increases, then the number of operations required to perform the factorization increases drastically (i.e., non-linear polynomial in time and is termed as NP–hard problem). We will discuss about some factoring algorithms, which classified in the following two categories:

1. **General-purpose factoring algorithm:** This is also known as *Kraitchik family algorithm* (after Maurice Kraitchik) (Bressoud and Wagon 2000). It has a running time based on the size of integer to be factored. These types of algorithms are used to factor RSA numbers that are based on the congruence of squares method, such as Dixon's algorithm, continued fraction factorization, general number field sieve, Shanks's square forms factorization, and quadratic sieve and rational sieve.

2. **Special-purpose factoring algorithm:** This algorithm's running time depends upon the properties of the number to be factored (i.e., size, special form, etc.) and varies between algorithms. For a given integer of unknown form, these methods are usually applied before general-purpose methods to remove the small factors. Examples of these methods are trial division, wheel factorization, Pollard's rho algorithm, Algebraic-group factorization algorithms, Williams' $p + 1$ algorithm, Lenstra elliptic curve factorization, Fermat's factorization method, Euler's factorization method, and special number field sieve.

6.5.2 Discrete Log Problem

As we have seen the integer factorization problem. Now, there is another problem based on discrete logarithms. This is defined with regard to multiplicative cyclic groups. A cyclic group is a group that can be generated by a single element in the group, which is termed as *group generator*. Let us consider **G** as a multiplicative cyclic group with g as a generator. Now, from the definition of cyclic groups, every element a in **G** can be written as g^x for some x. Given this, the discrete logarithm problem is defined as given group **G**, generator g of the group, and element h of **G**, to find the discrete logarithm to the base g of h in the group. Not all discrete logarithm problems are hard, but it depends on the group. For example, a popular choice of groups for discrete logarithm-based cryptosystems is Zp*, where p is a prime number. However, if $p - 1$ is a product of small primes, then the Pohlig–Hellman algorithm can solve the discrete logarithm problem in this group very efficiently. This is the reason why it is always recommended for p to be a safe prime when using Zp* as the basis of discrete logarithm-based cryptosystems. (A safe prime is a prime number that equals $2q + 1$, where q is a large prime number which confirms that $p - 1 = 2q$ has a large prime factor). Thus, the Pohlig–Hellman algorithm cannot solve the discrete logarithm problem easily.

As there is no known efficient classical algorithm for computing discrete logarithms, in general, it is considered to be computationally intractable. A general algorithm for computing $\log_{\mathrm{b}}(\mathrm{a})$ in finite groups **G** is to raise b to larger and larger powers k until the desired a is found. This approach is sometimes known as *trial multiplication approach* and requires running time linear in the size of the group **G**, exponential in the number of digits in the size of the group. Therefore, it is an exponential-time algorithm that is practically feasible only for small groups **G**. The inverse problem of discrete exponentiation is not difficult to compute, for example, using exponentiation by squaring. This is an analogous feature to one between integer factorization and integer multiplication. There are more sophisticated algorithms that are inspired by algorithms for integer factorization and are faster than the naïve algorithm. While computing discrete logarithms and factoring integers are distinct problems, both these problems seem to be difficult with no known efficient algorithms to break. Moreover, the difficulty of both the problems has been used to construct various cryptographic systems. Some of the popular choices for the group **G** in discrete logarithm cryptography are the cyclic groups (Zp)* where p is prime and this is used as Diffie–Hellman key exchange and cyclic subgroups of the elliptic curves over finite fields. (To learn more about elliptic curve cryptography, go to https://en.wikipedia.org/wiki/Elliptic-curve_cryptography).

To the best of our knowledge, in general, there is no publicly known algorithm for solving the discrete logarithm problem. However, the first three steps of the number field sieve algorithm only depend on group **G** and not on the specific elements of **G** whose finite log is desired. Thus, by precomputing these three steps for a specific group, the only need is to carry the last step. This step is much less computationally expensive than the first three, to obtain a specific logarithm in that group as investigated by researchers in 2015 (David *et al*.2015).

While both the symmetric and asymmetric cryptography have been popular ever since these were invented, they are probably most recognized for their usage in implementing blockchain. As we know, blockchain is essentially a distributed ledger, it is of the essence that the cryptography used in it is both reliable and functional. For example, most cryptocurrencies use keypairs from asymmetric cryptography to manage addresses on blockchain. The public key that can be viewed by anyone and referred to as the address holds the tokens. The private key can be used to access the address and authorize actions for the address. Asymmetric cryptography has many more uses in blockchain, ranging from implementations in simple smart contracts to advanced permission structures.

SUMMARY

Cryptography is a mechanism of protecting information by transforming the information into an incomprehensible format known as cipher text. Only those who possess the mechanism and the related secret key can decipher the message into plain text. There are two types of cryptography – symmetric-key and asymmetric-key. In a symmetric-key algorithm, both the sender and receiver share the key. The sender uses the key to hide the message and later the receiver will use the same key to retrieve the message. Asymmetric cryptography is costly in terms of computation and memory. Entities who want to use asymmetric cryptography uses a secret number (a “private key”) that is not shared, and a different number (a “public key”) that they can tell everyone. If someone else wants to send a message to this entity, then they will use the number they have been told to hide the message by the entity. Now, the message cannot be revealed, even by the sender, but the receiver can easily retrieve the message with his private or “secret” key. Asymmetric cryptography is often used for digital signatures. To achieve security, there are four primary cryptography primitives: (1) confidentiality or privacy, (2) authentication, (3) nonrepudiation, and (4) integrity.

SHORT ANSWER QUESTIONS

1. What is cryptography?
2. What are two types of cryptography?
3. How many keys are used in symmetric cryptography?
4. What is the main drawback of symmetric cryptography?
5. What is the difference between symmetric and public key cryptography?
6. What algorithms are used in asymmetric encryption?
7. How does asymmetric key encryption work?
8. How does a stream cipher work?
9. Which is faster – block cipher or stream cipher?

LONG ANSWER QUESTIONS

1. What are the challenges of symmetric key cryptography? List them by taking at least two symmetric key algorithms.
2. Prove mathematically that asymmetric key cryptography is still safe even if one key of sufficient length is shared in public domain.

CHAPTER 7

Smart Contracts

Learning Objectives

After reading this chapter, you will be able to:

- Understand the concept related to smart contracts.
- Discuss the characteristics of smart contracts.
- Analyze the utilities of smart grid.
- Discuss the proofs-of-origin.
- Understand about the supply chain management.
- Discuss the legal services and Darknet.

7.1 Introduction

Digital transformation is affecting business, law, government, and politics with its problems and potentials. Advances in the background of blockchain technology will have a significant effect on the strategy and execution of digital processes, e-governance initiatives, and ultimately social network processes. Various communities and associations have initiated to establish global practices that necessitate primarily third-party based involvement. Subsequent matters concerning legal, tax, business, education, R&D, and finance are elevated in this process. At the government position, an all-inclusive governing context needs to be formed, which permits modernization while taking into account community and civil constitutional rights. This involves scrutinizing the applicable government inevitability, which should aim at authenticating the permissible rationality of the statements archived in a blockchain, taking into ledger their purposeful and cryptography-based features. Since blockchain utilities are not restricted to a state or country level, researches, queries, proof-of-concepts (POCs), and steering of the lawful costs of transnational blockchains should also be initiated at the global level.

The time when Satoshi Nakamoto research (2008) and Bitcoin came into existence (2009), both research and cryptocurrencies of the blockchain have acknowledged an incredible volume of consideration. The advancement of blockchain use cases is often classified into the following phases:

1. **Blockchain 1.0:** It covers the use cases for cryptocurrencies.
2. **Blockchain 2.0:** It talks mostly about smart contracts in banking, financial services and insurance (BFSI) world hitting financial transactions.
3. **Blockchain 3.0:** It uses the extended smart contracts use cases for autonomous decentralized SBUs of an enterprise, with their individual regulations and high-level independence, with that spreading over in almost all the process area.

Consequently, several innovative arenas of use cases and application potentials for blockchain technology, which is more than cryptocurrency, are presently being established at a fast pace.

The foundation for this awareness is collected by the following features and numerous applications of blockchain concept:

1. In commercial developments, the method of shared consensus development can substitute the character of a trusted mediator/s in the fields of process implementation and validation. This affects mediators in the financial background as well as administrative roles in self-governing responsibilities. It consequently results in querying the business models of various enterprises and establishments that now works in these areas. Also, there are new models that would not be possible with regard to their financial aspects without blockchain technology. Assurance in a mediator is being substituted by the trust in a cooperative, confidence in technology, and consensus in cryptography.

2. In blockchain, standards can be recorded concerning which access privileges can be evidently and lastingly transacted among node users. Consequently, blockchain is perceived as the foundation of the “values” of Internet and as an enhancement to the earlier version of just the “information” Internet. Privileges to physical objects and entity values can be recorded in digital. This encompasses Internet platform that replicates and shares the information that login the backgrounds and ownership of properties and develops them as a transparent entity.

3. If innovation in smart contracts does not include contracts in the authorized and legal sense, the idea of smart contracts permits for guidelines and implementation directions to achieve prearranged developments on blockchain in an automatic and decentralized fashion. This unlocks huge possibilities for automation. The choice of use cases encompasses from transportation and trade to machine-to-machine and to IoT, such as smart objects, which can self-reliantly exchange and begin their use.

4. Fundamentally, the transactions characterized in a blockchain are noticeable to all node contributors in the network and are consequently transparent. Adding to this, blockchain offers irretrievability, which means transactions in blockchain cannot be later tampered or even erased. To inverse/retrieve a transaction, it is only imaginable at another time by agreement to submit the timestamp in the blockchain. This will provide testimonies of source and transactions protected for charted values in terms of assessing and audit logs. This unbolts an extensive range of potentials in the field of compliance and audits up to the process test that have been achieved manually so far, thus enquiring the novelties in an auditor’s business models.

Recently, these features have controlled to a volatile growth of new use cases as well as to an irresistible number of users. These values counted from various new start-ups, established technology enterprises, and freshly molded groups. However, people, administrations, NGOs, academies, R&D, and financers also investigate and evolve the next utility, similar to what the browser was to the Internet.

This enthusiasm cannot cover the element that there are presently a lot more ideas, philosophies, and thoughts than the existing instances that actually work. Since it is at its nascent stage, the complex technology carries multi-layered problems in the range of basics of information technology and also in the field of use cases and outbreak circumstances. Currently, the technology is deprived of infrastructures for an individual arrangement, satisfactory capacities, scale and short response times, a comprehensible administration model, and consistent lawful outline.

In contradiction to this circumstantial situation, an essential task for technology is the judgmental investigative assessment of advancements. The query of whether it is a propaganda or has adequate disruptive possibilities rest on numerous issues, which require to be examined thoroughly.

Queries to be considered should include the following factors:

1. Prospects and hazards.
2. Implementation problems and motivation factors.
3. Effects and influences on civic bodies, governments, and administration efforts.
4. Despite uncertainty, how can enterprises and state bodies start.
5. Classification of technical R&D requests, which comprises of categorization of early bird verticals where it is sought, and changes will happen.

A smart contract is the choice of solution to meet all the challenges and answer all the queries discussed so far.

A smart contract is a computing practice envisioned to digitally simplify, authenticate, or impose the cooperation, mediation, or negotiate actions of a contract. Smart contracts permit the actions of trustworthy transactions without intermediaries and mediators.

As blockchain is decentralized and exists between permitted nodes, there is no requirement to recompense mediators and it keeps a tab on time and consensus. Blockchains have their glitches, but they are valued, indisputably, quicker, inexpensive, and safer than old-style systems, which is why financial institutions and administrations prefer them.

In 1994, Nick Szabo, a cryptography-based legal researcher, comprehended that the decentralized register could be utilized for smart contracts, then known as self-reliant, self-regulated contracts, blockchain contracts, smart contracts or digital contracts. Using this, contracts could be changed to software code, archived and copied on the system and administered by the system of nodes that execute the blockchain. It will also affect ledger response such as transacting money and receiving goods or services.

7.2 Smart Contracts

At the first stage, blockchain empowers the decentralization of transaction. It also processes automation and establishes guidelines and corporate/administrative principles. The transactions can be accompanied by instructions for conserving reliability and then valued as smart contracts. They categorize what to investigate in a transaction and what continued activities are to be introduced.

With the combination of smart contracts and related process automation, many activities can be upgraded drastically as part of a transformation process. It can also be facilitated by certified auditing agencies if the uniformity of the data is guaranteed by a smart contract and inspection-proof archive. Standard philosophies of transformation strategy, such as replicate/archive/store only once and can consequently be executed in a standard manner using blockchain. When data has been established and confirmed, it is fed in a tamper-proof method and can be combined in a variation of situations. As a consequence, from a technical perspective, blockchain is a defacto means for process automation and optimization.

Quick Tip

A contract is a legally binding agreement which recognizes and governs the rights and duties of the parties to the agreement.

Thus, blockchain technology not only has various impacts on the processes, but also on the constructs of transparency and administrations that can meaningfully modify the delivery of transaction between process member nodes. But this elevates the request of new business models, and one process is reengineered for the new value chain.

Blockchain has disruptive potentials and old style-based process optimization does not seem to be suitable. A renewal of traditional optimization approaches appears conceivable as they have investigated processes from a tactical viewpoint and as the client value. Optimization also takes care of the role transformation of the participant/actor. In the meantime, process modeling requires to be combined with the advancement of new regulation and modeling constructs. Traditional process modeling methods are control-systems based flow structure. The questions that arise are as follows:

1. Can we integrate new auditing certification with traditional modeling languages governed by the state/regulatory agencies?

2. Is it possible that new actors in process control optimization provide substitutes to the new regulation in repositories?
3. What about the value chain integration with business modeling with various stakeholders and customer propositions?
4. What will the construct of new modeling language if it can be integrated with industrial, social, and economic successes and it is made workable through the groundbreaking processes of blockchain?

7.2.1 Definition of Smart Contracts

Smart contracts are arrangements/pacts/deeds/agreements that leverage blockchain to autonomously and safely accomplish duties/responsibilities when specific situations are encountered. Corresponding to other blockchain-based controls, the smart contract is intended with design consideration to execute without dependence on a centralized body.

A smart contract is considered as self-reliant/executing/enforcing (in simple terms, autonomous). It works on Boolean condition such as if-then-else.

Based on different schools of thought, each role has been given different definitions for smart contracts using its applications.

One school of thought states that smart contracts in the legal discipline reverberates utmost with lawyers/attorneys/notaries. Here, the concept "smart contract" is utilized to denote fundamental legal contracts, which are characterized and executed by software.

Another school of thought states that code-based smart contracts are relatively not related to lawyers/attorneys/notaries and it is a section of code works as an agent (software) that is architected to process specific activities if predetermined circumstances are met. Such activities are frequently rooted inside and achieved on a distributed centralized ledger.

For instance, one renowned smart contract application defines agents that generate cryptocurrency, deliver an elective instrument, and provide a computing-based silent auction instrument as smart contracts.

The discussed differences can create problems when the subject of smart contracts is discussed, and there is a peril that attorneys and software programmers/scientists merely exchange their concepts as cross-objectives. Though by observing smart legal contracts as well as contract code seems to be two distinct fields, the truth is that there is an association among them. For a smart contract that is legal in nature to implement, it will require to insert one or more fragments of code planned to perform specific activities if predetermined conditions are encountered and portions of code of that smart contract contains in it. Smart contracts, for purposes legal, are consequently efficiently constructed with a section of codes of the smart contract, but critically, under the roof of an inclusive association that produces lawfully enforceable privileges.

We will now differentiate between the two types of smart contracts – legal and code.

It is convenient in guaranteeing a precision of practice, but it does not give outcome in a basic definition. Understanding that there is no sole unanimously recognized definition, it is however suitable to establish a rudimentary explanation that attempts to depict a combined interpretation of what the subject smart contract summarizes.

Therefore, a smart contract is an agreement that is automated (using computing resources) and administered. Automation is achieved by software agents (computing resources), but it requires human interventions and control mechanisms. It ensures enforcement, either by legal bodies of governance and responsibilities or by using alteration/change/tamper-proof implementation of computer code.

This explanation has the benefit that it is comprehensive and sufficient to cover both the smart contracts — legal and code. It seizes the spirit of smart contracts, the capability of smart contracts to self-govern and to be self-reliant using Boolean logic.

7.2.2 Centralization–Decentralization Smart Contracts

In the old style centralized classical model of business associations, there is constantly a third body that is positioned among the two parties that are performing a transaction and confirming the regulations and clauses in a contract. This third body can be any of the following:

1. Finance institution.
2. Law implementation company.
3. Administration/state/national institution.
4. Mediator or any other intermediary.

When developing associations within a centralized system, industries are reliant on mediators that place patrons at peril. Also, central network cannot promise expenses/payments and execution of contracts.

The advancement of blockchain technology, which permits verticals to shape decentralized systems, unlocks new prospects for businesses to perform transactions and make contracts.

7.2.3 Smart Contract Concept

Although smart contract publicity has risen with the excitement towards blockchain technology, the concept of smart contract really came into the picture about 25 years ago. It was propounded by Nick Szabo, a well-known cryptography scientist, in 1995.

The Szabo offered idea exactly resembles what smart contracts give today, counting the innovation of using distributed ledger in executing and archiving smart contracts.

7.2.4 Smart Contract – Key to Trust and Security

A smart contract is similar to an agreement in the real world; it is digital and is characterized by small software codes embedded in a blockchain. More specifically, a smart contract is a portion of computer program that stores instructions/guidelines for exchanging the clauses of a contract, automatically authenticates execution, and then accomplishes the settled conditions.

Smart contract eliminates dependence on a third party when developing business associations and relationships; the parties developing a contract can perform activities among themselves.

7.2.5 Trust and Security

Smart contracts ensure that a blockchain is more than decentralized distributed shared autonomous encrypted storage and make probable the computerized and reliable alteration of information in the blockchain. For instance, smart contracts can be utilized in Bitcoin to execute several kinds of transactions, such as escrow, to comprehend the trust-based data repository.

The accuracy of smart contracts is of the highest importance. Meanwhile, with respect to web applications, for instance, constant changes of smart contracts are not voluntarily offered. This ensures that once a smart contract code has been input in the blockchain, it cannot simply be tampered and altered without enquiring the truthfulness of the data archived in the blockchain.

In the historical records, hacks on smart contracts have regularly been informed, some of which were made thinkable by tough-to-identify software design errors such as: (1) unchecked-send, (2) re-entrancy, and (3) solarstorm vulnerability for concurrency.

However, implementation ecosystems for smart contracts are also partly ambiguous. For example, processing ecosystems can utilize the authenticating peer node’s unrestricted computing resources; at the same time, smart contracts can simply be utilized as a denial of service (DoS) attack on the node.

Moreover, smart contracts being the code is not restricted to interact with the blockchain, but even calls the functions that are based on external services. Consequently, destructive smart contracts are also imaginable, which can send spam or attach bots within the blockchain.

While working with smart contracts, the following are to be safeguarded:

1. On its own, a smart contract has to be accurate and protected against outbreaks/attacks such as re-entrancy. This is not insignificant to safeguard as the decentralized autonomous organizations (DAO) attack is exposed.
2. It has to be guaranteed and certified that no vulnerable smart contracts pass in the blockchain. This is particularly true for blockchains with authoritative smart contract paradigms such as Hyperledger and Ethereum. Although Ethereum is succeeding in the primary steps in helping official authentication of smart contracts using the why mechanisms, such actions are still too awkward for most programmers and need too much contextual information in order for the measures to be utilized in an effective manner.

Overall, we need good R&D in the area of secure smart contracts both in by the means of officially provable languages as well as in supporting coders and the authentication of code previous to insertion in the blockchain.

Quick Tip

Ethereum implements a nearly Turing-complete language on its blockchain, a prominent smart contract framework.

7.2.6 Business Models

Because of the precise characteristic of blockchains particularly in distributed consensus reaching, digital value transactions, and automation, they are unalterable. First, it has the power to compete with wide-ranging business models of various enterprises and establishments. Second, it provides the option of new business models that would not be replicated without blockchain, at least not in a cost-effective method.

The power of blockchain is known for different kinds of backgrounds and use cases. It is presently uncertain how the structural features of business models assist privileged blockchain and how they can be best viable in economic ways.

In contradiction to this context, many queries arise for theoretical research regarding business models and the economic value of blockchain solutions, which are as follows:

1. What are the feasible use cases for blockchain technology, especially from a financial viewpoint, and how should they be considered and designed?
2. What are the attributes of business models, hidden from the explicit application, which can get an advantage from recognition with blockchain?

3. How can the influence of a blockchain operation on recognized business models be forecast, such as in the event of the removal of mediators?
4. How can business get advantage of blockchain solutions against old-style deployments?
5. How do we simulate and model blockchain business models and instrument them in an inexpensive and eloquent method?
6. Outside discrete businesses, what effect will blockchain have on current businesses or the finance industry?
7. What are the prospects and perils for developed and underdeveloped economies?

A methodical response to these queries reveals the power of blockchain in precise practical situations. In various scenarios, today's use case situations do not go beyond the example POC position. Only a logical study of appropriate business models and their economy will benefit the blockchain to brand a revolution in feasible application states.

7.2.7 Actions

By insertion, a smart contract on an open public blockchain that is permissionless, regulate over the implementation of the contract will not exist be in the possession of a sole party, such as a financial institution, and communication will not be restricted to closed systems, such as financial institutions' centralized accounts.

The appropriate accomplishment of contracts could be confirmed by the network of node systems linked to the blockchain. That same system would appraise the blockchain to update implementation of the agreement, and then supervise it for compliance with the conditions of smart contract.

A smart contract, shared and copied transversely a blockchain, could transform the manner in which business is performed. Possible practices are plentiful, which are as follows:

1. Assets proprietorship could be shifted automatically upon reception of deposited capitals.
2. Recognitions under service-level agreement (SLA) could be automatically compensated at the point of defilement.
3. Securities could be operated without the requirement for government securities stocks.

7.2.8 External Model

With respect to external model, the legal agreement would continue in its current form (such as a readable normal human linguistic artifact), external to the legal agreement. Specific restricted terms logic essentials of the legal agreement would be programmed/coded so that essential activities occur automatically when the applicable terms are fulfilled.

The program is not part of the legal agreement. Overall, it will offer a framework for the automatic act of an agreement inscribed in a natural human linguistic form. If there were any alteration in what occurs when the code accomplishes and what the legal agreement needs, the latter would take priority. In this external mode, consequently the programmed code would not lawfully work itself with the parties and would also not eliminate legal uncertainty. Therefore, the parties would essentially be content that the code precisely reproduces their requirements in the form of natural human linguistic legal contract.

In various aspects, at a minimum for solicitors/lawyers/notaries, the external model is only a trivial phase further than the working procedure offshoot of counterparts that already exists. Certainly, there are areas where these types automation already exists; for instance, daily security streams are already automated in the way of specific margining provisions.

That is not to disprove the possible influence a prevalent acceptance of the external model might have in execution, but just to note that it would probably be an insignificant phase for lawyers.

With respect to external model, it is consequently not only just the agreement that is “smart”, but it is the code/program development constructs that would go together with smart contracts and would be utilized to implement it.

7.2.9 Internal Model

In internal model, things do not change much and the legal agreement would possibly continue in its present-day form, but it is developed critically, with specific restrictive logic essentials of the legal agreement redrafted in a more official depiction than the present natural human linguistic format. A software system will provide more official and recognized illustration and implement the provisional logic using code.

The inscribed agreement would, as an outcome, look like a mishmash of methods. Certain sections would be conscripted in natural human linguistic, similar to what we have today. Nonetheless, other sections would successfully be set downcast on the folio in some procedure of code, or other recognized depiction. Instead of formulating the code or prescribed depiction within the written agreement itself, the inscribed agreement could mention to a recognized portion of code warehoused to another place and might tell that such a program is to be given lawful result among the parties.

7.2.10 Endorsements

Finances and policymaking are working rigorously with the opportunities and possibilities of digitization. Advances in the background of blockchain technology will have a lot of influence on the strategy and execution of digital processes and on social levels. Currently, global ideals are set by various start-ups and associations, making primary participation essential.

This should happen among various lines of business; meanwhile, it elevates questions connecting to R&D, tax, legal, and finance.

At the policymaking level, there is currently a nonexistence of all-inclusive legal framework which empowers innovation while also safeguarding citizens. To accomplish this, it is important to confirm legal inevitability. Although transactions in the next generation blockchain are not tampered, the usage of transactions or declarations archived in a blockchain, such as evidences of source, is presently uncertain. The following have to be studied:

1. What are the legal strengths of the statements stored in a blockchain ownership activity?
2. What practical and cryptographic necessities should a blockchain partake for this intention?
3. Blockchain use cases are not limited to a state level. It is supportive in global developments. So, inquiries and the inspections of the legal penalties of global blockchains should be started.

Currently, several enterprises are evolving POC solutions for both governed and ungoverned processes. For these initial level operations to discover their path into new business models, it is meaningful to scrutinize the usage of blockchains in extremely controlled areas, such as healthcare, energy, utilities, and BFSI. If required and if the experimental examinations outcome in a positive assessment, the usage of technology should be made conceivable by the help of judicial deviations that formerly spread over all applications.

These more administratively and policy-driven actions should be achieved under a state that is integrated in the global outline so that no solutions which are limited to any state are implemented.

7.2.11 Standardization

An important characteristic of blockchain technology is the capability to link transactions with software code to suppose smart contracts. It is predictable that designs and prototypes will arise for commonly used utilities as well as markets for smart contracts. Evolving consistent standard development blocks as an outcome, which entails the following:

Technical Stuff

Standardization is the process of implementing and developing technical standards based on the consensus of different parties that include firms, users, interest groups, standards organizations, and governments.

1. Auditing and accreditation agencies to authenticate their use cases and process truthfulness of smart contracts.
2. Reference repository and marketplaces providing smart contracts, mostly for small and medium enterprises.
3. Early cautionary systems for handling of recognized attacks and hacks.

These actions are predominantly pertinent for the usage of new technology by small and medium enterprises. Although large enterprises are capable to launch their own subdivisions for the advancement of smart contracts and blockchain use cases, small and medium enterprises are dependent on buying connected services and proficiency. Markets for smart contracts are likely to be established in future, much similar to application stores for mobile communities.

We already have open and extensively distributed blockchains that are principally utilized for cryptocurrency; private, entrée-governed blockchain requests will be developed in future. It is predictable that there will be a propagation of blockchain structures with overlying application frameworks. Therefore, it is essential to establish a blockchain record to register and proclaim blockchain structures in numerous use case areas. The goal of this work is to evade the duplication of actions and generate collaborations. This delivers the prospect to handover parallel actions to a mutual blockchain set-up. If the distribution of an infrastructure does not develop logic due to practical or business explanations, then it will be essential in future to normalize edges for interoperability. This means that an environment can mature in the long term that allows, for example, the interoperable usage of blockchain contracts for monetary transactions, assets tracking and the quality control of production information.

An execution intensive on small and medium enterprises would be significant in the quest of these commendations. This is mostly because small and medium enterprises often work closely in value conception systems, precisely where blockchain can completely progress its latent. For this motive, suitable recommended actions have to be started for small and medium enterprises to progress knowledge and to develop refined knowledge by POCs and sand boxes. This may comprise development and functioning of small and medium enterprises with intensive blockchain set-ups for different kind of use cases.

7.2.12 Benefits of Smart Contracts

Unambiguous programming codes and procedures in dominant and blockchain attributes such as transparent ecosystem, decentralized infrastructure, tamper-proof and other properties brand smart contracts a trustworthy substitute for founding business relations and executing transactions. As a substitute to old style contracts with centralized systems, the following are paybacks that smart contracts provide to businesses:

1. **It helps to establish direct transactions with clients:** Smart contracts eliminate the requirement for mediators and permit for transparent, straight relations with clients.

2. **It is failure resilient:** Since industries are not reliant on third-party mediators, no mere individual or unit governs information or finance. Decentralization signifies that even if any node/individual exits in the blockchain system, the system will remain functional with no damage on information or truthfulness.

3. **It ensures trust-based systems:** Business contracts are automatically implemented and obligated. Adeptly, these contracts are absolute and therefore indestructible.

4. **It helps to reduce fraud:** Because smart contracts are kept in a distributed blockchain system, their result is authenticated by everybody in the network. So, no individual can push to release other individuals' assets or information, as all other blockchain contributors would know this and sign such an effort as inacceptable.

5. **It is cost efficient:** Removing mediators eliminates extra charges, permitting businesses and their clients not only to communicate and transact straight but at a very low-to-no charges for transactions.

6. **It is based on ledger keeping:** All agreement transactions are kept in sequential order in the blockchain and can be retrieved laterally with a comprehensive audit track.

The advantages of this system will range to all major sections of the specific BFSI services business, athwart value chains and also create noteworthy value in three significant extents: risk reduction, cost savings and improved productivities.

Agreements or chronicles stored on blockchains or permissioned record books eradicate the requirement for a central middleman to deliver trust in the scheme. For marketplaces that do not use mediators, it is still more trust than existing actions such as:

1. **Finance:** Delivery of private equity to small and medium enterprises in a crowd funding as well as initial public offering (IPO).
2. **Structured economics:** Exchange and clearance of big, security loans such as consortium loans between a cluster of financial institutions, mutual and pension funds.

By automating sections of business processes in a shorter period of interval and perhaps complete processes in longer periods, smart agreements would meaningfully diminish the costs related with zones such as compliance, ledgers, and human intervention.

The actual advantage and supremacy of the technology is around reducing costs, perils, fault rates and settlement processes while permitting everybody to have a distributed mutualized structure.

7.2.13 The Lawful Standpoint

Technology often outperforms governing methodologies and legal perspectives, which is a tendency that is accepted out in the extent of smart contracts. To make smart contracts interoperable with the current legal system interfaces, originators of smart contracts infrastructure are vigorously working on numerous shades from a legal perspective.

7.3 Absolute and Immutable

Smart contracts developed as software code on distributed records would state that the agreements, once settled upon, cannot simply be tampered. This would originate in real-world glitches in several practical scenarios on how the conditions of the agreement could be altered once it exists. Agreement law provides requirements for the alterations, modification, or termination of agreements. Technical instruments in smart contracts can accomplish similar objectives. One conceivable tactic is what we often mention to as a "leakage access" of a precoded system of altering the condition of a smart contract. However, safeguarding that the exact authorizations are integrated into the leakage access itself is complicated, as is guaranteeing its right operation.

7.4 Contractual Confidentiality

Generally, a replica of smart contracts performed on a blockchain ledger is distributed with the nodes. The anonymity of the participants can be protected, but the confidentiality of contract accomplishment is not essentially secured. This is a zone that is getting consideration and where development will be made. It is required to resolve the challenge of confidentiality conserving information sharing inside enterprises, and between enterprises, by practicing cryptographic constructs. Likewise, a thought recognized as zero-knowledge-proofs is being discovered to change a method to detach the technique of authenticating a transaction from observing the data of that transaction.

7.5 Law Implementation and Settlement

The BFSI services business is extremely controlled, and precise licenses and endorsements are allotted to enterprises to contribute in a distributed ledger dependent market. Though the legitimacy of financial smart contracts is so far to be recognized. Truthful translation of legal clauses into software code is another important feature to deliberate. Start-ups are employed on a scheme that automatically translates legal papers into smart contracts, streamlining their understanding by both lawyers and software-based contract developers.

Lawmakers, government, controllers, and administrations have started to comprehend the value for distributed, decentralized ledgers in enhancing compliance, transparency, monitoring, evaluation, and reporting. The thrust from these establishments will be influential in soon overpowering lawful and governance sprints.

7.6 | Characteristics

There are some characteristics that smart contract provides. These are discussed in the following subsections.

7.6.1 Independence

Smart contract is developed by an individual (node), there is no requirement to depend on an agent, notary or other mediators to authorize. Parenthetically, this also blows out the risk of tampering by any third party. Meanwhile, implementation is somewhat accomplished automatically by the nodes of the network by one or more, perhaps prejudiced, nodes which may err.

7.6.2 Faithfulness and Trustfulness

Contracts are encoded and encrypted on a distributed shared ledger. No one can announce/broadcast that it has been lost.

7.6.3 Recovery and Backup

On a centralized network, a record may be lost, but in decentralized network the copies of information are backed up at each node. Information is redundant and updated many times.

7.6.4 Protection

Using cryptography, website and information addresses are encrypted, which preserves our contracts and prevents an attack. Actually, it would take an uncharacteristically smart attacker to bang the code and intrude.

7.6.5 Agility

One would normally have to devote huge amounts of time and form filling (paper based) to physically execute documents. Smart contracts leverage software program/script code to automate activities, thus saving time with a variety of business developments.

7.6.6 Reduce Expenditure

Smart contracts help to reduce the expenditure as they negate the existence of a middleman. For example, one needs to pay solicitor/notary to witness a transaction.

7.6.7 Truth-based Accurateness

Smart contracts are processed faster and cheaper. It reduces the chances of errors as it is not dependent on human intervention that arises from physically filling out loads of forms.

7.6.8 Smart Contract Applications

Presently, most of the real-world blockchain use case situations are associated to the BFSI sector. An outline of the detailed blockchain solutions clearly establishes that several developments in this arena can be allotted to the succeeding ranges of the application.

7.6.9 Cryptocurrency

Blockchain use cases as a transaction record for many cryptocurrencies, such as Bitcoin, Monero, and Ethereum.

7.6.10 Commercial Networks

Blockchain use cases in the range of smart contracting and information distribution and share, such as Ethereum (smart contract utilities), Hyperledger, and MultiChain.

7.6.11 Banking

Blockchain stimulated smart contracts in financial dealings, such as Corda and Ripple. Blockchain knowledge is utilized by many financial technology companies. R3 (a financial services provider) is an association of important universal financial establishments that is employed on the execution of a blockchain-based system for handling financial transactions among financial organizations. During a financial execution, they trust on a blockchain mechanism, such as Corda. Ripple delivers a message protocol for banks working and dependent on the blockchain like the traditional Society for Worldwide Interbank Financial Telecommunication (SWIFT) practice.

Fact Alert

SWIFT was founded in 1973 at La Hulpe, Belgium. It provides a network that enables financial institutions worldwide to send and receive information about financial transactions in a secure, standardized, and reliable environment.

Blockchains can be utilized in all extents that encompass seizure, proof, or dealings of any kind of agreement or object. For example, smart contract companies which are built upon an extraordinary use case situation. This enterprise generates or accomplishes digital documents regarding the source, credentials, and possession of diamonds and inscribes them into a blockchain with the objective of restrictive scam in the diamond trade.

In the same way, in the area supply chain management for contract requirement, the firms are making many blockchain-based contracts, like for instantaneous tracing of statements and transactions as well as for certification of constituent history in the supply chain. The subsequent segments take a closer depiction at some of the application use cases and businesses for these smart contract solutions.

7.7 Internet of Things

Physical smart entities are an important component of Internet of Things (IoT) and it requires digital networking. The purpose is to advance the value of the communication among human and machine or even among machine to machine. Subsequently, a central governance of IoT would perhaps be almost dreadful. It also aims at an extensive independence of the intelligent entities. This self-sufficiency-based autonomy can be maintained in numerous ways by blockchain technology.

For logistics and transportation as one of the important contract areas, this requires that possessions and properties network with each other, share their positions and transfer precise communications in the benefits of best added value using smart contracts. The consequential value-adding actions in the form of the smart contract have to be followed and stored visibly for all contributed actors. It is in no way related to internal organization processes such as defining the consumption of resources, tracking in the instance of a quality shortcoming, environmental footmark, or inter-organization processes (such as cost distribution, invoicing, proof of use based on smart contracts). Whatever may be the case, the value-added transactions that have been achieved have to be recognized, agreed, and the association between the possessions and the used properties and work apparatus have to be made accessible, in a contract form that cannot be altered, to all those who participated in the network. In the event of the proof of usage of a product or environmental footmark, contract also puts on to the client who has a curiosity in tamper-proof information.

In this case of decentralized system, blockchain contracts proposes a method that can substitute a central body which is problematic to execute and which is fundamentally not in the benefits of those who participated. A bigger test in this aspect lies in the protected linking of physical entities (like sensors) with their information in the form of contracts and the agreeing accesses in blockchain. It is significant to safeguard the virtual access in the network, and physical entities (goods) are exclusively connected to each other and can be allocated.

In combination with IoT, smart contracts deliver the likelihood of machines reaching contracts that confirm to be complied within two directions.

1. Guaranteeing the Internet of value, machines can formerly peak their facilities straight to their users and protect received rewards in a decentralized fashion in digital wallet.
2. Agreement and invoicing can also originate if a machine is not associated to the Internet at that particular moment. They will be harmonized in future using blockchain. For instance, upcoming business models would be imaginable in which companies let operating machines (such as self-directed vehicles) deliver their services (such as Uber taxi services) totally free of charge with the help of digital smart agreements (contracts). Machines then receive their currency in a straight way (by transporting individual), monitor and state maintenance necessities autonomously and invoice directly in both the ways. Excesses are finally dispatched to the producer. Even the current electorally debated revenue system of labor of robots would be relaxed to comprehend if we account the wages of robot using smart contracts. Like manpower, share of the machine's revenue would be sidetracked to the tax enforcing establishments and then to the citizens.

7.8 Utilities: Smart Grid

Smart grid is an example of the IoT and grants extensive problems due to the intricacies of electricity grid. The energy segment is presently being designed by two important tendencies, which are as follows:

1. With their unstable supply, recyclable and renewable utilities need improved harmonization between supply and demand within the grid.
2. Subsequently, this supply is decentralized. Very few central points-based large power plants exist. This is a perfect use case of decentralized smart contacts.

For instance, it is being deliberated that consumers of used or autonomous generators of electricity no longer do any business with their individual electricity board, but in its place with other consumers in the network, like the blockchain nodes. With respect to IoT (as defined earlier), this can be connected to discrete devices. Because of the restricted trustworthiness of such random performers in a decentralized electricity value chain, blockchain technology delivers a perfect foundation, appreciations to its capability to transact value using mutual contracts.

Alternatively, though, centralized solutions would continue to be available. For example, distribution system operators could coordinate decentralized trade and thereby represent a decentralized center of trust. It, therefore, remains to be seen to what extent solutions in the smart grid will be used in the well-developed and heavily regulated electricity market or to what extent will they be used particularly in other or more specific application scenarios.

7.9 Proofs of Origin

Providing, reviewing, and preserving proofs of origin (provenance) today represents a significant economic factor. Not only are auditing firms, auditors, and certifiers affected by the potential for change in a blockchain, so are manufacturers with respect to tracking their products.

Systems make it possible to track owners as well as the change of ownership, such as diamonds. Smart contract documents all ownership-related transactions for each diamond. As a result, the ownership history can be traced back beyond a doubt to the registration in the system. The possibility of identifying diamonds proves to be advantageous. Because of its optical behavior, each diamond is uniquely identifiable, similar to a fingerprint. When a diamond is examined, the fingerprint can be used to verify whether it is known in the blockchain, enabling ownership clarification. This service is of interest for a variety of business partners, such as banks, insurance companies, diamond dealers as well as police and courts.

Comparable proofs of origin are also needed for other industrial sectors and product types. First, in the case of product approvals, it has to be proven that certain conflict minerals (such as tin, tungsten, and tantalum) were not used in the production process. On the other hand, the use of manufacturer-certified spare parts is of interest in order to ensure that, for safety reasons, no counterfeit components are used. Both can be based on a clear identification of the products or audit-proof duplication of bookkeeping. At the core, it always has to be ensured that the origin of the products and raw materials is clearly traceable.

For example, when transporting dangerous goods, each vehicle is to be supplied with extensive documentation regarding transport containers, vehicle characteristics, training, etc. With smart contracts, regulations and provisions in the sense of an electronic contract management can be mapped, that is, rules and processes are formally described and automatically monitored. According to the newly established guidelines of supervisory authorities, the transport documents are now to be managed in digital form and are readable by various stakeholders. A closed system of information and processes for transporting dangerous goods is obtained, which can always show that the legal requirements for each transport have been followed. In addition, international transport chains can check national regulations while at the same time standardize safeguards in order to comply with the minimum requirements in each country.

Individual proofs of source also require a significant attention in the management of individual certificates. The digital journey of application method means that applicable certificates and papers are shared digitally. In future, evaluation of the originals can barely happen. A smart contract-based certificate in blockchain can guarantee that deposited certificates are not altered afterwards. For this resolution, it is essential that the originators of such certificates (smart contracts) such as universities, training institutes, and auditing companies list a digital finger impression of allotted papers in the blockchain. Proprietors of the documents can utilize this access to document the truthfulness of a deposited document that generates further assurances in the precision of an application or even of stamps of sanction.

7.10 Supply Chain Management

Supply chain management is an exciting arena of smart contract use case. Its value creation stakeholders include the following:

1. Providers (dealers and suppliers).
2. Manufacturers (producers).
3. Retailers (vendors).
4. Logistics and transportation.
5. Monetary service providers.

It requires technology for the safe sharing of information in inconsistency with the context of growing digital journey. Digital work in the background of IoT also helps to several new potentials of digital process governance in supply chain management and mainly in the financial area.

Today, we have touched the boundary of what is conceivable in the arena of physical service distribution of logistics and transportation execution in the supply chain by using automation, which are as follows:

1. Newest hardware and software.
2. Smart planning concepts.
3. Smart processes.
4. Digital technologies.
5. Digital financial processes.

The supply chain is yet too sluggish and consequently not integrated from the real-service formation course. The details for this can generally be credited to physical and error based human intervention processes that are error prone. Today majority of business transactions are dependent on paper statements still. By means of blockchain-based smart contracts, transactions can be managed self-sufficiently and freely. It also permits easy incorporation and protected networking between stakeholders of the supply chain. Figure 7.1 shows the basic smart card blockchain-based supply chain network.

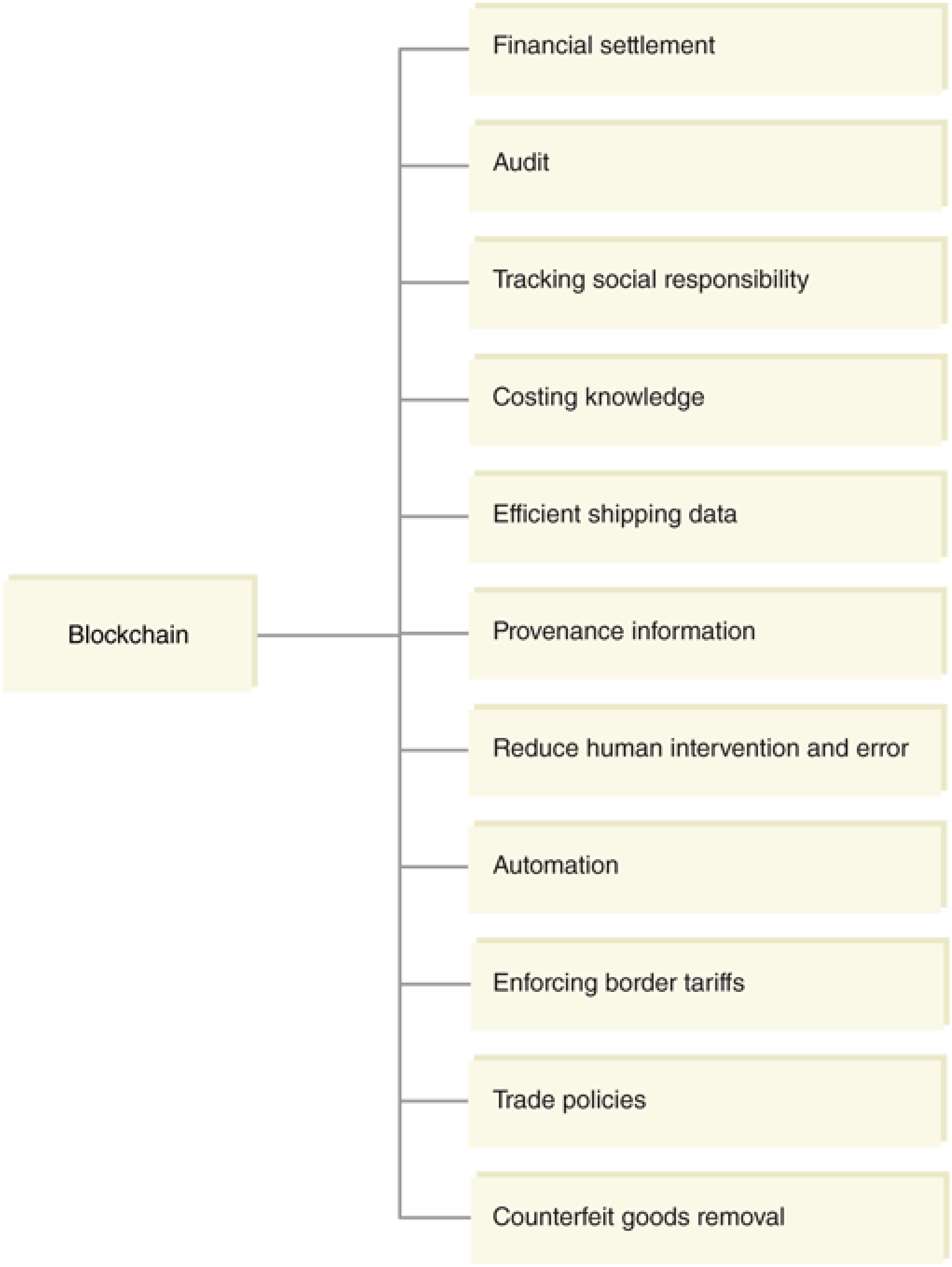


Figure 7.1 Basic smart card blockchain-based supply chain network.

The blockchain turns as shared decentralized distributed information storage openly and irretrievably protects all applicable information for the smart contract. Upon this data, the smart contract as code authenticates compliance with the individual information of the contract and autonomously allows financial transactions in the accomplishment of specific agreement term clauses. In blending with the use of decentralized regulator elements, logistics entities in the supply chain system can freely make scheduling choices and allocate orders autonomously.

Additionally, smart contracts deliver excessive probabilities for growing the competence of processes, particularly in operational and tactical buying, to independent scheduling choices and autonomous transaction executions. For instance, orders can be completed freely, and parent trees of values can be formed transversely at numerous levels of the supplier chain. The advantage in transparency for producers would be massive. By now, many examples exist of how rental contracts can be scrutinized using smart contracts. If an occupant miscarries to pay the rental fee on time, automobile can be gridlocked or further driving can be prohibited through the network.

7.11 Medical Sciences

Many use cases of blockchain smart contract are possible in the area of medical sciences. While no precise use case situations have been executed and published till now, research and development (R&D) in this area has been provoked, determined by the value creation of resolving central difficulties on the method to data based, adapted drug in an ecosystem of extremely sensitive information. It is the time digital medicine; extensive health-associated information occurs in diverse systems, which comprises of the following:

1. Sensor data (wearables like fitness watches).
2. Radiological images.
3. Clinical pathological information.
4. Digital health histories.
5. Confidential information like genetic tests.

Additionally, it is a prerequisite of current precautionary medicine to appraise such information across the masses. The traditional technique of modeling cannot be tracked in this ecosystem since information of this type cannot be composed and delivered in a centralized method. One conceivable methodology involves instruments that permit the possessor of the information to have transactional, governable control over the information's usage and can audit as required. This is exactly the value scheme of smart contracts.

It is important to note that all the features of blockchain are not required in the field of medical sciences. For instance, it is not categorically essential to decentralize the authentication. Conferring to the usual view, the following main technical problems have to be understood:

1. Continuous supervision of information practice has to be lined out so that it exists outside of the system. This may also necessitate the authorization of safe hardware.
2. Individual and group-based data usage authorization has to be conceivable to restrict the usage of information to explicit individuals or establishments or to precise fragments of the information.
3. Comprehensive decentralized information and incident log situations where the data is stored decentrally such as in the following cases:
    a. Hospitals.
    b. Insurance firms.
    c. Medical devices.
    d. Wearable digital devices.

This has to be moved between network systems. Safeguarding transmission with respect to manipulation is the main problem.

Among all of these areas, there are prospects that could deliver a foundation for generating an innovative method to custom data-based health administration using smart contracts-based blockchain.

7.12 | Finance

The finance industry is presently the segment with the major action in the blockchain field. Numerous well-known financial establishments are just opening to connect the possibilities of blockchain. Though it is not rarely highlighted that blockchain could, in concept, totally substitute financial mediators, these establishments are presently pushing much of their vigor into refining current financial schemes and services via the usage of blockchain smart contract technology. To deliver a valuation of the influence of blockchain on the business, particular cases of blockchain use cases in financial division are analyzed in the following section.

Currently, payment processes comprise numerous mediators such as financial institutions, treasuries, and central core banks are resource exhaustive. In addition, due to numerous mediators and diverse systems as well as motives linked with management and cost, clearance processes do not happen incessantly, but only a limited time during a day, instigating obvious time adjournments. Hypothetically, blockchains have the latent to eradicate time and cost shortcomings.

The finance sector is principally concentrating on global transfers, in which predominantly high charges are presently experienced. Other than this, rapid payment times would decrease the forex peril possible in worldwide transactions. Also, blockchain-reliant payment systems can raise safety and confidentiality, as payments are built on the push mechanism, that is, clients can eagerly start transactions without giving information such as bank data. Rewards for dealers may comprise the protection from scam because transaction irreversibility, which is a characteristic in blockchain smart contracts. Besides, there are little transaction fees as well as price and risk reduction since client expense data does not have to be kept.

Because transaction execution in stock market trading includes a large number of stakeholders, data has to incessantly be acquiescent and copied as part of authentication processes, consequential in high costs, extended transaction duration, and operational perils.

Hence, a hopeful arena of the use cases for blockchain is observed in the clearing of transactions. By means of a blockchain, controls could meaningfully reduce the cost and intricacies of transaction management and decrease the execution time. Reducing the time span decreases both operative and counterparty peril, possibly dropping the capital necessities for financial institution. The loan and liquidity peril could successfully be eradicated, as the working of blockchain systems include transaction being subjected to the previous ownership of agreeing funds.

Blockchain also permits for receipts of independent transactions in combination with the use of smart contracts. Though paper statements must be verified, accepted, and advanced with the help of long physical processes form more and more invoicing in the business to business part today, blockchains safeguard the contract clauses such as SLA, and smart contracts manage the contract implementation. The transaction related to the performance job can then be routinely activated. The transaction validation is also kept in the blockchain. These programmed transactions that are independent from the invoicing process are known as smart payments.

Particularly in the financial segment, excessive possibilities for blockchain are also observed in the field of compliance. In this background, two conceivable usages of the blockchain are being deliberated, which are as follows:

1. As a central record for combined accounting.

2. As an association blockchain for client information.

Financial institutions such as banks currently uphold diverse account records for various drives and instrument various actions to stop accounting delinquency. This characteristically includes executing numerous information truthfulness processes and distributing the accountability for comprising financial information in the account records. By using blockchain perceptions, these processes can mainly be computerized since blockchain allows the trusted merging of specific account records into a single data model. The evasion of the double-spending challenge in blockchain systems is particularly convenient in this respect. Management in accounting, like the changing dates of contracts to other time, can be prohibited by following the irreversibility and consistent timestamp of communications between nodes.

The accomplishment of numerous rules and guidelines for money laundering problems, such as fulfillment of KYC, includes high costs for banks and adjourn transactions, occasionally authoritatively. In addition, KYC developments are achieved separately in diverse financial establishments. A business wide client archive grounded on a blockchain system could eradicate the numerous expenditures of KYC evaluations and enable the encoded broadcast of client information. In amalgamation with the usage of smart contracts, numerous features could also be automated.

7.13 | Media and Entertainment

Bitcoin like cryptocurrency delivers methods to alternate payment and reward models and provides energy to imaginations in the media and entertainment (M&E) business. In the era of translucent data torrents, the ubiquitous metadata confusion in the M&E business, with its disjointed privileges, appears to be the initiating point for concentrating on blockchain. Considering all the ineffective efforts so far to familiarize unvarying recording and licensing ethics within the business, numerous shareholders are trusting principally on the promises of this technology.

Disparate in the traditional source pasts, even the minimum slices of the media disseminated in the market are delivered with IPs by many distinct applicants that make tracking and genuine authenticity almost dreadful for the applicants themselves.

In all deliberations of the market applicants for an early use of blockchain-based smart contracts, the not clear license streams in the M&E business models works as a predominantly large part. In the circumstance of the main M&E market contributors, owing to unknown rights owners, some percent of M&E license certificates hold their intermediate mediator roles between customer and maker. This contrasts with the condition in the financial business, where expenditures go into government reserves in a way to push invoice correctly and which is always preserved.

To use blockchain knowledge as a substitute solution for all performers and, on the top of that for a worldwide, transparent, and appropriate payment of imaginative persons, a rigorous conversation among all important contributors and shareholders and their agreement with each other is mandatory. The aims of this exchange are price investigation, the growth of a roadmap for shareholder partnership on standards, and the execution of social influence negotiations to provision the technology in addition to comprehend likely controllers and governing agendas. In this procedure, smart contracts can help companies to retain an eye on the most significant revolution potential of blockchain smart contracts – dropping the need for mediators to separately implement transactions. This translates to the following:

1. Creating contracts in a public platform with an assurance to execute them based on jointly settled terms and with a restricted number of obligatory countermeasures.

2. Removing the necessity for support in executing license transactions.

3. Dropping peril due to distrust of the properties or responsibilities of the parties.

4. Eliminating intermediate mediator roles and their possible conflict in the determination of present and developing digital transaction models.

The fruitful usage of the blockchain potentials, on top of that, the formation of distributed, transparent records for information of data such as license certificate, digital content, etc., which is mutually agreed by numerous associates. This happens on the foundation of the concepts of all associations of those who have write permission to all contributors and their mutually agreed distributed transactions. The licensing system is measured and maintained by a few numbers of sanctioned and approved market contributors and functions in a totally transparent fashion.

7.14 Public Services

In case of public services, smart contract-based blockchain technology is both a hazard and a prospect. The digital services of citizen management service have so far been considered by fast-tracking present processes or converting them more effectively. Blockchain technology enhances a novel side by substituting government planned executions with confidentially prearranged ones. At this situation, the usage of blockchain smart contract technology delivers the possibilities to reinforce transparency and reliability in public governance processes. Blockchain also delivers the chance to streamline processes for interdepartment governance communication, particularly for managerial processes along several levels.

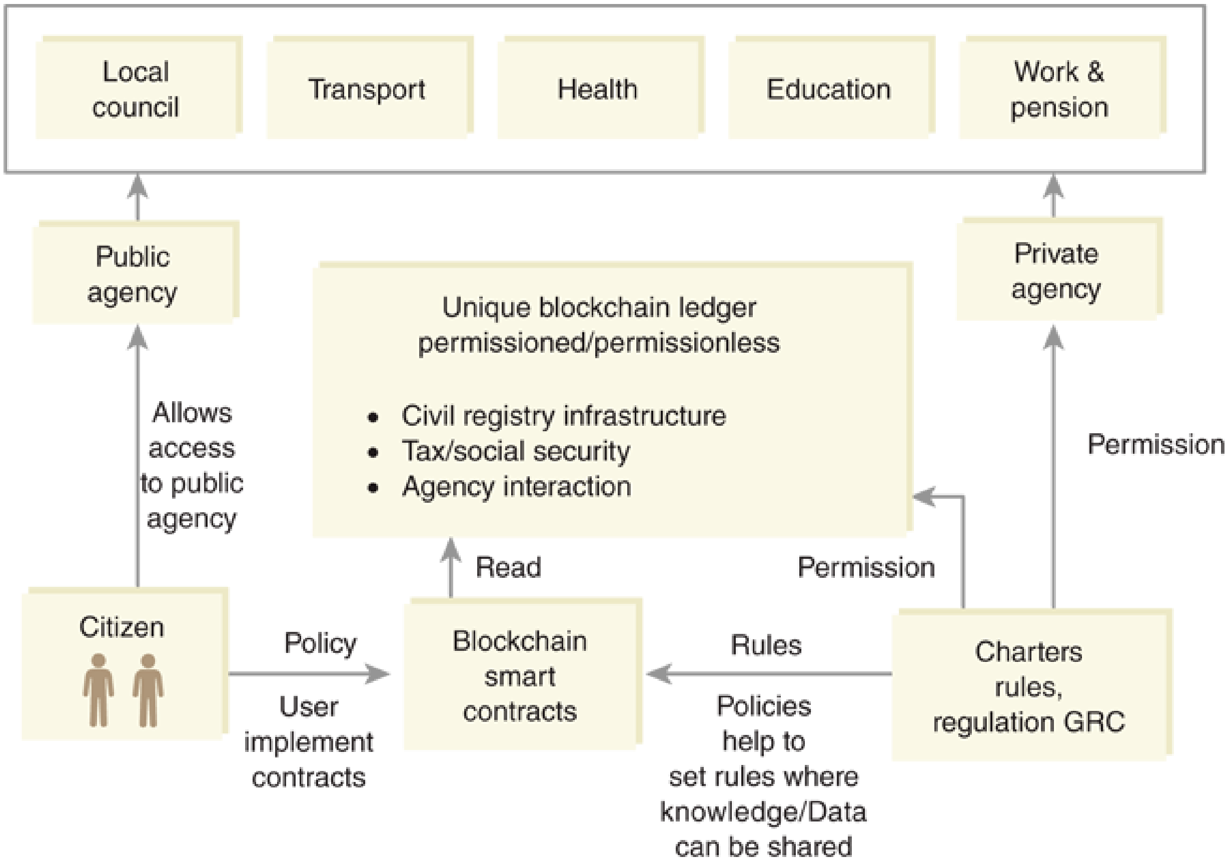

Local council
Transport
Health
Education
Work & pension
Public agency
Unique blockchain ledger permissioned/permissionless
• Civil registry infrastructure
• Tax/social security
• Agency interaction
Private agency
Allows access to public agency
Permission
Read
Permission
Citizen
Policy
User implement contracts
Blockchain smart contracts
Rules
Policies help to set rules where knowledge/Data can be shared
Charters rules, regulation GRC

Figure 7.2 Smart contract use cases in public service.

Nowadays, in numerous scenarios, performers of public services play the character of mediators. The public management preserves records to document possession constructs, while solicitors/notaries via their distinct situation of reliability confirm ownership assignments. In addition, the government body assists in many circumstances as a reliable trusted partner such as when it comes to authorizing individualities of people or belongings or validating the legitimacy of papers. Figure 7.2 shows the smart contract use cases in public service.

Therefore, various possible use cases in the public service are being deliberated these days in citizen charters. The range of the present conversation include the following:

Fact Alert

Electronic voting (e-voting) uses electronic means to either aid casting and counting votes and may use standalone electronic voting machines or computers connected to the Internet.

1. e-Payment.
2. e-Governance.
3. Public ownership accounts.
4. Origin guarantees.
5. Authentication and validation services.
6. Digital identities.
7. e-Voting.

These are frequently theoretical deliberations or archetypes. However, there are also instances in which the smart contract technology has been in fruitful use for many years. One instance can be that one can guard the truthfulness of medical information of citizens with blockchain-like technology. The states can deliver the blockchain-based emergency service lines with e-governance program.

The understandable use case of e-payment is just as small as it is easy to execute. Many countries started to accept Bitcoin as a payment mode for their administrative services. The other zones of use cases are more realistic but related with more multifaceted process alterations. Blockchain is stated with specific occurrence in the background of records and the handover of ownership. Its capability to register transactions authenticity, transparency and immutably originates very near to the necessities of traditional registry administration. The motives for using blockchain technology in this situation may differ. It is stimulating for those areas where traditional government constructions for registering or belief in them are missing. Wherever government developments are recognized, smart contracts can make the ownership handover development even more clear and, if needed, swifter.

Other than this, the smart contract can also be utilized for interdepartment collaboration, such as to validate whether specific information or papers are deposited at an organizational division or not. Also, it is conceivable to safe the truthfulness of data and papers that, at a minimum from a handler standpoint, can be a trivial alternate to digital signatures.

Technical Stuff

Blind signature is a digital signature and secret-key encryption protocol that has the mathematical property of commutativity.

Elections and e-voting is also another use case. Usually, in such a background, each elector obtains a token that characterizes his ballot. Each aspirant gets a receipt of the address such as the digital ballot. The voting itself is characterized by a business of the token to the aspirant's receipt address. For the arena of governmental elections, though, conversation regarding the use of high-tech supports is debated too much.

In several states, the public management has been experiencing thoughtful fluctuations for a number of years, owing to the growing number of activities joined with improved necessities and limited finances. For many years, digital transformation has been realized as a way out of this predicament. Because of its decentralized characteristics, blockchain technology consequently delivers a thought-provoking viewpoint for the national structure of secretarial areas.

Several globally debated use case situations in the public services are convoyed by a new numeral problem. Developing and executing a blockchain require knowledgeable specialists, together with cryptography and IT researchers. Other than the diverse open practical facts, essential queries should also be present. Characteristic mediators generate trust through administrative actions. Mediators in the public services are focused to distinct necessities with respect to accuracy and reliability. Blockchain substitutes this administrative faith with an assurance in knowledge and its cryptography-based transactions. In this background, it has to be measured in each occasion whether the usage of blockchain is sensible and maintainable for a longer period of time.

7.15 | Legal Services

Blockchain can essentially contest the legitimate entitlement of law, while initiating new resources of law implementation. The decentralized and typically autonomous planning of the networks performs the conclusive character in such a procedure.

Most of the legal scheme is systematized in a technically neutral fashion. The datum that a specific transaction is held using smart legal contracts typically does not disturb the legal characteristic of the business. However, there are also omissions to this opinion.

Usually, the autonomous blockchain networks function worldwide and willingly permit interborder dealings. United with autonomous buildings, this frequently develops the classical law implementation method, which is practically dreadful.

Traditional governing law assumes that it is likely to outline an exact addressee and, if required, to be capable to get a grip of him/her. This statement is completely positioned in query by both the decentralization and anonymous autonomous nature of blockchain. It is consequently essential to describe new methods to efficient law implementation.

Other circumstances apply to closed systems. The enforcement of legal principles and standards is easier there. The appropriate gatekeepers are suitable regulatory addressees. Traditional approaches to regulatory law can be used without further ado in this extent.

The automation of contract management has been discussed since mid-1990s under the heading of smart contracts. Contrary to the term, smart contracts are not contracts in the legal sense, but rather the linking of contracts with reality. Certain features of blockchain technology can now prove to be very useful in automating this performance. These are as follows:

1. The decentralized validation of transactions allows for an automated contract execution on a peer-to-peer basis.

2. Blockchain technology makes it possible to embed values in the form of tokens directly into a contract, guaranteeing the contract execution without any credit risk.

3. Blockchain technology can automate the execution of contracts so that subsequent unilateral changes to the process are no longer possible.

4. Blockchain is credited with the potential to dramatically lower transaction costs, particularly through the elimination of intermediaries. This creates potential for new contracts and types of contracts that would have been unviable to date, especially in the context of micropayments. However, in all of this, it is important to note that the automation of law has (and must have) limits. Value decisions cannot be replaced by the technology, and it is important that property rights are not circumvented.

In addition to individual contractual relationships, some relationships in the field of corporate law can also be settled on the basis of blockchain. So-called DAOs are based on this idea. Tokens are used as voting rights within the "company". Innovative organizational forms and financing fundamentals can economically be stimulating. However, the resulting issues in terms of society and the law of obligations are still largely unexplained.

This becomes particularly clear in consideration of open systems. On one hand, the database and trust in it are precisely based on the transparency of all the transactions. On the other hand, the technology is based on the use of pseudonyms and therefore ultimately incorporates the idea of privacy-by-design. Conflicts between the blockchain approach and general data protection regulation may particularly arise with regard to accountability for data processing and the right to be "forgotten".

In order to foster innovation, it is possible to define navigating sandboxes, although the general regulatory framework necessary for such a purpose would create market access barriers that would be too high, without incurring irresponsible dangers. Therefore, it is important to ensure that appropriate due diligence measures are applied when predefined thresholds are exceeded.

In the present time, the policy of leaving things to take their own course will not work. Transformation to the lawful system will be compulsory, both to stop threats as well as to permit added novelty.

On one hand, the robust swift development of discrete utilities may affect in new systemic problems, along the legal law system. The liberty essential for the growth of the technology consequently must be convoyed by funds in an intensive care and the growth of pressure tests. This also predominantly spreads over in relation with smart contracts.

On the other hand, modifications to the governing framework are required in order to make modernization conceivable. One understandable instance apprehends recognized necessities, such as blockchain has latent power in the field of transaction authentication. However, the law is not precisely scientifically neutral in this aspect. The latent power is consequently dependent upon the acknowledgement of the technology as an appropriate method of the particular business. Governing limitation is not enough to encourage the novelty in this admiration.

7.16 Darknet

In the relationships of social fight of benefits, Darknet as the arena of use case of blockchain smart contracts and of the cryptocurrencies it allows to position amid the unrestricted, independent, free, and unseen sharing of data, goods, and services in the benefits of law implementation.

Today, all of us work on the indexed WWW, which is reachable to the public by using search engines. Another form of the Internet also exists now. In this community, some are considered to simplify the message, information exchange, and business transactions that are problematic or terrible to comprehend. These are frequently described today under the concept of Darknet. These are networks in the Internet that are also utilized for unlawful drives. These comprise markets in many mediums in the open source anonymous communication such as tor network, where linking data is unidentified, as well as sharing medium for software and broadcasting.

Technical Stuff

Darknet is an umbrella term which describes the portions of the Internet purposefully not open to public view. They are hidden networks whose architecture is superimposed on that of the Internet.

The Darknet has increased in significance. It delivers an apparently rule-free medium that is not directly available from the visible Internet, which delivers platform for radical communications, illegal philosophies, and unlawful trading and which also helps as a resource of message and communication. As an outcome, the Darknet has shaped an ecosystem that is very striking to extremists and offenders due to the potential of secrecy and no tracking. It is significant to not oversee the circumstance that there are, in fact, plenty of lawful and observable forms of utilization in the Darknet, which solely follows the feature of independent and unnoticed sharing of data and which even delivers defense from maltreatment in exploitive and autocratic arrangements.

The cryptocurrencies popular today practices databases as blockchains. This blockchain is continuously observed to every exchange companion. As we already know, objects amid which Bitcoins are exchanged are termed as wallets. These are not knotted to the individuality of a being per se and can be created in large number so that associates of a Bitcoin deal can generally endure anonymity. Therefore, Bitcoin is also utilized as the money for steering unlawful jobs in the Darknet.

Law implementation bodies have a curiosity in spotting unlawful items in the Darknet. The goal is to discover to what degree the legal compliant thought of Bitcoin transactions of the recognized exchange associates such as wallets. The legitimately compliant set of information on interchange settings in the Darknet license additional decisions concerning the following:

1. Characteristics of the deal.
2. The dealt service and goods.

3. Supplementary data about the individualities of the specific exchange associates.

7.17 The Future

We need to reach there. A blockchain with enough interrelationships among financial network and properties records does not yet occur and might never occur. Even if it is possible, smart contracts are yet to face limits.

It is an inevitable handling of inputs to generate a predictable output. The inevitability of a smart contract is consequently perceived as an important benefit. If the code executes, there should be no misinterpretation as to its meaning and it can be simulated continually, and efficiently, for equivalent transactions.

The disparity among the inevitability of code and the uncertainty of language may occasionally be a myth. It can be stated that code at no time can be incorrect, but it can have unintentional penalties, and where there are accidental penalties, there is obligation. However, it is difficult to accept the danger of such inadvertent penalties.

Numerous attorneys would also propose that, occasionally, thoughtful uncertainty and the capability to meet the requirements can be important parameters.

An interrelated blockchain and a public code language for writing smart contracts that implement transaction across that blockchain would generate race in a variety of fields.

New applicants could communicate with public ledgers in a way before restricted to the authorized administrators, and isolated (private) transactions among entities would be less dependent on mediator arbitraries. It would also increase the capability of revolutionary financial technology organizations to communicate with the network financial system.

But is it genuine to imagine conservative financial establishments to escort in such a new stage of development of the revolutionary race condition?

SUMMARY

Although smart contracts cannot critic if a state has been judiciously encountered, or if a file has been agreeably delivered, there is substantial latent power in generating standardized procedures for reiterated transactions, where the agreed transferred contracts are not the standard.

In the financial ecosphere, such conventions may be established through the implementation of derived business across a distributed database-based record account ledger.

In the real-world scenario, one can envision a deal for the entity among two isolated vendors. Once the price is set, they can practice a public standard (with a nominal fee) smart contract to give the price and the entity record-keeping particulars. The smart contact will run the contract authenticating that the vendor is the right proprietor of the entity, depositing the money, and recording the registration of the entity in the name of a buyer.

Eventually, smart contracts are rational procedures reliant on the blockchain nodes with whom they communicate.

With the acceptance of blockchain technology in its beginning, smart contracts are not presently skilled of communicating with the variety of systems essential to begin them as the defacto choice for the implementation of contracts over the current closed schemes.

This will transform when the public ledgers and financial infrastructure will be connected and will start to exchange information freely. Also, the probable phase where smart contracts systematize the implementation of agreements, in specific to those areas with no obligation for individual human interference.

SHORT ANSWER QUESTIONS

1. What is trust in security?
2. How does a smart contract work?
3. What is a smart contract in blockchain?
4. Are smart contracts legally binding?
5. How are smart contracts stored on blockchain?
6. Where can smart contracts be used?
7. Why does blockchain need a smart contract?
8. What is a smart contract in Hyperledger?
9. Which is the first smart contract platform?
10. What are the benefits of smart contracts?

LONG ANSWER QUESTIONS

1. Consider two use cases, one in finance and the other in media and entertainment. Demonstrate the effective use of smart contract in these cases.
2. Explain the advantages and disadvantages of centralized and decentralized smart contracts.

CHAPTER 8

Bitcoins

Learning Objectives

After reading this chapter, you will be able to:

- Describe the working of Bitcoin.
- Define Merkle trees.
- Identify the structure of Bitcoin block.
- Understand Bitcoin address, transaction, network, wallet, and payments.
- Discuss about the Bitcoin client and Bitcoin supply.

8.1 Introduction

Much before the proliferation of Internet, several attempts were made by many people to have digital cash technologies in place, such as ecash protocols in the 1980s, that were developed by David Chaum (Chaum 1983; Chaum 1997) and Stefan Brands, which could not take up for the wider circulation. The first proposals for the distributed digital scarcity-based cryptocurrencies were b-money by Wei Dai (Dai W 1998) and bit gold by Nick Szabo (Szabo 2013).

But it was Bitcoin which got huge popularity as a digital and global currency system that allows people to send or receive money across Internet without being linked to a real identity. It is a grey area as to who invented Bitcoin. It was in August 2008, when the domain name bitcoin.org was registered. Later, in the same year on 31 October, a link to an article authored by Satoshi Nakamoto titled “Bitcoin: A peer-to-peer electronic cash system” was posted to a cryptography mailing list. This article described a system for electronic transactions without relying on a trusted third party. On 3 January 2009, Satoshi Nakamoto mined the genesis block (block number 0) of Bitcoin, giving rise to Bitcoin network having a reward of 50 Bitcoins. Hosted at SourceForge, the first open source Bitcoin client was released on 9 January 2009. Hal Finney, a programmer, was one of the first supporter, adopter, contributor to Bitcoin and receiver of the first Bitcoin transaction. He was the first person to download Bitcoin software the day it was released and received 10 bitcoins from Nakamoto in the world’s first Bitcoin transaction on 12

January 2009. Nakamoto, before disappearing into the virtual world, handed over reins to developer Gavin Andresen, who then became the Bitcoin lead developer at the Bitcoin Foundation.

Fact Alert

SourceForge is a web-based service that offers software developers a centralized online location to control and manage free and open-source software projects.

Bitcoin blockchain is developed as a public ledger that records Bitcoin transactions. In this chain of blocks, each block contains a hash of the previous block up to the genesis block of the chain. Communicating nodes form a network, where each node runs Bitcoin software and maintains the blockchain. In addition, network nodes can also validate transactions, append them in their ledger, and then broadcast this updated ledger to other nodes. In order to achieve independent verification of the chain, every node stores its own copy of the blockchain. Roughly every 10 minutes, a new group of accepted transactions, known as *block*, is created without requiring central oversight. These are added to the blockchain and published to all nodes. To prevent double spending, Bitcoin software determines when a particular Bitcoin was spent and is recorded in a conventional ledger at the node. These nodes are also known as *miners*, which are motivated by rewards in the form of releasing new Bitcoin and transaction fees. These miners are decentralized authority enforcing the credibility of Bitcoin network. Bitcoin is the unit of account in Bitcoin system with unicode character ₿. Because of the high valuation of Bitcoin, it was proposed to have smaller denomination that are termed as millibitcoin (mBTC) and satoshi (sat), and till date these are the smallest fraction amount within Bitcoin. 1 mBTC represents 0.00000001 bitcoins, that is, one hundred millionth of a Bitcoin. Let us now try to understand the legal status of Bitcoin.

Technical Stuff

Unicode is a computing industry standard for consistent encoding, representation, and handling of text expressed in most of the world's writing systems. The standard is maintained by the Unicode Consortium, and as of May 2019 the most recent version, Unicode 12.1, contains a repertoire of 137,994 characters covering 150 modern and historic scripts, as well as multiple symbol sets and emoji.

The year 2013 was very active from the perspective of policy and guidelines on virtual currencies. A report regarding centralized and decentralized "virtual currencies" and their legal status within "money services business" and Bank Secrecy Act regulations was issued by the bureau of the United States Department of Treasury. This report classified digital currencies and other digital payment systems such as Bitcoin as "virtual currencies" as they are not legal tender under any sovereign jurisdiction. However, they gave a decision that would need Bitcoin exchanges. This provides a platform where Bitcoins are traded for traditional currencies to disclose large transactions and suspicious activity, comply with money-laundering regulations, and collect information about their customers as the traditional financial institutions are required to do. The US Treasury subsequently extended its antimoney laundering regulations to include the processors of Bitcoin transactions. This was followed by the industry group committee for the Establishment of the

Digital Asset Transfer Authority, to work with regulators and policymakers for adapting the existing currency requirements to the digital currency along with business models and risk management standards.

8.2 Working of Bitcoin

Satoshi Nakamoto defined Bitcoin as a chain of digital signatures. In this chain, each owner of the currency has a pair of keys, one public and one private, that are stored locally in a file.

A chain of Bitcoin transactions depicts simple illustration of a chain of transactions from one node to another. The Bitcoin represented in Figure 8.1 is the same but at different points in time. In order to initiate the transaction, future owner P1 has to first send his/her public key to the original owner P0. This owner transfers Bitcoins by digitally signing a hash of the previous transaction and the public key of the future owner. Every single Bitcoin contains complete history of all its transactions in past, and change in the ownership becomes part of the code. However, because of its storage design, it only allows for the new owner to spend it. All the signed transactions are sent to the public network without disclosing the involved parties. To avoid double spending without intermediary, validating the transactions is achieved by implementing online timestamping solution. It is one of the first digital currencies using peer-to-peer technology for payments, and it also consists “miners”. These miners could be independent individuals and/or companies who own the governing computing power and participate in the Bitcoin network. These are decentralized authority enforcing the credibility and motivated by rewards (by means of releasing new Bitcoin) and transaction fees paid. The miners used to ensure that a series of data have existed and not been altered since a specific point in time, in order to get into the hash. In its hash, every timestamp includes the previous timestamp, thus forming a chain of ownership. The network can verify the transactions once broadcasting the new transactions takes place.

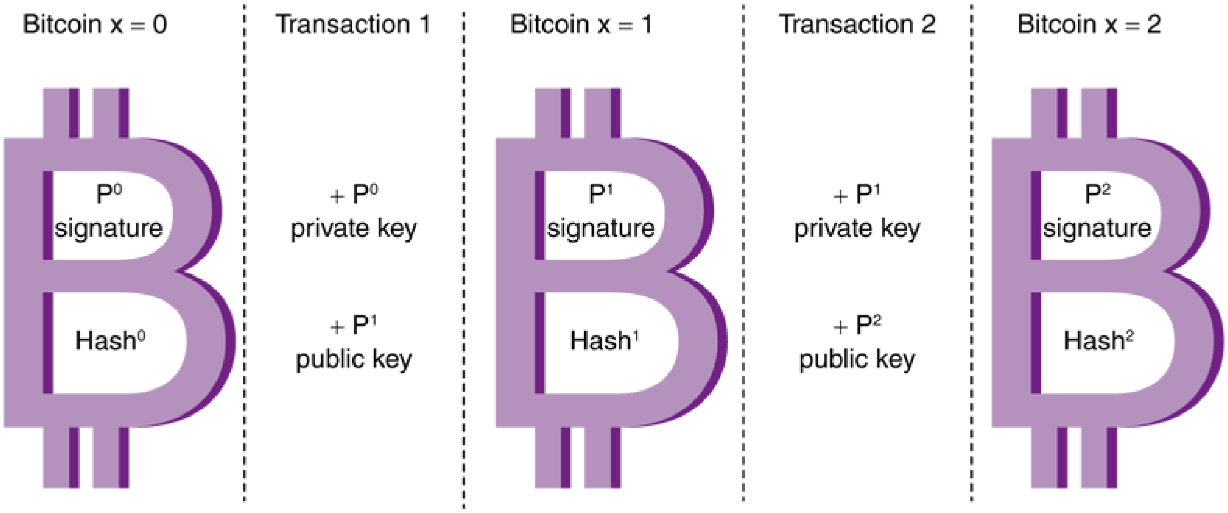


Figure 8.1 Chain of Bitcoin transactions.

Technical Stuff

An application-specific integrated circuit (ASIC) is an integrated circuit (IC) chip customized for a particular use, rather than intended for general-purpose use.

Thus, "mining" is the process of validating transactions by using computing power to find valid blocks (based on proof-of-work). Please refer to Chapter 4 for more on this. Furthermore, it is the only way to create new money in the Bitcoin scheme. The miners are rewarded with new Bitcoin at a fixed, but periodically declining rate; however, the total supply of Bitcoin is approaching 21 million. Basically, it involves solving a computationally difficult puzzle to discover a new block, which is added to the blockchain and receives a reward in the form of a few Bitcoins. In 2009, the block reward was 50 new Bitcoins and it decreases every four years. The difficulty of the mining process increases as more Bitcoins are created. The mining difficulty began by Satoshi Nakamoto at 1.0 with Bitcoin's debut back in 2009 and by year end it was only 1.18. However, as of February 2019, the mining difficulty is over 6.06 billion. Initially, an ordinary desktop computer sufficed for the mining process; however, now to combat the difficulty level, miners have started using faster hardware such as application-specific integrated circuits typically designed for a particular type of computation and also using more advanced processing units such as graphic processing units.

Technical Stuff

A graphics processing unit (GPU) is a specialized electronic circuit designed to rapidly manipulate and alter memory to accelerate the creation of images in a frame buffer intended for output to a display device.

8.3 Merkle Trees

There are a few building blocks for Bitcoin – Hash Function, Digital Signature, TimeStamp, and Merkle tree. In Chapters 3 and 6, we have discussed the first three in detail. Now, we will elaborate on the Merkle tree, which is considered to be an important element in Bitcoin. Proposed by Ralph Merkle (Merkle 1998), it is a binary tree of hashes that is used to verify data integrity efficiently and securely. Figure 8.2 depicts a Merkle tree.

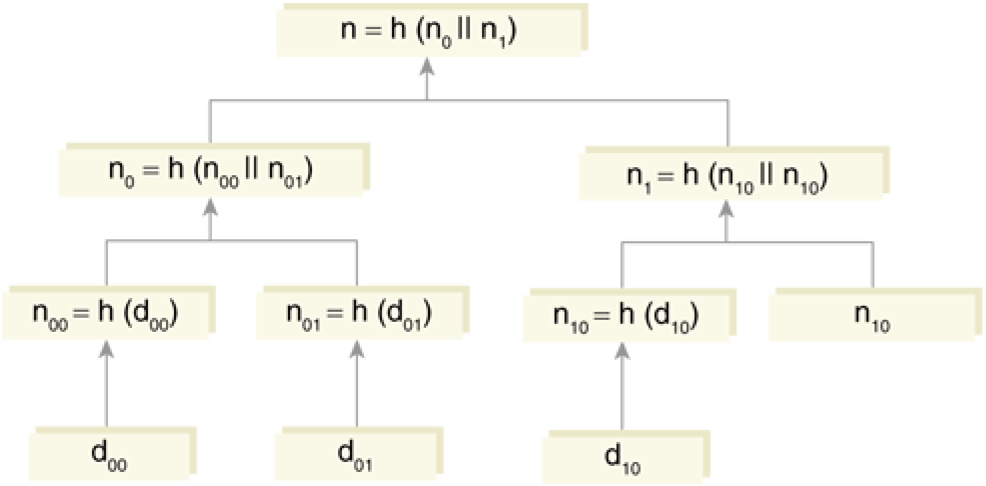


Figure 8.2 Merkle tree.

In this tree, every non-leaf node is the hash value of its concatenated child nodes, while the leaves are computed over data blocks. The final hash value of the root node is known as *Merkle root*, which is stored in the block header. Since the non-leaf node is expected to have two child nodes, the missing child node is a special case of Merkle tree, just like child node n_1. The solution in Bitcoin is straightforward in this kind of situation. When forming the nodes in a row, if there are an odd number of nodes, the last node will be duplicated.

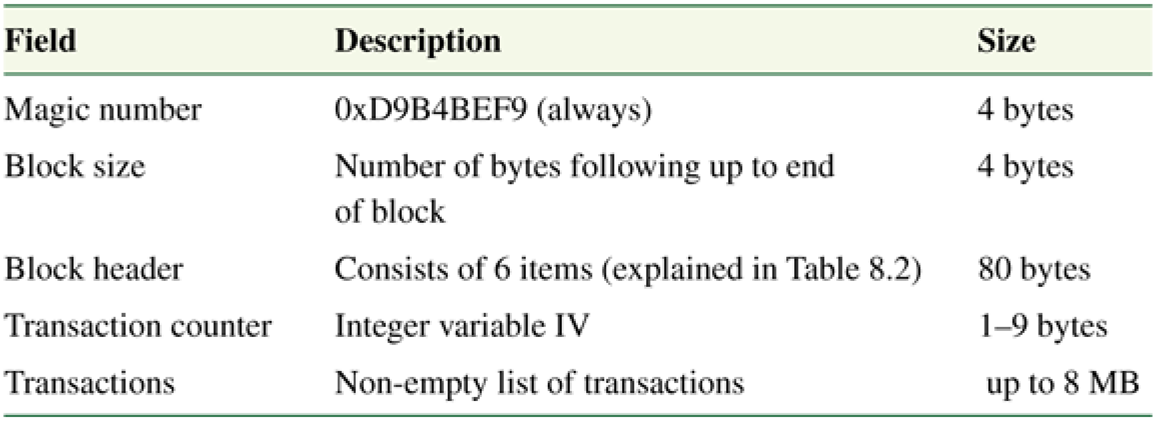
One of the strengths of Merkle tree is that there is no requirement of recomputing the hash of all data if one data block changes. For example, if the block d_{00} changed, only one branch from n_{00}, n_0, and finally root n will be recomputed. It requires a much smaller number of hash computations and makes the process more efficient, which makes it as an important building block in Bitcoin.

8.4 Bitcoin Block Structure

Bitcoin block holds recent transaction contents, proof-of-work and reference to the hash of previous block. A Bitcoin block can be considered as a container data structure that contains more than 500 transactions on an average. Average size of a block is approximately 1 MB and can go up to 8 MB, which enables more transactions to be processed per second. Transaction data is permanently recorded in these blocks. Blocks are organized into a linear sequence or chain over time by adding new transactions which are constantly being processed by miners. A block is composed of a header and a long list of transactions, and its structure is given in Table 8.1.

Among other things, each block also contains the current time, a record of some or all recent transactions, and a reference to the block that came immediately before it. It also contains a unique answer to a difficult-to-solve mathematical puzzle. New blocks will be added only when the correct answer is found. The process of "mining" is essentially the process to solve the current challenge in order to find the answer that "solves" the current block. The mathematical challenge in each block is very difficult to solve. However, once a valid solution is obtained, it is relatively very easy for the rest of the network to confirm that the solution is correct. Theoretically and practically, there can be multiple valid solutions for any given block. However, only one of the solutions needs to be found for the block to be solved. If two miners arrive at two different valid solutions for the same block at the same time, then the peer-to-peer network is designed in such a fashion that resolves these splits within a short span of time. This facilitates for only one branch of the chain to survive and continue. Subsequently, the client accepts the longest chain of blocks as valid chain. The length of blockchain refers to the chain with the most combined difficulty and not the one with the most number of blocks. This prevents someone from creating a large number of low-difficulty blocks, and forking the chain in their favor.

Table 8.1 **Block structure**



Field	Description	Size
Magic number	0xD9B4BEF9 (always)	4 bytes
Block size	Number of bytes following up to end of block	4 bytes
Block header	Consists of 6 items (explained in Table 8.2)	80 bytes
Transaction counter	Integer variable IV	1–9 bytes
Transactions	Non-empty list of transactions	up to 8 MB

Flash Quiz

A man lives in a house whose number is the reverse of his brother's house number. The difference between the house numbers ends in 2. What are the lowest possible numbers of their houses?

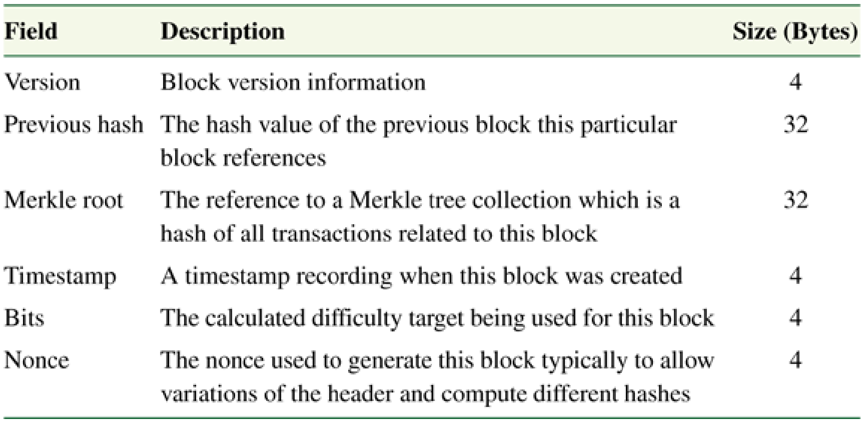
Now, let us discuss the block header that contains metadata of a block with three different sets in six fields:

1. Hash of the previous block.
2. Merkle tree root.
3. Mining challenge.

The block header structure is shown in Table 8.2.

When a node creates an outgoing connection, it will immediately advertise its version in the first field otherwise used the current version. *Previous hash* is the hash value of the previous block header. *Merkle root*, as explained before, are binary trees of hashes. In Bitcoin, it is used to verify transaction integrity by twice SHA-256, that is, computing SHA-256 hash of the computed SHA-256 hash of some transaction contents. *Timestamp* is the time when the block was generated. *Bits* designate the difficulty of the proof-of-work, and different *Nonces* are tried to meet Bitcoin proof-of-work requirements.

Table 8.2 **Block header structure**



Field	Description	Size (Bytes)
Version	Block version information	4
Previous hash	The hash value of the previous block this particular block references	32
Merkle root	The reference to a Merkle tree collection which is a hash of all transactions related to this block	32
Timestamp	A timestamp recording when this block was created	4
Bits	The calculated difficulty target being used for this block	4
Nonce	The nonce used to generate this block typically to allow variations of the header and compute different hashes	4

These series of blocks form a blockchain, which is a shared public ledger. The sequence of the blocks in the blockchain is based on the time that the transactions confirm, and every block is built on top of a previous block.

8.5 Bitcoin Address

Like e-mail addresses, Bitcoins can be sent to a person having a Bitcoin address. However, unlike an e-mail address that receives many e-mails from several senders, people have different Bitcoin addresses; and a unique address should be used for each transaction. A Bitcoin address is an identifier of 26–35 letters and numbers, it begins with the number 1 or 3 or bc1 and is case sensitive. The address represents a virtual destination for a Bitcoin payment. There is no cost associated with generation of Bitcoin addresses; for example, using the desktop client of “Bitcoin Core”, one click on “New Address” will assign an address that was developed by Wladimir van der Laan based on the code by Satoshi Nakamoto. One can also generate a Bitcoin address using an account at an exchange or online wallet service. Addresses can be created even without an Internet connection; further it does not require any contact or registration with Bitcoin network. It is possible and recommended to create large batches of addresses offline with freely available software utilities such as Bitcoin core. Generating batches of addresses is useful in multiple scenarios such as e-commerce websites, where a unique pregenerated address is dispensed to each customer who chooses to pay with Bitcoin.

Currently, there are three address formats in use:

1. P2PKH that begins with number 1, for example: 1BMSEYstWetqTFn5Au4m4GFg7xJaNVBvN2.
2. P2SH type starting with number 3, for example: 38t1WpEZ73CNmQviecrnyiWrnqRhWN J9Ly.
3. Bech32 type is case insensitive and starts with bc1, for example: bc1rPsrrr7xfkvy5l643lydnw9re59gtzzwf5mqadq.

Most Bitcoin addresses consist of random digits, with uppercase and lowercase letters. A Bitcoin address uses a base58 encoding, consisting of numbers and alphabets of characters 0 … 9, A … Z, and a … z, but without these four characters: alphabet “I” (uppercase “eye”), alphabet “l” (lowercase “ell”, alphabet “O” (uppercase “oh”), and numeric “0” (number zero). This is to prevent visual ambiguity. Moreover, to avoid any typographical errors, which need to be automatically identified and rejected, some of the characters inside the Bitcoin address are used as a checksum. With this encoding, a Bitcoin address encodes 25 bytes: the first byte is the version number, which will be zero; the next 20 bytes are RIPEMD-160 digest that can be considered pure arbitrary data; the last 4 bytes are treated as checksum digits. These are the first 4 bytes of a two times SHA-256 digest of the previous 21 bytes. With this in order to check the Bitcoin address, the first 21 bytes are captured, the checksum is computed, and checked that it corresponds to the last 4 bytes. There are open source codes at http://rosettacode.org/wiki/Bitcoin/address_validation, which is available for performing in over 30 programming languages. There is also an online tool available at http://lenschulwitz.com/base58.

Technical Stuff

In computer science, Base64 is a group of binary-to-text encoding schemes that represent binary data in an ASCII string format by translating it into a radix-64 representation.

The reuse of the same address for multiple transactions is discouraged as reusing allows others to reliably determine that the address being reused is an already-known subject. If it is reused, it is considered to be abusing the privacy and security of the participants of the transactions as well as the future holders of Bitcoin. Therefore, after the received coins have been spent, the address should not be used again. It has been argued that the phrase “bitcoin address” was a bad name and a better name such as “bitcoin invoice” was suggested in the community.

Addresses in Bitcoin are neither accounts nor wallets. They do not carry balances and are being used only to receive funds. Bitcoin never uses the field send “from” an address at any time. Many times users are fooled by various confusing services and software display Bitcoins received with an address, minus Bitcoins sent in random unrelated transactions as an “address balance”. This number is meaningless as it does not imply that the recipient of Bitcoins sent to the address has spent them, or that they have Bitcoins received.

It is hard to tell who owns the address unless someone opts to link their name in Bitcoin. Bitcoin by design does not keep track of users; it keeps track of addresses where the money is. Each address has two important pieces of cryptographic information in the form of keys: a public key and a private key. The “bitcoin address” is created from the public key, similar to an e-mail address. Anyone can look it and send Bitcoins to it. The private address, or private key, is similar to an e-mail password; the owner can only send Bitcoins with it. Therefore, it is very important that the private key is kept secret.

The easiest way to create a Bitcoin address is to use well-tested, open source, peer-reviewed wallet software. The technical steps involved in generating the address are presented here. It uses the same elliptic curve cryptography for generating the public and private key, as explained in Chapter 6.

Step 0: Let us consider having a private ECDSA key, the same as what we have generated in Chapter 6.

18e14a7b6a307f426a94f8114701e7c8e774e7f9a47e2c2035db29a20632172
5

Step 1: Take the corresponding public key generated with it (33 bytes, 1 byte 0x02 (*y*-coordinate is even), and 32 bytes corresponding to *x*-coordinate):

0250863ad64a87ae8a2fe83c1af1a8403cb53f53e486d8511dad8a04887e5b2
352

Step 2: SHA-256 hashing is performed on the public key:

0b7c28c9b7290c98d7438e70b3d3f7c848fbd7d1dc194ff83f4f7cc9b1378e9
8

Step 3: Perform RIPEMD-160 hashing on the above result of SHA-256:

f54a5851e9372b87810a8e60cdd2e7cfd80b6e31

Step 4: Add version byte in front of RIPEMD-160 hash (0x00 for main Bitcoin network):

00f54a5851e9372b87810a8e60cdd2e7cfd80b6e31

(As explained earlier, the following steps are the Base58Check encoding, which has multiple implementing options.)

Step 5: SHA-256 hashing is performed on the extended RIPEMD-160 result:

ad3c854da227c7e99c4abfad4ea41d71311160df2e415e713318c70d67c6b41
c

Step 6: SHA-256 hashing is performed on the result of previous SHA-256 hash:

c7f18fe8fcbed6396741e58ad259b5cb16b7fd7f041904147ba1dcffabf747fd

Step 7: The first 4 bytes of the second SHA-256 hash is considered as the address checksum:

c7f18fe8

Step 8: Add the 4 checksum bytes from step 7 at the end of extended RIPEMD-160 hash from Step 4. This finally results into a 25-byte binary Bitcoin address.

00f54a5851e9372b87810a8e60cdd2e7cfd80b6e31c7f18fe8

Step 9: This result is converted from a byte string into a base58 string using Base58Check encoding, resulting into the most commonly used Bitcoin address format.

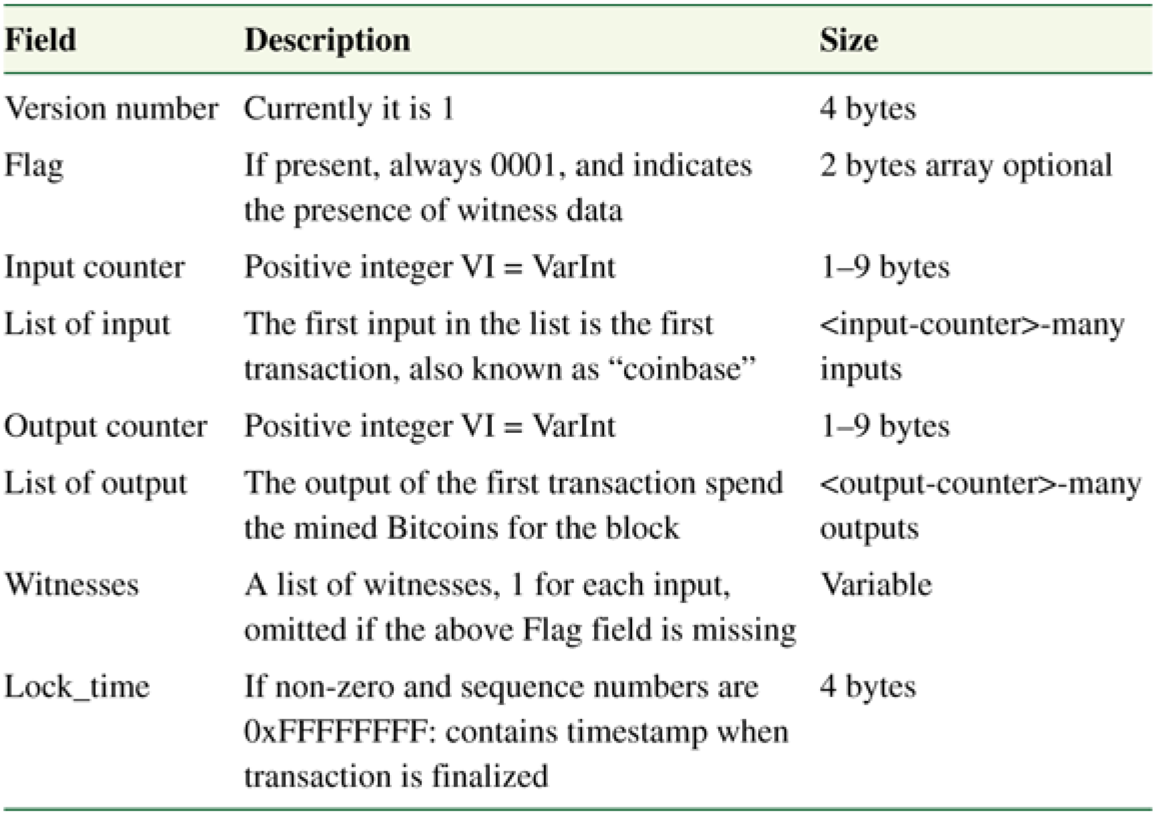
1PMycacnJaSqwwJqjawXBErnLsZ7RkXUAs

Unlike the recovery of the losses in the centralized systems, the losses in Bitcoin are usually unrecoverable. Therefore, it is not recommended to manually handle the keys (or addresses).

8.6 Bitcoin Transactions

A transaction is referred to as a transfer of Bitcoin whole or fraction value that is broadcast to the network and collected in the blocks. Any current transaction contains previous transaction output as new transaction inputs and dedicates all the input Bitcoin values to new output. No encryption is done on the transactions and therefore it is possible to browse and view every transaction as a block using any hex editor, the transaction format is depicted in Table 8.3. A blockchain browser provides human-readable terms from the Bitcoin site where every transaction included within the blockchain can be used to verify payments. In simple words, after enough confirmations from many nodes, the transaction is, said to be buried and these can be considered as immutable.

Table 8.3 **Bitcoin transaction format of a block**



Field	Description	Size
Version number	Currently it is 1	4 bytes
Flag	If present, always 0001, and indicates the presence of witness data	2 bytes array optional
Input counter	Positive integer VI = VarInt	1–9 bytes
List of input	The first input in the list is the first transaction, also known as "coinbase"	<input-counter>-many inputs
Output counter	Positive integer VI = VarInt	1–9 bytes
List of output	The output of the first transaction spend the mined Bitcoins for the block	<output-counter>-many outputs
Witnesses	A list of witnesses, 1 for each input, omitted if the above Flag field is missing	Variable
Lock_time	If non-zero and sequence numbers are 0xFFFFFFFF: contains timestamp when transaction is finalized	4 bytes

EXAMPLE 8–1

A simple Bitcoin transaction with only 1 input and only 1 output

INPUT:

```
Previous                                     tx:
f5d8ee39a430901c91a5917b9f2dc19d6d1a0e9cea205b009c
a73dd04470b9a6

Index: 0

scriptSig:
304502206e21798a42fae0e854281abd38bacd1aeed3ee3738
d9e1446618c4571d10
```

```
90db022100e2ac980643b0b82c0e88ffdfec6b64e3e6ba35e7
ba5fdd7d5d6cc8d25c6b241501
```

OUTPUT:

```
Value: 5000000000

scriptPubKey: OP_DUP OP_HASH160
404371705fa9bd789a2fcd52d2c580b65d35549d

OP_EQUALVERIFY OP_CHECKSIG
```

Let us understand this in detail. The input in this transaction imports 50 BTC from output #0 in transaction f5d8ee39a. Then the output sends 50 BTC to Bitcoin address (expressed here in hexadecimal 404371705…). When the recipient requires spending these Bitcoins, he/she will refer output #0 of this transaction in an input in his/her transaction.

8.6.1 Input

An input has a reference to an output of a previous transaction. If the current transaction value is not the same as one input to it, then several inputs are often listed in a transaction. All the values of new transaction's input are added up, that is, the total coin value of the previous output is referred by the new transaction's inputs. Then the total is computed after subtracting any transaction fee. `Previous tx` is a hash of a previous transaction while Index is the specific output in the referenced transaction.

The script contains two components, a signature part and a public key part. ScriptSig is the first half of a script. The public key as the second component is used to verify the redeemer signature that is an ECDSA signature over a hash of a simplified version of the transaction. Using the public key it proves that the transaction was created by the real owner of the Bitcoins in question. Different flags define how the transaction is simplified and can be used to create different types of payment.

8.6.2 Output

An output section contains instructions for sending Bitcoins. Value is measured as number of Satoshi (as mentioned earlier, 1 BTC = 100,000,000 Satoshi), which this output will be worth when claimed. `ScriptPubKey` is the second half of a script. When there is more than one output, they will share the combined value of the input. The entire combined input value has to be sent in an output, as each output from one transaction can only ever be referenced once by an input of a subsequent transaction. If the input is worth 50 BTC but only 25 BTC were required to be sent, then Bitcoin will create only two outputs worth 25 BTC each. In this scenario, one to the destination and one back to the current owner (sometimes known as "change", though it is being sent to self). Any input Bitcoins that are not redeemed in an output is considered as a transaction fee. Those who generate the block can claim it by inserting it into the coinbase transaction of that particular block.

Bitcoin uses a custom Forth-like scripting system to verify that inputs are authorized to collect the values of referenced output. The input's scriptSig and the referenced output's scriptPubKey are evaluated, with scriptPubKey using the values left on the stack by scriptSig. The input is authorized if scriptPubKey returns true value.

Transactions are allowed to contain multiple inputs and outputs, enabling Bitcoins to be split and combined. Normal transactions will have either a single input from a larger number of previous transactions or multiple inputs combining smaller amounts. By using Bitcoin, it is possible to design more complex types of transactions. These can be linked together into cryptographically enforced agreements known as *contracts* (this has been discussed in Chapter 7.)

8.7 Bitcoin Network

The Bitcoin network is a peer-to-peer payment network based on unambiguous cryptographic protocol. Users of this network can send and receive Bitcoins by broadcasting digitally signed transaction to the network using Bitcoin wallet (specific software). These transactions are recorded in a distributed public database with consensus achieved by proof-of-work approach. To share transactions, a minimal structure of the network is required and an ad hoc decentralized network of volunteers appears to be feasible and sufficient. Messages on the network are broadcasted on a best effort basis while the nodes can leave and rejoin the network depending on their will. On rejoining the node, update themselves by downloading and verifying new blocks from other nodes to complete its local copy of the blockchain. In order to form a distributed peer-to-peer network, Bitcoin uses proof-of-work approach, which is often known as Bitcoin mining and considered to be an energy-intensive process. Roughly speaking, electricity bill consumes more than 90% of operating costs for miners.

Fact Alert

While peer-to-peer systems have previously been used in many application domains, the architecture was popularized by the file sharing system Napster, originally released in 1999.

By now, we are well versed that requiring proof-of-work to accept a new block to the blockchain was Satoshi Nakamoto's key innovation. For the Bitcoin network, a valid proof-of-work is found by incrementing a nonce until a value is found that provides the block's hash the required number of leading zeros. The block cannot be changed without redoing the work provided the hashing has produced a valid result. As blocks are chained one after the other, the work to change the block would include redoing the work for the preceding block. A majority consensus in Bitcoin network is represented by the longest chain that required the greatest amount of effort to produce. Therefore, when the majority of computing power is collectively controlled by honest nodes, the honest chain will grow fastest. To modify a past block, an attacker would have to redo the proof-of-work of that block and all the blocks prior to it and then surpass the work of honest nodes. The probability of a slower attacker catching up in the Bitcoin network diminishes exponentially as subsequent blocks are added.

Apart from unauthorized spending, Bitcoin network also solves a specific problem of the Internet payment system (i.e., double-spending), whereby a user pays the same coin to two or more different recipients. The Bitcoin network guards against double-spending by recording all Bitcoin transfers in an open ledger that is visible to all users, and ensuring that all transferred Bitcoins have not been previously spent.

8.8 Bitcoin Wallets

In simple terms, a Bitcoin wallet is a collection of private keys of the public key cryptography. It is also referred to as client software that is used to manage those keys and make secure transactions on the Bitcoin network. However, theoretically the wallet can be a software, hardware, online web service, or even piece of paper that can work by generating necessary public-key–private-key pair to conclude the transaction. The public key (used for generating an address) and the private key (sometimes known as seed) are large integer numbers represented as a Wallet Import Format consisting of alphabets and numbers. As stated in Section 8.2, every public key should only be used once. While the address is public, the private key should only be seen by the owner of the wallet and never shared with anyone. When a user wishes to use Bitcoins, private key is combined with the transaction request, which includes the receiver’s public address to digitally “sign” transaction.

In general, the Bitcoin wallet provides the following functionality:

1. Storage of Bitcoin addresses and corresponding open/closed public keys on users computing device in a typical file “wallet.dat”.
2. Conduct both credit and debit transactions of Bitcoins even when Internet connection is not available.
3. Provide information about the balance Bitcoins at all available addresses, transactions history, user preferences, etc.

Bitcoin Core is considered to be the original Bitcoin client wallet that stores private key information in a file named wallet.dat. The Bitcoin data directory contains wallet.dat file that may be encrypted with a password. The intended usage of the wallet file is only for one installation of Bitcoin at a time. However, if this is cloned for use on multiple computers, it may result in strange behavior.

Wallets are classified based on their features that may include physical, digital, online, and offline. Physical contains hardware Bitcoin wallets and paper wallets meant to be used offline. Digital wallets can be mobile or desktop computer software or web services. These digital wallets are online by default, and if not then these might have potential to be online at some point. However, a hardware wallet can connect to, or print a paper wallet from, a computer.

Desktop (Thick client) wallets are installed on computer and provide complete control of the currency, download of network blocks, and control of their authenticity. In this category, there are two which are prominent: Bitcoin-Qt and Armory.

1. **Bitcoin-Qt:** It is available for Windows, Mac OS and Linux OS. It has high stability, reliability, privacy, resource capacity, and provides signing of messages, keys export/import, wallet.dat file is protected by password encryption.

2. **Armory:** It is available for Windows, and Linux OS. It functions over Bitcoin-Qt and expands its usage by managing several Bitcoin wallet, storing them in offline, protecting from hacker attacks, creating messages and signing them.

Desktop (Thin client) wallets do not need to be downloaded. These are easy to install on the tablet/cell/smart phone. In this category, there are two which are prominent: Multibit and Electrum.

1. **Multibit:** It is available for Windows, Mac OS, and Linux OS, and it is a multicoin wallet.
2. **Electrum:** It is available for Windows, Mac OS, Linux OS, and Android.

Custodial Bitcoin wallet, also known as *hosted wallet*, is very similar in its functions to online bank applications. Unlike other wallets that keep users responsible for security, these wallets provide backup and guarantee access to the wallet even in crisis situations such as when the user loses his/her phone or forgets a mnemonic phrase to access. Usually, custodial wallets support enhanced security measures such as 2-Factor Authentication (discussed in Chapter 3) and multisignature. Without charging commission fee, some of them (Coinbase, Freewallet) provide instant transactions between the wallet's users. On the contrary, just as in a bank, users have no direct access to their funds and informs the custodial to make transactions on their behalf. The funds are not fully secured of being seized by a court decision as well as targeted hacking activities or a scam. Freewallet is a family of custodial wallets working on iOS, Android, and Web. It is a multicurrency wallet that operates 30+ currencies, and has been recently listed as the third biggest online wallet for Android.

Flash Quiz

How many factors of Authentication have you read in this book?
(*Hint*: Refer Chapter 3)

In the hardware, Bitcoin wallets category is by Denarium, which is a physical Bitcoin coin manufacturer. Denarium produces easy, handy, and secure wallets in the form of a coin. In this, a security seal instead of password protection is being used to protect the private key. Denarium also eliminates the need to trust the manufacturer by offering trustless multisignature coins. TREZOR is an isolated hardware environment for signing offline transactions. It uses a small display to visually verify the transaction contents. Another company named Ledger Wallet is also a manufacturer of various crypto hardware wallets.

There is another rarely used Bitcoin wallet known as *paper wallet*. It is a document containing the data necessary to generate any number of Bitcoin private keys and forming a wallet of keys. However, people often use the term as a physical document for offline storing of Bitcoins. People by themselves can create paper wallet by using printers and Bitcoin address generator.

The choice of the user for the wallet will depend upon the usage scenario of Bitcoin stored in it. It is difficult to propose the safest and best Bitcoin wallet as there are different wallets for different goals. For saving larger amounts, Bitcoin hardware wallets may be a better option, provided the user can afford it. On the other hand, software wallets are free, and may be more convenient for frequent transactions and handling smaller quantities. Users may also use a combination of wallets such as using a mobile wallet for checking account, software wallet for transactions, and hardware or paper wallet as a savings account.

There is no safest cryptocurrency wallet, but in order to ensure safety of Bitcoin wallet, it is necessary to adopt the following measures:

1. Perform wallet encryption with a strong password that is difficult to hack. Though it does not give absolute protection. For example, if the computer where the wallet is stored is infected with a keystroke recording virus, the password used for protection may become known to an attacker and compromise the wallet contents.

2. It is important to take regular back up of the entire wallet. We should note that some addresses are used to store changes in transactions and may not be visible to the end user. So, while taking the backup, care needs to be adhered for the completeness.

8.9 | Bitcoin Payments

As we already know, Bitcoin is a decentralized, peer-to-peer network that enables its users to conduct transactions related to money. It facilitates to send and receive digital payments from anyone, in any part of the world, and that too almost instantly.

Since its inception in 2009, Bitcoin has grown a lot and adoption is growing day-by-day. Many of the online merchants are thinking to consider accepting Bitcoin alongside other payment methods, such as credit card, as it is easy to receive Bitcoin payments. The only requirement is to give an address and indicate how much Bitcoin users need to pay. Furthermore, in case the merchants do not want to receive payments directly, there are plenty of payment processors available that can handle it for a nominal fee and convert the received Bitcoins into regular currency. Unlike similar online payment methods, Bitcoin does not take a percentage of the transaction value. It requires attaching a small fee for the payment, irrespective to its value, to get processed by the network. Though technically, any business (online or offline) can accept Bitcoin. However, because of the nature of Bitcoin, it is ideal for those websites which are engaged in selling digital goods and services. Bitcoin is also well-suited for those online sites that need to receive international payments, as it is much faster than bank transfers from one country to another.

Now, we will discuss a few methods to accept Bitcoin payments on merchant site (such as Coinbase and BitPay) and manual payments.

8.9.1 Coinbase

Coinbase is one of the largest Bitcoin exchanges that also offers a merchant service. It enables regular users to buy and sell Bitcoin using their credit cards and bank accounts. This platform provides an easy sign up feature for merchant service, either as a business or as an individual. Once registered, it enables to accept Bitcoin payments on the site, which will go directly into the Coinbase account. Post this, the received Bitcoins can be converted to regular currency based on the current exchange rate and then can be withdrawn to the bank account for a fee. It also provides several features to customize percentage earnings in Bitcoin and liquidate the rest.

8.9.2 BitPay

BitPay is similar to Coinbase, which enables to receive Bitcoin payments and transforms them into bank withdrawals with a nominal percentage fee. Unlike Coinbase, this platform has a different approach as it does not pull double duty as an exchange. BitPay is the Bitcoin payment processor of choice for platforms such as Shopify and Steam and it is gradually picking up. Over a dozen of integration with major platforms is possible with this. It can be used in functionalities such as to accept one-time payments, set up subscriptions, and even receive donations along with invoicing and record-keeping. All this makes it ideal for websites that want an all-in-one solution.

8.9.3 Manual Payments

One of Bitcoin's main selling points is that it overcomes the requirements to rely on third-party services to use them. If anyone wants to accept Bitcoin payments on a website, it is possible to do so even without going through another platform. It is a simple two-step process:

Step 1: Generate a Bitcoin address for each sale you make on your site.

Step 2: Convey the instructions for how much money people should pay for it.

The coins will be received in the destination wallet, and it can process the order as soon as the transaction is confirmed. However, invoicing, billing, and delivery is not a part of it and the system may follow the existing method for them. This does cut out the middleman, but it is potentially a lot of work. Therefore, this method is suitable for websites that accept Bitcoin payments for short durations from time to time, or experience small sales volumes. The Bitcoin received in this form can be stored in any wallet and can be used any moment as soon as the transaction is confirmed, just as for any other Bitcoin.

8.10 | Bitcoin Clients

Actually, Bitcoin client is a super set of Bitcoin wallet, which is discussed in Chapter 5. It is a software for the end user, which facilitates private key generation and security, payment, and typically provides the following information:

1. About the state of the network and transactions.
2. Related to the private keys under its control.
3. About syndication of network events to other peer clients.

The security of the wallet is considered to be paramount. Therefore, clients with focused private keys protection from people with access to the machine where the wallet is stored are considered to be useful. Furthermore, clients who completely implement the Bitcoin network protocol are safer, and it is less likely that they can be easily tricked by powerful attackers. A client that fully implements the protocol will always use the correct blockchain and never allow double-spends or invalid transactions to exist in the blockchain under any circumstances. Clients who partially implement the protocol typically trust that 50% or more of the network's mining power is honest. To protect them from double-spends and other network attacks, some clients trust one or more remote servers. Table 8.4 compares the features of the different open-source (available on GitHub https://github.com/bitcoin/bitcoin) clients in chronologic order.

Table 8.4 **Bitcoin clients in chronologic order**

Client	Audience	Backup	Set-up Time	Disk Space	Started
Airbitz	Everyone	Automatic	Instant	20 MB	Oct 2014
Armory	Power users	One time	Hours	150+ GB	Jul 2011
Bitcoin Core	End users	Manual	Hours	120+ GB	May 2011
Bitcoin Knots	End users	Manual	Hours	5 GB	Dec 2011
Bitcoind	Programmers	Manual	Hours	120+ GB	Aug 2009
Bitcoin Explorer	Power users	BIP39	Instant	3 MB	May 2011
libbitcoin-explorer	Programmers	BIP39	Instant	3 MB	May 2011
Bitcoin Wallet	End users	Manual	Instant	15 MB	Mar 2011
Blocktrail	Everyone	One time	Instant	7.5 MB	Sep 2015
Electrum	Power users	Memorized	Minutes	5 MB	Nov 2011
Gocoin	Power users	Memorized	Hours	120+ GB	May 2013
GreenAddress	Everyone	Memorized	Instant	None	Apr 2013
MultiBit	End users	Automatic	Seconds	50 MB	Jul 2011
Mycelium	Everyone	Manual	Instant	10 MB	Sep 2013
My Wallet	Everyone	Automatic	Minutes	None	Dec 2011

8.11 Bitcoin Supply

Typically, currency is issued by a central bank of a country in a centralized economy. The issuance rate is supposed to match the growth of the amount of goods that are exchanged such that these goods can be traded with stable prices. The central bank of the country actually controls the monetary base; for example, in the US, the FED increases the monetary base by issuing currency. Contrary to this, in a fully decentralized monetary system such as Bitcoin, there is no central authority that regulates the monetary base and currency is created by the nodes in the network. The Bitcoin generation algorithm by Satoshi Nakamoto had already defined well in advance on how currency will be created and at what rate. However, any currency generated by deviating the rules will be rejected by the network.

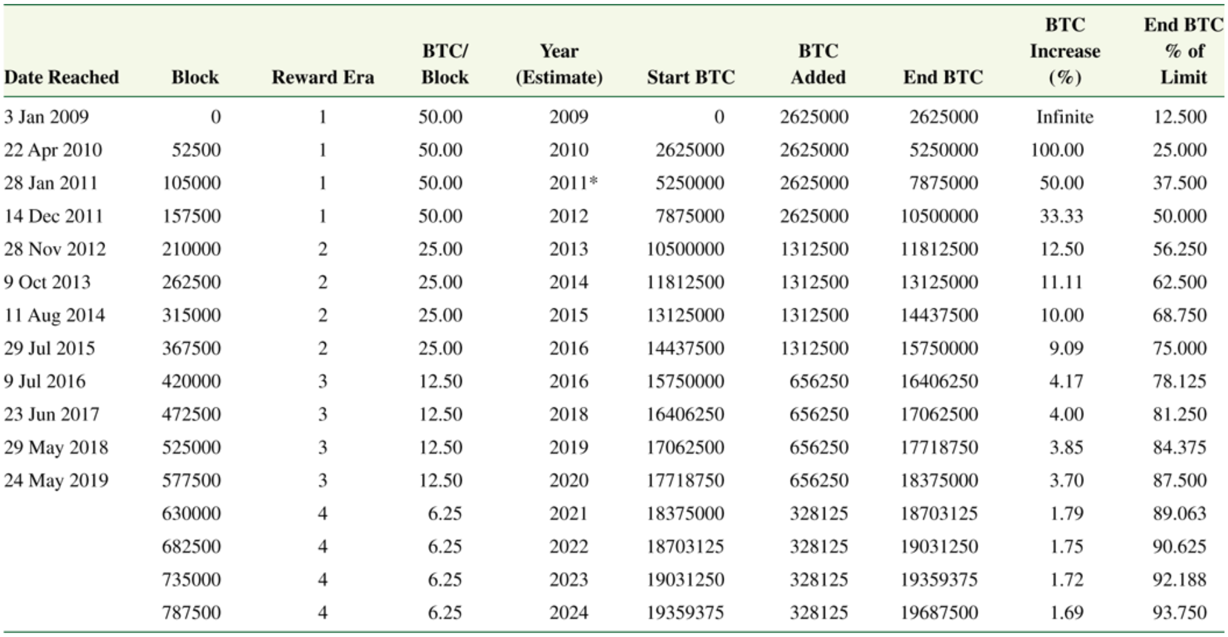
Whenever a user discovers a new block, Bitcoins are created. The rate of block creation is adjusted after every 2016 blocks. The number of Bitcoins generated per block is set to decrease geometrically, with a 50% reduction every 210,000 blocks, or approximately in the time span of 4 years. Table 8.5 presents projected Bitcoins. It resulted into the upper limit of Bitcoins in existence that will not exceed 21 million. Satoshi has never really justified or explained many of these constants.

The halving day that occurred on 28 November 2012 was the day of first subsidy halving, when block 210000 was solved. A Radeon HD 5800 Series graphics card was used to solve block 210000 and was broadcasted on 28 November 2012 at 15:24:38 UTC. After mining for less than a week, a miner named *laughingbear* contributed the solution while Slush's pool was responsible for finding the solution. Later until block 420000, the block reward would be 25 BTC instead of the original 50 BTC.

SUMMARY

Bitcoin is a peer-to-peer decentralized digital currency that enables instant payments to anyone, anywhere in the world, with no central authority. Here, transaction management and money issuance are carried out collectively by the network. Bitcoin software was released by Satoshi Nakamoto under the MIT license. Most client software derived or "from scratch", also uses open-source licensing. The first successful implementation of Bitcoin was described in 1998 by Wei Dai on the cypherpunks mailing list. It was designed around the idea of using cryptography to control the creation and transfer of money, rather than relying on the central authorities, while having all the desirable properties of currency. Bitcoin is portable, durable, divisible, recognizable, scarce, and difficult to counterfeit.

Table 8.5 **Projected Bitcoins short term**



Date Reached	Block	Reward Era	BTC/ Block	Year (Estimate)	Start BTC	BTC Added	End BTC	BTC Increase (%)	End BTC % of Limit
3 Jan 2009	0	1	50.00	2009	0	2625000	2625000	Infinite	12.500
22 Apr 2010	52500	1	50.00	2010	2625000	2625000	5250000	100.00	25.000
28 Jan 2011	105000	1	50.00	2011*	5250000	2625000	7875000	50.00	37.500
14 Dec 2011	157500	1	50.00	2012	7875000	2625000	10500000	33.33	50.000
28 Nov 2012	210000	2	25.00	2013	10500000	1312500	11812500	12.50	56.250
9 Oct 2013	262500	2	25.00	2014	11812500	1312500	13125000	11.11	62.500
11 Aug 2014	315000	2	25.00	2015	13125000	1312500	14437500	10.00	68.750
29 Jul 2015	367500	2	25.00	2016	14437500	1312500	15750000	9.09	75.000
9 Jul 2016	420000	3	12.50	2016	15750000	656250	16406250	4.17	78.125
23 Jun 2017	472500	3	12.50	2018	16406250	656250	17062500	4.00	81.250
29 May 2018	525000	3	12.50	2019	17062500	656250	17718750	3.85	84.375
24 May 2019	577500	3	12.50	2020	17718750	656250	18375000	3.70	87.500
	630000	4	6.25	2021	18375000	328125	18703125	1.79	89.063
	682500	4	6.25	2022	18703125	328125	19031250	1.75	90.625
	735000	4	6.25	2023	19031250	328125	19359375	1.72	92.188
	787500	4	6.25	2024	19359375	328125	19687500	1.69	93.750

* In Block 124724, user midnightmagic mined a solo block to himself which underpaid the reward by a single Satoshi and simultaneously destroyed the block's fees. This is one of two only known reductions in the total mined supply of Bitcoin. Therefore, from block 124724 onwards, all total supply estimates must technically be reduced by 1 Satoshi.

Source: https://en.bitcoin.it/wiki/Controlled_supply

SHORT ANSWER QUESTIONS

1. What is Bitcoin and how does it work?
2. Who created Bitcoin?
3. Can Bitcoin be converted to cash?
4. What is the purpose of Bitcoin?
5. Why is Bitcoin so valuable?
6. Why is Bitcoin untraceable?
7. Who controls the Bitcoin network?
8. Is Bitcoin real money?
9. Is Bitcoin mining legal?
10. How long does it take to mine 1 Bitcoin?

LONG ANSWER QUESTIONS

1. Explain the complete cycle of Bitcoin payment. Also explain how anonymity is achieved.
2. What are the advantages and disadvantages of Bitcoin?

CHAPTER 9

Decentralized Applications

LEARNING OBJECTIVES

After reading this chapter, you will be able to:

- Understand web applications and requirements.
- Discuss about mining in blockchain Bitcoins.
- Understand blocks validation and identification.
- Outline about mining hardware and software.
- Understand the running of miner software and executing several miners.
- Identify Bitcoins management and reason for Bitcoin mining and swarm.
- Understand the robotic possibilities and sidechain hopping.
- Define hard forks and soft forks.

9.1 Introduction

Decentralized applications (DApps) are utilities that are executed on a peer-to-peer network instead on a single system or a computer. They are not new and have existed since the time of peer-to-peer systems came into existence. DApp is a type of software code developed to exist on the Internet network in a method that is not governed by any sole computer or system or entity. DApps do not require executing on the distributed blockchain network architecture. Some of the peer-to-peer traditional DApps are as follows:

Technical Stuff

A Torrent is a fast-flowing stream. However, in the web world it gives addresses for computers around the world which can send parts of the requested file.

1. Tor
2. BitTorrent
3. BitMessage
4. Popcorn Time

In contrast to basic smart contracts, characteristic definition of Bitcoin that transfers currency from one point to another, DApps have a limitless set of members on all borders of the marketplace. We differentiate between DApps and smart contracts by stating the DApp as a blockchain-aided website, whereas a smart contract permits to link the nodes of blockchain using code on the basis of the agreement. A simple and easy way to comprehend this is to consider how old-style websites, typically static, function compared to the dynamic nature of the current websites which are context sensitive.

9.2 Today's Web Applications Requirement

A traditional web application uses the following to extract the webpage:

1. Hypertext Markup Language (HTML).
2. Cascading Style Sheets (CSS).
3. JavaScript (or any web-scripting language).

Technical Stuff

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML.

DApps require getting details from a database exploiting an application program interface (API) utilities. So, whenever any user accesses DApps-based traditional application, there is a call to the API that connects the webpage to the database for retrieving information and represent it on the webpage. In this case, traditional DApps application uses the following to extract the webpage.

1. User frontend.
2. API utilities.
3. Database.

DApp is a predictable and straightforward webpage-based application. The frontend practices the precise identical technology (such as traditional websites) to extract the webpage content. Here, the most crucial difference between frontend and DApp is the use of smart contract rather than API utilities to the database. In the case of DApps, this database is blockchain. In this case, the blockchain enables DApps application by using the following to extract the webpage.

Technical Stuff

Hypertext Markup Language (HTML), developed by WHATWG with initial release in 1993, is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading

Style Sheets (CSS) and scripting languages such as JavaScript.

1. Frontend.
2. Smart contract program code.
3. Blockchain.

In contrast to old-style centralized applications, where the backend program is executed on the central servers, DApps execute program code on distributed decentralized peer-to-peer network. DApps involve the entire suite that comprises backend (actual logic), frontend (what we see), and a smart contract.

Smart contract is at the backend on the peer-to-peer system, which is a very small section of DApps. To create DApps based on the smart contracts, it requires linking many smart contracts based on the conditions and logics with a dependency on other third-party systems for frontend systems. Design of DApps utilizes a smart contract based on the blockchain along with technologies such as Swarm and Whisper, discussed later in this chapter.

DApps can have the frontend program, and the end user programs which are written in a language such as traditional applications. It requests the system call of the smart contracts to connect and retrieve information from backend decentralized storage such as InterPlanetary File System (IPFS) and Swarm.

9.2.1 Qualification of Blockchain DApps

Let us go into more detail, for any application to be termed as DApps and powered by blockchain should follow the given guidelines:

1. It will mostly be open sourced (not proprietary).
2. It must function freely, unidentified (i.e., autonomous in nature), and with no unit governing most of its operations and keys. DApps may acclimatize with its practice in reply to anticipated developments and system feedback, but the agreement of its operators (nodes) must agree to all alterations.
3. It must store data records using cryptography principles.
4. It must use cryptography-based storage in a public decentralized distributed blockchain to evade any failure/choke at a central point.
5. It must use cryptographic token keys.
6. It practices reward-based systems where miners or farmers are rewarded based application tokens on their work to reach any node for consensus building.
7. It produces tokens and unique timestamps.
8. It works on the consensus algorithms (such as proof-of-work, proof-of-value, etc.), where codes contribute to the network (application).

9.2.2 Whisper DApp Message Communication

Whisper is the mechanism used to perform the communication between DApps based on Ethereum. DApps requires the following three basic parameters:

1. Computing resource.
2. Storage (distributed database).
3. Message communication.

Ethereum builds the above platform of compute, storage, and message based on the envisaged strategy of developing public decentralized platform. This is achieved by the following:

1. **Ethereum Virtual Machine:** Helps to provide computing resource.
2. **Swarm:** Helps for storage.
3. **Whisper:** Helps in message communication.

Whisper is a communication protocol used for messaging in a peer-to-peer network of DApps. It works like an API to communicate (send and receive) between nodes of the network systems with complete confidentiality. This feature of confidentiality forces developers of Whisper to make some compromises to impair the performance for confidentiality. Therefore, it makes sense and evaluates to understand the use case suitable for Whisper. The following cases suits Whisper:

1. Pub–sub governance is best suited for publishing and subscribing (pub–sub) communication coordination. DApps can do lot of collaboration using pub–sub activities.
2. Protected, undetectable, and confidential communication is a trait of Whisper communication. It is developed with the design patterns of private and protected messaging with reasonable deniable capability.
3. If we are looking for real-time response over the communication channel with low latencies, Whisper should not be the choice of messaging with respect to time delicate systems.
4. Whisper is not at all good for messaging where we want to send large chunks of data. It is meant for situations where data packets are typically less than 64 KB.

Whisper is a procedure (configuration)-based communication (message) protocol that provides DApps programmers a lot of elasticity in governing the safety and confidentiality constraints of their communications. Let us now try to understand the construct.

9.2.2.1 Whisper Node Network

Whisper communication protocol is comprised of system nodes that are linked to each other in a decentralized manner. To create this network system, a node discovers its connecting nodes in the network using peer-to-peer network protocol/algorithm.

Peer-to-peer network protocol/algorithm is a technology mechanism in the Ethereum environment that offers the platform basis for which procedures in the Ethereum heap are developed.

Whisper and peer-to-peer network work on wire practice/protocol. Whisper and its allied practices/protocols Swarm and Ethereum do not operate between them. The circumstance that both Ethereum and Whisper incline to execute on the same deployment is merely a chance driven by suitability.

9.2.2.2 Identity-Based Messaging

As soon as a node has been linked to Whisper, a DApp can get the communication messages by launching an individuality (identity) in that node. An individuality (identity) is not stringently essential to direct communications, though it is required to deploy two-way messages. This offers interesting applications and problems.

Quick Tip

Identity-Based Encryption (IBE) is a public-key encryption in which the public key of a user is some unique information about the identity of the user (e.g., user's email address).

Generally, a Whisper identity is an object or a group of nodes that works on communication messages. An identity object can be considered as a container of an encoded encryption key. To accept Whisper communications, it is essential to generate an encryption key. Both symmetric and asymmetric encryption keys are used for diverse applications.

Encryption safeguards that only the designated recipient(s) can receive the intended message. If a node can decode a section of the communication, then that communication is planned for the receiver using that node.

9.2.2.3 Sending Messages in Darkness

Whisper advocates for its power as being intelligent to transfer messages in darkness. This signifies that two nodes in a Whisper system can message without giving any noticeable indication to traffic observers and other network nodes, even if those network nodes contributed in the communication routing. This is realized by transaction routine for confidentiality.

To attain complete darkness in message, two elementary standards must be followed:

1. The content of the communication is unreachable to those who interrupt the communications.

2. Messaging nodes cannot be straightforwardly recognized.

The safety of the communications is realized by Whisper adopting an encryption-based method to communicate. However, it is not possible to direct unencrypted communication using Whisper. The sending data must also be concealed to cover the circumstance that two nodes are communicating with each other. This is achieved as the communication is not addressed to anyone.

To transfer a part of communication, the transferring DApp attempts an API call to node of Whisper to encode (encrypt) the communication by means of a distributed symmetric key or the receiver's key (public) and seize the encrypted communication inside a cover (envelope). Similar to its practical counterpart, a Whisper cover comprises metadata to support it to get directed to its last terminus and managed.

In contrast to practical world covers, Whisper covers do not comprise any data about its receiver. A characteristic Whisper cover has the following content:

1. Version
2. Expiry
3. TTL
4. Topic
5. AESNonce
6. Data
7. EnvNonce

The only portion of data motivating to a hack/attack would be the items of the data field, which embraces the encrypted information.

It is logical as part of the dark messaging policy that the receiver of a portion of the communication should not be willingly recognized. Without such a vital portion of data, the only conceivable policy to have any confidence for the communication to reach its proposed receiver is to transfer the communication to each node.

9.2.2.4 Routing

Whisper deals with traffic investigation by having every communication directed to each node in the network. Logically, Whisper acts similar to the user datagram protocol functioning in transmission mode. Here, each node accepts a replica of the same communication, and it is not possible to tell which receiver the sender is attempting to connect with.

Technical Stuff

Routing is the process of selecting a path for traffic in a network or between or across multiple networks.

Though it is hard to express which receiver a communication is addressed to, a controlling opponent may still be able to identify if two nodes are messaging. In this situation, if node A sends a communication to node B, the opponent will be able to distinguish that messaging occurred between the two. Such a hack/attack can be overpowered by familiarizing noise into the network by taking dutiful nodes transferring junk communications encoded with a random encryption key into the system.

9.2.2.5 Message Filtering

For a node to agree if a part of communication is planned for an identity utilizing its service, it requires to be capable to decrypt the communication. Since the cover (envelope) comprises no metadata on the proposed receiver, a node must attempt each key it possesses before it can identify whether a part of communication is directed to its users. Decryption is not cheap and requires a lot of work. Moreover, with the circumstance that a node will accept every part of communication of the network, it is not viable to decrypt each part of incoming communications in most real-world use cases.

This problem is resolved by requiring each communication to be linked with a theme (topic). An identity records the themes it is concerned with by means of its encryption key to produce a communication filter on a node. This well-organized likelihood filter, known as *bloom*, can voice to a good level of certainty if a portion of communication fits to a theme of attention (interest). A node will try decryption if a filter indicates a likely match.

9.2.3 Service Quality Assurance

With every Whisper communication being carried to each accessible node, it is not difficult to arrive to the decision that Whisper is vulnerable to denial of service (DoS) attacks. A hack/attack can be classified in following ways:

1. Continually pumping communications messages into the network causing flood attack.
2. Manage communication messages to stay around for a longer time by putting a long time to live (TTL) on the cover, causing an expiry attack.

Technical Stuff

Ping of death is a type of denial of service (DoS) attack where an attacker attempts to crash, destabilize, or freeze the targeted computer or service by sending malformed or oversized packets using a simple ping command.

Flood attack is prohibited by the Whisper node. Communication filters need the source to accomplish proof-of-work (PoW) calculation and count the consequence as the Env Nonce field in the communication message cover. A node may deny receiving a part of communication if PoW is extremely low. Communication filters may choose for more rigorous PoW prerequisite than the node requires.

Expiry attack is prohibited by a communication rating system that takes PoW, communication size, and TTL into interpretation to calculate a communication message rating. The principles of the rating are somewhat modest for the higher scores in message. The following points regarding communication messages should be noted:

1. Smaller communication messages have higher scores.

2. Communication messages with higher PoW has higher ratings.

3. Communication messages with lower TTL have higher scores.

The score of a part of communication touches both its advancing priority and the amount of time it will be recorded in the system. It is consequently related to corresponding sender to make certain communication ratings as high as possible to realize its objective. For instance, when a node's communication pool reaches its upper memory limit, the node may trash communications that it contemplates to be of low score. This scoring system confines the influence of a DoS attack.

Whisper is an identity-based autonomous low-level communication message system, which ensures the following for its network system (DApps) and users:

1. Hash based identities.

2. Confidentiality declarations.

3. Encrypted communications.

4. Cryptographic assurances for senders.
5. Communication with a definite TTL.
6. Integrated confidentiality and secrecy.
7. Agreements of "darkness".
8. Select in or out of diverse confidentiality characteristics.
9. Uses Ethereum network.

Whisper client and Ethereum nodes utilize the peer-to-peer wire protocol for their node-to-node communication.

9.2.4 Design Decisions

A design decision is a building block in next generation distributed DApp, which requires the following:

1. Comprehensive many-to-many data discovery.
2. Signal cooperation.
3. Simple transmission.
4. Judicious declaration of comprehensive confidentiality.

As per the design decision, there is a significant difference between encryption of communications and darkness, which the developers of Ethereum are attempting to realize. For numerous determinations, just having a clue of the destination of someone's message can finish the desirable confidentiality assurances without ever breaching the encryption of the information (e.g., Wikileaks). Metadata around a message, which is deliberated in appropriate and large amounts can offer a lot of information and occasionally cancel the result of encrypted content.

9.2.5 Whisper Elements

The key elements of Whisper include the following:

1. Envelopes (covers).
2. Messages (communications).
3. Topics (themes).

Envelopes (covers) are packets that comprise TTL (in seconds), expiry, topics (themes), and nonce that delivers for PoW needs for communication senders coupled with message data field. The message data field contains the following:

1. Actual message.
2. Payload.
3. Flags.
4. Signature.

Payloads are encoded by the sender and decoded by the receiver. The practice offers for ratings of peers by the nodes, and rating of the communications themselves by the effort consumed in finding the nonce. Proof of larger effort must afford a communication with larger significance on the network.

Nodes can publicize their themes (topics) of interest to each other. Senders and receivers can select in or out of diverse confidentiality characteristics against performance characteristics.

9.2.6 Acceptance Blockades

Although Whisper is an effective communication protocol, there is not much acceptance. This is partially because of the strategy selections of

Whisper and the substitute communication opportunity from inside the Ethereum environment.

9.2.6.1 Key Management

Whisper presently trusts on nodes possessing on to users' undisclosed (secret) key, so that communications proposed for the users can be decrypted. This restriction can be improved by sharing Whisper nodes executing with the help of wallet utility on the user's computing device.

9.2.6.2 Disabled (By Select-In)

Presently, Whisper is a select-in provision. To practice it, one has to permit it at the command line. As a consequence, Whisper is not executing on most Ethereum nodes. This choice to have Whisper inactivated by default is a practical one, as it is still an in-trial practice.

9.2.6.3 Whisper's Family-Based Postal Services Over Swarm

In 2017, a novel communication protocol termed as Postal Services over Swarm (PSS) was announced. Superficially, it appears to be similar to Whisper but with a diverse set of balances, such as having probabilistic routing. In contrast to Whisper, PSS needs a discrete Swarm client to function in a Swarm collection.

While PSS may not be in competition with Whisper but having PSS in the overall combined structure may weak interest and resources available to the growth of Whisper. In addition, PSS may discover a substitute resolution to the key management challenges of Whisper at the instant.

In brief, PSS signifies both a danger and a prospect to Whisper. Both ways, designers' take advantage from taking more choices when it arises to communications. Whisper is a nascent, experimental messaging protocol, but it has now instituted its roots into production application.

9.3 Mining in Blockchain Bitcoin

Bitcoin constitutes the following:

1. It is a set of rules or procedures or protocol that describes how the network should function.
2. It is a software application/utility that implements a protocol.
3. It is a network of computing devices executing software that practices to a set of rules to generate and accomplish Bitcoin currency.

Mining is described in the set of rules or procedures or protocol executed in software, and it is a vital role in governing the Bitcoin network. Mining helps in the following activities:

1. Authenticating transactions.
2. Avoiding double spending.
3. Collecting transaction rewards and fees.
4. Producing the money supply.
5. Defending the network by supporting tons of computing power to process on top of historical transactions.

Mining authenticates transactions by assessing them against historical or past transactions. Transactions cannot disburse Bitcoins that do not occur or that were consumed/expended. They must direct Bitcoins to validated and verified addresses and follow each well-defined instruction by the protocol.

With the occurrence that is the best at every 10 minutes, mining generates new blocks from the newest transactions and generates the number of Bitcoins demarcated by the present block reward. Miners also confirm blocks formed by other miners to permit the entire network to continue development on the blockchain.

9.4 Blocks Validation and Identification

To discover a validated and verified block, the miner develops a series of current transactions and computes some immediate (summary) statistics about the anticipated block. This statistics based summary is joined with a number termed as *nonce* to generate a block header. The block header's hash is then computed and understood if it is adequate to conquer at the existing difficulty. If it is not, the nonce is altered and the new hash is computed and verified.

There is no method to generate a validated and verified block except by brute force to generate and test search algorithm. It describes the attempts of the miners as one nonce and reiterating the process till it gets fortunate. Throughout the search, the miner cannot forecast if the subsequent nonce will offer a smaller hash than the previous one, as it is defined by brute force execution. The only method to intensify the likelihoods of conquering is to increase the speed with which one can attempt nonces. Additional computing power is required to process for clearance, the sooner one can search the more probable it is to discover a conquering block.

When a validated and verified block has been created, it is transmitted to the network and swiftly confirmed by the other nodes in the system. The trouble of discovering a conquering number is accustomed every 2016 blocks. Thus, blocks are created after every 10 minutes on an average.

9.5 Bitcoins Creation

Once a miner discovers a new block, it contains a new address from where new Bitcoins and any transaction rewards are to be given as a fee. This payment is the financial motivation for individuals to execute miners. If the circumstances are correct, one can deploy mining hardware to execute, recompensing for time, power, and computing resources and make an income by trading the ensuing Bitcoins that were granted.

At present, 50 Bitcoins are granted to the miner who discovers each block. This will endure till block 210,000 is discovered, at this time the block repayment will split into half to 25 Bitcoins. Subsequently, the payment/award will then become half once more every 210,000 blocks. This signifies that the number of Bitcoins to be generated will be at the maximum (i.e., 21 million) by 2040.

Form where are these Bitcoins generated? These are generated by the network as the section of the Bitcoin protocol.

9.6 Mining Hardware

As already mentioned, the word miner defines an individual who implements mining computing systems, the computing hardware executing the mining, or the software that performs the algorithm needed in mining.

As we recognize, some computing resources are faster than the other computing resources. These resources can have faster or slower processing units, the RAM can be of different configurations, and volume of the hard drives can be bigger and smaller. It is also right that some categories of processors are superior at mining over others. However, the verification of nonces is a very recursive/iterative task, and computing hardware that does iterative tasks speedily is best suited for mining.

The constituents of the CPU are organized in a manner that helps the computing resource one application to another. The CPU is enhanced for job swapping (switching) since that is how it devotes most of its time.

In addition, a computing resource graphics processing unit (GPU) is meant for general graphical activities such as draw a square, shadow, pixel, etc., in an iterative fashion as and when required. It is organized internally so that it can be much faster and more competent for Bitcoin mining. Actually, general video cards can defeat general CPUs by hundred times or more.

The most recent advancement in Bitcoin mining is an additional processor called field-programmable gate array (FPGA). The FPGA is an extremely programmable processor. It is more exorbitant than GPUs and CPUs but it is also relatively well-organized in its usage of electricity power. Miners who are viewing to function where electricity is more costly can finance more money at the beginning to purchase the FPGA miner and then recompense less in continuing electricity expenses.

Technical Stuff

A field-programmable gate array (FPGA) is an integrated circuit designed to be configured by a customer or designer after manufacturing. Hence the term "field-programmable".

9.7 Mining Software

An individual mining with a GPU will require software to indicate the computing resource to mine Bitcoins. Mining software is accessible for the following:

1. Windows.

2. Mac.

3. GNU/Linux.

Most software is freely available under open source umbrella that one can download and install on their own or with the help of forum.

When an individual installs the mining software and executes it, it will describe how fast it is mining. It is described with a number signified in hashes per second with general speeds available in mega or giga hashes per second.

The objective is to become as many hashes accomplished by hardware as conceivable per unit time (per second). The finest software for computing hardware will assist in doing this. Suitable software offers a best hash rate but is also steady, which denotes mining should not break because of a malfunction in the software.

Once the mining software is installed, it may also ask for the precise settings that will be used by the miner. These situations may differ from one system to another, and will support to get maximum performance from the miner. A miner can discover these settings for specific GPU on online forums related to Bitcoins.

9.8 Running Miner Software

Once equipped to mine, a user can initiate the mining software. This may include initiating the existing software program several times if one possesses several GPUs or one may select to use mining software that executes as a bootable disk that will cover the complete computing system and accomplish the mining required.

It is important to note that we should evaluate the miner continuously to be certain it is executing and receiving a decent hash rate. This is the situation if one has performed any specific actions or changes to optimize hashes per unit time (second). It is also equipped with mobile applications and websites to provide updates of the status of the miner.

When a block is won, the Bitcoin account balance will rise by the quantity of the block award/payment, including transaction charges that were compensated.

9.9 Executing Several Miners

If one is motivated to capitalize more computing resources into mining Bitcoins, then it is likely to attach several miners organized on a network. To accomplish this, some elementary network tools such as a router and a computing resource to execute the Bitcoin daemon is required. It is required to implement an authentication system using password, and consequently miners can interact to the running instance of the Bitcoin daemon. Once a block is discovered by one of the miners, the Bitcoin daemon will comprise the wallet with the key that ciphers the block and asks for the block payment/awards and payment.

Executing several miners has electrical power and heat inferences that need to be considered. A high configuration based mining computing resource can utilize as much power as consumed by the big electrical appliance.

9.9.1 CPU Overclocking

Computing resources such as CPUs possess speed at which they execute, which is termed as *clock speed*. For the current set of processors, this is specified in gigahertz. Once a CPU is mass produced, it is verified to observe how fast with the specified clock speed it can consistently help. Processors that are not constant at higher clock speeds are traded as lesser clock speed systems. User of exploratory computing resources have been pushing out more power from their processing units by growing the clock speed, which is a method termed as *overclocking*.

Quick Tip

The term underclocking reduces power consumption and resultant heat generation of a device, with the trade-off being lower clock speeds and reduction in performance.

Usually, GPUs have a software utility that is utilized to modify the clock speed of the processing unit as anticipated. GPUs also consider a RAM speed that can be attuned.

As already mentioned, the objective is to receive as several hashes as possible per unit time. It is important to note that we need a good hash rate but with consistency. The challenge with overclocking is the extent one can overclock the GPU without instigating the miner to stop/halt. When halted/stopped, the hash rate will come to zero. In the situation of a multiple GPU implementation, the hash rate will come to a portion of what it should actually be.

It is required to simulate with the hardware computing resources under different conditions to discover what offers the best situation with most stable clock rate.

With respect to GPU RAM speed, this is repeatedly decreased to save power consumption and aid graphic cards to execute cool at a specified clock speed and ensuing hash rate.

9.9.2 Single Mining

Mining is a probability based endeavor in which the likelihood of awarding a block within a specified period of time can be designed. Regrettably for several systems, the time needed to have a decent probability at mining a block can be long, may be months or even years.

While running many computing resources at high hash rates, one may discover blocks in a better frequency and may be able to grip the rate disparity/variation of mined blocks. On the other hand, solo mining can help if the hash rate is low and it can wait to be executed longer while expecting to get fortunate. If the aim is for safer bet and consistency of rewards, mining in a pool model is favoured.

9.9.3 Pooled Mining

Conquering a block will most probably be uncommon, so mining pools were formed as a method to level out the payouts/rewards. Those who link a mining pool collaborate to mine blocks as a cluster and when a block is resolved by one of the members, the pay-outs/rewards are mutually shared.

The payout reaching to each member takes into interpretation the hash rate and time for pool mining. Mining pool workers may take a fraction of Bitcoins as payouts/rewards for generating and executing the pool. There are several pools to select from and mining patrons can effortlessly be swapped from one pool to another.

It is useful to link with at least two pools in case the first one becomes inaccessible for some time. In absence of the backup pool, the hash rate will effectually touch zero till the pool becomes accessible again.

9.10 | Bitcoins Management

It is immaterial if mining is performed by a single miner or a set of disjoint solo miners, with single computing resource or with several, ultimately one will earn some Bitcoins. When it is received/awarded, it is significant to manage them appropriately.

Once the Bitcoin client is executed for the first time, it generates a wallet for Bitcoins. This digital wallet comprises private keys, formed of long blocks of arbitrary alphabets and numbers, which are intended to be reserved undisclosed. Private keys permit Bitcoin clients to perform operations, spend, and efficiently possess Bitcoins.

Corresponding to each private key, a Bitcoin address and related public key are formed. When someone directs Bitcoins, their corresponding Bitcoin client digitally signs Bitcoins over one of the addresses using the private key. This operation is transmitted to the Bitcoin network and logged in a block at the later time.

It should be noted that Bitcoins are not really directed anyplace but are actually allocated to addresses. To transfer Bitcoins to another address, the private key to the initial address is required. This shows that we need to safeguard the key and have a backup wallet to shield it to counter theft (data), virus outbreak/attack, damage because of the storage failure, and natural disaster.

The general method to back up a digital Bitcoin wallet is to utilize the backup characteristic of Bitcoin client program. If such capability is absent, one should look for a supplementary backup, discover Bitcoin configuration directory, and perform a full backup of it. It is important to switch off the Bitcoin program before developing this standby replica. We can keep the data safe using USB drives, CDs, or any other storage media. We can preserve backups in a locker till they are required to be used.

Fact Alert

The USB disk was invented by Amir Ban, Dov Moran and Oron Ogdan from M-Systems, an Israeli company, and they were granted a US patent on 14 November 2000.

It is best to examine the backup on an alternative protected system, and periodic backup testing is also required. It is also good to verify it on an alternative system because we may lose the private keys while bringing (restore) a backup over a digital wallet and miss the Bitcoins.

Another threat comes from the online world, and the data needs to be protected. This signifies maintaining Bitcoin wallet offline or on a system that is separated from the Internet. For rational yet safe entry to the Bitcoin wallet, we can use a bootable disk with computing system to enter and operate Bitcoins. The bootable disk would comprise all the software required to manage Bitcoins.

Encryption is also a good option when the backups are placed in a situation where it is accessed by other individuals. Once encoded, a possible attack would require password to get entry to the digital wallet file and attack Bitcoins. There are several applications that offer a general method to encrypt files on any disk drive or computing system. Bitcoin website comprises wallet encryption program that can also be used.

9.10.1 Bitcoins Usage

Bitcoin is currency, and one can use it to settle any sum. It can be used to carry out any business or even perform operations. It is used as an online digital currency to buy any services, goods, and products. Today, many businesses have already started to accept Bitcoin as the acceptable mode of payment.

For commercial operations that only receive state currencies such as dollars or euros, it will need to exchange Bitcoins. This can be completed on any online exchange that accepts Bitcoin. This requires an exchange account to sell, transfer, or buy the Bitcoins. Using this account, Bitcoins can be traded with the currency one prefers. In order to accomplish this, one needs to transfer the money that is compatible with Bitcoin. A significant characteristic of Bitcoin is irreversibility. When Bitcoins are transferred to the corresponding address, there is practically no method to inverse the transaction.

Bitcoin is consequently unsuited with economic facilities that permit payments to be undecided or can be reversed back such as credit cards, cheques, and bank drafts. In the business that revolves around Bitcoin, repayments should be addressed by the sender of the capitals and purchasers should use escrow facilities with the retailers.

Mining requires various aspects to be considered such as: (1) which hardware to use, (2) any upgrade, and (3) how to mine the first coin.

By offering swift, low-cost, global money exchange, Bitcoin has the ability to transform the current time notion of both currency and business. Bitcoin was initiated as an open source software based on the research article published by Satoshi Nakamoto (2009). Bitcoin software appears to have been developed explicitly to deliver Bitcoin, and developed an online system to transfer value that provisions a potential digital currency.

Bitcoin is developed based on the software program and network mechanisms. A software program termed as *Bitcoin client* concurrently accomplishes and supports Bitcoin transactions. This software program keeps a ledger that records each transaction, which is validated by the network nodes.

The network of Bitcoin, which involves numerous computing resources that execute the software program has the following key responsibilities to complete.

1. Transmit transaction data.
2. Authenticate and confirm transactions to safeguard Bitcoins that cannot be expended for a second time, thus avoiding double spending.

The first activity is performed in a straightforward manner as the Bitcoin network functions as a peer-to-peer node based network system. In this manner, the distribution of information is easy. By working on various nodes across the world, the network guarantees that it will function as long as it offers a beneficial service.

The second activity is a bit more complex and is resolved through mining, which is performed by various computing resources that execute the mining software application or program.

9.11 Reasons for Bitcoin Mining

There can be various reasons to mine Bitcoins. These reasons can vary over time as one understands about Bitcoin and track its development. It is important to understand others' motivation to be able to accept the Bitcoin network and the money with which it is compatible.

Most programmers/users initiate mining as an ordinary extension of the work they are performing. For instance, Bitcoin mining is comparable to other advanced grid computing developments because they are useful and offer a chance to collaborate with others in resolving a large challenge.

When Bitcoins are mined, miners support to solve the challenge of generating a currency and compensation/reward network that does not depend on a central delivery agency.

Those who work on blockchain realize it is imperative to stay well-versed with innovative technologies. Miners mine Bitcoins to shape the careers with these technologies as Bitcoin is a novel amalgamation of numerous new technologies such as cryptograph, peer-to-peer networks, and distributed systems and databases.

Meanwhile, Bitcoin purposes as a currency and mining can be functioned as a business process, and most of the miners perform it for income. However, it is a difficult business because Bitcoin values can alter extensively and required capital investment for a mining operation can be some small amount. If you can function proficiently and want to mine for income, be certain to perform assessment assignment before executing any big procurements.

Mining is a method to obtain Bitcoins and this may be a reason for those who desire to obtain Bitcoins without using facilities such as exchanges or by doing any regular business requiring goods or services. Another reason for mining is for secrecy. If one resolves a block and is vigilant to attach to the Bitcoin network utilizing a Tor router, mining is a method to get Bitcoins while being completely incognito.

Acknowledged as a payment based spend and expense network, Bitcoin is also a software program project. There are several software implementations that depend on Bitcoin technology for their achievements.

9.12 | Swarm

Swarm is a storage platform that works on a distributed system, content sharing and distribution service, and an Ethereum based native services existing at a base layer.

The main goal of Swarm is to deliver an adequately decentralized and identical replica ledger stock of Ethereum's public information to store and share distributed applications code, data, and blockchain information. From a financial perspective, it permits contributors to proficiently pool compute resources such as storage and network bandwidth to offer these facilities to all the members of the network, while being compensated and rewarded by Ethereum.

9.12.1 Goal of Swarm

The main goal of Swarm is to offer infrastructure services to developers and testers in the DApps world, particularly the following:

1. Communication messaging.
2. Data streaming.
3. Peer-to-peer ledger record accounting.
4. Alterable resource changes.
5. Storage assurance.
6. Proof of guardianship assessments.
7. Payment gateways.
8. Distributed peer-to-peer database services.

With respect to users, Swarm is similar to the World Wide Web, with the exclusion that uploaded data is not held (hosted) on an explicit server. It delivers peer-to-peer distributed storage and possesses the following characteristics.

1. DDoS resilient.
2. Zero downtime.
3. Fault tolerant.
4. Censorship resilient.

5. Built-in autonomous incentive system that practices peer-to-peer ledger keeping and permits transaction of resources based on the expense and reward system.

Swarm is developed to intensely work with the following:

1. Peer-to-peer network layer protocol of Ethereum.
2. Domain name resolution (using ENS) for blockchain that ensures payments/rewards/incentives of services and content accessibility assurance.

9.12.2 Basics of Swarm

Swarm was developed to offer foundation base of infrastructure services for a new decentralized application powered by the Internet. It is a peer-to-peer node of networking based distributed services by participating resources among the nodes. The digital services can be storage, message forwarding and payment processing.

These participations are precisely managed on a peer-to-peer foundation, permitting nodes to transact resource, but delivering financial reward to nodes using less than they assist.

Swarm Testnet is operated by Ethereum, which can be utilized to assess the functionality in an analogous method to the Ethereum Testnet. Anyone can connect the network by executing the Swarm client node on a computing device such as server, desktop, laptop, or mobile device.

The Swarm client comes as one of the constituents of Ethereum stack. Golang is officially used to write implementation references. It is available in Swarm repository that comes under Ethersphere.

The data items that are uploaded are not certain to continue on the Testnet till storage assurance (insurance) is executed. All contributing nodes should reflect contributing an intended service with no prescribed responsibility whatsoever and should be anticipated to erase content at their determination. Consequently, users should under no condition treat Swarm as assured and protected storage till the reward system is in place.

Swarm delivers an API, that is, local HTTP proxy and interaction can be established by DApps or command line utilities. RPC JSON API is used establishing message communication. Testnet based servers deliver public gateways, which assist to simply validate characteristics and permit unrestricted entree so that individuals can attempt Swarm without even executing their individual nodes.

Swarm nodes can link with one or many blockchains for resolution of domain names and one blockchain for network bandwidth and storage rewards. Nodes executing the identical network id are agreed to link to the same blockchain for rewards incentive system. A Swarm network node is recognized using network identities, which are random integers.

Swarm permits for upload and delete, which signifies that any specific node can upload data to the Swarm and then is permitted to be disconnected after deletion. Up to that time, nodes do not vanish or become inaccessible, the data will still be available because of the "synchronize" process in which nodes incessantly transact along accessible data among each other.

Swarm can handle encryption also. It is not fortified to upload the data that is private in sense and not encrypted as it is possible to unfasten it. Swarm users should avoid uploading prohibited, debatable, or unscrupulous data.

It is recommended to use encryption for sensitive data. Only users who have access to the root (Swarm hash) and decryption key can access the data that is encrypted and protected. This requires an extra step before publishing the data, and users are slightly safe in contradiction of uncaring publishing if they use encryption. Although there are no agreements for elimination, untouched data that is not obviously assured will ultimately vanish from the Swarm, as nodes will be rewarded to trash it in case of storage volume limits.

Swarm does not support any delete or remove operation as it is a persistent data structure as data is distributed to Swarm nodes that are compensated to help it.

9.12.3 Robotic Swarms

Nowadays, industries may possess just one drone, there are days to come when large sections of an industry can be observed by a number of robots. How will these robots achieve suitable activities such as gathering data or establishing around a mutual objective?

It is realized in robotic swarms that each robot works on rudimentary guidelines and design motivation from beings such as ants that frequently cluster together and move. These trivial instructions then augment up to joint tendency such as distributed detect and sense, or exploration and saving missions, which are developed as an outcome of the communication between the robots.

Until now, these idea have not been implemented in a comprehensive manner. But researchers have great expectations for applications that are now termed as *precision farming*. For example, convoys of drones can be utilized to assess the condition of crops and depict a finer course of image for agriculturalists. As researchers touch on this innovative idea, they are coming up with several safety and operational glitches that have barred robot swarms from being implemented in the practical world.

The blend of blockchain with robotic swarm based distributed systems can deliver essential competences to develop more protected, independent, elastic, and even lucrative robotic swarm logistics.

Use of Bitcoin to robotics has been conceived for self-directed networks of driverless vehicles such as cars and drones that would transport correspondences. But here, the idea is for a blockchain-based network where robotic nodes are deployed in a safe and distributed method. One possible use for the blockchain is to benefit the robotic clusters to arrive at an agreement about a conclusion without a central agency.

This results in a prototype where the blockchain is used by robotic swarms by executing as nodes in a system and capturing their transactions within blocks. Blockchain based robotic swarms use cases comprises secure interaction among robots, distributed policymaking, behavior variation, and business models.

9.13 | Robotic Possibilities

At present, robotics investments are decreasing, whereas computing resources are increasing. Robotics is not limited to research and development (R&D), though we can make thousands of robots to work together for an industrial purpose. The only challenge is swiftness and precision. The notion is to use the blockchain for R&D in the industrial world.

Robots derive the track based on the contract and help develop consensus on the blockchain. This can be experimented, even by political parties to develop different use cases.

Suppose a cluster of drone sensors are trying build consensus whether an image is of type A or type B. One robot (drone) of many will perceive the necessity to make a choice and initiate a vote using a distinct transaction generating two addresses that signifies each choice: type A or type B. The result is obtained by the majority regulation. Each robot casts its decision based on the consensus vote in the arrangement of a transaction directed to the address that characterizes their decision. As it is a public blockchain, it can be viewed by all the robots, and they can swiftly authenticate the outcome of the vote. The robots reiterate this procedure for each choice the robotic swarm requires to make. Practically, each vote casted would require occurring at a particular time, which is one of the challenges with blockchain technology, as public blockchains often need the smallest amount of time to guarantee that the transactions are protected.

9.14 | Sidechain Hopping

Robotic swarms work on behavior. That is, it requires jumping from one behavior to another to achieve their aim. For this, the clusters of robots practice interoperable blockchains to switch or hop to control algorithms in a procedure of behavior variation.

Consequently, robotic swarms may hop behaviors when they transact to another sidechain. A specific blockchain may trust decentralized governance using round-robin mining, and other blockchains can follow the leader node robot based control mechanism.

A chain can be a private organization blockchain leveraged for testing objectives and occupied with organization data, but it can be attached to a public blockchain if required.

Blockchains have some basic boundaries that can stop extensive proliferation of blockchain-reliant robotic swarms. For instance, the 600 seconds transaction authorization window time of Bitcoin becomes a problem in the method of fast voting on choices. Robotic swarms perhaps cannot wait for 600 seconds to identify which path to move if the swarms are near any obstacle.

In addition to this, if the Bitcoin blockchain grows in the same way, it may be too troublesome for each swarm robot to hold and fly with a replica of the entire record ledger.

9.15 | Blockchain Forks

A blockchain fork is a mutually approved software update, or in simple words, a patch. Blockchains rely on decentralized clusters of computing systems, all employed jointly. Each distinct computing resource, generally known as a node, executes the software required to authenticate the public ledger of blockchain and preserve the protected network. If the number of full nodes increases and works simultaneously to execute the software, the network will be safer and more protected.

Every full node requires executing the same part of software to get access to the public distributed ledger. In order to understand this better, we can say that any full node executing Bitcoin central software has authentication rights to access the Bitcoin ledger (based on blockchain) and can consequently validate Bitcoin transactions, history, and access. Though any of the computing node only executing central software cannot get the entry in the blockchain.

9.15.1 Hard Forks versus Soft Forks

Any deployment based on distributed systems is dependent on the core software and the version as well. It is important to note that developers can push new updates to install new features in the core software. But a challenge still exists where all the nodes of the older version software will not be compatible to new updates.

Hard forks are forks that are mismatched with older versions of the software. Hard forks characteristically alter consensus guidelines such as block size, mining algorithm, and consensus procedure. Hence, the older version becomes incompatible and mismatched. Whenever the newer version is updated it changes the protocol rules; these types of updates are treated as hard fork. Computing nodes that are not chosen for upgrade will not have their software extended to the family of compatible blockchain nodes. Hard forks are not typically combative, and all network nodes mostly settle to update. Otherwise the compute node executing the upgraded software will not have the precise identical ledger as the compute node executing the older version. Meanwhile, each computing node will have diverse consensus instructions, and they will be principally executing a discrete blockchain.

There are a specific variety of forks that are well matched with older versions of the software known as *soft forks*. It is a software update that works well with older copies without any major problems.

9.15.2 Argumentative Hard Forks

Not all forks are consistently settled upon by the network of full nodes. Actually, most forks are well thought out as contentious or argumentative hard forks.

These types of hard forks happen when a noteworthy percentage of computing nodes do not agree on which version of the software to execute. Because of this disagreement, contentious hard forks are typically observed as disadvantageous to the main chain.

Mostly, hard forks generate value instability. It is significant to distinguish the background of each fork in order to take benefit of occasionally radical and sudden variations. It is important to understand the vital actions for blockchain ecosystems and consequences as part of broadcast message to all computing nodes.

SUMMARY

Decentralized applications (DApps) are a noteworthy change from traditional applications, mostly because of the fundamental principle of decentralization that moves the entire lifecycle. Trust and data immutability are additional features that distinguishes the two. Dissimilar to traditional application that have a centralized regulatory agency, DApps run on a decentralized blockchain and the data once transacted on the chain cannot be altered or deleted. Consequently, at each phase of the DApps development lifecycle, everybody looks to ensure to use the philosophies of the decentralized network.

SHORT ANSWER QUESTIONS

1. What does DApps mean?
2. What is DApps blockchain?
3. How does DApps work?
4. What is Whisper protocol?
5. What is Swarm protocol?
6. What is Swarm in blockchain?
7. How long does it take to mine 1 Bitcoin?
8. What does overclocking a CPU do?
9. What is a fork in blockchain?
10. How many forks does Bitcoin have?
11. What is a hard fork?
12. What is a soft fork?

LONG ANSWER QUESTIONS

1. What are the differences and similarities between single and pooled mining?
2. Explain Whisper with the help of a use case and note how it is good for the following:
   - **a.** Publish–subscribe coordination signaling.
   - **b.** Secure, untraceable communication.

CHAPTER 10

Blockchain Vertical Solutions and Use Cases

Learning Objectives

After reading this chapter, you will be able to:

- Explore various verticals where blockchain technology can be utilize.
- Several use cases will enable to draw analogy in different areas.
- Draw analogy in different areas after studying the use cases.
- Appreciate the cross intersection of different technologies in blockchain applications.
- Explore the cases where further research can be applied.

10.1 Blockchain

Blockchain is being extensively discussed in recent years. Several prototypes are being developed to convert next generation use cases for various possible applications. The appeal of blockchain and its use cases can be best evaluated by the type of consideration it reaps from new start-ups and competitors.

10.1.1 Blockchain in Financial Institutions

Nobody can deny the importance of blockchain technology in financial institutions. It is very important that how financial institutions across the world have positively comprehend this technology with focused approach and resolving the pain points in the present enterprise setup process.

Nowadays, financial institutions are confronted with matters such as follows:

1. Increasing operations expenditure.
2. Growing vulnerability to duplicitous outbreaks on centralized servers.
3. Problems in safeguarding transparency.

There are various financial institutions operations that involve the following:

1. Worldwide financial institutions-based customer account opening.

2. Exhaustive labour-intensive human intervention-based paperwork.

3. Encompass expensive mediators.

4. Inefficient long-time based transactions authentication.

5. Non-existence of scam-resistant concurrent resolution.

Financial institutions are constantly discovering new methods to accomplish transactions more swiftly for heightened customer service, while guaranteeing cost optimization in its execution and transparency to patrons and controllers. To accomplish this, blockchain offers a solution for financial institutions as it characteristically benefits to eradicate mediators, preserve incontrovertible record of transactions, and facilitate instantaneous and concurrent transactions executions.

It helps to reduce the amount of time in investment of human intervention, and most importantly, to improve client service and gratification. Like any other business, selecting the correct “use case” is crucial for financial institutions to get evolving growth of blockchain.

Digital cryptocurrency has been a subject of discussion since the first Bitcoin operation was carried out in 2009. While financial institutions and controllers have mostly been suspicious of Bitcoin, the fundamental methodology of blockchain and distributed ledger gained their attention in the past few years.

The appeal of blockchain was its technique of authenticating and tracing transactions. In place of reliable third-party mediators or centralized financial institutions, it trusts on consensus amongst a peer-to-peer node network of computers based on multifaceted algorithms.

Instead of being stored in a centralized database, timestamps of a block’s transactions are stored on all nodes of the network. This elimination of mediators and decentralization of belief has presented a potential to make global payments, trading, and settlements faster, more reliable and less costly.

The following are some of blockchain's features:

1. **Decentralization:** Instead of one central agency monitoring the entire transaction within an environment, blockchain dispenses governance amid all peers in the transaction chain, thus generating a common public infrastructure.
2. **Digital signature:** Blockchain empowers an agreement or contract of transactional worth using exclusive digital signatures that depend on public (known to all the nodes) and private keys (known only to addressee) to generate an evidence of rights.
3. **Mining:** This is a distributed voting-based consensus mechanism where the system incentivizes miners for authorization and authentication of transactions and records them in blocks using stringent cryptographic guidelines.
4. **Data integrity:** The application of multifaceted algorithms and agreement among users safeguards that transaction information cannot be altered once decided upon. Information stored on blockchain works as the sole form of fact (truth) for all contributors working together, thus diminishing the peril of deception.

In addition, blockchain's attack and tamper resilient model can also be used for non-financial transactions. As it eliminates faults and redundancy, blockchain is perfect for altering a swarm of digital developments.

Significant benefits of blockchain comprise of the following:

1. Decrease in clearance time to mere fraction seconds by eliminating mediators.

2. Sparing reliable intermediaries with access by all contributors' node in the value chain to digital assets that authenticate each individual's identity.
3. Substantial security augmentation in fields such as payments and credit card scam using decentralized public ledgers that provisions particulars of each transaction and endures constant authentication by miners.
4. Decrease in physical cost through elimination of exclusive patented and registered structure.
5. Elimination of error management using simultaneous concurrent tracing of transactions, thus avoiding double spending.
6. Complete automation of transactional developments, from expense to clearance.
7. Elimination of paperwork-based jams caused by redundancy.
8. Risk management using data integrity safeguarded by sequential storing of data prescribed with cryptography. It decreases the regulatory load and slashes compliance investment in field such as know your customer (KYC) activities.

Blockchain can also empower access into marketplaces that have conventionally been subjugated by financial institutions. In the current digital age, financial institutions (banks) have realized a growth in rivalry from non-financial institutions companies such as digital wallets, mobile digital payments, and loaning. Blockchain is expected to strengthen such a race, as it will diminish technical blockades for digitally knowledge-based non-financial institutions. The following are some instances of these use cases.

1. **Permissioned blockchains:** Enterprises can generate blockchains limited to hand-picked customers for a precise drive. They can create a permission-based ecosystem that can transfer currency and digital properties online to settle business transactions.

2. **Liquidity originator:** A blockchain-based ecosystem can permit enterprises to become liquidity creators and initiate money exchange for carrying a global payment transaction at a lesser exchange rate. This will allow non-profit units to contest with traditional financial institutions.

3. **Equity capital:** A blockchain smart contract ecosystem can offer crowdfunding of equity capital.

4. **Hybrid loaning:** Enterprises can seek financing from blockchain-based node of networks as potential investor. Since such investors will have lesser functioning budgets than old-style financial institutions, they can work with minimal interest rates.

For financial institutions, this ought to be an indication to match the gap in this field, conceivably by generating individual arrangements of these platforms on a blockchain, as modern enterprises armed with knowledge and independent of governing compliance necessities could make swift pavement into their old-style headlocks. This indicates that blockchain is also anticipated to generate a new set of prospects for financial institutions to association with next generation digital enterprises, discovering role in new business arenas. These prospects include the following:

1. **Internet of Things (IoT):** Smart devices can be permitted to transact independent operations using blockchain-based smart contracts.

2. **Following healthcare payments:** Blockchain environment could safeguard that care payment is expended completely on healthcare transactions. The system can save time duration of settlement after every operation, supported by straightforward processing.

3. **Exchange anything:** Blockchain could empower exchange for any pooled digital asset or computing resources in exchange for a service or goods already contracted upon.

10.1.2 Business Transformation Using Blockchain

Blockchain's radical nature is consequential from its capability to change practically any process, from elementary paperwork to work for intricate contracts/agreements across the globe. This integral competence is appealing to financial domain policymakers, who have faith in its innovative power with respect to their business. Their assurance is reproduced in many surveys that many enterprises are in watch and act mode, investigating or already working on solutions using blockchain technology. Blockchain's innovative effect will encompass financial institution back offices to worldwide financial organizations.

10.1.3 Decentralized Business Payment Clearance

Business payment clearance (settlement) processes presently need many days (at least 2 to 3 days) for expenses and securities to exchange transactions. Transferring this process of payment clearance to a decentralized ledger can have a revolutionary outcome on investment marketplaces. This necessity is not restricted to stock shares and debt tools, and it can also be protracted to multifaceted tools such as financial offshoots. Important incentives for financial institutions to position blockchain in investment markets include the following:

1. **Reduced operative investment:** A business payment clearance platform could eradicate or alter the character of mediators, ensuing in reduced charges and other expenses.
2. **Worldwide business:** Working on this model will permit continuous business worldwide by preserving securities on a decentralized ledger, permitting business outside prevailing provincial schemes.
3. **Augmented faith:** With all the transactions stored on blocks being visible, it even provides the distributed alteration proof account book, while enhancing trust based on hash arrangement and at the same time helping the investment markets to grow.
4. **Diminished risk:** By performing the stored individuality, transactions in instantaneous online, decentralized system rally movability would eliminate attack on data, decrease cluster risk, and

recover the time taken for guidelines of practice of short-selling a tradable asset.

5. **Compliance publications:** Relaxed entry to transaction data for compliance auditors would diminish the cost of compliance reporting for market contributors.

10.1.4 Decentralized Business Finance

A business finance platform with memo of credit, receipt of shipping, and signature solutions built on blockchain comprises of the following characteristics:

1. Shippers receipt of shipping on the blockchain as a digital property.
2. Financial institutions provide memo of credit as a digital property on blockchain platform.
3. Multiple party-based digitally signed smart contracts.
4. Smart contract empowered, event reliant capital release to guarantee swiftness, and transparency.

10.1.5 Paper Authorization (Signing) and Accounts Administration

Decentralizing paper authorization would permit organizations to perform signature on the latest documents and validate their legitimacy. Such a platform would permit the following:

Quick Tip

Authorization is the function of specifying access rights/privileges to resources, which is related to information security and computer security in general and to access control in particular.

1. Easy distribution of validated papers.
2. Decreased time to take users on to the platform.
3. Definite execution of the latest version of papers.
4. Faster shared multiple members-based system authentication.

10.1.6 Distributed Identity and Access Management

Currently, the customer information is stored in a centralized fashion. Identity management is a tedious task and increases operational cost. Decentralized distributed identity and access management solution would lessen the pressure on the existing centralized method of storing customer data. By recording information in blocks and consuming it in a hash format that cannot be tampered, financial institutions can optimize the safety of stowed identity, advance portability of information, and decrease the time taken for KYC pains.

10.1.7 Blockchain Deployment

Despite the rapid increase in interest and activity over the last few years, blockchain is still in its primary phase. The novelties in the current financial institutions blockchain setup are at several phases of internal experimentation. Variations sustained by blockchain, such as storing information in multiple positions rather than in a single central node, characterizes a sweeping swing in the technique of financial institutions functioning. This could be the biggest obstacle to overcome in terms of administrative characteristic. Nonetheless, given its revolutionary latent, financial institutions would be misguided not to initiate actions toward integrating blockchain into their financial systems.

The following are a few significant early measures financial institutions should deliberate when realizing a blockchain platform together with current systems.

1. **Innovation identification:** An important query to request before initiating an experimentation is the process to transform to blockchain, which can be complicated. Blockchain is fundamentally a public distributed database, and financial institutions have generally trusted database management methodologies to store and regulate access to information. Making an innovation team that discovers the advantages and disadvantages of transferring a process to blockchain would be the perfect solution to initiate the blockchain journey. Such a team would function like an independent organization and discover parts where blockchain can add worth, while adhering with the deliberate objectives of the financial institutions.

2. **Feasibility analysis:** This includes considering the paybacks and expenses of transferring a process to blockchain. Captivating the viewpoint of key shareholders and associates wedged by the change is important.

3. **Experimentation and prototype:** Not all actions will have the same potential as storing data at multiple nodes. At this stage, proof of concept (POC) will be required and present the real-world simulation for the application ready to be deployed as a decentralized application.

4. **Governing and data security ecosystem:** Outside parameters such as compliances play a vital role in the blockchain age. The present governing framework does not have necessity for accepting a technology that could eradicate mediators. Storing information on servers in various countries will also require financial institutions agreement with data privacy regulations, which may differ from one nation to another. Likewise, there is no outline of regulations to make smart contracts work in the investment markets as they happen today. While controllers will ultimately change, it will be vital for

premature movers to entrench this feature into their longstanding strategies.

5. **Blockchain characteristics open/permissioned:** Most financial institutions are recognized to be functioned on closed blockchain solutions. Once agreed on the technology's growing state, it is logical for financial institutions to hold control by deploying a central manager to approve blockchain transaction. The full benefits of decentralization, such as lesser operation costs, cannot be realized without leaving control. This permissioned method will make a lot of sense in future, but as platforms develop self-sufficiently, business players will be stressed to comprehend the right paybacks of a blockchain platform.

6. **Economies of scale:** The Bitcoin forum looks for the finest method to increase the transaction processing volume of blockchain from the existing transactions per second, as practical situations would need financial institutions to process much higher number of transactions.

10.2 Blockchain in Insurance

Blockchain can be offered across the insurance sector due to its capability to provide lasting tactical aids such as reduce operative costs in the form of lesser replication of processes, condensed counterparty perils, augmented process automation, and protected and decentralized operations. Figure 10.1 shows the relevance of blockchain in the insurance sector.

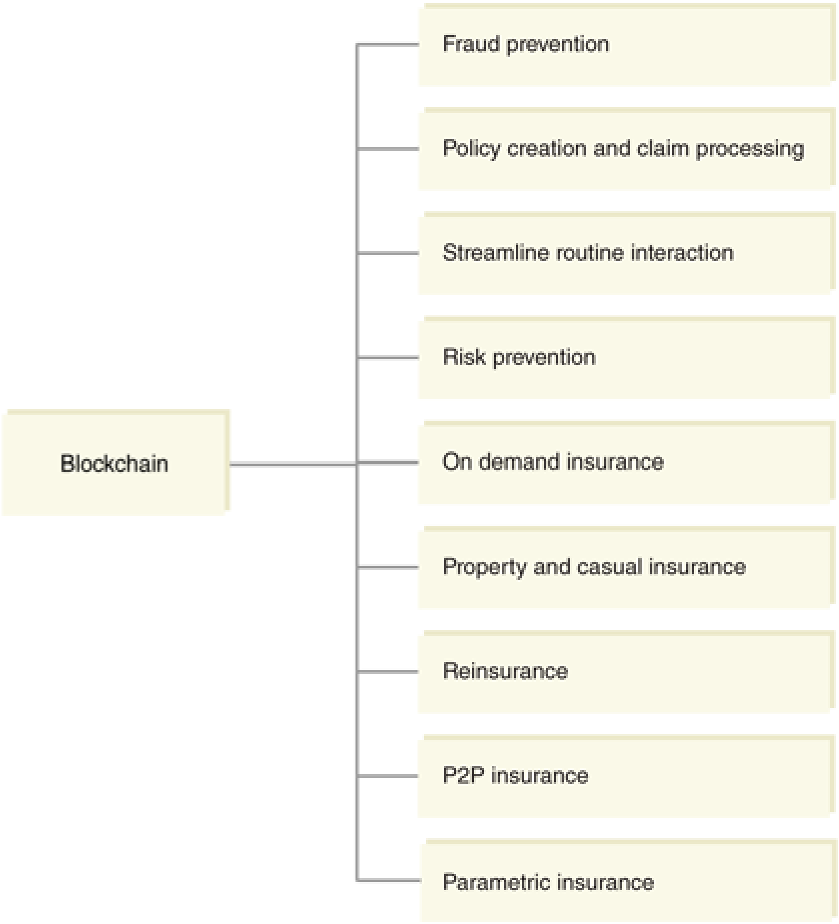


Figure 10.1 Blockchain in insurance.

Likewise, to each sector, the growth of the insurance has experienced a big development in which it has exhibited to adapt to several risks, kinds, and the amount insured. It contracts with persons and produces agreements between the two parties – the insurer and the insured.

The objective is to cover either their assets against mishap or the insurance of life. In the subject of a contract, a premium is to be paid conferring to the type of agreement, the position, and the worth of the product, good, or asset to be insured.

In relation to insurance, the assets remain indeterminate as only some of the agreements will experience some indemnities, at the same time most agreements can go a long time without any mishaps. Consequently, the agreement will be recognized for some initial years. It will depend on an automatic renewal based on the accident it suffered and the subsequent premium rise to protect the solvability of each agreement. Thus, some agreements can be declined due to health conditions, age, and period of the agreement. Life agreements have numerous durations that can be selected based on the demand and cautious understanding, from a one-month old baby to a whole life. Premiums amount differ depending on the demanded period and the customer.

Today, the market is changing rapidly, even insurance offerings are accessed through portals and applications. This atmosphere consequence leads to a low-interest rate inclination that diminishes the product available in excess. Focus is made on marketing and promotion as belief in insurance is the least compared to other customer-based businesses.

10.2.1 Health Insurance

As mentioned above, application blockchain in the healthcare sector has the potential to cover the ledgers in the form of records of patients within the operational blockchain. Application of such a technology in the healthcare sector would not only enable history of patients but it would also speed up the insurance process, thus decreasing a patient's concern regarding clerical dealings while at the same time taking care of the patient's health.

10.2.1.1 Use Cases

In this subsection, we will discuss several use cases in this sector, which will help draw analogy for usage.

1. It can be a scheme for generating healthcare tools and platform on the Ethereum blockchain and it offers patients management over their information. It can take shape to store permissioned blockchain that can be leveraged in healthcare facility to retrieve a patient's data

visible to insurance companies to receive in a real time to quickly process the claim.

2. It can be decentralized node-based system for medical histories. The scheme catalogues medical histories on blockchain, and key to the related parties to envisage these accounts. Using this technique, it can help to develop an audit trail that is easy to discover and authenticate, while at the same time maintain patient confidentiality.

10.2.2 Asset Insurance

Blockchain can be leveraged to catalogue, record, and examine the accountability for high-value properties such as jewellery. It can also authenticate the proprietorship of these possessions.

10.2.2.1 Use Case

Asset insurance can use technologies such as blockchain, smart contracts, and M2M vision to mitigate risk and diminish scam. Asset time-lapse method can be used as a tracking action reliant on a blockchain-based ecosystem for a high-value asset (such as jewellery) industry. The aim is to link all trade associates' counting manufacturers, sellers, and consumers to recognize a high-value asset transaction history from the origin to the ultimate customer; the business goal is to be able to establish legitimacy, transparency, and origin.

There is a requirement to guarantee the asset time-lapse method availability for all, safeguarding transparency along the whole asset life that assures the customer and enhances business performance. The methodology needs to develop a global digital ledger that traces and safeguards noteworthy assets through their period by collecting and registering on the blockchain with relevant data such as characteristics, history, and ownership.

This encrypted data is accessible for investors to guarantee the source of their assets' legitimacy. Data is also accessible for insurance business and titleholders. With this information, it can have the ability to trace high-value assets (e.g., if it was bought on e-tailers). It also helps with insurers when assets are stolen and are perhaps passing boundaries and entering black marketplaces.

10.2.3 Marine Insurance

The shipping method often includes a substantial number of members that create the processes, which are very complex, and increases the fault boundaries due to deficiency of documentation, data gaps, etc.

10.2.3.1 Use Case

Marine insurance utilizes blockchain technology as the foundation for supply chains that enable the association of businesses by deploying an exclusive shared sight of the transaction including specifics, secrecy, and discretion. Communication is relaxed among transporting channels, cargo movers, transporters, harbour, and terminal workers by providing access to delivery documents and online information by making use of IoT devices to access varied data such as temperature, vessel weight, etc. This comprises of the following parts:

1. The blockchain launches a method to trace possessions in its different phases (workshop, field, and destination), allowing every contributor to examine the development of the product online. It enables auditors as well as consultants to examine the complete process by inspecting the block instead of demanding information for every object.
2. A collection of fundamentals to ease information distribution within diverse units such as harbours, shipping firms, etc.
3. Application programmer interface to write new tools on top of the blockchain-powered platform.

10.2.4 Bail Bond Withdrawal

A bail bond is a document of credit guaranteeing imbursement of the entitlement. Even if the claim entitlement process takes an extended period, this document cannot be cancelled except if the entitlement is entirely rewarded. For example, if a ship smashes into a harbour and damages it, a bail bond is to be exercised. The problem is that these documents are developed on paper and require to be physically cancelled which is a complex procedure, particularly in developing nations.

A blockchain can simplify the procedure by recording the bail bond, relating it to the claim procedure, and inevitably cancelling it as soon as the entitlement is fully rewarded. The bond could be stated within the additional clauses of the insurance agreement either by a delivery authorization from the harbour's financial institution or authorization of closing payment application.

10.2.4.1 Home Insurance

Assessing and dispensing entitlement-based claims can be somewhat problematic due to the non-existence of data and information that is completed manually, which results in numerous mistakes. Though tracing assets insurances are independent in most situations, blockchain attached with smart contracts and a perpetual audit trail can simplify entitlement claim procedures and lessen the chances of error.

10.2.4.1.1Use Cases

Some of the use cases are discussed below:

1. The use case lets owners to get a provisional temporary insurance guaranteed by the insurance company and which is liable on the rent period. The blockchain offers a timestamped, tamper-proof, online immutable record, and low-cost policy in contrast to old-style home commercial policies.

2. Another use can work as a captive insurance powered by blockchain. It helps to consume API. The POC can be written to agree to guidelines and disbursement agreements, and it is recognized for real estates and specialized insurance. The tool is selected to record policy regenerations, claim dispensation, and premium payment.

10.2.4.2 Auto Insurance

When an individual takes an insurance policy on a used automobile they recently purchased, both owner and insurer would preferably try to assess the entire past of the automobile. This would be irrespective of whether it covers original equipment manufacturer parts or common forms, whether it was involved in an accident or suffered damages due to flooding, and even whether earlier possessors recurrently changed the oil.

While using the blockchain-based transparent framework, the following roles can work like nodes:

1. Automobile manufacturer
2. Parts ancillary
3. Sole dealer
4. Repair workshop
5. Guarantor
6. Driver

Even the specific parts that go into a vehicle can be accounted for, which makes a solid platform for maintaining automobile history in comparison to the present system, in a formation that is distinctly protected from data breaches.

Just as significant, however, is that this information would be accessible to a customer to authenticate that the automobile is in good condition and that in case it was involved in a crash, it was correctly and carefully restored.

10.2.4.2.1Use Cases

Some of the use cases are discussed below:

1. The blockchain-based car insurance use case can work like an IoT review, with better-quality telematics and an experienced cluster of professionals to offer disciplined drivers a decentralized application that recognizes cryptographic currency payments for their automobile insurance. The blockchain methods are modified in order to shape an application that provides the driver with a rudimentary and reasonable insurance resolution with exclusive transparency.

   The application can be a novel insurance plan on-request based on safe driving history. The use case goal is to generate a safe driver forum and increase awareness of safe driving. Having an option with a utilization-based insurance product, the use case can offer a distinct method to contact with each customer, improved risk mitigation, improved fraud examination, and complete driver safety.

2. Another use case can work with a system of insurers to communicate the history of past records among insurers for automobile insurance.

   Nowadays, a customer should demand his past records history from the earlier insurer to transfer it to the next insurer, keeping in mind that each insurer has its own arrangement of report. This new use case would simplify the data transmission and speed up the process. The challenge is if a scheme is to be recognized within the blockchain, the entire process must be transformed in advance, which is not the case at present. The solution, therefore, is to standardize the record of information and transform it.

10.3 Life Insurance and Claim Processing in

Case of Death

Insurance due to death and claim processes has not changed much in years, regardless of the numerous technological advancements and increase in automation. For most insurers, this practice is an exhaustive, unproductive process that is sensitive to fraud and results in claim experts being overburdened with paperwork. As consumer care takes focus for most insurers, the process is anticipated to change to guarantee least disturbance to recipients. Thus, the emphasis for most establishments is to advance the recipient experience by diminishing physical touchpoints, dropping deception, and improvement in on-time payment of claims.

Developing technologies are driving insurers to reconsider their instruments and practices across the system.

1. **Documentation-based human intervention reliant processing:** This includes the following parameters.
   - **(a)** Physical authentication and data entry.
   - **(b)** Non-existence of structured process.
   - **(c)** Data in different systems.
   - **(d)** Multifaceted preservation of papers.
   - **(e)** Bulky labour-intensive manual effort for claim payment.
2. **Automation-led processing:** This includes the following parameters.
   - **(a)** Guidelines-based systems for claim settlement.
   - **(b)** Lessening of manual intervention.
   - **(c)** Automated credentials systems.
   - **(d)** Resource reduction for claims settlements.
3. **Recipient-focused processing:** This includes the following parameters.

(a) Refining recipient experience.

(b) Enhanced on-time payment.

(c) Diminished touchpoints.

(d) Improved fraud discovery.

10.3.1 Death Registration

Death registration across the world has an analogous flow, with roughly nation-oriented deviations. The process works at a high level as follows.

1. Reason and type of death report/declaration qualified by the last appearing physician is required. A coroner declaration is also required if the death is not normal/unusual.
2. The above-mentioned doctor declaration is sent to an administration agency state-sanctioned cremation centre, which issues the declaration of cremation. The municipality issues the death certificate.

The above process may differ from one state to another.

10.3.1.1 Process

In receipt of the death certificate, the recipient can give it to the insurance company to start claim payment. The insurance company authenticates the particulars to execute the claim. This procedure – from claim instigation to claim disbursement – takes a long time, sometimes even months, depending upon the insured's data and reason of death given by the party.

Occasionally, insurance enterprises authenticate public death record ledgers to see whether any departed individual from the record holds polices with them, and if an entitlement has been started for the covered deceased. If there is no entitlement administered for the insured, the insurance benefactor then takes the essential steps to start and execute a death claim.

Death registration and claim filing are long, monotonous processes for the recipient, particularly at the time when an he/she has just suffered the loss of a loved one. Because of manual interventions, various claims are overdue up to several months, which can dishearten the recipient. In addition, the probability of fraudulent claims are increased since there are numerous information sources existing in separation across the sequence.

10.3.1.2 Blockchain Solution

We understand that there are numerous labour-intensive contact points in death and claim registration and settlement processes. The business desires a combined single process to protect information availability and transparency to several autonomous bondholders. Developing an integrated solution would necessitate deliberation on the following:

1. **Information security:** This requires the following parameters.
   - **(a)** Regulation-based execution to guarantee information security, compliance governance, and access-reliant control.
   - **(b)** Transaction information audit to obey guidelines and to achieve compliance trials.
2. **Shared network:** This requires the following parameters.
   - **(a)** Deliver change/modify access to transaction information to various contributors.
   - **(b)** Guarantee information access to proposed contributors at any juncture, thus reducing the turnaround time.
3. **Disintermediation:** This requires the following parameters.
   - **(a)** Start a public platform by permitting numerous shareholders to directly change transaction information.

(b) Create trust between members by guaranteeing safe and transparent transaction procedures.

4. **Integrated, rationalized process:** Integrate distinct but reliant death registration and claim processes into a sole, rationalized process.

10.3.1.2.1Use Cases

Smart contract-powered permissioned blockchain eradicates death registration and death claims processing challenges. As discussed in the previous sections, a blockchain-based proposal could support and produce an updated, integrated solution.

One conceivable method would shape a private blockchain solution that would integrate both death registration and claims processing. The following nodes can be in the blockchain system – insurers, hospitals, cremation homes, and health departments.

Let us consider a case scenario in the event of a death. The phases in the anticipated process comprise the following:

1. Assume a death occurs in a hospital or clinic. The hospital records the particulars of the deceased such as time, cause, and nature along with other specifics into the hospital accounts. The IT system of the hospital would be unified with the blockchain system. As soon as the data is recorded into the hospital's account, it would be passed onto the blockchain nodes. The blockchain would guarantee that the information is communicated in a safe manner by means of cryptographic hashing practices.
2. This data is recovered and used by an insurer to evaluate a probable insured record. Blockchain would permit guidelines-dependent transaction processing to safeguard that the data is acknowledged by planned receivers only. For a positive match, the insurer associates the recipient and requests for a cremation home using online web portals.

3. The recipient chooses a cremation home using an online portal where a series of accessible cremation homes exist. The blockchain system would assure that this information is concurrently accessible to the insurer and the cremation home in a safe manner, thus safeguarding the continuous movement of the data.

4. The cremation home accepts the request from the recipient and starts the death registration. The cremation executive logs in to the online portal, which is connected with the blockchain system. Rules are made in the blockchain system to guarantee that the particulars now exist on the death registration system that has to be entered by the cremation executive. The particulars are recovered from the information that was entered in the hospital. The executive does not have to communicate with the hospital staff regarding the accounts, thus diminishing turnaround time.

5. The death registration format is distributed to the national health department via the network. This format helps create the death certificate laterally with the funeral authorization. The certificates are distributed via the system to the insurer, cremation home, and recipient. The blockchain system guarantees that the death certificate is accessed by concerned members simultaneously in a safe method, thus avoiding several interactions. This decreases the turnaround time significantly, without risking information security.

6. The insurer obtains this data, executes the death claim, and pays the claim sum to the recipient. The claim sum is calculated by processing smart contract on the blockchain system. When the claim sum is identified, the obligatory sum is paid to the recipient.

7. On getting the claim sum, the recipient recognizes the receipt, thus closing the process.

As already mentioned, the whole procedure could be processed by a shared smart contract present within the blockchain system. All documentation produced as part of the execution would be recorded on a distributed database, and the hash address stored on the blockchain. This automated process would guarantee that the responsibility is no longer on the recipient to process all the required paperwork and to follow up with managers for executing death claims. Now, the responsibility would lie on the insurer to execute claims. In this way, a troublesome workflow with several procedures is transformed into a well-organized workflow.

The operation choices comprise of the following:

1. Several insurers have claims systems that could be united via APIs and governed by event reliant on causes in smart contracts. In one state, the death registration could be agreed entirely on the blockchain, and the policy and claim processing could be managed on current systems, with data passed using interfaces.
2. The unconnected method would have the logic integrated for death registration, claim processing, and policy on the blockchain. Information could be provided to pre-approved contractual conditions kept as smart contracts.

The following are advantages of such a system:

1. The time of the entire death claims execution is significantly decreased to 3–4 days, from several weeks and months.
2. The recipient is not troubled with gathering death-related papers, depositing numerous forms, and working with the insurer for payment of the claim sum. Death registration and claims are started and executed by the nodes in the network.
3. All the accounts are accessible on a public ledger and can be inspected at any time with simple tracking.
4. The solution can be ascended to comprise several agencies such as governing bodies, insurers, hospitals, and citizen charter services, which will increase the scope of the insurer.

10.4 Healthcare

Nowadays, healthcare scholars and consultants are facing challenges due to fragmented and isolated information, deferred messages, and dissimilar workflow operations. A one-side provider feels hesitant to share information due to the following main reasons.

1. Observations that patient health and credentials data protection guidelines avert such distributions.
2. Possible accountability and financial penalties related to information distribution.

On the other hand, vendor explicit and mismatched health systems generate breaches in healthcare interactions, making it tough to organize and deliver patient care.

Nowadays, a challenge in production healthcare arrangements is the absence of protected relations that can link all self-governing health systems together to start an end-to-end accessible system while shielding healthcare specialists with an appropriate degree of secrecy. Though information standards deliver basic interoperability for information sharing among trusted networks, this degree of interoperability is restricted to the comprehended standards and desired information mapping among systems. Management of these systems is also complicated as change in communication in one system requires other parties in the trusted system to accept the transformation as well.

There are various complications to realize interoperability in healthcare system, including mismatched software and absence of access to information external to a healthcare system.

A possible solution to these glitches encompasses the use of blockchain technology, which delivers trustless transactions through decentralization with autonomous functionality. However, intricacies within the healthcare industry create additional difficulties to deploy blockchain technology.

This section discusses the central characteristics of blockchain technology, which can support in starting trusted associations and other competences in numerous possible applications and examines important development deliberations for developing decentralized applications in the healthcare industry (Figure 10.2).

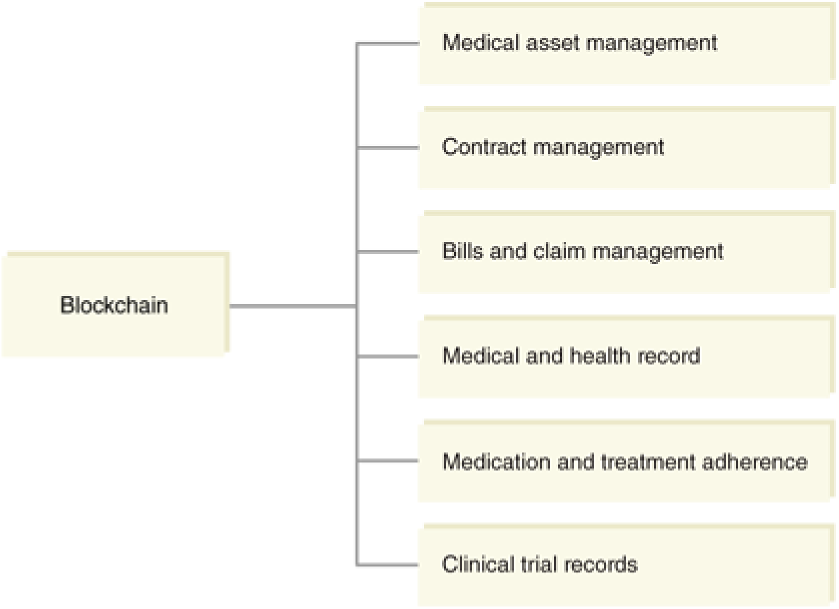

Blockchain
Medical asset management
Contract management
Bills and claim management
Medical and health record
Medication and treatment adherence
Clinical trial records

Figure 10.2 Blockchain in healthcare.

10.4.1 Issues

Nowadays, the persistent matters in healthcare is interoperability. It is encountered with restricted information distribution and patient-focused maintenance adopted by healthcare but interoperability would permit patients to access and regulate their own health data.

10.4.2 Interoperability

Healthcare interoperability defines the capability for diverse IT systems and software tools to interact, distribute data, and practice the shared information. Permitting data systems to work collaboratively internally and beyond the enterprise firewalls is important to improve real maintenance of delivery for people and entities. For instance, interoperability empowers providers to firmly distribute patient medical data in scalable fashion, irrespective of provider place and trust associations among them.

Safe and accessible information distribution is crucial to deliver effective cooperative healthcare diagnosis for patients. Information distribution supports improved investigative precision by collecting validations or commendations from a team of medical professionals, as well as averting shortfalls and mistakes in treatment and medication. Similarly, collected patient history records help doctors comprehend patient requirements and results in more effective treatments.

For instance, a team of physicians with diverse subjects in cancer care create tumour panels that regularly work on cancer cases, share information, and govern real cancer handling and care strategies for patients.

Another instance, if a cancer patient under treatment is admitted to an emergency room in a different hospital, then it would be life-threatening if the healthcare provider cannot get details of the patient's medical history to recognize possible drug administration. Moreover, the patient's main cancer care provider should be informed of the situation.

Notwithstanding the rank of medical information distribution, today's healthcare systems need patients to attain and distribute their individual medical records with other providers either via physical documents or electronic forms. The course of finding and distributing medical records faces many problems, such as the following:

1. The process is sluggish. Meanwhile, replicas of medical information must be ready, distributed, and chosen by patients. The rule permits providers up to 30 days to stream medical histories to patients, though some providers may send non-critical health data in 5–10 days.

2. Information replicas may go missing or stolen during their physical transfer from one place to another.

3. Patient health record may be disjointed because their information is stored in contradictory and isolated systems. There is no single system that stores all the medical histories of a patient, so a patient must be accountable for keeping track of all the medical records and information of health facilities where they received treatment.

4. Nowadays, healthcare arrangements are more provider focused than patient focused, thus discouraging patients from regulating their individual health histories and assessing if their health records is complete or who has retrieved their information.

The unproductive information distribution process in the healthcare industry results in part from the absence of faith between providers and the non-existence of interoperability among health IT arrangements and applications. Healthcare interoperability includes three stages, which are listed below from least to most reliable.

1. **Introductory:** Introductory interoperability permits data transfers among healthcare systems. It does not require providers accessing the data to be able to understand it.

2. **Operational:** Operational interoperability defines formats for transferred medical data and confirms that the received data is conserved and can be explained at the data field by means of various set-ups.

3. **Semantic:** Semantic interoperability requires explanation and meaning of transferred information.

These three stages certify that dissimilar health systems provide data with necessary information quality and security. Introductory and operational interoperability are rudiments for semantic interoperability, which is most difficult to accomplish but most expected to develop excellence of care.

Now, we will focus on discovering blockchain-based applications that support introductory and operational interoperability from a practical infrastructure's viewpoint. Semantic interoperability needs medical domain expertise and medical plans that implement the acceptance of metadata such as mutual information standards and to understand innumerable foundations of health data.

10.4.3 Patient-Focused Care

The healthcare industry is moving from volume-focused care to value-focused care. In volume-focused care, providers are rewarded to deliver more actions because reimbursement is directly related to the amount of care. On the other hand, value-focused care endorses patient-focused care with advanced excellence or value, in which patients are conversant and joint together in medical decision-making. In patient-focused care, patients can manage the experience standards and verified results, such as indications or health position, gathered from their wearable devices. Patients can also get easy access to their medical histories with complete information of their entire health past, which would decrease information division and imprecision caused by message interruptions or management mistakes and in turn advance care.

Preferably, all health systems would deliver automatic statements for patients to access their medical data instantaneously, as data is entered into the system or when pathological reports are accessible. Also, in patient-focused care it is essential for patients to govern the time and receive their collected health information and to select what fragments of data they would be willing to distribute. However, healthcare systems nowadays do not deliver the resources for patients to adjust or withdraw a provider's access to their statistics. As a consequence, once a provider has taken an interest for a patient or has attained access to patient information, that information is forever in the ownership of the provider. When a patient goes to different healthcare providers throughout their life, their health and other personal information is accessible at numerous places. This dispersal increases chances of data theft since it only warrants one provider that lags adequate security measures to put patient data susceptible to outbreak. Interchangeably, a patient may request to issue his/her medical history to a new provider, which is mostly not attained in a straightforward manner.

Interoperability is also essential to provide a patient-focused model that progresses excellence in care for specific patients. Problems occur in the healthcare infrastructure, which obstruct interoperability and therefore patient-focused care, including the following:

1. **Information security and confidentiality apprehension:** Notwithstanding the necessity for information sharing, lack of a safe infrastructure increases the peril of confidential data attacks. Providers could encounter lawful penalties when data is negotiated.
2. **Non-existence of faith between providers:** Due to security guidelines, care providers must be capable to recognize other providers and also believe their individual access rights before any patient health associated statement occurs. Trust relations often happen among providers that use the same compatible health system with a mutually agreed provider directory, such as in an isolated exercise or a linked hospital network.
3. **Scalability apprehensions:** Medical information may comprise large volumes of statistics such as medical large volume images, particularly for cancer patients or for chronic circumstances. This large volume of data is problematic to transfer digitally due to limits in bandwidth or preventive firewall policies such as that in remote areas.

In order to overcome the above-mentioned shortcomings, we will discuss some use cases of blockchains in the healthcare industry.

10.4.3.1 Use Cases

Some of the use cases are given below.

1. Treatment tracing to distinguish overdose and overtreatment.
2. Information sharing to integrate telemedicine with old-style care.
3. Distributing cancer information with providers by means of patient-approved access.
4. Cancer records sharing to collect results in cancer cases.
5. Patient identity and access management for efficient patient account record matching.
6. Individual health histories for retrieving and governing comprehensive health history.
7. Automation of health insurance claim settlement.

10.4.4 Treatment Tracing to Distinguish Overdose and Overtreatment

It is extensively recognized that there is widespread illegal medical practice still existing in the world. While several efforts are being made to handle these concerns present in the prescription tracing system, there is still deficiencies in the technology to do so efficiently.

The following are some problems in the present prescription opioid market:

1. Digital hoarding.
2. Doctor shopping to obtain multiple prescriptions.

3. Provider unawareness.
4. Susceptible and centralized data.
5. Prescription in excess.

The decentralization, audit, and trail mechanism of blockchain technology delivers a hopeful method to prescription evaluation that not only handles prescriptions in a protected manner but also motivates (rewards) for advising less prescriptions.

Quick Tip

An audit trail is a security-relevant chronological record, set of records, and/or destination and source of records that provide documentary evidence of the sequence of activities that have affected at any time a specific operation, procedure, or event.

Unfortunately, healthcare benefactors nowadays are rewarded to recommend opioids to patients. For instance, benefactors experience lesser face-to-face interactions with patients, less costs linked with patient care, and therefore better returns from higher revenues. Similarly, chemists are rewarded to yield and dispense opioids. Subsequently, the more they trade, the more their revenues and the better their profits. Furthermore, patients are rewarded to use more opioids. Handling problems of pain, physiotherapy, or post-surgery recuperation can be unsatisfying and peppered with dissatisfaction. Opioids deliver a temporary respite, although at the risk of making a patient habituated to the drug.

To balance the enticements that lead to the growth of opioid epidemic, a blockchain-based arrangement can launch a reliable system of hospitals and drugstores to stock opioid-related dealings with prescriptions and fulfilling the orders in a safe and answerable manner. Such shared and public/private permissioned blockchain mechanisms would permit attached providers to use other information silos without unambiguous trust associations among each other. Shareholders within the system such as hospitals, chemists, and drugstores are similarly rewarded to board new affiliates to the association because with each extra participant, they can make a more comprehensive dataset. Rules can be jointly prescribed so the association can steadily board new benefactor affiliates to the system.

More comprehensive past opioid prescription can be accessible to excessively recommended opioid by benefactors and also outlines of doctor shopping patterns can be judged. Therefore, benefactors will be rewarded to meet the necessities to join the association of providers through possible access to information that will improve the excellence of their care. Most significantly, by tracing the past of opioid treatments, patients will get care more suitable to their ailment and thus be directed away from the hazards of opioids to less habit-forming lifelong treatment arrangements.

10.4.5 Information Sharing to Integrate Telemedicine with Old-Style Care

Conventionally, telemedicine delivers available treatment to patients who are positioned in remote parts and far away from local health amenities or in zones which lack health amenities and specialists. This has become progressively prevalent among patients who require to accept suitable medical care. Linked patients can avoid wasting time at clinics and get instant treatment for minor ailments. Nowadays, due to mounting approachability to smart devices, many organizations deliver 24 × 7 access to upkeep, and various accessible tools have been shaped for patients to evaluate, govern, and state their health using technology.

Telemedicine facilities are typically equipped with more cutting-edge technologies and are much more extensively associated to old-style physical health facilities. Because of online services, it is mutual for providers from diverse areas or systems to treat patients, resulting in a decrease in care endurance. Health information gathered throughout telemedicine care may be unapproachable by care providers, which generates an inadequate medical past and may result in improper treatment.

By eliminating the requirement for a third-party agency and authorizing communications directly among involved members, blockchain can link the message barrier among the providers. However, blockchain solely cannot handle the multifaceted information sharing task. It must be combined with different health systems and health information standards. We can expect a tangible infrastructure where blockchain is linked to dissimilar health database systems. Each database system can depict a new protected information channel, comparable to what is utilized to share information with other analogous systems. A smart contract can be utilized to manage the information transactions among systems grounded on common contracts and also generate past transaction histories. In the end, a strong architecture will comprise several more design mechanisms to implement this.

10.4.6 Distributing Cancer Information to Providers with Patient-Approved Access

Cancer detection and treatment are seldom written in detail, that is, they may not include numerous deliberations due to complication of the illness and availability of various existing treatment choices. Receiving new viewpoints from various experts can help narrow down the choices and may decrease the time to find appropriate treatment for the patient. Today, hospitals are linked to at least one cancer association, which comprises a diverse team of medicinal, clinical, radiation oncologists, experts, and care providers to assess and deliberate a cancer patient's illness and treatment choices. The cumulative determination to inspire cancer care teamwork among cancer specialists, patients, and relations remains unreceptive in the treatment decision process. In the treatment process, a big hospital may involve experts from a broader range of expertise areas, while a smaller hospital may have inadequate possessions to increase their cancer boards. Also, the excellence of care for cancer patients may be ignored due to patient detachment.

In reality, patients may demand to talk to a new provider for a second opinion on their medical condition and existing treatment case. Nowadays, to provide crucial information, patients have to get replicas of their medical data from their existing provider. This information includes their family history, visit information, prescriptions, present treatment choices, etc. All the information will then be distributed to the new provider in a physical form. In this technically progressive civilization, patients with serious conditions should have minimum participation in information sharing process. The sensitive information should be shared in an appropriate method to avert interruptions in treatment. In the present health systems, patient-governed information distribution is absent for cancer patients to demand a second opinion and also share their data by choice.

Blockchain technology delivers the prospect for trustless conversation and disintermediation that permits present trust associations to be combined and proliferated across several establishments and providers. This method is analogous to patient recommendation procedure, especially if the recommendations are not restricted to a sole provider's system. In its place, they might be extended across diverse areas, zones, and even nations. A blockchain can also seize prevailing trust associations between patients and providers, thus permitting patients to choose the provider that can distribute their information and data.

10.4.7 Cancer Records Sharing to Collect Results

Information distribution is especially significant in cancer care, where situations are generally intricate and treatments are infrequently similar.

Capability to provide information guarantees integrity of outcomes collected from experimental trials by empowering specific authorization and justification, but it can also collect intellect accumulated to lessen unjustified replication in medical trials. It permits dispersed medical trials to accomplish a noteworthy unit size and thus speeds up the detection of more operative cancer cure. As a consequence, cancer patients receive treatments based on explanations received from a small group of extremely choosy patients with dissimilar geographies, family clinical past, second opinions, etc.

The records are efforts to gather basic information from cancer incidents across topographical zones and to develop inclusive cancer control. As with all health systems, cancer records are frequently isolated and disjointed, which can correspondingly influence blockchain technology for accelerated data exchange. In total, with augmented accessibility of more accurate information gathered from countless patients, cognitive sciences can be utilized to concept predictive and extrapolative models for supporting care benefactors with decision care. A knowledge environment can be envisioned using blockchain to provide analytical models and advance accuracies of scholarly medical insights.

10.4.8 Patient Identity and Access Management

An essential element in health data transfer is patient identification, which matches a patient in a database by means of an exclusive set of data. Systems have been formed to accomplish patient individualities inside a healthcare enterprise or inside a trusted system. Regardless of the augmented progress effort, matching patient information accurately and reliably is still a major problem. Patient identity mismatch leads to repeated patient accounts and flawed or unsuitable medical information.

There are likewise notable expenses to healthcare enterprises who preserve redundant records and rectify incorrectly merged mistakes and patients who undergo repetitive examinations or treatment postponements. These mistakes also influence compensation, as entitlements may be deprived because of old or inappropriate data, and the safety hazards becomes complex when patients reveal their personal data.

Lacking general standards for gathering patient identifying data can result in a patient's records to differ from one hospital to another. For example, geographical information, such as name, date of birth, address, and unique identification number (UIN) are frequently used to identify a patient.

Names may be kept in numerous formats such as first and last name, combination of nickname and last name, with or without middle initial, and identical or analogous names. Similarly, date of birth can be taken into the system in several ways, address can vary as patients transfer to a new place, and patients may not agree to give their UID or do not possess one. Also, patient data physically entered into a system may contain errors or mistakes, and the extra information collected might result in more chances for errors. A patient's information may be accumulated into a single UID within an enterprise, but at the same time the UID may not function in other enterprises.

Deprived of a practical, integrated identity and access management, patient credentials at several care units may be mismatched, except when he/she accepts care from an enterprise. The main feature of blockchain integrates such a decentralized, integrated identity network. At present, several blockchains use cryptography-based protected addresses to characterize individualities. Each address is scientifically associated with a unique identity-based key that is utilized to confirm the possession of an address or a uniqueness that does not reveal any specific data linked to the patient. The decentralized audit trail features of blockchain can implement consistent confirmable individualities for patients via a worldwide patient directory archive distributable across all healthcare units within a country and even outside of it. In case of a misplaced or stolen identity, a new report can be created and reallocated to the patient.

10.4.9 Retrieval and Governance of Individual Health History

In contrast to the present standard exercise of using provider-focused health records to preserve and govern patient information, individual health accounts are requests used by patients (the factual information owners) to access their health data. The goal for individual health records is to benefit patients steadily and to suitably gather, trail, and regulate their comprehensive health accounts gathered from numerous sources such as health provider appointment information, vaccination history, prescriptions archives, and information gathered by smart mobile devices.

Individual health account permits patients to govern how their health data is consumed and distributed, authenticate the correctness of their health accounts, and correct probable mistakes in the information. Organization and technology establishments have started discovering centralized resolutions with their health products tools. Centralized methods do not determine the information sharing challenge at its central point and may consequently face analogous point-to-point prevailing incongruent health systems.

In deviation, blockchains permit sharing of governance to entities via decentralization empowered by consensus procedures. By generating a generally accessible and safe data sharing service that links to present health blockchain network, patients can effortlessly combine their medical past without demanding a replica from each provider. Links to individual smart mobile devices are also conceivable as blockchains establish trust among healthcare experts and third-party well-being tracking applications and tools. In addition, permission-dependent information delivery can be established with smart contracts to assure that patients persist in governance of their information access, they are conscious of the source of accumulated information origins, and they are well-versed when their information is retrieved by providers.

Information source and access records are made transparent to patients through unassailable audit trails to constantly keep them informed of when and by whom their health data is repossessed.

10.4.10 Automation of Health Insurance Claim Settlement

Health insurance is utilized to shield specific possessions from the shocking expenses of a major health trauma or treating a chronic illness and to safeguard that care is offered when needed. It can incur the charge of doctor appointments, medicinal and operating expenditures, dependent on the category of the health insurance contract. Patients may have to bear out-of-pocket healthcare expenses but a major portion is treated as entitlements to health insurance organizations. Providers are then compensated using the claim settlement procedure, where the insurer regulates their fiscal accountability for the compensation and the imbursement quantity is given to the provider.

The insurer may choose to recompense the entitlement in full, refute the claim, or decrease the sum paid to the provider. The choice to decrease expense to the provider is characteristically done when the insurance establishment has identified that the payable facility is unsuitable or medically redundant for the treatment or practice codes. Consequently, it is important to safeguard that all claims meant for imbursement are encoded truthfully. As an insurance provider obtains a medical entitlement, they start a systematic evaluation. Occasionally even minor mistakes such as incorrect patient name may result in a claim to be rejected. Nowadays, most claims are processed automatically without human interference, but claim settlement still remains a daunting task as entitlements become further multifaceted and are inhibited by fault and scam.

Presently, one fourth of the claims are excluded either since they are not acknowledged by the insurer or they comprise imperfections such as unfinished or inappropriate information or non-existence of proof. Additionally, smart contracts provide a prospect for automating the settlement process by allocating and thus making entitlements transparent to the benefactor and insurer, revealing possible mistakes and deceptions that can be amended or examined in a more appropriate method. An advantage of generating these predetermined arrangements using smart contracts is to safeguard that participants are currently updated and correctly informed of the guidelines or any change in instructions.

10.5 | Assets Management

Assets are a comparatively fresh expansion linked to the diffusion of information technology (IT) in various sectors and services. Asset is a variable entitlement of a specific service or goods guaranteed by the asset provider, which is not connected to a specific account, and is administered using computing technologies, counting asset issuance, entitlement of possession, and transference. Assets have numerous after use cases (as given in Figure 10.3), including financial securities, smart property, authorized fiat money, limited communal currency, digital vouchers, digital figurines, and access and payment to specific resources.

Blockchain infrastructure for asset management permits generation of autonomous digital assets that could be transformative disruptions. Asset management could use security features of blockchains, which comprise the following:

1. Inconceivable of imitation.
2. Unassailability.
3. No mediation and effortlessness in transfer.
4. Transparency and effortlessness of inspecting with audits.
5. No burden associated to transaction dealing.
6. Network result carried by the integrated infrastructure for several types of tokens.

10.5.1 Assets on Blockchain

We must first consider the following minimum requirements for bookkeeping (ledger) of digital assets:

1. **Safety:** The ledger must consist of acceptable agreement rules to recognize possession and authorize assignment or exchange of assets.
2. **Forged confrontation:** The structure needs to have instruments to guarantee inconceivability of forging assets.
3. **Auditability:** The network must store all operations in order to allow audit of transactions as required by compliance agencies.

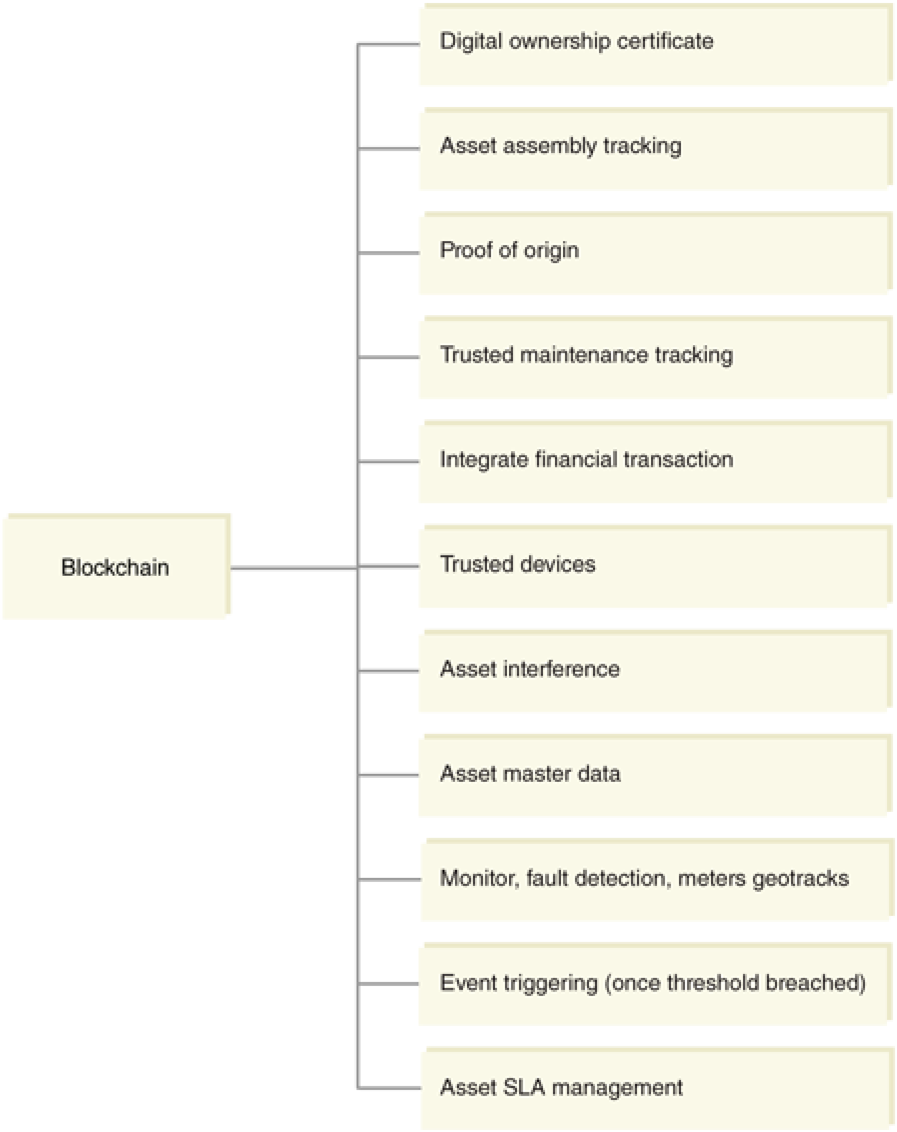

Blockchain
Digital ownership certificate
Asset assembly tracking
Proof of origin
Trusted maintenance tracking
Integrate financial transaction
Trusted devices
Asset interference
Asset master data
Monitor, fault detection, meters geotracks
Event triggering (once threshold breached)
Asset SLA management

Figure 10.3 Blockchain in asset management.

Figure 10.3 shows the relevance of blockchain in asset management.

There are numerous additional features that, while not obligatory, can influence acceptance of arrangement by users and governing regulators such as core unassailability. These regulators are confirmed by inherent features of fundamental-computing infrastructure from individual network operators (nodes).

One class of these assets is digital currency, where the asset is an entitlement to a practical currency. Centralized digital currency systems are generally performed in e-commerce transactions. In core unassailability, a centralized asset assignment organization offers its users with an online portal and a backend database to maintain ledger balances and operation history. The arrangement could practice a two-factor authentication verification that is susceptible to numerous attacks such as phishing. The arrangement also delivers traders with patented accelerators such as standard interface and toolkits in order to accept payments from clients.

Centralized asset network requires investments to backend computing systems, user certification, and governing regulatory agreement. Consequently, it is problematic to install and support the systems for small and medium establishments. The users and assessors of such arrangements could be worried about conceivable variability of transactions, disaster recovery of the infrastructure and its transparency. These apprehensions would be understandable in case the provider is a moderately small enterprise without adequate public status. For analogous motives, centralized asset-computing systems (with a sole body governing all characteristics) are improbable to seize customer-to-customer markets.

Blockchain ledgers deliver a substitute to centralized electronic asset management. Blockchain permits to lose (but binds) the elementary activities achieved by centralized digital currency providers. These tasks include the following:

1. **Transaction control:** This could be achieved in a decentralized fashion by physically dispersed shared nodes of the system. Furthermore, describing the guidelines for transaction operations could be fragmented from the process.
2. **Asset distribution:** Usually, this could be achieved by any node of the blockchain system.

3. **Safeguarding user's assets (investments):** This could be achieved by mediator parties using supervised or unsupervised digital wallets.

4. **Service identity and access management:** This could be deployed by developing public key-based token infrastructure reliant on blockchain.

5. **Application management:** This does not require support of blockchain developers.

Blockchains deliver a decentralized asset management mechanism, which could be less challenging and more tempting for asset suppliers, facilities, and clients. For asset suppliers, blockchain-reliant ledgers could be a specific platform-as-a-service. Though blockchains are not the only kind of specific platform-as-a-service, their fundamental characteristics such as augmented audit, log facility, and user safety make them more appealing than other possible all-purpose substitutes. Moreover, non-availability of dependence on a single provider and the linked reduced cost of transactions could be an added benefit in the situation of permissionless blockchain with a varied contribution of transaction operators.

The core elements of a blockchain-reliant ledgers are discussed in subsequent subsections.

10.5.1.1 Blockchain Description

Blockchain description controls the technique by which data is transmitted between blockchain network nodes and how the timestamp of the blockchain is devised reliant on the acknowledged information, such as grammar of transactions conditions. The conditions comprise of the following:

1. **Transaction conditions:** These are lawful transactions with respect to the current system condition. The instructions are on how communications alter the system state, and so on.

2. **Unassailable conditions:** These conditions show how transactions are clustered into blocks and how their headers are safeguarded.
3. **Agreement logic:** These conditions show how nodes consensus upon the condition of the structure, how forks (in blockchain) are fixed, etc.
4. **Network algorithms:** These conditions show how any activities, blocks, and other information are communicated/transacted between network nodes, etc.

Guidelines that are difficult to follow could be prescribed by a restricted sequence of asset operation authenticators. Moreover, centralized operations would present safety susceptibilities. Completely automated operations would be more useful with the perspective of security and due to augmented regulations, which could be accomplished by establishing requisites for controlling regulatory compliances such as audit in the blockchain description.

In an ideal situation, the requirements would include all guidelines of operations, empowering participants to produce direct agreements among themselves with blockchain systems defined as a resource of value chain other than an active participant. Logically, public blockchains could be associated with the World Wide Web (WWW) as a means of information transfer. The protocols such as HTTP and HTTPS do not replicate any financial condition. However, they are extensively applied in current digital financial services such as encryption using HTTPS. Similarly, a public blockchain could enable regulatory amenability for financial offerings without executing service-centric regulations such as mandatory customer credentials validations.

A publicly accessible blockchain description composed of open entree to blockchain tools could produce an ideal ecosystem for novelties and third-party use cases. While dealing a public description could be more difficult than registered/private description, the former makes straight relations with the overall essence of blockchains as consensus motivated networks.

10.5.1.2 Asset Attorneys

An asset provider using blockchain is not commonly needed to process transactions or to transfer information to the blockchain. These activities could be deputized to blockchain attorneys. Moreover, attorneys could be identified objects (in permissioned/private), or any users fulfilling technical abilities forced by a blockchain consensus procedure with respect to public/permissionless blockchains. Permissioned blockchains could be more helpful for financial organizations in the current scenario due to elasticity of the blockchain description and augmented compliance. On the other hand, permissionless could be more appealing for customer-to-customer markets and other M2M, IOT use cases due to characteristic without trust and permissionless access and exit.

Permissionless blockchains need incentives for nodes contributing in structuring and safeguarding it. This objective is achieved by familiarizing tokens, which are produced by generating blocks and by receiving transaction charges. In the case of a permissioned blockchain, attorneys are concerned in possessing the secure blockchain as it delivers them with a torrent of returns by executing services.

10.5.1.3 Network Nodes

A public network offers three safety methods for integral nodes:

1. Complete authentication nodes that validate and store each transaction in the system. This safety approach could be utilized by blockchain attorneys, compliance controllers, auditors, diagnostic services, and devoted blockchain service providers.

2. Easy payment authentication nodes, which can be utilized by a mainstream user, as this safety mode needs slight infrastructure resources.

3. Limited authentication nodes made conceivable with the support of separated observer and scam proof. The nodes can authenticate a minor proportion of transactions, while underwriting to the complete safety of the blockchain network. Limited authentication nodes could be functioned by service providers on the network.

In case of a blockchain with limited read rights, its architecture can be discovered by operation processors. For instance, operation processors can function as complete nodes, and users can be offered related transactions through easy payment authentication nodes or corresponding web interfaces. Therefore, blockchains with limited access can be less accessible or consistent due to irregular delivery of transaction processing.

There is a significant difference between easy payment authentication nodes and API rights to blockchain information. Though easy payment authentication nodes do not ensure the safety of the blockchain, they are widely accessible and possess the following characteristics:

1. **Exclusivity:** There will be a sole replica of a blockchain; numerous replicas can be inconsequentially distinguished and can only be qualified to the dynamic agreement of blockchain attorneys.
2. **Unassailability:** The blockchain cannot be altered or backdated, even by an agreement among attorneys, if it is done illegally. Such modifications should not be acknowledged by payment authentication nodes.

It is important to note that these features can be accomplished by pushing block formation using proof-of-work algorithm.

In case access to the blockchain is offered through APIs without revealing its architecture, consistently attesting exclusivity and unassailability becomes increasingly more difficult. Even if the controller or assessor has comprehensive entree to the blockchain, information offered to the controller can fluctuate from information assisted through API due to an eclipse attack achieved by conspiring blockchain attorneys.

10.5.1.4 Assets Authentications

Blockchain-dependent assets are carrier assets, and possession of an asset is discovered by the information of a private key. Two-factor verification and other safety procedures analogous to centralized e-wallet can be realized by means of devoted wallet services. Safety characteristics of public key can be enhanced by utilization of focused hardware wallets for authorization of transactions. Comprehensively, blockchain-computing resources deliver safe decentralization and eradicates central failure points characteristic to centralized e-wallet/currency ledgers.

With respect to preserving user privacy, blockchain nodes can employ tree level folders and recompense to agreement guidelines, which permit the formation of publicly unrelated addresses that will help demand-based assessment. Transaction sums can be concealed by means of range proofs.

As blockchain-computing resources offer comprehensive timestamping of events, it can be utilized to deploy decentralized public key, which will associate individualities of users and objects of public keys. Public key can be systematized as a portion of the blockchain description, or as a distinct intersection procedure. The key would permit for legitimately acknowledged value assignment and asset allotment.

10.5.1.5 Asset Allotment

Commonly, assets can be allotted by any blockchain node and grammar of assets would be executed by the allotter. As asset allotment is a distinct type of transaction; the individuality of the allotter could be described by conferring with usual user credential guidelines. A controlling agency can clearly admit asset allotment by signing the agreement transaction collected with the allotment, or by yielding the allotter a distinct type of the digital certificate.

Asset allotment can stipulate the type and sum of allotted assets and the individuality of the allotter. The following are additional asset characteristics:

1. An asset can be sealed or inaccessible, denotating that assets of similar type cannot be allotted in upcoming time by anyone, counting the primary allotter. This type of asset is beneficial for producing non-dissolving shares.
2. An asset can be noticed as divisible to larger decimal places.
3. An asset can be formed as non-movable in order to restrict secondary marketplace.
4. Added metadata can be linked with the asset, either straight or in the form of a hash.

10.5.1.6 Asset Deployment

An individual digital asset or a collection of assets preserved by the same allotter can possibly have its individual blockchain as permissionless or permissioned. Safeguarding a small-scale permissionless blockchain can be more expensive, as the fee of an attack on the network is relative to the fee of the blockchain token. A permissioned blockchain can be more susceptible to attacks, but it will still partake a central failure point in the arrangement of a sole operation processor.

10.5.1.6.1Isolated Assets Blockchain

From an assessing and auditing perspective, characteristics of an allotter-governed blockchain can be analogous to current asset systems. Due to centralization, the asset allotter working as a blockchain node can have a reward reporting false data throughout audits.

In this situation, recognized asset business pairs by means of distinct blockchains for each asset can be unproductive. The price of functioning an allotter explicit blockchain can be analogous to old-style asset systems due to the prerequisite to grow end-user services such as ledger tools. Moreover, isolated blockchains could hinder the formation of third-party tools and reduce the network outcome by demanding supplementary tools to communicate with other digital assets.

10.5.1.6.2Blockchains Based on Several Assets

Several assets can be naturally maintained by a blockchain. Associated to other implementation models, several assets blockchains have more space-competent proof of ownership, as easy payment authentication can be employed for all naturally maintained blockchain assets. In addition, acknowledged instruments of allocating blockchain safety increases security perils in permissionless background.

Blockchains with the federation-based regulatory model can eliminate the above-mentioned security perils. The federation-based regulatory model positions more accountability on blockchain preservers. As the preservers can successfully regulate the state of the blockchain, they could legitimately be capable to reserve transactions and lien accounts as per the direction of regulatory bodies.

A blockchain based on several assets could be united into present blockchain infrastructure by means of sidechain technology.

10.5.1.6.3Smart Contract Assets

Assets can be characterized by the support of a smart contract. The agreement can store the plot of addresses of present owners of the asset versus consistent and related accounts. These accounts can be modified with the support of communications directed at the agreement encrypting asset allocation/allotment. The agreement can use the usual approval system of the original blockchain to evaluate, assign, and allot permissions or to stipulate new guidelines for asset operations.

Assets Use Cases Asset blockchains can be exploited by different users and use cases. The different categories are asset allotters, blockchain attorneys, controllers, application developers, and customers.

In specific situations, a user can fit into numerous classes mentioned above in use cases. One of the rewards of blockchain technology is that the specific user roles could be evidently detached. For instance, an asset allotter can give transaction operations and application development to other parties that are not part of the nodes. Furthermore, the public blockchain ecosystem can deliver competences such as asset marketplace and third-party-based tools development without any expectations from the asset allotter.

Diverse classes of users would have different needs as to the process of a blockchain. The requirements would also rely on the features of digital assets logged and accounted on the blockchain. For instance, legal apprehensions for digital securities would be more complex than other assets, and the access blockade for these kinds of digital assets is anticipated to be fairly high. Overall, digital asset application comes under the following sections:

1. **Organizational assets:** These are considered by established transaction processors. The authorized needs take preference of ease of access and worldwide range. Digital assets that characterize securities commonly fall in this section.
2. **Peer-to-peer assets:** These are assets with immature or almost missing market of devoted transaction processors and a need for easy access and worldwide range of technology. This type of digital asset comprises use case assets, business-to-consumer (B2C) assets such as vouchers, discount coupons, subscription, and gift assets.

User classification requires asset allocators counterfeit struggle, access permissions, cost of transaction, openness, blockchain attorneys, definitiveness and comprehensiveness of transaction processing rules, controller’s auditability, transaction conclusiveness, and unassailability.

Application developers include ease of application development, accessibility of manuals, APIs, SDKs, frameworks, technology roadmap, access permissions, and reach. End users include ease of use, user security, access permissions, privacy, transparency, range, and legitimacy.

Classification is uncertain in case of smart assets. There are organizational registries for certain assets such as real estate, though for most assets centralized possession registries should not be present.

Legal and compliance needs for organizational assets would need the use of isolated or stringently controlled permissioned public blockchains, which would be preserved by current transaction processors. In this situation, blockchain technology could deliver a pioneering application implementation model with distributed database and code-distributed between members, integrated audit trails and perhaps more third-party contribution in the system of autonomous verification services. In contrast, peer–to-peer assets could effectively use public blockchains due to the need for easy access and worldwide range, and the cost of transaction would be less for asset allocators and application developers.

10.6 Financial Institutional Assets

Digital assets could characterize publicly-operated financial Institutional assets such as share and securities. These assets are deeply controlled and utilized in business-to-business (B2B) backgrounds, consequently necessitating permissioned blockchains in a short duration. Moreover, numerous types of securities can help wide-ranging smart contract competences, which are usually not utilized by other categories of digital assets. Financial institutional assets can be operated in a decentralized fashion without necessitating mediators.

Permissionless blockchains can be suitable for innovative financial institutional assets such as crowdfunding. For example, an enterprise can allocate digital assets and its shares, and trade these stocks in a crowdfunding or a scheme operation. Moreover, the enterprise can recompense proportionate payments to respective shareholders.

10.7 Smart Assets

Smart assets characterize the possession of practical entities with the benefit of blockchain information. For instance, a blockchain-powered car can function only if the driver has the blockchain-reliant possession token. The possessor can use an app on his/her smart device to link the car using near-field communication (NFC). The possession demarcated in this method could be shifted using an operation with an input holding the token.

Smart assets have slow operation speed and require safety prior to scalability. Consequently, smart assets could reasonably be deployed with the assistance of devoted possession procedures (to be developed) on extremely protected public blockchains, which do not essentially support the idea of smart assets.

10.8 Electronic Currency

Digital assets can characterize e-currency such as substitute currencies in digital games or entitlements of sanction currency. Electronic currency attached to practical currencies usually have high-operation speed. Consequently, they would need scalable, high-end compute infrastructure availed by blockchains based on several assets.

10.8.1 Business-to-Customer Assets

Digital assets can be utilized to characterize discount, coupons, vouchers, gift cards, and loyalty points. The assets can be allocated by a trader and moved to buyers when procurement is completed. The trader describes a transparent guideline of how assets can be exchanged or encashed for goods. A large vendor can allocate several kinds of tokens and trace their delivery and possession, which can be beneficial for the study of consumer base. Related to current deployments, blockchain-computing resources would deliver an integral secondary marketplace for properties though asset allocation could be controlled with the support of allocation metadata.

10.8.2 Event Management Assets

An event venue such as cinema, theatre, concert hall, and stadium can allocate digital assets that match tickets (access) for an explicit event. This permits patrons to purchase or vend their tickets safely and swiftly, without worry of forgery. To verify the possession of a ticket, an individual joining the event would transfer it to the nominated address; this would be deployed as a mobile app in a simple manner. By accumulating metadata to tenders, the allocator could encrypt data about a precise ticket such as a seat for the event.

10.8.3 Digital Registration and Subscription

Digital assets can be utilized to leverage entry to digital possessions such as digital content streaming. For instance, a digital content streaming provider can offer specific unrestricted usage to music articulated as a digital token, which patrons can purchase for a specific sum of currency. Due to the blockchain's transparency, the content benefactor could effortlessly verify when the patron token was allocated and whether it is legal and still existing. The benefactor could allocate several types of tokens that match several levels of rights such as read, write, or both to explicit possessions and categories of resources.

Like digital registration and subscription, irremovable digital assets can be beneficial for role-based access and control such as giving access to different personas of administration for developing/using the web service.

10.8.4 Digital Democracy

Digital tokens can be utilized to instrument voting by transferring tokens to one of numerous chosen addresses. Though the current asset systems are not adequately safe to embrace government elections, they could be utilized for voting between stockholders or in competitions. In the latter situation, voting procedure is effortlessly rewarded. Constitutional voting would perhaps need more intricate techniques such as blockchain-voting cryptosystem utilized with zero-knowledge evidences (proofs) to impose autonomy and non-pliability.

10.8.5 Transportation and Logistics

From a transportation point of view, there are numerous use cases that could help the grouping of blockchain and IOT. In relation to telematics, it would permit producers to augment additional sensors to support service centres to safely seize and store engine health information and other automobile performance data. This data can be utilized with machine learning processes to govern when an automobile needs care. In this instance, blockchain delivers decentralized data storage, removes a central failure point, provides a non-intermediary record, and allows smart devices to autonomously connect with one another.

Similarly, the combination of IoT and blockchain can be utilized to safeguard transfer of perishable foodstuffs by regulating and monitoring the temperature during transference process. It can also be utilized to systematize order completion, billing, and payment of smart contracts. Each of these abilities make blockchain a perfect constituent of IoT architecture.

10.8.6 Cargo Tracking

Tracking in transportation cargo is not unusual. Industries have been using GPS to trace cargo-carrying vehicles for decades. Initially, position updates were made by calls and the practice of fax. Afterwards, these practices were substituted with computerized systems such as electronic data interfaces. As businesses face ever-increasing client opportunities and vendors are gradually guaranteeing one- or two-hours delivery, old-style practices will not work. Hence, the subsequent trend of delivery will originate through blockchain, which not only faces the latest necessities but also current legitimacy subjects.

Enterprises prefer updated data so they can take early choices and distribute information so other users can replicate the same process. The validity and authenticity of the information is essential for decisions. As information moves through numerous systems, there are probabilities for it to be misunderstood, changed, or hampered with or without the holder's familiarity or agreement. This leads to chaos in a worldwide supply chain. Blockchain by which the complete system backs information authentication instils trust to the whole environment. Also, due to information recorded in a decentralized fashion, the amalgamation of information is made easy, as all arrangements are attach to a sole node to access reliable information.

10.8.7 Carter Onboarding

Onboarding a carter is not simple. This process includes authenticating the carter's driver credentials. To safeguard the process, they preserve a robust security ranking, authenticate insurance reporting, and authorize their capability to meet service-level agreements (SLAs). Conventionally, transporters and agents depend on carters to deliver this information either in physical or electronic form.

As cargo agents look for volume, they frequently discover new transporters who are nearer to the pickup place but they are required to onboard these transporters before allocating loads. Usually, transporters and cargo agents have a number of resources devoted to authenticating carters and handling data within a transactional record. The challenge with this method is that this information is not distributed with other carters. As a consequence, each trade must capitalize time and resources in boarding several similar carters.

It is possible to solve this issue by using blockchain to authenticate transactions and record and distribute data. More than depending on inner resources to authenticate carters, we can use the blockchain system itself. Network node participants can be rewarded for their work similar to the way in which miners are presently incentivized with ethers. A noteworthy benefit of this model is that the network augments the new node to a public blockchain, thus permitting other members to use the information without added struggle.

10.8.8 Load Board Reliability

One of the major apprehensions related to load boards is the necessity to rely on the information itself. In several situations, load board data is obsolete and false. When a carter executes with a collection of cargo agents to carry goods, the data is characteristically recorded in numerous load boards, triggering data replication and erroneous demand predictions. For this purpose, the usage of load boards has been discouraged in several enterprises.

In the core of this model, information legitimacy challenges linked with load boards can be addressed by deploying a blockchain. For instance, when a carter publishes a load in the blockchain, it is timestamped with vital data.

In the case of blockchain, when Agent A tries to publish a load, the mechanism first authenticates to observe if the load previously occurs by associating the timestamped data. If Agent B tries to publish an identical load, the board would classify it as match and observe the user. This can safeguard that each load board will exist with a single record for a specified load. An advantage to this scheme is that it also addresses the subject of out-of-date data. When a load is engaged by a carter and reorganized in the blockchain, all load boards will repeat the restructured positions.

10.8.9 Rationalized Factoring

Small-scale transportation enterprises with inadequate cashflow cannot afford lengthy delays and commonly trust on factoring enterprises to be compensated quicker for a small proportion of returns share. There are two key features to this operation:

1. **Authentication:** Other than the evidence of delivery vouchers recorded by drivers, factoring enterprises trust on cargo agents and delivery acceptors to recognize acceptable delivery. Consequently, factoring enterprises require more data from reliable participants before compensating drivers, and they require it faster. Also, information authentication could be managed by a network node relatively than an internal source.
2. **Expenses:** With cryptocurrencies acquiring significance in today's world, a time will come when drivers and merchants will be rewarded through this model. This change will necessitate key alterations in the logistic ecosystem, though stakeholders will be required to be qualified to operate cryptocurrencies and transportation systems will be required to help them.

10.9 | Manufacturing

The potential for blockchain-enabled solutions to produce value by serving enterprises to face challenges is apparent. If properly used, blockchain could intensify transparency in the entire supply chain, trace the individuality and identifications of key employees, permit for more unified audit and regulatory purposes, and more. Manufacturing enterprises are now seen as frontrunners in evolving this technology.

Blockchain is employed to deliver applications to long-lasting business challenges, which include the following:

1. Supply chain assessment for better transparency into intricate, cross-community supply chains where adjournments and tracking

limitations influence production and cost-effectiveness.

2. Origins of materials and forged detection to decrease the effects of forgery and illegal copy on the worldwide economy.
3. Actions such as long-term engineering plan, high-intricacy based goods, postponements in distributing restructured engineering provisions or portions replacements can increase revised work and postpone ultimate transfer.
4. Identity and access rights for when it is vital to recognize who is taking a delivery and what are their identifications, including lawyers, accountants, engineers, and operators.
5. Asset tracing to examine multifaceted and exclusive gear transfers or in different modes of logistics across carters.
6. Quality management that can see transversely a manufacturing lifecycle to measure credentials, excellence, outlines of faults, etc.
7. Governing compliance improved by ineradicable accounts of activities taken, assets' transfers demonstrated by consensus that is permission-based accessible in seconds.

Blockchain-enabled solutions can flawlessly combine all of this data, distributing noteworthy value for industrial establishments, and can also benefit the full potential of other cutting-edge technologies such as augmented reality, IoT, and 3D printing (Figure 10.4).

Car manufacturers achieve immense and multifaceted supply chains. Raw resources are shaped into distinct parts, which are collected into bigger systems that are finally assembled as vehicles. Though automobile manufacturers are eventually liable to clients and controllers for their automobiles' consistency and security, they lack adequate visibility in the origin of the automobiles' parts and their drive from the manufacturing unit to the outlet floor.

Blockchain-powered systems would support automobile manufacturers to trace every phase of the manufacturing process into all supply chain layers. And if a constituent is faulty, it can regulate the reason, whether it was substandard steel, the supplier's material, or an issue in their factory. When integrated with associated sensors and other IoT components, a blockchain-enabled system could even account the problems to a delivery of equipment onboard a train or carter ship. This would imply that travel through rough oceans or ill-controlled temperatures could have triggered automobile impairment.

Because the enterprises that provide automobile supply chain develop an inter-reliant web, it could be helpful for all participants to have some discernibility in the temperament of goods. This can be problematic when the data is stored in numerous, incompatible systems.

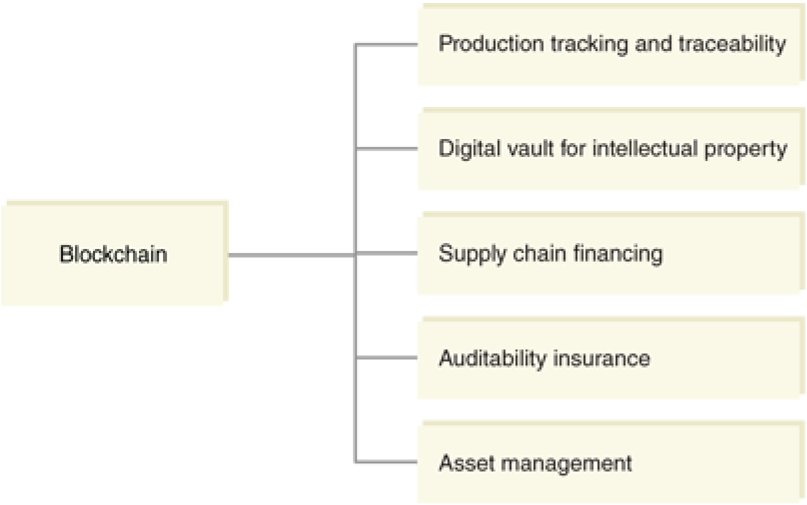

Blockchain
Production tracking and traceability
Digital vault for intellectual property
Supply chain financing
Auditability insurance
Asset management

Figure 10.4 Blockchain in manufacturing.

Further, blockchain can support automobile manufacturers and their supply chains due to the following reasons:

1. **Advance service management:** It is a daunting task to recognize which automobiles have machines emphasised to a recall, as it can be a laborious and resource exhaustive activity. The information is

characteristically recorded among several systems and involves settlement for precision to track goods. By using blockchain, automobiles with faulty parts can be swiftly gauged, maybe even before they have left the manufacturing unit. The defective part's trip from the dealer's workshop through concluding assemblage could be measured.

2. **Reinforce inventory control:** A blockchain-enabled system can support automobile manufacturers and their dealers trace choke points in real time during the supply chain and empower improved inventory scheduling.

SUMMARY

Blockchain use cases go far beyond cryptocurrency and bitcoin. With its capability to produce more transparency and impartiality while also helping in execution time and money, the technology is influencing a variety of sectors in many ways, ranging from how agreements are imposed to making administration work more competent.

SHORT ANSWER QUESTIONS

1. Why are financial institutions important?
2. What are the functions of financial institutions?
3. What is a financial system? What are its functions?
4. What are the different types of financial institutions? How can they use blockchain technology?
5. What are the different types of insurance?
6. What are the main functions of a healthcare system?
7. How does blockchain technology help in reducing the time for claim settlement?
8. How can a smart contract be used for asset management?

LONG ANSWER QUESTIONS

1. Prepare a use case with actors and activities of a healthcare system. How can blockchain technology be of help to ease their activities?

2. What are the functions of an insurance company? Discuss where they can use blockchain technology to help their customers.

CHAPTER 11

Blockchain and Allied Technologies

Learning Objectives

After reading this chapter, you will be able to:

- Appreciate blockchain and its integration with other technologies.
- Understand the finer points of technological applications of blockchain with other domains.

- Understand practical real-life situations where blockchain and allied technologies can be envisaged together.
- Understand the complexities of technology integration and how it can be converted in the service offerings.
- Comprehend the governance mechanisms as required for single technology platform.

11.1 Blockchain and Cloud Computing

Blockchain mechanism is a method to model data without the requirement for a central governance. A blockchain works similar to dispersed database that hosts an incessantly increasing quantity of records. These records store details in blocks instead of producing them in a single entity. Individually, a block is linked to a subsequent block in a lined, sequential order by means of a cryptography-based digital signature. Consequently, records cannot be altered, and somewhat tried variations are noticeable to all members. This method permits blockchains to be utilized as ledgers that can be distributed and validated by anybody with suitable authorizations. These shared ledgers can be distributed across various locations, nations, or organizations. Shared ledgers characterize smart contracts that frequently have recently developed in the form of code that is archived, authenticated, and implemented on a blockchain, thus delivering a platform for self-administering, self-actioned contracts.

Infrastructure evolving drifts such as software-defined networks, network virtualization, and containers working on microservices are all moving towards to a distributed-computing standard that depends on peer-to-peer communications.

Blockchain requires network deployments attempting to look for scalable, accessible, and synchronized transactions. It confirms highly available, comprise compatible mechanisms such as distributed databases and other consensus models. Successful executions provide multi-level privacy and confidentiality which is realized through multichannel communications, various ledgers, and distributed participants for transaction transparency founded on auditability if required.

The drive of distributed-computing requirement to the edge, determined by the ever-increasing charges of cloud computing, needed a novel method of creating trust and expense and charge back management across comprehensive and vastly distributed ecosystems. Blockchain, as it is commonly demarcated, has the potential to be develop as the primary platform on which all edge communications are executed because blockchains can generate unidentified, peer-to-peer levels of trust while giving an infrastructure to harvest distributed-computing paradigms. Every congregated technology has its own potential of market advancement forecasts, which when reserved in entirety recommend a pattern change in how customers and providers will communicate.

Telecom vendors and providers as a market will be clear recipients owing to the growth of networking bandwidth for increasing volumes of instruments and consumers. Moving the investments from central data centers and migrating workload to a public cloud have been debated and conferred for many years now. This is not a comprehensive migration of the present networking infrastructure to the peer-to-peer shared model, but it will be a noteworthy migration where most of the applications will get the advantage of availability and match of the exact computing resources to execute precise workloads. The evolving inclination today is that most use cases of data analytics is getting migrated from development and processing nearer to the edge.

There are new latency-based use cases such as augmented reality, autonomous vehicles, and remote hospital and patient care. These new lower latency services will only increase the movement.

Consequently, network and telecom providers are gradually chasing smart distributed-computing platforms such as centralized radio access network, cloud radio access network, and mobile edge computing. It even includes data center networks interconnection layouts accepting shared and dispersed functionality and peer-to-peer use cases to increase performance and architecturally diminish the total expenditure of network execution. Present attempts to optimize distributed environments leveraging blockchain are only concentrating on trivial applications or slightly impactful results. These attempts commonly comprise of shared endeavours among businesses or small associations.

At present, classification of blockchain is not properly defined to put appropriate systems in control. In addition, confidentiality guidelines may be a problem to conform with an agreed unchallengeable characteristic of blockchain. Information management necessities in diverse geographies may be dissimilar and may risk confidentiality guidelines such as the European Union General Data Protection Regulation (EU GDPR). Following confidentiality principles of diverse geographies may be the most significant task. For example, as the communications on a blockchain are thought to be absolute and immutable, this does not match with the GDPR necessities of information removal and data erasure after legitimate usage.

Blockchain technology can generate boundless prospect in achieving underutilized computing resources and equipment and delivers the infrastructure to provide a completely new environment.

Containers such as networking area is migrating to a software-defined ecosystem for cloud-native workloads. The concept of microservices-based containers has only been developing and growing these days. This mechanism permits running workloads and processes in containers and permits them to function as an inaccessible element of application distribution. Containers optimize inclusive developer understanding, enrich code and module reuse, and streamline processes for cloud-native workloads. Containers are extremely useful in resolving scalability problems.

There is an amplified risk in archiving large amounts of data on centralized records in cloud or even in data centers. The susceptibilities that can be a single point of failure have been suppressed as demonstrated by the augmented occurrence of big data cracks. Decentralized data outages break up data into portions, encodes and allocates it across numerous networks. This delivers rewards by not relying on centralized databases, whose ecosystems could be exposed. There are added advantages to decentralized databases (storages), including reduced operating cost, improved accessibility, and faster delivery using peer-to-peer nodes in the networks.

However, there are boundaries that are required to be resolved such as security that is relational to the magnitude of a decentralized network. Network, communications, and content delivery network (CDN) methods will have to expand to license huge amounts of information to target diverse nodes. Present storage mechanisms permit users to engage third-party to store their information as well as protect their data's privacy, truth, and accessibility. There occurs several compliances and rules such as Health Insurance Portability and Accountability Act (HIPAA), GDPR, etc., that enterprises must follow while executing storage projects. Eventually, these solutions are disposed to manual intervention and lead to errors. In addition, the safety of centralized mechanism is constantly at jeopardy of getting bargained by unanticipated occurrences or events.

Middleware technologies will be comparable to offchain programs written and coded in any language, run within a protected and reliable container, and interconnected using safe channels. They can work as independent mediators, cooperating with the ecosystem off the chain while preserving the truthfulness of the blockchain. Open source tools will help as the structure blocks this ecosystem. Relying on open frameworks for protocol deployments of peer-to-peer networking, consensus, storage (database), and VMs/containers is crucial in creating trust within the bigger environment and fast tracks innovation. A modular standard and mechanism will permit associations to choose the best elements and shape their dispersed applications, irrespective of the underlying characteristics.

Moreover, it will also permit the elements to modify limiting any reliance formed above the main layer. To permit a blockchain-enabled business transformation, the blockchain at the fundamental technology layer should not be the single section of distributed infrastructure. The following are technologies and services that need to be distributed in the cloud model.

1. Middleware.
2. Storage.
3. Routing.
4. Published services.
5. Compliances.
6. Network.
7. Software development platforms such as containers.

Several issues help the likelihood of fully established, decentralized peer-to-peer shared networks attempting significant large volume amid huge populations in the future. These issues comprise of the following:

1. Increasing social cognizance of the hazards and susceptibilities of keeping citizens' confidential data on huge centralized computing resources.
2. Inactive income prospects that will payback entities to change the essential drive to contribute and collaborate.
3. Faster interpretation of emergent technologies that will empower contributors to involve in cooperative mutual networks.

As already discussed, peering involves a noteworthy transformation to the interpretation of the user's characteristics. This helps to suggest an architectural model that permits to rent computing resource in as-a-service model such as server and storage inside a peering node of network. The business is adjacent to technology-based edge, where it is possible to divide the compute activities and have these transactions executed in a similar manner by several nodes of a distributed network. Distributed networking grid helps in significantly reducing the operational cost needed for multifaceted applications, scientific scheming, machine learning, and high-powered computing proficiencies.

Subsequently, the foundation of a distributed network is related to existing peers such as service providers and consumers, modelling will bring about all types of resources available to the application needed. The building phase of the architecture will have to be intelligent enough to distribute application requirements to competent delivery nodes as per the contract. The blockchain and middleware elements will work in the background as connecting architecture. In order to access network services, a consumer must be associated and liked to a node. The peer-to-peer mechanism permits for ingesting both the network's computing resources and servicing of the nodes requests through leasing the neighbouring node's surplus computing power.

Each linked device will work as a peer-to-peer node within the network that will record prominent proceedings such as costs and sums as well as communications with smart contracts required on the network.

The application can either be the requirement of server or storage transactions. The generic availability of computing and storage resources will require knowledge of the probability of a distinct peer and entire network.

Despite their exclusive potential, blockchains deliver very restricted compute resources to execute decentralized workloads, including small storage, incompetent virtual machine (VM), and a protocol that requires high latency. Ultimately, blockchain technology will advance to solve some of these issues, but there will be an ever-increasing requirement to deliver added capacities to blockchain workloads.

The existing clouds cannot achieve the needs of decentralized applications that require wholly decentralized computing resources for their execution. In the meantime, there is an increasing mandate for computing capacities from businesses and scientific consortiums to execute large workloads and run large data capacities. The computing resources to execute big data workloads is most frequently delivered by cloud service providers and high-performance computing (HPC) resources. This signifies that state-of-the-art small start-ups frequently do not possess the resources and knowledge to obtain and execute HPC infrastructures, although old-style cloud service providers are still unapproachable for challenging workload requirements such as GPU rendering. In addition, data centers utilize enormous quantities of energy for working on servers and air conditioning systems. It is not only costly, but it is not environment friendly as well. We require a novel form of cloud technologies for decentralized workloads that can enable reducing the operational cost of infrastructure used for blockchain computing.

11.2 | Characteristics of Blockchain Cloud

This section will cover the cloud services and characteristics that provide blockchain service offerings.

11.2.1 Blockchain Cloud Self-Service Portal

The self-service portal-based marketplace delivers an easy-to-use API so that consumers can use and observe how the market and environment is rapidly developing. This will give flexibility to consumers to see the combination of worker pools, available resources, pricing tables, smart contracts for leasing, and rewards and penalty systems.

11.2.2 Worker Resource Pools

Resource pools in blockchain cloud should have the likelihood to be unified into the network, and work like a worker node and incentivize any work for performing blockchain workload transactions. There exists a scheduler whose accountability is to allocate tasks to worker nodes. Thus, it replicates in the same method as mining pools. A specific miner regularly links a mining pool to take full advantage of their probabilities of receiving recompense for computations. Correspondingly, as a worker node, it will link to a public worker pool that will make certain that it has enough capacity to distribute.

11.2.3 Task-Based Charging Models

Regular cloud-based chargeback models will not work in this scheme. Blockchain computing is based on tasks and transactions. So, it is important to device chargeback based on tasks. A complete set of transactions types with its boundary conditions and execution limitations. An associated chargeback should be available on the self-service portals. Regular test beds should be available to benchmark a developer's submission of task-related computing infrastructure requirements.

11.2.4 Decentralized Application Store (Appstore)

If the blockchain cloud exists with decentralized application store (Appstore), it provides a gateway to new group of blockchain technologies that rely on application. The first use case can emerge from areas such as artificial intelligence (AI), big data, Internet of Things (IoT), and Fintech.

Using these areas, consumers will have the capability of browsing in library where there is a decentralized application store. Also, developers can contribute their individual decentralized application store and monetize or deliver it in an open source. Decentralized application store can be realized as a group of readymade workloads crossing all possible cases. These applications are developed, modified, and maintained in various categories.

11.2.5 Blockchain Cloud Governance

The measures and guidelines that govern the working process of blockchain cloud are known as governance. The network nodes contributing to the work signs digital to abide by the policies. It also discusses the security rule and benchmarks required to safeguard the blockchain platform. Security and confidentiality in blockchain arrangements must deliver both infrastructure security as well as operational security requirements. Network management and monitoring abilities comprise areas to respond deviations in the platform and ecosystem, including monitoring, analytics, automation, service management, dashboarding tools, capacity, configuration management database (CMDB), business process metrics, management interface, and change configuration.

11.3 Blockchain and Artificial Intelligence

Blockchain is a revolutionary next generation paradigm authorizing the innocuous and consistent storage and communication of data, along with other rewards and benefits. Artificial intelligence (AI) is a radical knowledge that can learn on its own by examining and determining forms in immense volumes of (big) data. Consequently, there occurs an accepted bridge between blockchain and AI.

Blockchain secures records/communicates truthful data, while AI needs large volumes of consistent data to determine patterns and acquire knowledge based on its learning. Both blockchain and AI are complementary to each other and there would be ground-breaking innovations when these two are integrated. The possible paybacks are anticipated in the field autonomous automobiles, healthcare, smart contracts, Internet of Things and edge analytics, decentralized autonomous organizations (DAOs), Fintech, logistics and transportation, and security and compliance.

In various situations, AI cannot be utilized without the guarantee of the protection and consistency of the data delivered by blockchain. The worth of various blockchain use cases will be restricted and deprived of AI.

Though at present, blockchain and AI models are not effectively united, the probability of them to integrate together in the near time is exciting, driven by considerable predictable paybacks. Blockchain and AI technologies are no exclusions, predominantly when AI is considered as the next-gen technique and is still evolving. This is apart from some use cases in entertainment (gaming), connecting language parsers, processing and image recognition. There are predictable advantages, which would also result in huge profits.

It is not clear as to how AI can be united with blockchain technology utilized in cryptocurrencies, though there are possibilities that this can be performed when robots will be familiarized and possess goods and property assets. In this scenario, the robot will use AI to make the essential communications with Bitcoin.

As per research, the future holds noteworthy discoveries delivering significant paybacks that can be realized by reducing medical expenses and cultivating excellence in healthcare. This is so because all public cloud providers with small consortiums are increasingly discovering AI for healthcare use cases, meant at utilizing patient data more effectively.

The advantages of AI for Fintech are prevalent, as huge historical volumes of data is accessible. Considering the long time (more than two decades) that AI has been in the market, security, confidentiality, and fraud detection were predicted with excessive success using statistics-based logical reasoning. Blockchain can play a vital role to achieve this goal.

Moreover, AI is being utilized to discover the swiftest way to perform stock market trading to make wagers on market impetus, and to X-ray press notifications and financial metadata that could indicate that a stock would go down or up. Unfortunately, stocks and supplies that work similar to random processes cannot be forecasted so easily for the current price or upcoming rates. Obviously, current AI use cases in banking, financial services and insurance (BFSI) are just the start and the supremacy of AI to bring better knowledge, reduce costs, lower hazards, and grow revenues will become a certainty.

Supply chain processes are already using blockchains, which may be integrated with AI. The test is in the future to spread AI to enduring fragments of the supply chain. The applications of AI in supply chains will eventually outcome in laying an environment where supply chains associate with many interfaces and technologies, empowering unified movement of goods and data from one interface to another, thus entirely automating the method and attaining noteworthy advantages in the process.

Two significant features of blockchain, smart contracts and decentralization, are getting used for self-learning patterns using AI and even adapt based on the scenarios. This can be achieved by identifying data pattern changes, recognition of languages and parsing, image processing, and sensing suspicious events. This can also be achieved for DAOs where digital contracts can be applied to improve its value and efficacy. Currently, efforts are made to integrate blockchain and AI. The advantages of such a paradigm will fetch distributed ledger technology (DLT) to manage blockchain smarter and enhance its capability to transform through self-learning.

Retail sector has gone through a big transformation by using blockchain and AI. Similarly, finance (account receivables/credit/debit) services can use AI and blockchain technology to transform. Nowadays, the technology exists for commerce applying for loan using online services to buy the financial instrument and consume it. Moreover, one can think of creating smart digital contract using AI and blockchain, thus further automating and then dispatching using cryptocurrency.

Rudimentary credit and accounts receivable errands can and will be computerized in future. Some of the automation potential use cases in this area include payment applications, data storage, risk analysis, and credit and accounts receivable functions.

AI and blockchain technology are going to help various use cases of decentralized organizations. Both AI and blockchain have their own differentiators, but they will be more impactful when they are integrated, realized, and used as single augmented technology. Developers will have to think further at a higher level than before. They will drive analysis for their enterprises, pursue results with the latest technology, and recommend transformation by discovering prospects to keep progressing and rolling.

11.4 Blockchain and IoT

IoT is a stimulating developing structure that delivers limitless advantages, but there are various problems with the present centralized IoT systems such that all devices are recognized, validated, and linked to centralized servers. This architecture was used to link an extensive series of computing resources for various years and will last to deliver small-scale IoT ecosystems. However, it will not accomplish delivering the requirements to extend the IoT system for future generations. There are various advantages of both these technologies, which can be integrated to get an enhanced result.

IoT has limitless advantages and implementing a decentralized method for it would resolve several problems, particularly safety. Accepting a uniform peer-to-peer communication system to execute the infinite number of transactions among devices will meaningfully lower the budgets linked with mounting and sustaining large centralized data centres and will allocate capacity requirements across the devices that develop IoT networks. This will avert disaster in any node in a network from making the whole network to a tentative failure.

The decentralized, self-governing, and immutable abilities of blockchain brands it as a perfect constituent to become an introductory component of

IoT architectures. It is no surprise that organizations with IoT knowledges have rapidly become one of the primary accepters of blockchain technology. However, launching peer-to-peer messaging will result in a number of problems, particularly compliance and security. IoT security is bigger than just about shielding sensitive data. Consequently, blockchain methods will have to preserve confidentiality and safety in IoT networks and practice authentication and agreement of members for transactions to avert delay and fraud. Actually, blockchain technology is measured as an important mechanism to resolve confidentiality and consistency matters in IoT. It can be utilized to trace billions of linked devices, thus empowering the execution of transactions and harmonization among devices; this permits for noteworthy investments for IoT business producers. Furthermore, this decentralized method would eradicate single points of failure, producing a more robust structure for devices to execute on. The cryptographic logic system utilized by blockchains would develop more isolated customer data.

Within IoT network, blockchain can retain an immutable information of the past that might be of various types of devices. These characteristics allows the self-directed working of smart devices without the requirement for a centralized agency. Consequently, blockchain will expose a sequence of IoT situations that were tough, or even dreadful to deploy without it. For instance, by using blockchain, IoT solutions can empower safe communication among devices in the network. In this method, blockchain will take communication interactions among devices similar to Fintech transactions in a cryptocurrency network. To permit communication interactions, devices will use smart contracts which then use the contract among the two bodies (agency or party).

One of the most interesting abilities of blockchain is the capability to preserve a fully decentralized, reliable ledger of all transactions that occur in a network. This competence is vital to empower various agreements and govern necessities of industrial IoT use cases without the requirement to trust a centralized mechanism. Various big enterprises have initiated to accept blockchain with IoT ecosystems to make use of all the features of blockchain.

11.4.1 Blockchain and IoT Combination Advantages

Both blockchain and IOT has its own advantages. But when it is integrated its power will increase exponentially. The advantages of the IoT combination are enumerated here.

1. **Transparency:** All member nodes have the capability to observe all transactions and a node block has its own ledger. The data of the transaction is safeguarded by a member's private key, so even if all members visit the data, the nodes remain protected. IoT is an active system in which all linked devices can distribute information simultaneously and concurrently while protecting members confidentiality.

2. **Decentralization:** Most members must confirm the transactions to authenticate and integrate it to DLT. There is no sole agency that can authenticate the transactions or establish precise instructions to have transactions acknowledged. Consequently, there is an immense amount of belief encompassed since mainstream members in the network must reach a contract to authenticate transactions. So, blockchain will deliver a protected platform for IoT. In addition, blockchain eliminates centralized stream of flow and failure at a choking point of the existing centralized IoT models.

3. **Disaster recovery:** Each node has its individual replica of the ledger that comprises all communications that have ever executed in the network. Therefore, blockchain is capable to endure an outbreak. Even if one network was negotiated and attacked, the blockchain would be preserved by the other nodes in the system. Maintaining a replica of data at individual nodes in the IoT will advance information distribution requirements. However, it does have distribution and storage problems.

4. **Protection:** Blockchain has the capacity to deliver a protected network over autonomous trustless agencies (nodes), which is required in IoT with numerous and diverse devices. Thus, the entire IoT network participants must be malevolent to make an attack.

5. **Swiftness:** A blockchain transaction is dispersed in the network in a fraction of time and will be administered at any period of interval.

6. **Optimization:** Current IoT architectures are not cheap due to high-compute resource and managed service expenses linked with centralized model and networking gears. The overall volume of messages that will have to be managed when there are infinite IoT devices will increase these expenses considerably.

7. **Unalterable:** Working on an immutable ledger is one of the foremost rewards of blockchain technology. Any alterations in the distributed ledger must be tested by almost all network nodes. So, the transactions cannot be changed or erased easily. Running an ecosystem in an immutable ledger for IoT information will increase the safety and confidentiality, which are the main features in this area.

8. **Secrecy:** To execute a transaction, both consumer and vendor use unidentified and inimitable address pointers which keep their individuality isolated. This characteristic has been opposed as it intensifies the practice of Bitcoins in the illegal grey market. However, it could be a benefit if used for other situations such as citizen charter systems.

11.4.2 Blockchain and IoT Combination: Limitations and Problems

There is no uncertainty that assimilating blockchain would have many rewards. However, blockchain technology is not a perfect model and it has its own faults and problems. These challenges can be summarized as follows:

1. **Infrastructure capacity:** A blockchain system requires the computing resources and time required to realize encryption for all entities. IoT ecosystems have varied types of devices with dissimilar computing abilities, and every device will be able execute a similar encryption logic at the mandatory speed.

2. **Storage capacity:** The chief advantage of blockchain is that it eliminates the requirement for a central system to record transactions. Nonetheless, the ledger has to be stowed on the individual nodes. The distributed ledger will intensify the storage size with time and with growing number of members in the network. In comparison, IoT devices consume less processing time and requires very little storage capacity.

3. **Knowledge scarcity:** Blockchain is a new technology and it is difficult to find skills pertaining to blockchain. So, there is only a handful of resources with limited knowledge in this area. In other use cases, there is an extensive deficiency of knowledge of how blockchain runs. IoT is universal, so accepting and diffusing blockchain integrated with IoT will be very problematic without community cognizance about the blockchain.

4. **Regulations:** Blockchain is a distributed database that has the aptitude to attach diverse individuals from diverse nations without having any legal regulations or agreements to track, which is a significant appeal to both producers and consumers. This is the main problem for accepting blockchain in various industries and use cases.

5. **Device identification and discovery:** Blockchain as a model is not designed for discovery of the devices that is the heart of IoT. In other words, nodes never discover each other in the network.

6. **Blockchain and IoT next steps:** Blockchain has transformed the perception of centralized establishments. The incorporation of blockchain in IoT will be the start for introducing novel businesses and use cases. We can imagine some use cases in logistics, supply chain, fintech, and healthcare, which are discussed in Chapter 10.

7. **Smart contracts:** These are codes embedded in blockchain. They can encode and archive data safely, confine access to data to only the anticipated users and then be scripted to operate the data within an autonomous rational workflow of processes among members of the node. Smart contracts interpret business logic into an automated process, significantly cultivating functioning competence. By means of smart contracts used in IoT, ecosystems will deliver an effectual way to expand safety and truthfulness of IoT data. This needs to be adopted and diffused across the systems.

8. **Compliance and regulations:** These are measures shaped by establishments and local charted agencies to outline lawful customs of working with goods and services within a specific nation or province. The next step is how to leverage compliances in an integrated system of IoT and blockchain across borders.

9. **Security:** Assimilating blockchain with IoT can advance safety of the ecosystem as it leverages consensus agreement of the participants to authenticate transactions to avert delay and fraud. However, IoT devices possess very less computing capacity that cannot be developed to process big cryptography-based logic. The question is how to use less capacity IoT devices for cryptography-based logic algorithm executions.

Nowadays, IoT technology is present in every walk of life throughout the world. It can link commonplace entities to the Internet. Using inexpensive sensors, much of the data and knowledge can be derived from neighbouring ecosystems that results from civilizing our lives. However, the existing IoT architecture that grounded on three-tier computing model has various matters that is essential to be explored, particularly scalability and safety.

11.5 | Blockchain and Machine Learning

Today’s AI and machine learning based applications require rapid development, data quality, and data relevancy. As compared to publicly available data, privately possessed data is more pertinent and appropriate for machine learning. Private controlled data is typically unexploited and distant as it is recorded in distinct smart and intelligent devices. Big corporates and businesses attempt to access privately possessed data by delivering unrestricted (free) service to consumers. Nonetheless, these organizations can only acquire a portion of the privately possessed data, which is a subsection of the enormous unexploited data maintained by all entities.

Machine learning deployed in a decentralized ecosystem is considered to develop and target unexploited privately possessed data and release their potential to enable machine learning development while delivering financial enticements and shielding data confidentiality. Machine learning algorithm will be executed on smart devices without mining the data, to have greater set of data intuition. The machine learning result will be accumulated with consequences created from other devices to develop an impartial, complete, and precise self-derived autonomous analytics and forecasts that is the backbone of blockchain. It considers a block as the devices to manage privately owned data and computing capacity in decentralized fashion as algorithms are executed on discrete devices by using their spare-processing power.

The excellence of the algorithms is as serious as data value to produce good functioning applications and perform precise forecasts using machine learning. It is desired that this works as cooperative intellect, a machine learning community will be developed by envisaging the marketplace for brilliant developers to publish their logic as a charged or free service. Any contributors (nodes) that required to subscribe any service from the marketplace to run forecast through machine learning can discover or demand appropriate algorithms. Revolution from the margin are fortified and huge potential in machine learning are released through decentralizing the algorithm thinking, self-governing, immutable, autonomous nodes (devices/locations).

Monetary enticements and a consistent ecosystem are required to enable contributions in decentralized algorithms development. With the assistance of blockchain smart contract, a decentralized autonomous system can be imagined that links contributors of the machine learning community directly without the necessity of a centralized agency. It guarantees members who successfully contribute their work such as permitting algorithms to be executed on discrete data and develop machine learning algorithms to be incentivized using tokens and identifiers.

In this way, machine learning with blockchain will provide a scalable decentralized ecosystem that purposes to link possibly infinite number of devices and numerous developers to simplify machine learning expansion while leveraging decentralized possession of data. Various potentials will be unleashed through unused data and cooperative intellect. One can imagine and provision novelty from the edge to endorse eccentric, revolutionary, and assorted inventions across geographies. Such dispersed machine learning protocol is too authoritative to be measured by a limited individual. It should be preserved cooperatively and permit mass contribution using decentralization inherited by blockchain. The entire machine learning in decentralized mode will keep developing. The system permits every contributor (node) to correspondingly and impartially participate in the machine learning community.

11.5.1 Machine Learning Infrastructure Optimization

At present, machine learning is frequently steered through a centralized server with restricted computational power. If extra processing power can be exploited to execute more machine learning logic, the market potential is incredible.

If machine learning, in a decentralized way, can link the unused computational capacity of these devices and develop a decentralized super system for machine learning, the additional market income foreseen to be produced can be substantial.

An enormous growth can be predicted if several enterprises can deploy machine learning algorithms and mechanisms in their industries. This is possible given that the preliminary cost needed to attain machine learning algorithms and platform or even human resource to develop standard algorithms can really be cheaper. One can forecast mass diffusion in machine learning in the marketable world when the accompanying investment is reduced and when the forecast made by machine learning is more precise and affects the daily walk of life use cases.

Enterprises that demand to develop analytical forecasts through machine learning can either select to grow their internal systems or obtain readily accessible ones, which will be made available by developers either free or with an applicable charge.

Algorithms are characterized conferring to diverse types of industry use cases or subjects on the self-service portal where users can obtain the suitable algorithms that are best suit to the situation. Subsequently, users require to define their requirements and principles such as area in terms of geography, compliance, and security. Scope clarity permits machine learning algorithms to execute on a small dataset that allows for a more precise forecast, which is suited to a client's requirement.

The principles, scope delivered by all the contributors (node), and incentive to the nodes occur with the help of tokens and identifiers. This can be achieved using smart contracts delivered using blockchains. It is signed digitally to eliminate hazards associated among users, developers, and decentralized nodes.

11.5.2 Machine Learning Powered by Smart Contracts

Smart contracts operate as accelerators that are not trust based and eliminate the use of agents to enable the decentralized platform for machine learning. The following are factors that propel machine learning algorithms powered by smart contracts.

1. Users who wish to subscribe for machine learning algorithms pay to node tokens to execute it on data possessors and approved dataset in their specific devices. These decentralized participants will simplify by allocating algorithms and accumulating, combining and be around the limited forecast outcomes to yield the concluding forecast report.
2. The marketplace that publishes developers machine learning services to sell are also paid by subscribers using tokens and identifiers.
3. Machine learning algorithms and modelling nodes (teams) that train the network are also incentivized by consumers who exclusively desire to customize their algorithm to make it suitable for specific use.

11.5.3 Smart Contract Usage in Machine Learning Datasets

A smart contract executes on decentralized participants that link data holders (on distinct devices/systems). Once it is executed, the concluding report is formed by participant nodes, which is ultimately recorded in the smart contract. A smart contract can also be used as a governance tool in the marketplace for subscribing, provisioning, amending, upgrading, and model training of machine learning algorithms.

The agreement will run based on the situation demarcated under the background. The parameters comprise the volume of tokens that users are ready to pay for possessing the machine learning algorithms. Upon the conclusion of the machine learning outcome by the reporting node, smart contract runs and transfers the tokens to the worker node.

Huge volume of data is required in order to produce good models in machine learning. This is so because big volume data increases the total throughput, and supports in developing a more comprehensive inference, and produces a more consistent system. This is one of the reasons why big data in machine learning cannot be exaggerated. However, integrating blockchain storage volumes in machine learning signifies shared data platform, safe and large data, and better learning models.

Blockchain's decentralized characteristics empower data to be distributed between a participating node. This delivers easy availability to data for associated machine learning system deployments. The subject of data attainment has been the main obstacle to machine learning evolution.

When data is available on decentralized systems, it is bigger in volume as well as secure and originates from internal and external sources. Internal origins of data can be congregated into limited and civic agencies. Using blockchain, data can be distributed across and when utilized as a seed value to machine learning model yields high-competency rate as linked to privately held data.

External data may be the input from related enterprises. When this data is utilized by analytical machine learning models, it is certain that the model will make more accurate forecasts. Apart from obtaining large amount of data using blockchain at almost no expense, it is also safe from being decentralized.

The undulating outcome of acquiring big volume of shielded data for machine learning models is the expansion of improved and more consistent models for various purposes, such as forecasting.

11.6 Blockchain and Robotic Process Automation

We are at the centre of a digital journey in which we are gradually relying on technology to resolve glitches. There is no scarcity of use cases for how we can trust technology. While the stride of technological progression is impressive, it still faces issues related to administration.

Robotic process automation (RPA) is a software that automates labour intensive, instruction-based monotonous repeating activities. RPA software can optimize execution time, increase data throughput, advance accuracy, and optimize and reduce time.

These technologies are self-assured to demonstrate an important role in the future of many technological advances. It is important to understand the ability to automate labour-intensive activities and deliver novel models; business paradigms will likely change the method in which we accomplish our use cases and create an improved experience for users.

As we adopt these technologies to resolve issues, we need to be certain that the individuals and the processes required to deploy and manage these new methods are updated. To take complete benefit of the RPA software, we need to ensure that the corresponding individuals and process modifications are updated on it.

11.6.1 Individuals and Processes

From an "individual" standpoint, it is important that we provide skills to users to identify and discover processes that are ready for automation, impart them how to script, debug the software when it breakdowns, and skill them on how to achieve a digital workers based enterprise as well as how to communicate with digital workspaces.

From a "process" viewpoint, we will require to comprehend the latent effect on the guidelines and measures and apprise the processes to replicate the practice of new software. It will also require new documentation to look for internal and external inspections, and we will have to get acquainted with the RPA software that may affect the greater governing context.

The fast transformations in technologies are compelling industries to reconsider the mode in which they are delivering their services. Trades are continuously looking to utilize cloud-based technologies to revolutionize and provide newer modules of services while cultivating the competence of present services to withstand and nurture their businesses. These cloud-based technologies include everything-as-a-service, machine learning, dark analytics, virtual/augmented reality, blockchain, IoT, and RPA.

RPA in its basic form is the automation of labour-intensive processes by reproducing repetitive tasks with a software-based application. The technology imitates user activities or a group of activities and can be realized in a couple of weeks using such software. RPA can be entrenched into present applications or can be seated on top of various ecosystems or platforms to automate a group of processes. On the other hand, blockchain technology will deliver an integrated platform with RPA with characteristics such as security, unalterable system (immutable), scalability, distributed ledger, decentralized governor, digital asset movement, and shared distributed database.

Blockchain technology makes it almost impossible to change or alter any transaction data, thus lessening the probability of a scam. Adopting these two technologies in an organized format will address some important client apprehensions predominant in various use cases such as safety and interoperability with its unified recompense reclamation abilities. RPA will permit rationalization and implementation of business processes while blockchain will deliver the platform to empower safe accomplishment of these reorganized business processes across various enterprises, untrusted agencies, and administration ecosystem.

A confederation of blockchains may transform various industry applications. The notion of an RPA-enabled alliance of private blockchains can be additionally prolonged to manage industry pain points. The predicted concept will initiate various prospects that an enterprise can further execute in business process. Hypothetically, the knowledge has the latent to substitute/complement bodies such as civic administration processes with transparency, optimization, and governance.

As industries and technologies get settled, we will realize a network of blockchains evolving where entities-based assets reliant on the understanding, privacy, and safety deliberations are stored on a private blockchain for in-house usage, but with APIs to countless public blockchains as per the enterprise or administration requests.

Technologies such as RPA and blockchain are fragments of the evolving technology framework and own incredible latent to enhance use cases and unite them within an enterprise. But there are specific apprehensions and deliberations including governing and compliance problems, slow acceptance, and high investment on assets that requires attention to be judiciously assessed in order to completely exploit RPA and blockchain competences.

RPA empowers business process automation by means of software-programmed script robots, commonly termed as *digital workforces*. In recent times, RPA has advanced from execution routine tasks to perform intelligent-based tasks such as integration of other technologies on blockchain platforms. These are visual perception, machine learning, collaboration, and analytics.

RPA and blockchain already exist in Fintech. However, the use case examination has stretched into various industries such as insurance, BFSI, retail, supply chain, healthcare, and even government.

Notwithstanding innumerable applications, the objective is equivalent to the following:

1. Emphasis on human skills relevant to persuasive tasks using imagination, design, method, clarification, decision-making, and compassion.

2. Reliable digital set-up levers all monotonous mundane tasks and backend activities processing.

11.6.1.1 Benefits

User experiences are manifesting and compelling businesses to accept a fundamentally distinct standard when it comes to digitally renovating the processes technology. Technology solutions require to flawlessly communicate with end users and ecosystems, while robotically sanitizing through huge volumes of data, and delivering a reliable transactions execution at the backend. The combination of RPA and blockchain can help with these requirements.

RPA with cognitive abilities can represent the dominant character as an analytical gateway to communicate with end users, apprehend data inputs, and cooperate with IT ecosystems to distribute information in instantaneous fashion. Blockchain can deliver distributed ledger and shared infrastructure for the transaction-processing competences between enterprises. From a process viewpoint, blockchain plays an enabler role for intelligent, trusted, and smooth value contract management between executing agencies.

RPA can empower blockchain to transfer information with prevailing IT systems as well as support in launching monitoring and governance across transaction-processing tasks. Blockchain can guarantee trust to robotic process automation arranged process flows by delivering protected ecosystem to authenticate all communications and uphold audit trail in an immutable distributed ledger technology. If integration is architected harmoniously, RPA and blockchain can completely automate digital workflows to realize enterprise-wide objectives.

While process automation using RPA is perfect for usual large amount data and policy-based processes, blockchain is perceived as a model technology for asset inventory management, records keeping, and transactions management application and it has repetitive tasks-based processing that calls for insights from RPA. The advantages are include the following:

1. **Automated transaction processing:** RPA-motivated processes communicate with end users and platforms to acquire data in the form of customer requirements, order details, and existing documents utilizing bots and orchestrating the process flow. Blockchain can deliver a ledger to store business communications and processes for intelligent autonomous immutable decisions such as identity and access management, information authentication, and payments processing.

2. **Governance and compliance:** RPA can automate mundane compliance activities for governing flows as well as automating business process. Blockchain can be utilized to generate an immutable storage of the governing actions that can be retrieved in case of internal or external assessments for compliances such as GDPR, HIPAA, and SOX.

11.6.1.2 Constraints and Growth Factors

Both RPA and blockchain are innovative technologies and are growing in diverse directions and speed. Hence, the operation of RPA and blockchain does not exist without problems. Several enterprises are not completely exploiting RPA as an enterprise innovation competence that scales across numerous lines of business. These enterprises are identifying benefits from the use of digital workforce, but their inclusive value proposal is moderated by not taking the proper steps. When choosing an application scenario for RPA and blockchain strategy, it is important to prudently assess an enterprise's practices and reflection of technology. It is good to prefer a scenario that has a rational, mundane routine-based processing, and paybacks from immutable data storage as a potential applicant for RPA and blockchain pilot.

SUMMARY

The achievement of digital transformation depends upon rational thoughtful activities such as integration of advance technologies with blockchain through all-inclusive necessities, adhering to business objectives, recognizing key success criteria, and finding strong governing regulatory guidelines. It is important to invest time on examining the existing operating model, scheming and architecting a target state, and documenting the process integration interfaces before initiating the execution.

SHORT ANSWER QUESTIONS

1. Will blockchain replace cloud computing? Explain your answer.
2. Does blockchain use cloud?
3. What is blockchain cloud service?
4. What is the difference between blockchain and cloud?
5. Why is cloud governance important?
6. What is security governance in cloud computing?
7. How does blockchain relate to AI technologies?
8. How is AI big data and blockchain interrelated?
9. How do you secure IoT with blockchain?
10. How can blockchain disrupt IoT?

LONG ANSWER QUESTIONS

1. How can blockchain and AI help in robotics technologies?
2. Prepare a use case to demonstrate how machine learning can be powered by smart contracts.

When Blockchain Smart Contracts Need Large Files

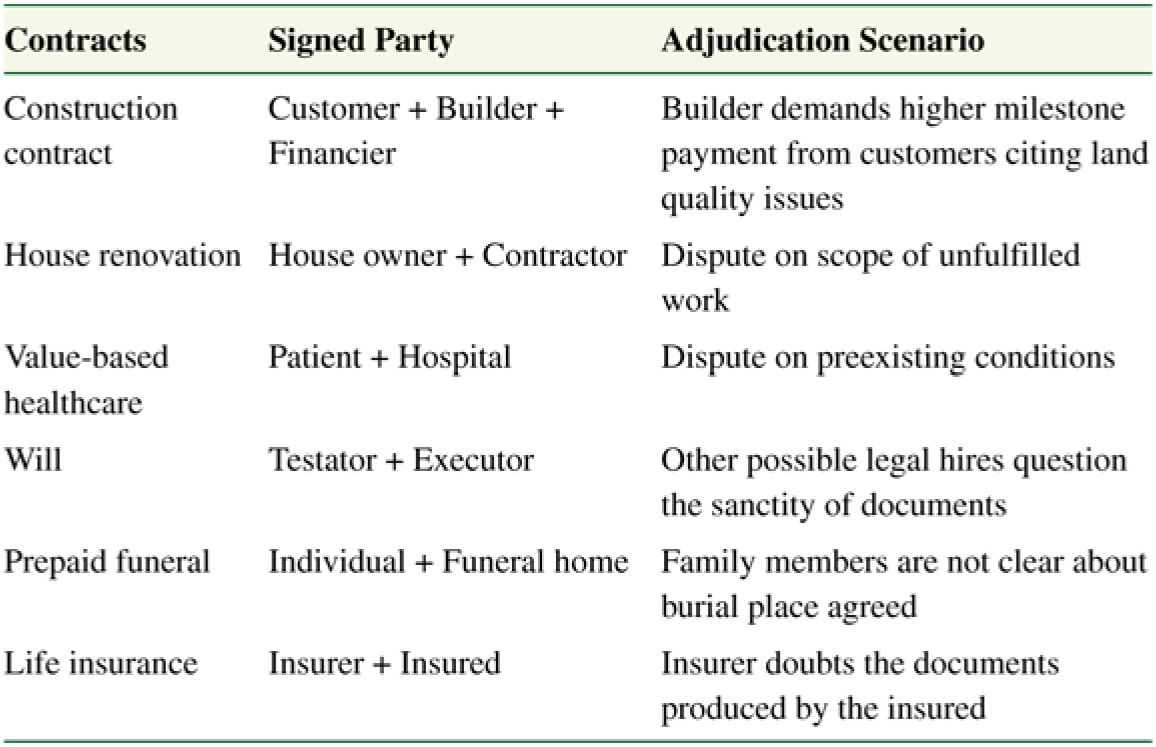
A ANNEXURE

A.1 Spectrum of Smart Contracts

Can a block of code with a few if-then blocks formalize self-enforcing agreements between individuals or institutions? People today trust a ticketing machine to take in their bills, spit out the ticket, and give the balance due. People trust the machine during the few moments they neither have their bill nor the ticket. There are many such examples of contracts around us. People do not question the trust when the consequences are small or the trust is well established.

Is this situation the same in a marriage alimony? Can a black box with some if-then logic fulfill the alimony requirements, possible even after a few decades from signing? These and many other countless real-life examples exist around us, and we examine a few in Table A.1.

Table A.1 **Few examples of real-life situations with long running contracts**



Contracts	Signed Party	Adjudication Scenario
Construction contract	Customer + Builder + Financier	Builder demands higher milestone payment from customers citing land quality issues
House renovation	House owner + Contractor	Dispute on scope of unfulfilled work
Value-based healthcare	Patient + Hospital	Dispute on preexisting conditions
Will	Testator + Executor	Other possible legal hires question the sanctity of documents
Prepaid funeral	Individual + Funeral home	Family members are not clear about burial place agreed
Life insurance	Insurer + Insured	Insurer doubts the documents produced by the insured

One may argue that these are extreme situations and not even a candidate for electronic automation. Are contracts enforced by a piece of code legally binding in the first place? Should we even consider such contracts for automation? There are many other contracts that want to remove transaction charges, and possibly third party and middlemen.

There is a trustable black box in the form of a peer-to-peer (P2P) network of computers that manages a self-enforcing piece of computer code that we can all trust. This P2P network managed by blockchain is referred to as smart contract, which can self-enforce agreement between parties under obligatory rules. The following are features of a smart contract:

1. A block of simple computer code and data.
2. Contractual clauses are embedded in the computer code as simple human readable code with if-then branching rules.
3. Technology and rule based operations can initiate real-world actions such as banking transactions.

4. A proof-of-stake or proof-of-work based blockchain execution framework provides irrevocable execution framework for the piece of computer code.

Institutions are aware that smart contracts able to address many simple real-world requirements such as transferring securities, clearing derivatives, handling cross-border financial transactions, handling loyalty programs, etc. In such cases, removing the intermediaries can largely reduce transaction costs involved. Blockchain with immutable files become necessary when the value to the parties and effective time span for contract execution is high (Figure A.1).

All situations are not the same; some transactions are initiated and completed in seconds while others take longer, some can even take years or decades. On the other hand, the value of the transaction can be a few rupees, hundreds of rupees or sometime a few lakhs. Trust on a smart contract black box by any individual or institution varies based on the value of the transaction and life of the transaction. Uncertainty arises when the duration of transaction is longer. The entire circumstance may be different at the time of contract execution versus the time when the contract was signed; the parties could have changed, geopolitical scenarios could have evolved, etc.

When this uncertainty combines with higher stakes, then the situation becomes more complicated. The action initiated by a smart contract is prone to trigger litigation and adjudication proceedings by parties involved or even parties outside of the contract that are effected by the smart contract action. The logic that triggered the action may be questioned by the circumstances and rationale for the contract; for example, an heir who is not included in a will as beneficiary can question the sanctity of the circumstance during contract initiation. Law proceedings may call for documentary evidences to prove or disprove the sequence of events amounting to the circumstance. In the circumstance, the digital repositories are valid and the stored documents will need to be extracted and exhibited.

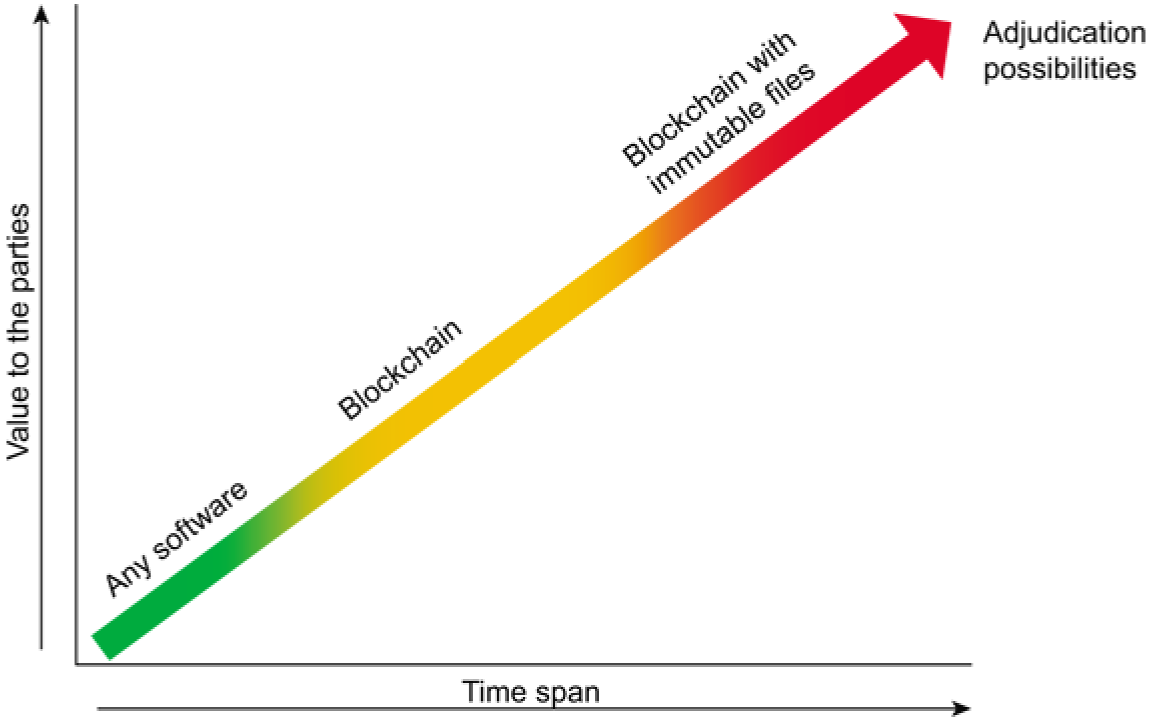


Figure A.1 Blockchain with immutable files.

It should be noted that in a heated dispute, anything and everything would be under scrutiny. In such cases, scientific logic and rationale would not be considerations. Therefore, immutability of any of these digitally stored files has to be at the same level of the smart contract itself.

A.2 Solution Options

We accept the fact that the blockchain based smart contract platform can reliably store and execute contracts. The underlying science provides the necessary assurance to ensure the sanctity of platforms such as Ethereum. The same platform can also store large files and embed them into a smart contract; the size of these contracts can be a few megabytes to even gigabytes.

It would have been an ideal scenario to have one platform providing end-to-end solution. While Ethereum does provide option to store data, it is not meant for large data and it can become very expensive. A rough calculation with the current price of Ether at $213 shows that 1GB of data can cost close to a million USD.

It is clear that storing large data permanently on Ethereum is extremely expensive. While we maintain the minimum data required for the smart contract's working on Ethereum, we have to delegate storage of large files to other external solutions. We require an off-chain solution to store and maintain these files for long term.

We need to understand where and how are these files stored. Most storage options are centralized in some form such as file share, cloud storage, tape drives, optical mediums, etc., which are all susceptible to manipulation. Censorship, network firewalls, and local service outage scenarios can render most of this common content source inaccessible; a consequence of this can be a smart contract failing and triggering serious reparations. It is paramount that smart contracts are protected from such situations.

Controlling content modification can be controlled to some extent by access control at various levels, restricting privileged access at various layers of storage that includes physical disk blocks, caches, storage network, file system, logical data, and application level access. Unfortunately, any possibility of changing content at one or many of these layers can be subject of scrutiny and an immediate reason for rejection. The following are potential reason for concerns:

1. Photographs can be digitally manipulated.
2. Videos can be morphed using deepfake techniques.
3. Text documents can be edited and saved.
4. Records in large databases can be purged.
5. CAD blueprints can be manipulated.
6. A property registration document can be purged in entirety without any trace.

There will always be new solutions that will tempt a smart contract designer, it is always advised the designer go with the most popular choice at the point of time, particularly so when the time span and the value to the parties is high.

A good option is to look at an off-chain solution that uses Merkle tree root hash as a tamper-proof data store and accessed as an independent external service. We then capture a permanent link of this externally stored file as part of the smart contract data.

Table A.2 shows the immutable data storage options for blockchain. Figure A.2 is a representation of using IPFS as an external storage. This external storage could be any of the various options shown in the figure.

Characteristics that both IPFS and Swarm share are hash-based content addressing, immutability, version control of content, and censorship-resistance. One key difference between IPFS and Swarm is that the latter uses a content-addressed chunk store rather than the distributed hash table (DHT), which is used by IPFS. IPFS is generally considered more mature as compared to Swarm, even though it is closely associated with the Ethereum stack.

Table A.2 **Immutable data storage options for blockchain**

Option	Description
InterPlanetary File System (IPFS)	It is a protocol and P2P network for storing and sharing data in a distributed file system. It uses content addressing to uniquely identify each file in a global namespace connecting all computing devices.
Swarm	It is a distributed storage platform and content distribution service, a native base layer service of the Ethereum web3 stack that aims to provide a decentralized and redundant store for decentralized application (DApp) code, user data, blockchain, and state data.
Filecoin	It is an open-source, public cryptocurrency and digital payment system intended to be a blockchain-based cooperative digital storage and data retrieval method. It is made by Protocol Labs and builds on top of IPFS, allowing users to rent unused hard drive space.
BitTorrent	It is a communication protocol for P2P file sharing, which is used to distribute data and electronic files over the Internet.

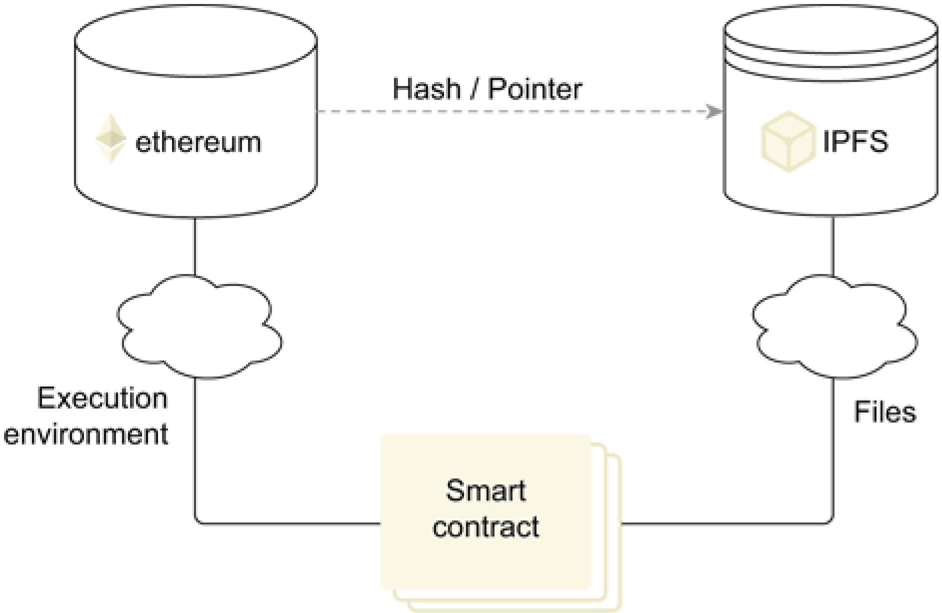


Figure A.2 Using IPFS as an external storage.

Both are promising platforms if they succeed with their development roadmaps and achieve wider adoption. There is no doubt this will bring big changes to the smart contract world and the overall Internet itself.

For the conceptual understanding, we will focus more on the IPFS stack. The two platforms are however similar and can be largely interchanged.

A.3 InterPlanetary File System – An Overview

InterPlanetary File System (IPFS) resembles a single BitTorrent swarm that exchanges objects within a single Git repository. Files are distributed through a BitTorrent-based protocol. IPFS acts as a sort of combination of Kodemila, BitTorrent, and Git to create a distributed subsystem of the Internet.

The aim of IPFS is to provide an immutable general decentralized storage layer and a content delivery protocol using P2P networks composed of participating nodes that are able to store and retrieve content. The DHT of the files also serve as the key to access their content. Chunks of files are used to store large files similar to BitTorrent. The IPFS network is sustained using the incentive mechanism for people to run clients. Filecoins are the monetary reward provided for hosting the clients (http://filecoin.io).

The files can be further encrypted using public- and private-key cryptography to prevent others from misusing the data. Figure A.3 is a block diagram illustrating a request with the hash of a file returning locations of data; this set is read subsequently from the P2P network.

IPFS uses a generic P2P solution implemented in the libp2p network layer. It is developed on the mainline BitTorrent DHT implementation that has been well tested over the years. To summarize, the following are key features of IPFS:

1. Low-latency retrieval.
2. Efficient auto-scaling.

3. Reliable, fault-tolerant operations, and resistant to multiple node's disconnections.
4. Zero-downtime.
5. Global scale, thus censorship-resistant.
6. Potentially permanent version archive of content.

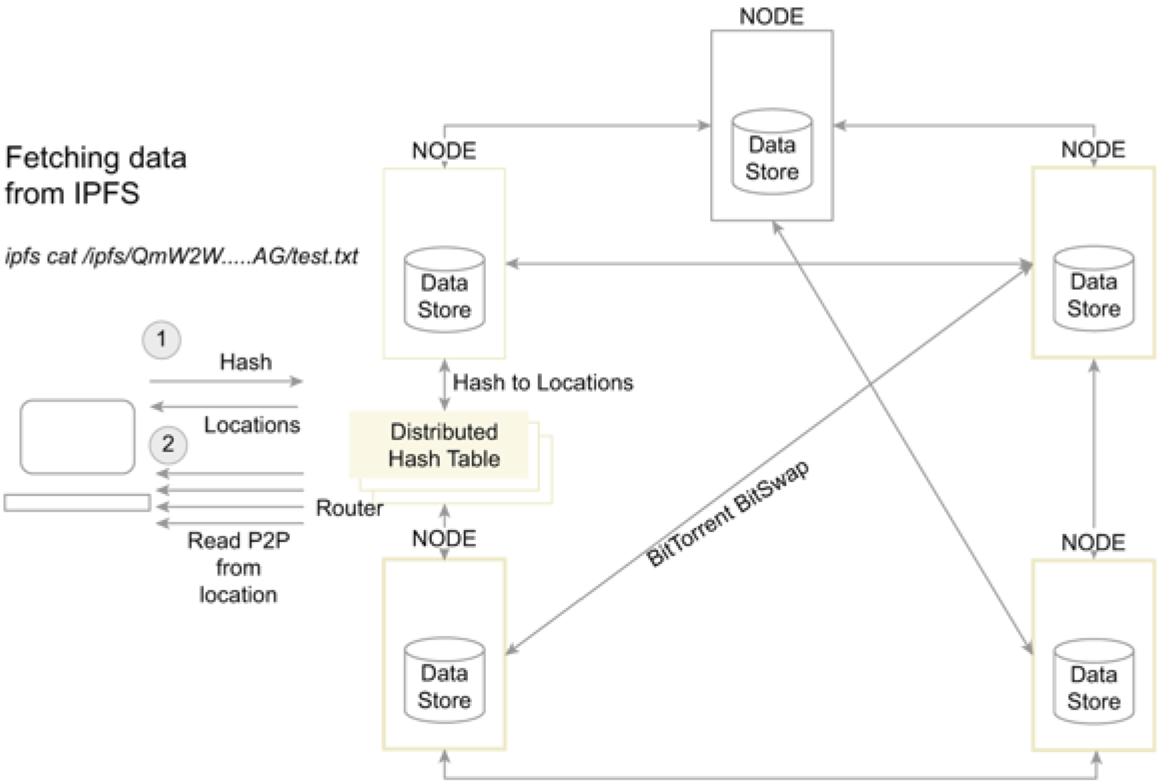


Figure A.3 Block diagram illustrating request with hash.

A.4 Using IPFS in Ethereum Smart Contracts

Smart contracts have a reputation to be complex, requiring full appreciation of the domain involved, with technical and non-technical considerations in many areas. However, the piece of code that implements some of these smart contracts are not enormously large in line of code, they are relatively small in comparison to many other systems. A simple way to understand a complex smart contract could be through examination of some actual code. We examine a simple DApp that will upload a document to IPFS, store the IPFS hash on the Ethereum blockchain, and verify the upload through a successful transaction receipt.

Figure A.4 shows the logic flow depicting initialization of the smart contract till its registration into Ethereum. Also shown is the requisite file stored in IPFS.

We look at the solidity code snippets that implement the above smart contract with the required file updated into the IPFS and linked to the smart contract (Table A.3). The Google Chrome browser with metamask plugin (https://metamask.io/) was used here to interact with the DApp. Remix (https://remix.ethereum.org) was used to deploy and test the code, Ether from Rinkeby Testnet was used as this was meant only for development purpose.

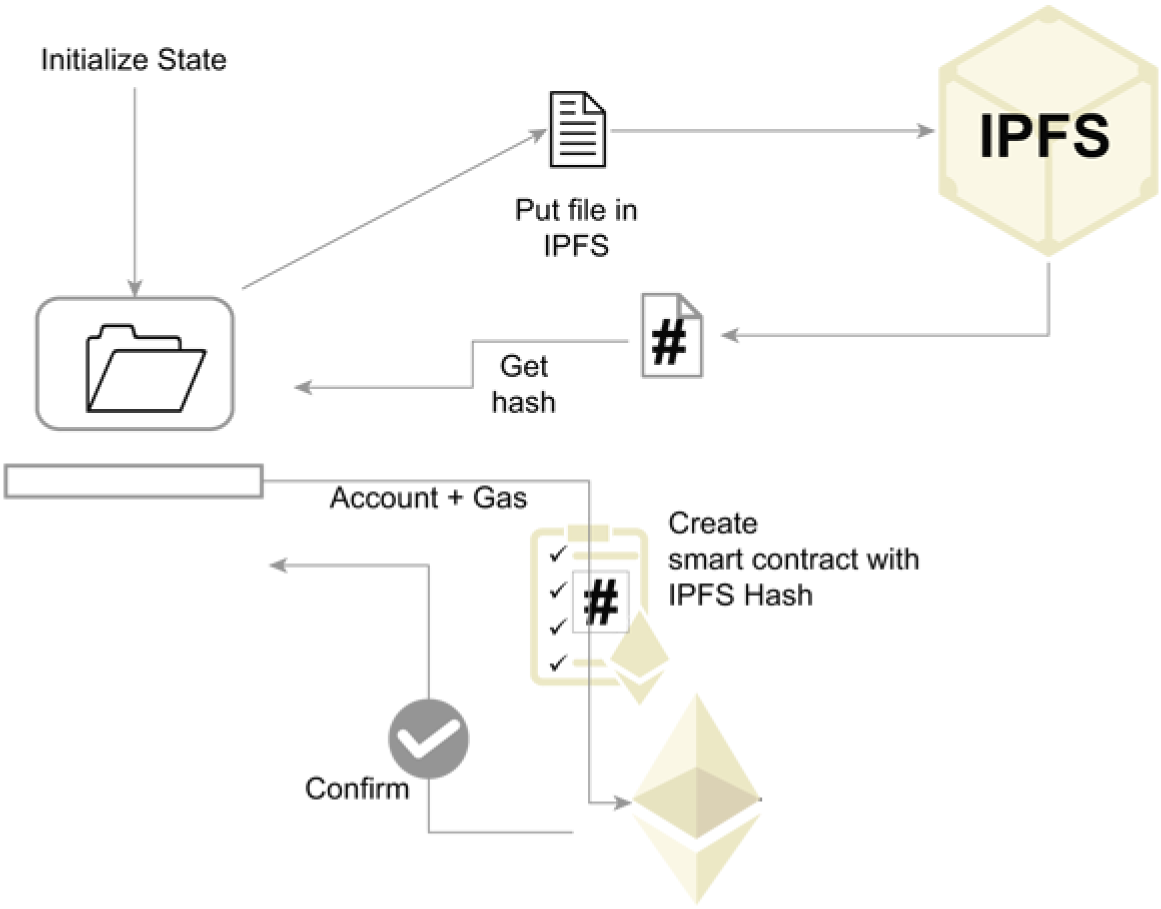


Figure A.4 Logic flow of initialization till registration of smart contract.

A.5 Key Considerations

There are many challenges for smart contracts that are expected to execute a contract sometime in the future. Developers and other stakeholders need to understand these challenges and incorporate some best practices as they go about implementations. It is good practice for developers to explore already available code in the community, which was deployment tested. As the system develops, such contributions will be far more in number, including the following:

1. Use of the latest version of compiler.
2. **Oversight of overflow and underflow conditions:** Usage of unit8 type is common in a long-running program. A developer can sometimes overlook the possibility of arithmetic operations, reaching the 8-bit limit of 255 and performing united operations.
3. **Invoking functions:** Solidity provides for functions that are external, public, private, internal. Any external functions are vulnerable to manipulation. It is recommended that the developers stick to only private and internal functions. Further to this, to avoid unexpected logics, any exceptions from these functions call need to be handled accordingly and not assumed that it will always execute successfully.

Table A.3 **Solidity code snippet implementing IPFS in Ethereum**

Set the state variables

```
class App extends Component {
    state = {
      ipfsHash:null,
      buffer:'',
      ethAddress:'',
      blockNumber:'',
      transactionHash:'',
      gasUsed:'',
      txReceipt:''
    };
```

Capture the user's file

```
captureFile =(event) => {
        event.stopPropagation()
        event.preventDefault()
        const file = event.target.files[0]
        let reader = new window.FileReader()
        reader.readAsArrayBuffer(file)
        reader.onloadend = () =>
this.convertToBuffer(reader)
      };
```

Convert the file to a buffer

```
convertToBuffer = async(reader) => {
      //file is converted to a buffer for upload to IPFS
        const buffer = await Buffer.from(reader.result);
      //set this buffer -using es6 syntax
        this.setState({buffer});
    };
```

Send the buffered file to IPFS

```
try{
        this.setState({blockNumber:"waiting.."});
        this.setState({gasUsed:"waiting. . . "});
//get Transaction Receipt in console on click
//See: https://web3js.readthedocs.io/en/1.0/web3-eth.
html#gettransactionreceipt
await web3.eth.getTransactionReceipt(this.state.
transactionHash, (err, txReceipt)=>{
          console.log(err,txReceipt);
          this.setState({txReceipt});
        }); //await for getTransactionReceipt
```

IPFS returns a hash

```
await web3.eth.getTransactionReceipt(this.state.
transactionHash, (err, txReceipt)=>{
          console.log(err,txReceipt);
          this.setState({txReceipt});
        }); //await for getTransactionReceipt
```

Get the user's MetaMask Ethereum address

```
    //bring in user's metamask account address
     const accounts = await web3.eth.getAccounts();

     console.log('Sending from Metamask account: ' +
accounts[0]);
```

Send the IPFS for storage on Ethereum

```
await ipfs.add(this.state.buffer, (err, ipfsHash) => {
        console.log(err,ipfsHash);
        //setState by setting ipfsHash to ipfsHash[0].hash
        this.setState ({ipfsHash:ipfsHash[0].hash});
```

Using MetaMask, user will confirm the transaction to Ethereum

```
storehash.methods.sendHash(this.state.ipfsHash).send({
          from: accounts[0]
        }
```

Ethereum contract will return a transaction hash number

```
 (error, transactionHash) => {
          console.log(transactionHash);
          this.setState({transactionHash});
        }); //storehash
```

4. **Running out of gas:** Ensure the program logic such as reentrancy does not cause a situation when the gas runs out sometime in the future. Recursive call attack is a possibility by malicious actors on smart contracts.

5. **Eye on resource consumption:** Have a clear idea on the logic and the gas consumption, particularly the looping logics and implications for the future scenario, which if not properly anticipated could result in gas exhaustion.

6. **Multiple external calls:** External calls can sometimes fail accidentally or deliberately, which can cause a denial of service condition in the smart contract. To minimize the damage caused by such failures, it is always better to isolate each external call into its own transaction that can be initiated by the recipient of the call.

7. **Pitfalls with external calls:** External calls are always unpredictable; particularly in the case when multiple transactions are required, the developer cannot assume a call or a series of calls will behave predictably. A failure may be due to a deliberate act by a malicious actor to get a specific behavior of contract execution. The situation can be extremely serious when it involves financial transactions.

8. **Track on bugs:** We never know the kind of bugs that can emerge in future; there are many layers in computing stack that can create vulnerability. Therefore, it is always recommended to use the latest compilers (e.g., Solidity). Also, developers should check forums such as DeFi for the latest information on bugs and appropriate precautions.

Ethereum Essentials: MetaMask and Remix

B ANNEXURE

Disclaimer: Blockchain is the work in progress. This annexure discusses several technologies from various providers. Most of these are open source; however users may check for any license, if any, and authors are not responsible for any eventuality. The purpose of this annexure is to provide relevant information under one umbrella.

B.1 Installation and Personalization of MetaMask

We know that Ethereum is a global blockchain computer that allows anyone write programs that run in public and is at the same verifiable in open, thus making the applications fully auditable and tamperproof unlike traditional "centralized" websites, where the trust lies on the provider company.

In this annexure, we will discuss how to create an Ethereum Wallet in your Browser, namely MetaMask. It is an extension for accessing Ethereum-enabled distributed applications (Dapps) in the browser. The extension injects the Ethereum web3 API into every website's JavaScript context, so that Dapps can read from the blockchain. This also enables you to create and manage your own identities (via private keys, local client wallet, and hardware wallets). When a Dapp wants to perform a transaction and write to the blockchain, you get a secure interface using MetaMask to review the transaction before approving or rejecting it. It also enables access to Web

3.0, Dapps, NFTs, tokens, etc. It generates passwords and keys on your device, so that only you have access to your accounts and data. You can always choose what to share and what to keep private; it provides an essential utility for blockchain newcomers, token traders, crypto gamers, and developers.

We will first learn how to install the MetaMask extension in the browser (we have used Version 7.7.9, which was updated on 5 May 2020, having a size of 10.14 MB). This extension is available in 52 languages. You can find the latest version of MetaMask on the official website https://metamask.io/

After installing MetaMask, you will be able to set up your own set of accounts, or "keys" that will open a whole new set of possibilities. This uses a BIP-44 12-word seed phrase to generate the account. You can enter any valid 12-word seed phrase while installing an account, use the key that are associated with it, and after installation start unlocking the power of the blockchain.

Installing MetaMask

Step 1 – Go to the Chrome webstore.

Step 2 – Click "Get Chrome Extension" to install MetaMask as shown in Figure B.1.

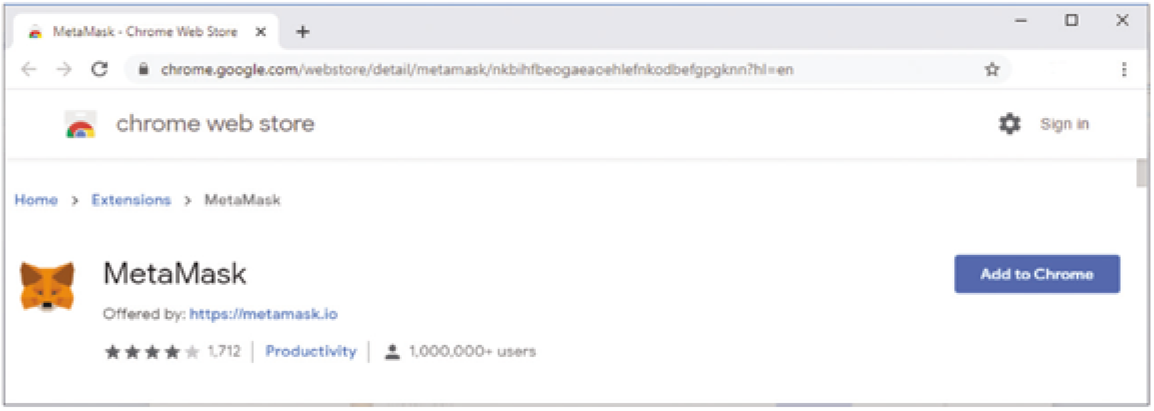
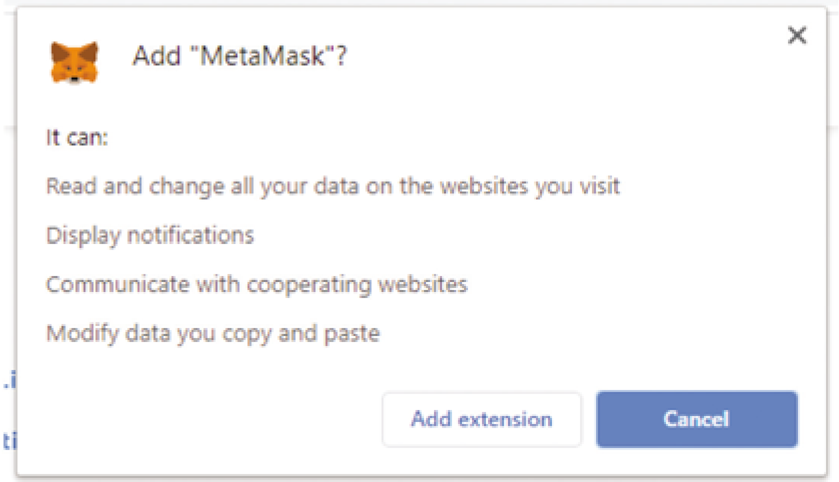

MetaMask - Chrome Web Store
chrome.google.com/webstore/detail/metamask/nkbihfbeogaeaoehlefnkodbefgpgknn?hl=en
chrome web store
Sign in
Home
Extensions
MetaMask
MetaMask
Offered by: https://metamask.io
Add to Chrome
★★★★☆ 1,712
Productivity
1,000,000+ users

Figure B.1 Installation of MetaMask – Step 2.

Step 3 – Click “Add to Chrome” tab and you will get a message box (Figure B.2).



Add "MetaMask"?

It can:

Read and change all your data on the websites you visit

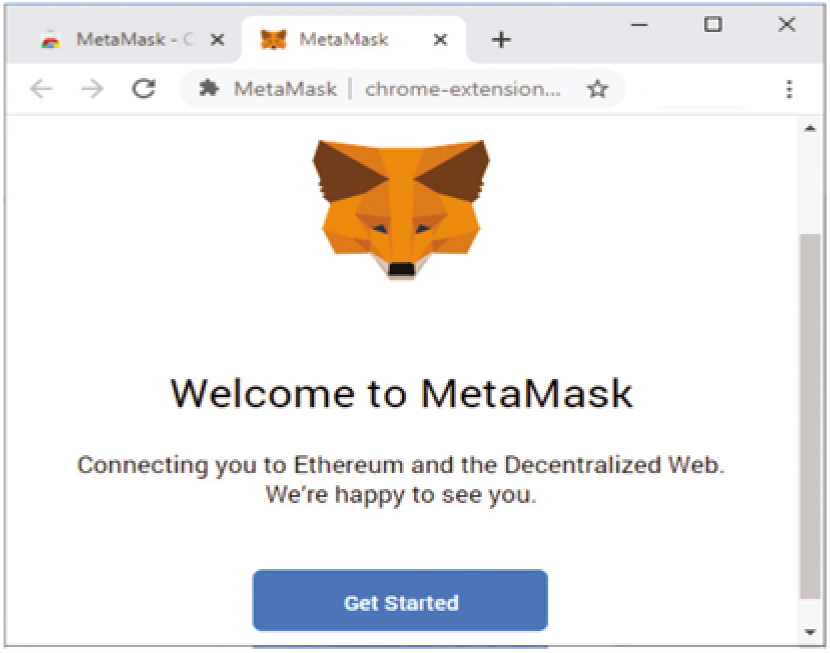
Display notifications

Communicate with cooperating websites

Modify data you copy and paste

Figure B.2 Installation of MetaMask – Step 3.

Step 4 – Click “Add Extension” to complete the installation and after a few clicks you will see that the installation is completed as shown in Figure B.3.

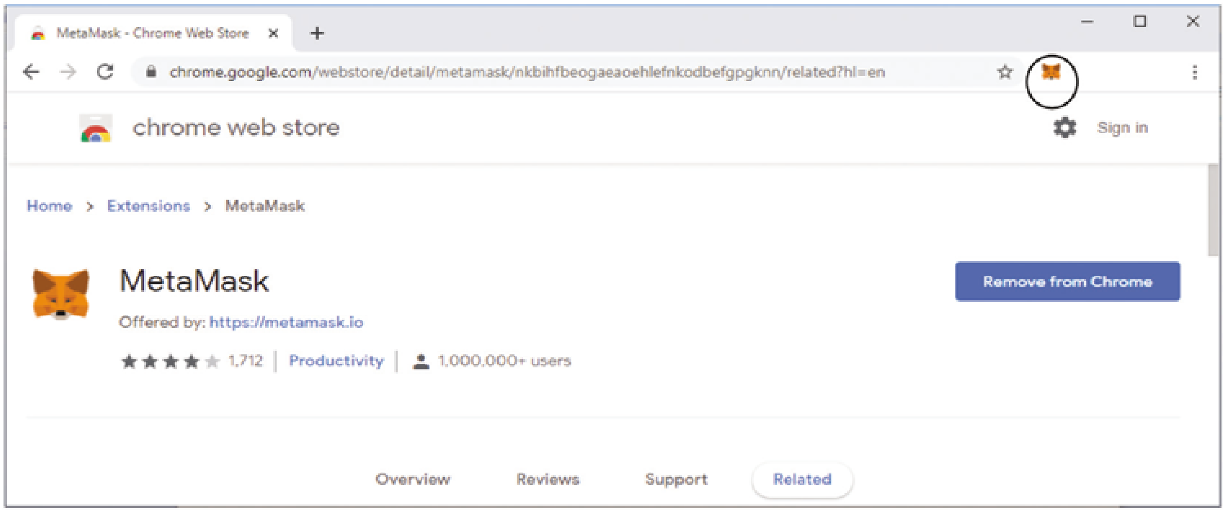


Welcome to MetaMask

Connecting you to Ethereum and the Decentralized Web.
We're happy to see you.

Figure B.3 Installation of MetaMask – Step 4a.

This conveys that MetaMask has been installed when you see the fox logo on the upper right-hand corner (circled in Figure B.4).



chrome.google.com/webstore/detail/metamask/nkbihfbeogaeaoehlefnkodbefgpgknn/related?hl=en

Home > Extensions > MetaMask

MetaMask

Offered by: https://metamask.io

★★★★★ 1,712 | Productivity | 1,000,000+ users

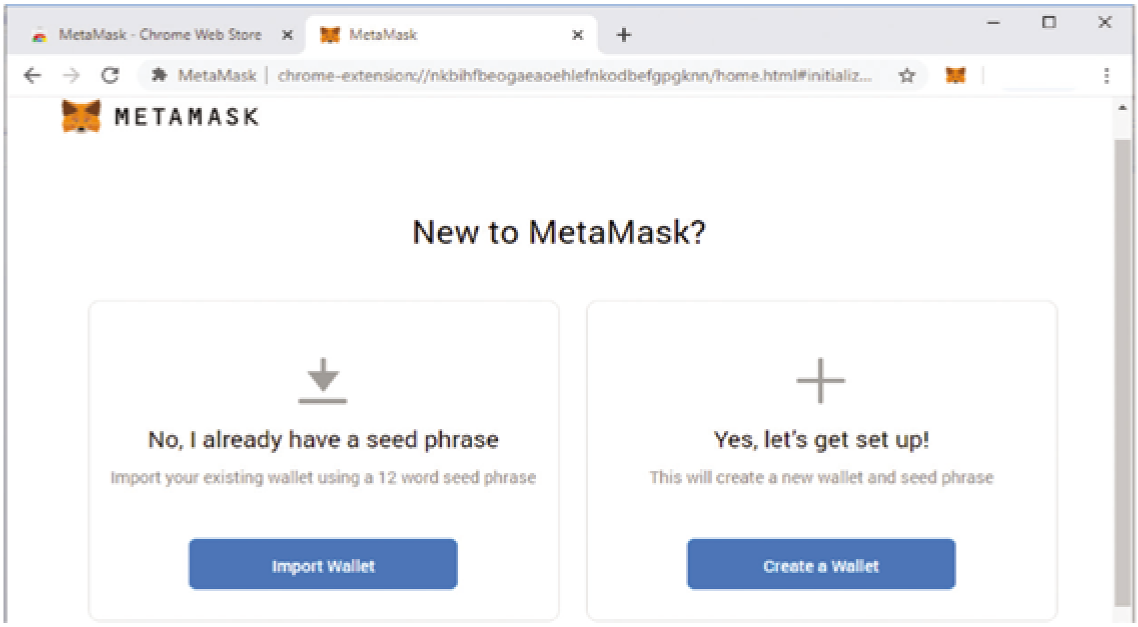
Figure B.4 Installation of Metamask – Step 4b.

Once installation is complete, install a wallet (or "vault") to hold the cryptocurrencies. To do this, you have to follow the steps below:

Step 5 – Click on the MetaMask logo in the upper right-hand corner of the Google chrome browser.

Step 6 – Read and agree to the terms and conditions. You may have to agree to 2 to 3 pages worth of terms.

Step 7 – Creating a wallet in MetaMask. In case you have already created a wallet in past and you may like to import it into a new machine, use the Import tab otherwise create a Wallet in the right tab of the screen, as shown in Figure B.5.


MetaMask - Chrome Web Store
MetaMask
MetaMask | chrome-extension://nkbihfbeogaeaoehlefnkodbefgpgknn/home.html#initializ...
METAMASK
Import Wallet
Create a Wallet

New to MetaMask?

No, I already have a seed phrase

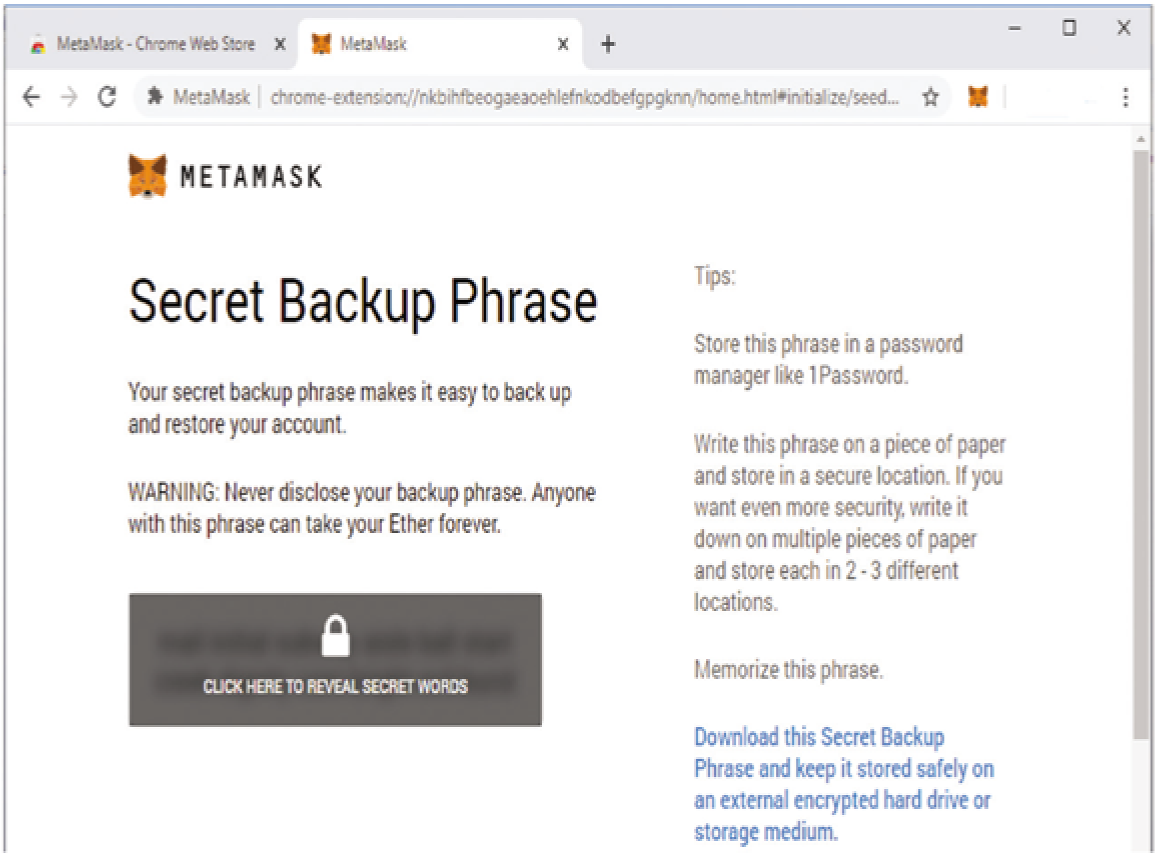
Import your existing wallet using a 12 word seed phrase

Yes, let's get set up!

This will create a new wallet and seed phrase

Figure B.5 Installation of MetaMask – Step 7.

Step 8 – The password should be a minimum of 8 characters. A secret backup phrase is used to reproduce password in case the password is forgotten. Once this activity is over you will be shown something similar to Figure B.6.



MetaMask - Chrome Web Store
MetaMask
MetaMask | chrome-extension://nkbihfbeogaeaoehlefnkodbefgpgknn/home.html#initialize/seed...
METAMASK
CLICK HERE TO REVEAL SECRET WORDS

Secret Backup Phrase

Your secret backup phrase makes it easy to back up and restore your account.

WARNING: Never disclose your backup phrase. Anyone with this phrase can take your Ether forever.

Tips:

Store this phrase in a password manager like 1Password.

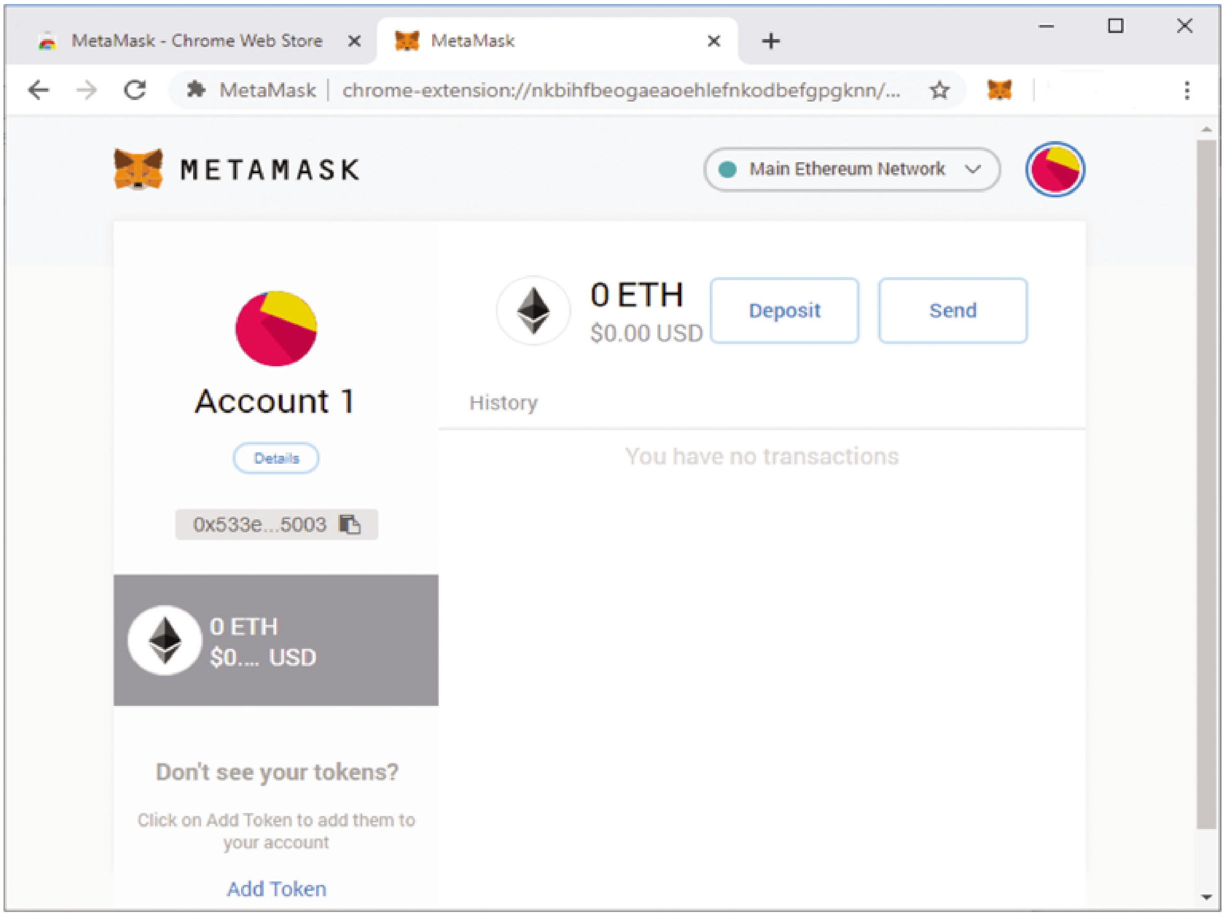
Write this phrase on a piece of paper and store in a secure location. If you want even more security, write it down on multiple pieces of paper and store each in 2 - 3 different locations.

Memorize this phrase.

Download this Secret Backup Phrase and keep it stored safely on an external encrypted hard drive or storage medium.

Figure B.6 Installation of MetaMask – Step 8a.

The process is complete after verifying the pass phrase and you can use MetaMask as shown in Figure B.7.



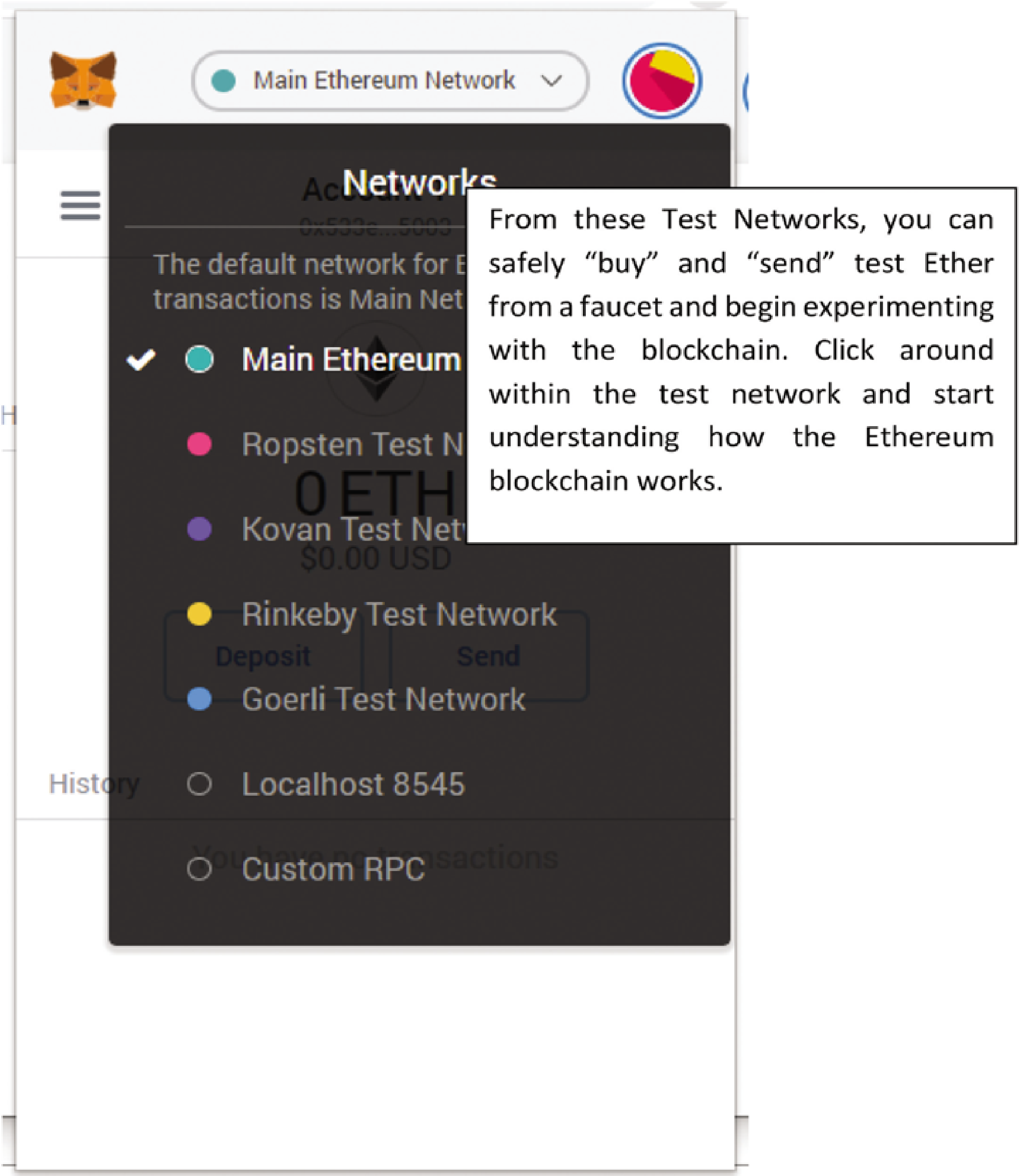
You have no transactions

Don't see your tokens?

Click on Add Token to add them to your account

Figure B.7 Installation of Metamask – Step 8b.

You are now in the Ethereum Mainnet network. To start experimenting with MetaMask, you can switch to one of the Testnet networks by clicking "Main Network" in the right-hand corner of the wallet pop up screen, and selecting one of the Testnets such as Ropsten Test Network or Kovan Test Network (Figure B.8).



From these Test Networks, you can safely "buy" and "send" test Ether from a faucet and begin experimenting with the blockchain. Click around within the test network and start understanding how the Ethereum blockchain works.

Figure B.8 Installation of Metamask – Step 8c.

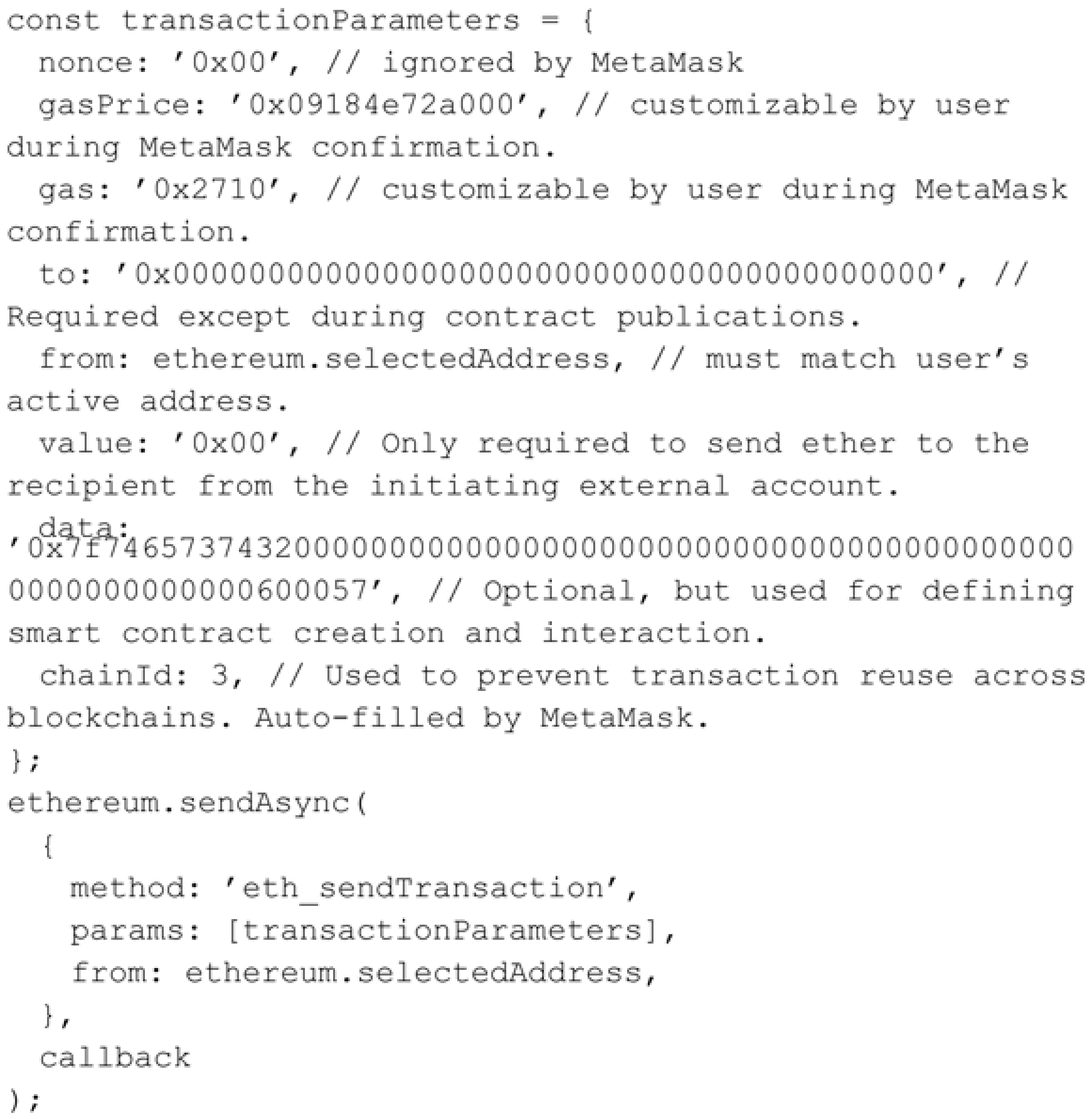
B.2 Initializing Dapps

Once you have your basic dev environment set up, you are ready to start interacting with some smart contracts. There are some basic things you will need regardless of which library you are going to use when communicating with a smart contract. It is always recommended that you should deliberately connect to a test-network version of your smart contract initially, which makes it easy for you to “try out” on your system without using real money.

Every account in Ethereum has an address, whether it is an external key–pair account or a smart contract. In order for any smart contract library to communicate with your contracts, they will need to know its exact address. The application binary interface (ABI) specification of Ethereum is created to encode the interface of a smart contract. It is an array of method-describing objects, and when you feed this and the address into a contract-abstraction library such as web3, Truffle, ethjs, and Embark, this ABI tells those libraries about what methods to provide, and how to compose transactions to call those methods. Your account can be used in a variety of context in Ethereum, including as identifiers and for signing transactions. In order to request a signature from the user or have the user approve a transaction, you must be able to access the user’s accounts. The wallet methods below involve a signature or transaction approval and all require sending the account as a function parameter.

1. eth_sendTransaction
2. eth_sign (insecure and unadvised to use)
3. eth_personalSign
4. eth_signTypedData

Transactions are a formal action on a blockchain. They are always initiated in MetaMask with a call to the eth_sendTransaction method. They can involve a simple sending of ether and may result in sending tokens, creating a new smart contract, or changing state on the blockchain in a number of ways. They are always initiated by a signature from an external account or a simple key pair. In MetaMask, using the ethereum.sendAsync method directly and sending a transaction involves composing an options object such as follows:



```
const transactionParameters = {
  nonce: '0x00', // ignored by MetaMask
  gasPrice: '0x09184e72a000', // customizable by user
during MetaMask confirmation.
  gas: '0x2710', // customizable by user during MetaMask
confirmation.
  to: '0x0000000000000000000000000000000000000000', //
Required except during contract publications.
  from: ethereum.selectedAddress, // must match user's
active address.
  value: '0x00', // Only required to send ether to the
recipient from the initiating external account.
  data:
'0x7f746573743200000000000000000000000000000000000000000
0000000000000600057', // Optional, but used for defining
smart contract creation and interaction.
  chainId: 3, // Used to prevent transaction reuse across
blockchains. Auto-filled by MetaMask.
};
ethereum.sendAsync(
  {
    method: 'eth_sendTransaction',
    params: [transactionParameters],
    from: ethereum.selectedAddress,
  },
  callback
);
```

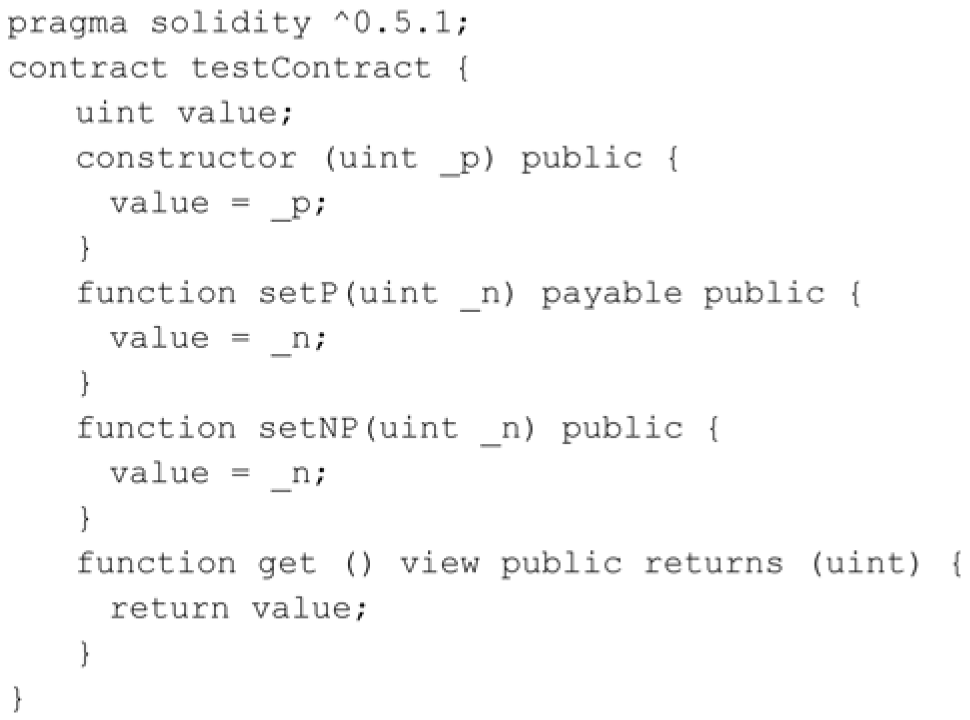
B.3 Creating and Deploying a Contract Using Remix

There are three types of environments in Remix (http://remix.ethereum.org/), which can be plugged to the browser:

1. **JavaScript VM:** In this, all the transactions will be executed in a sandbox blockchain in the browser. This means nothing will be persisted when you reload the page. The JsVM is its own blockchain; on each reload it will start a new blockchain and the old one will not be saved.
2. **Injected provider:** In this, Remix will connect to an injected web3 provider. MetaMask is an example of a provider that inject web3.
3. **Web3 provider:** In this, Remix will connect to a remote node. You will need to provide the URL to the selected provider: Geth, parity, or any Ethereum client.

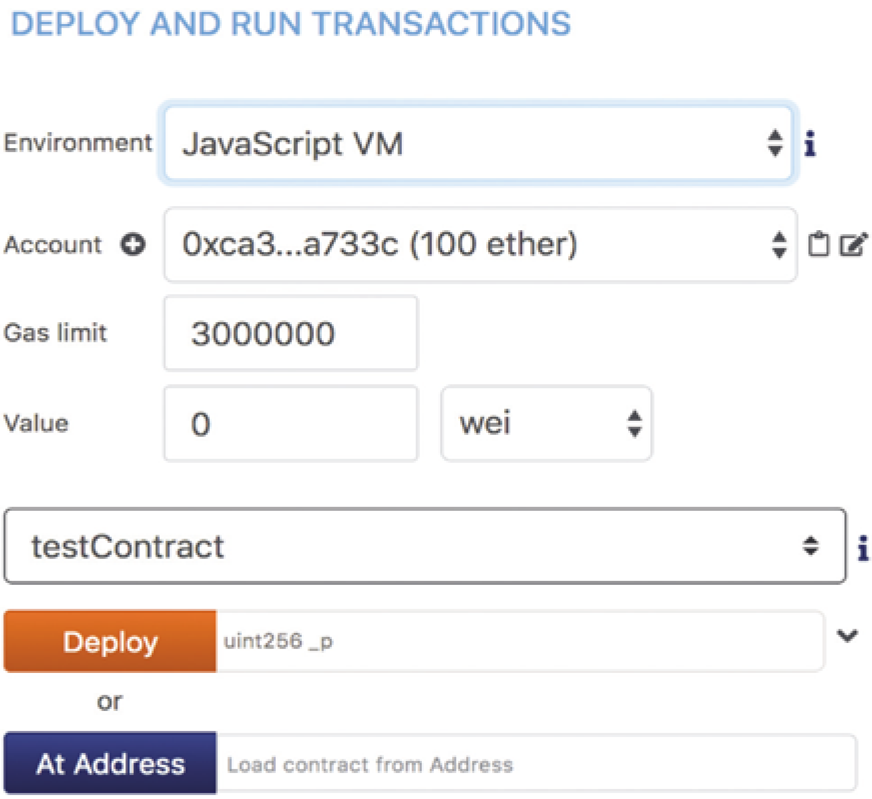
Both Web3 provider and Injected provider require the use of an external tool. The external tool for Web3 provider is an Ethereum node and for Injected provider is the MetaMask. The JavaScript VM mode is convenient because each execution runs in your browser and you do not need any other software or Ethereum node to run it. Thus, it is the easiest test environment as no set up is required. But keep in mind that reloading the browser when you are in JavaScript VM will restart Remix in an empty state. Therefore, make sure the VM mode is selected and all accounts displayed in accounts should have 100 ether.

We are now going to show you a sample contract which is very basic. The goal is to quickly start to create and interact with a sample contract. Go to the Remix site and you will be prompted an IDE in which you can enter the following code. However, you will see several *. sol files on the left pane, which are avaliable by default.



```
pragma solidity ^0.5.1;
contract testContract {
    uint value;
    constructor (uint _p) public {
      value = _p;
    }
    function setP(uint _n) payable public {
      value = _n;
    }
    function setNP(uint _n) public {
      value = _n;
    }
    function get () view public returns (uint) {
      return value;
    }
}
```

After creating an instance, the Compile tab displays information related to the current contract. Go to the Run tab and select JavaScript VM to specify that you are going to deploy an instance of the contract in the JavaScript VM state (Figure B.9).



DEPLOY AND RUN TRANSACTIONS

Environment JavaScript VM

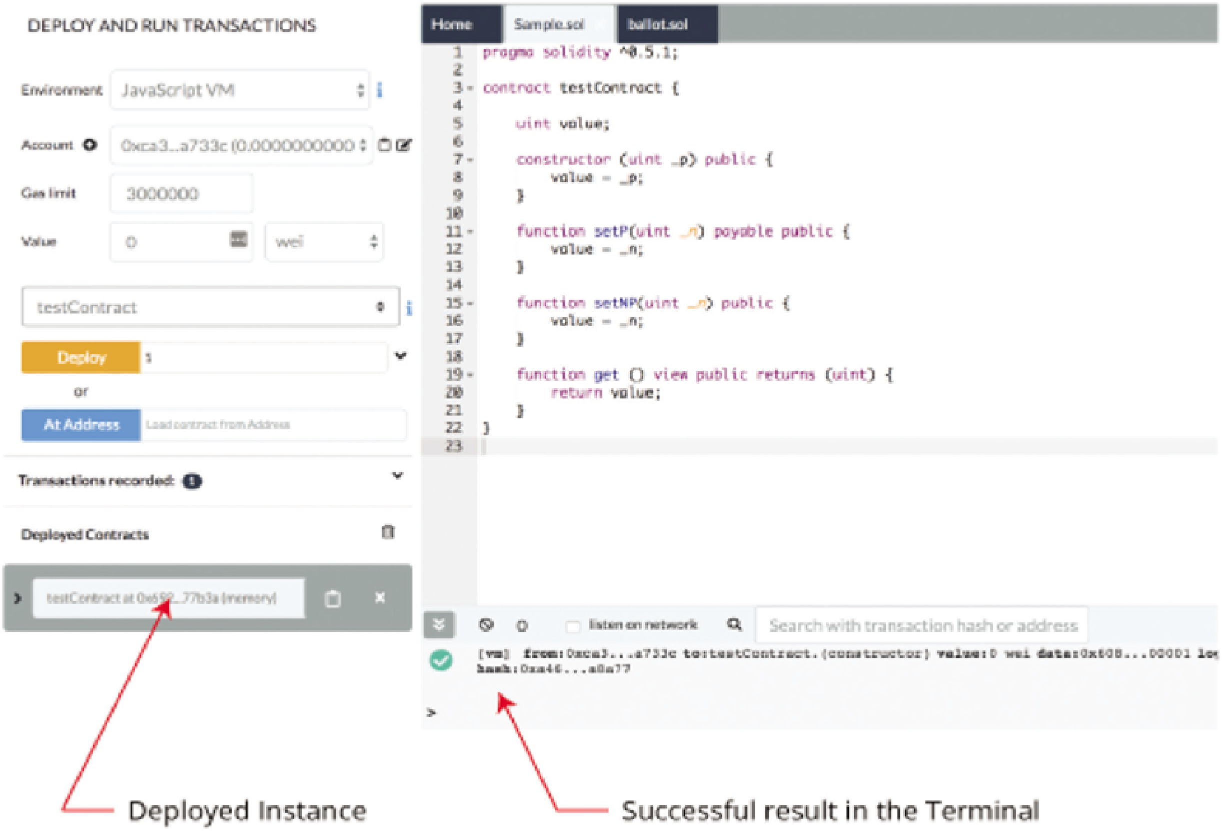
Account 0xca3...a733c (100 ether)

Gas limit 3000000

Value 0 wei

testContract

Figure B.9 Deploy and run transactions screen.


DEPLOY AND RUN TRANSACTIONS
Home
Sample.sol
ballot.sol
Deploy
or
At Address
Load contract from Address
Deployed Contracts
listen on network
Search with transaction hash or address

Environment: JavaScript VM

Account: 0xca3...a733c (0.0000000000

Gas limit: 3000000

Value: 0 wei

testContract

Deploy: 5

Transactions recorded: 1

testContract at 0x659...77b3a (memory)

```
pragma solidity ^0.5.1;

contract testContract {

    uint value;

    constructor (uint _p) public {
        value = _p;
    }

    function setP(uint _n) payable public {
        value = _n;
    }

    function setNP(uint _n) public {
        value = _n;
    }

    function get () view public returns (uint) {
        return value;
    }
}
```

```
[vm] from:0xca3...a733c to:testContract.(constructor) value:0 wei data:0x608...00001 lo
hash:0xa46...a8a77
>
```

Deployed Instance

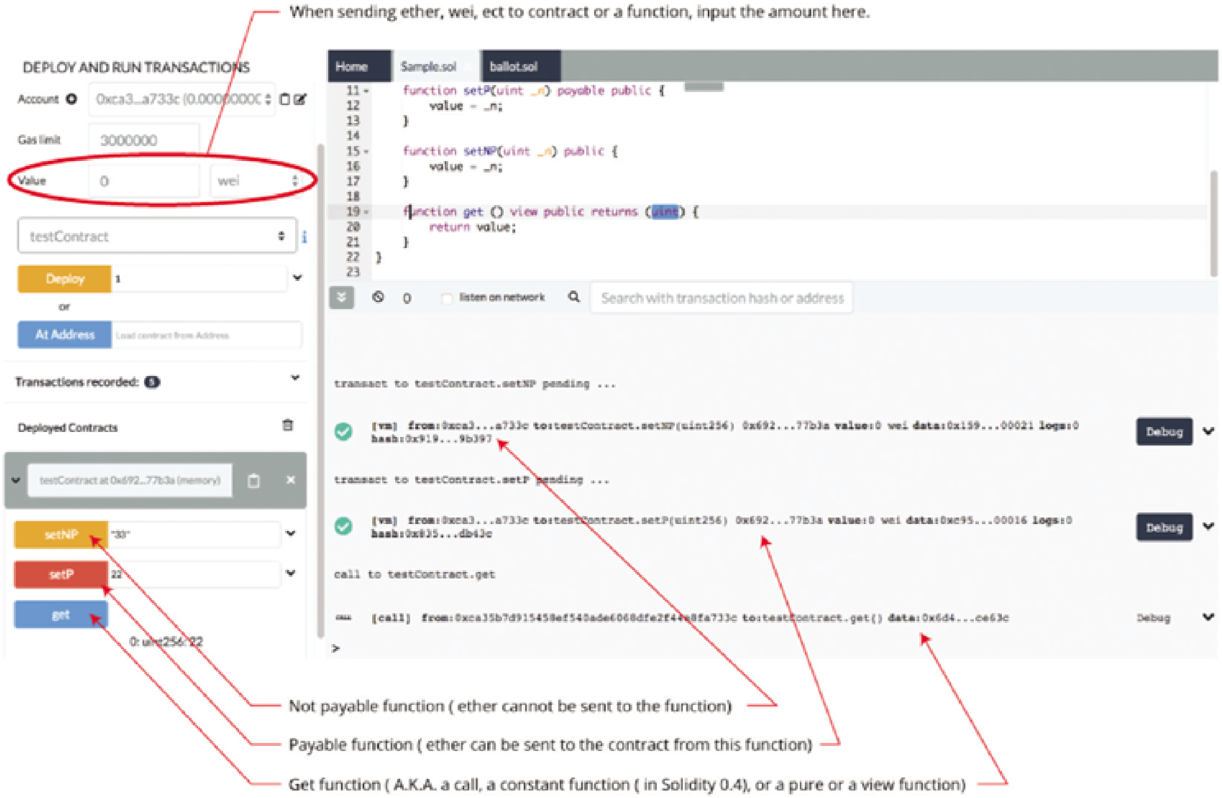
Successful result in the Terminal

Figure B.10 Newly created instance.

This newly created instance is displayed in the Run tab (Figure B.10).

This new instance contains 3 actions which corresponds to the 3 functions – `setP`, `setPN`, get (Figure B.11). Clicking on `SetP` or `SetPN` will create a new transaction. However, note that `SetP` is payable: it is possible to send value (Ether) to the contract. `SetPN` is not payable: it is not possible to send value (Ether) to the contract. Clicking on `get` will not execute a transaction: it does not execute a transaction because `get` does not modify the state (variable value) of this instance. As `get` is view you can see the return value just below the action.

Now, let us look at the debugging transactions. There are two ways, and each one corresponds to a different use case.



When sending ether, wei, ect to contract or a function, input the amount here.

DEPLOY AND RUN TRANSACTIONS

Account: 0xca3...a733c (0.000000000
Gas limit: 3000000
Value: 0 wei

testContract

Deploy 1

or

At Address: Load contract from Address

Transactions recorded: 3

Deployed Contracts

testContract at 0x692...77b3a (memory)

setNP "30"
setP 22
get

0: uint256: 22

```
    function setP(uint _n) payable public {
        value = _n;
    }

    function setNP(uint _n) public {
        value = _n;
    }

    function get () view public returns (uint) {
        return value;
    }
}
```

```
transact to testContract.setNP pending ...

[vm] from:0xca3...a733c to:testContract.setNP(uint256) 0x692...77b3a value:0 wei data:0x139...00021 logs:0
hash:0x919...9b397

transact to testContract.setP pending ...

[vm] from:0xca3...a733c to:testContract.setP(uint256) 0x692...77b3a value:0 wei data:0xc95...00016 logs:0
hash:0x835...db43c

call to testContract.get

[call] from:0xca35b7d915458ef540ade6068dfe2f44e8fa733c to:testContract.get() data:0x6d4...ce63c
```

Not payable function (ether cannot be sent to the function)

Payable function (ether can be sent to the contract from this function)

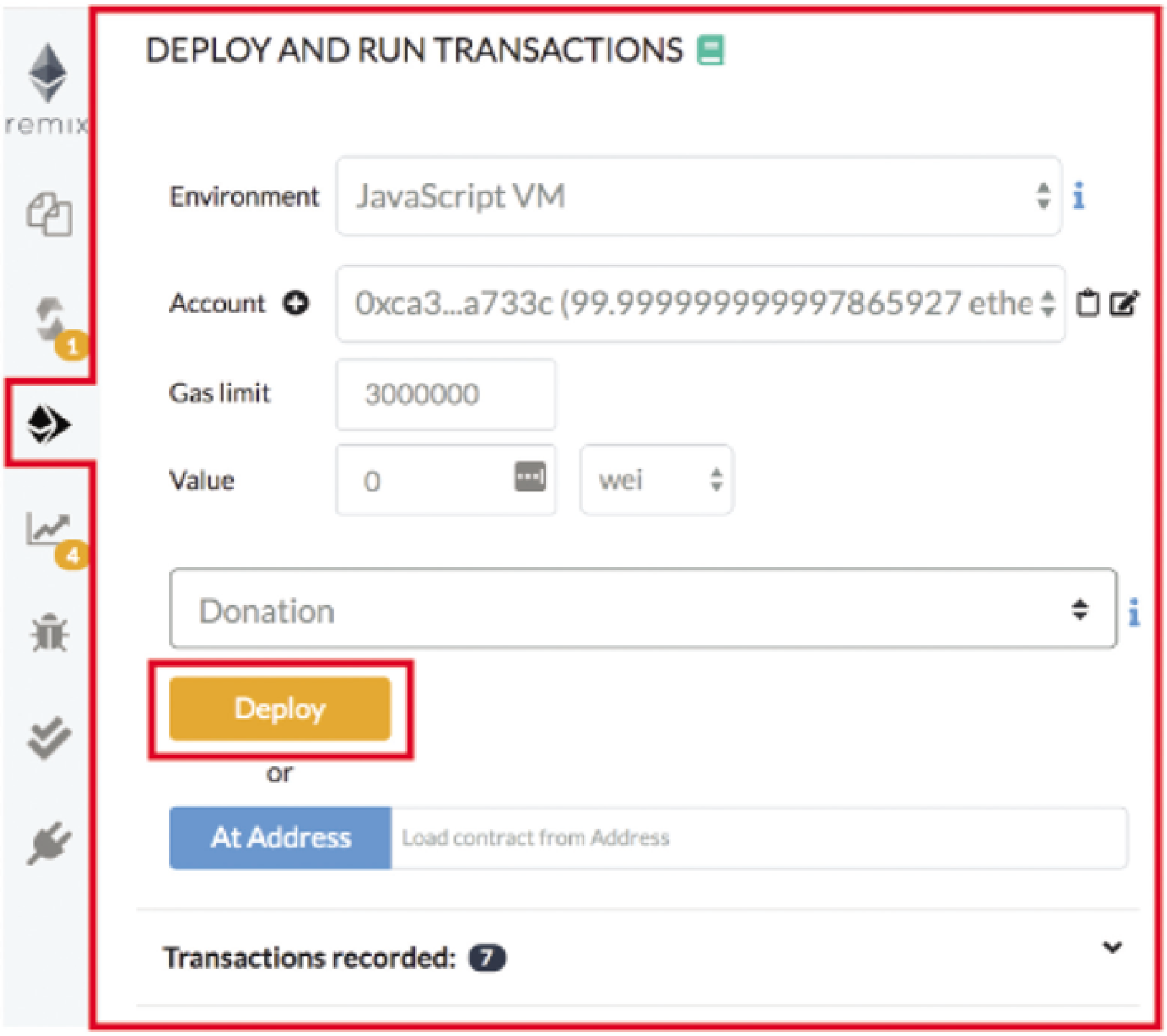
Get function (A.K.A. a call, a constant function (in Solidity 0.4), or a pure or a view function)

Figure B.11 The three functions – `setP, setPN, get.`

1. **From the transaction log in the Terminal:** Use this when you want to debug a transaction.
2. **From the Debugger:** Use this if you have a transaction hash.

Let us start with a basic contract and initiate Debugging from the transaction log in the Terminal. First create a blank file in the file explorer (by clicking the + icon) and give it a name. Then copy and compile the following code and click the Run & Deploy icon in the icon panel.

We will run the JavaScript VM as a simple example to understand this better. This simulates a custom blockchain. You could do the same using a proper backend node. Let us deploy the contract by clicking the Deploy button (Figure B.12).



DEPLOY AND RUN TRANSACTIONS

Environment: JavaScript VM

Account: 0xca3...a733c (99.999999999997865927 ethe

Gas limit: 3000000

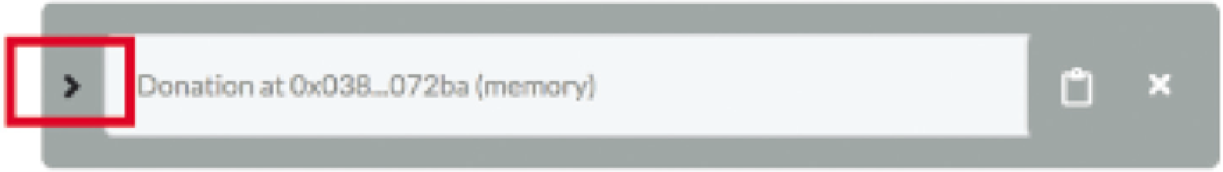
Value: 0 wei

Donation

Transactions recorded: 7

Figure B.12 Deploying the contract.

Figure B.13 shows the deployed instance.



Donation at 0x038...072ba (memory)

Figure B.13 The deployed instance.

Then open it up (by clicking the caret) Figure B.14.

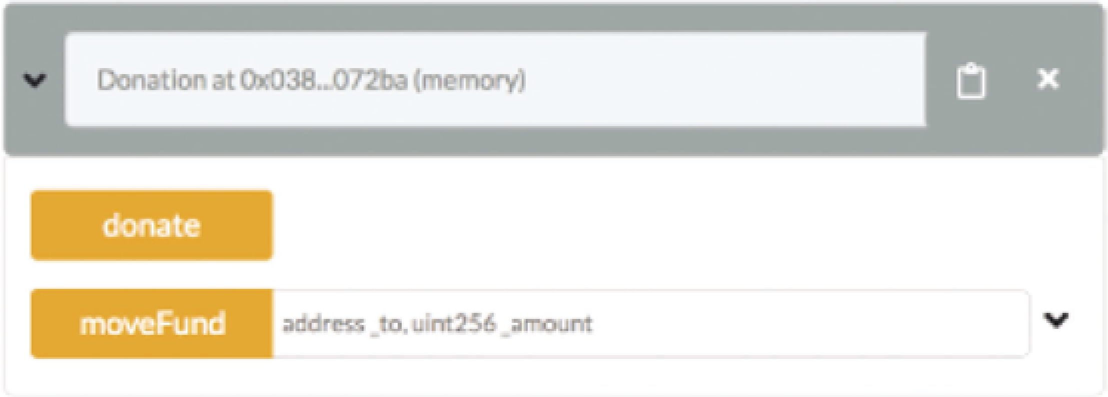
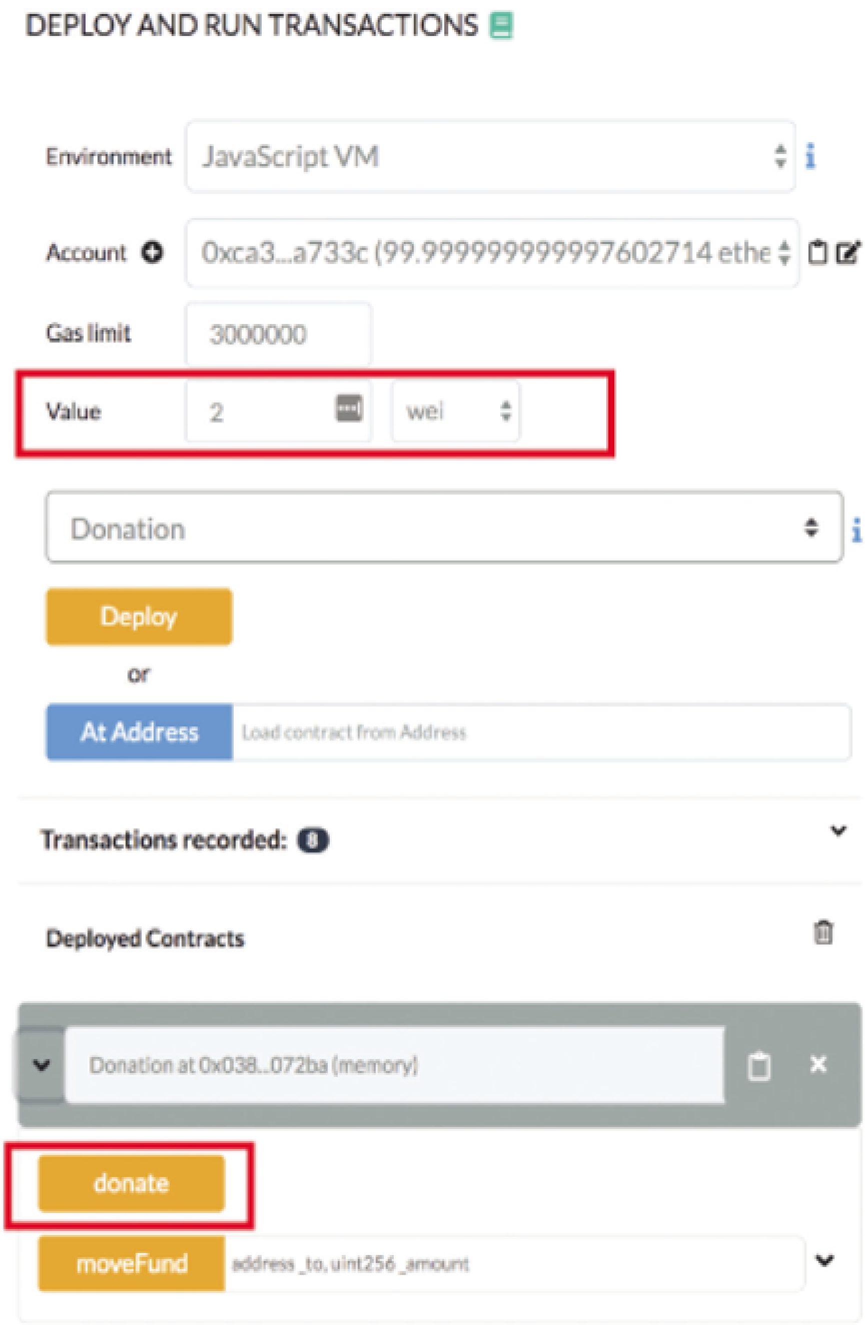

Donation at 0x038...072ba (memory)

Figure B.14 Opening the deployed instance.

Then call the Donate function to send it to ether. To do this, put 2 in the value input box and select Ether as the unit (and not wei). Click the Donate button (Figure B.15) to send ether to the this function.



DEPLOY AND RUN TRANSACTIONS

Environment: JavaScript VM

Account: 0xca3...a733c (99.99999999999997602714 ethe

Gas limit: 3000000

Value: 2 wei

Donation

Transactions recorded: 8

Deployed Contracts

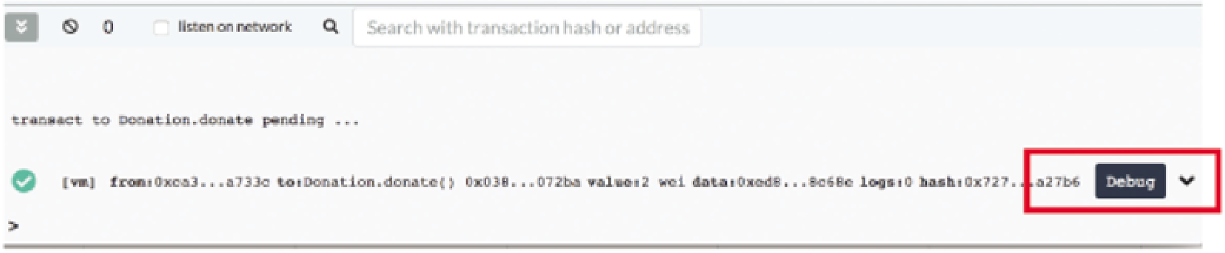
Donation at 0x038...072ba (memory)

moveFund: address _to, uint256 _amount

Figure B.15 Donate function to send to ether.

Because we are using JavaScript VM, everything happens almost instantly. However, if we had been using Injected Web 3, then we would need to approve the transaction, pay for gas and wait for the transaction to get mined.

Remix displays information related to each transaction result in the terminal. Therefore, check in the terminal where the transaction you just made is logged and click the Debug button (Figure B.16) to start debugging.



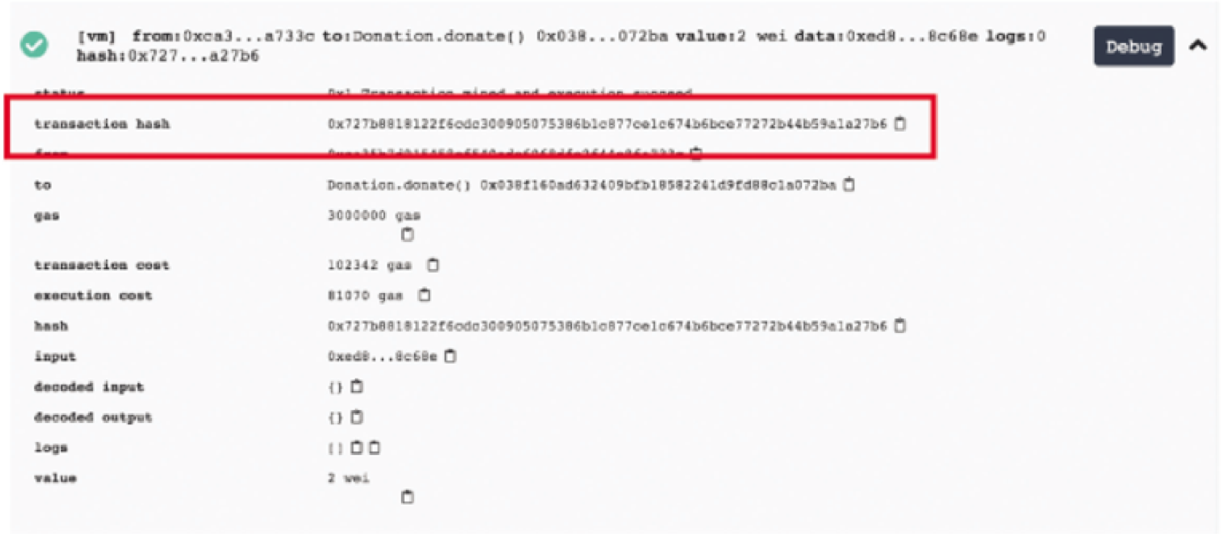
```
transact to Donation.donate pending ...

[vm] from:0xca3...a733c to:Donation.donate() 0x038...072ba value:2 wei data:0xed8...8c68e logs:0 hash:0x727...a27b6
>
```

Figure B.16 Initiating debugging.

When a compilation succeeds, Remix creates a JSON file for each compiled contract. The JSON file contains the compilation's artifact. For this to happen, you need to check the **Generate contract metadata** box in the **General settings** section of the **Settings** module. The .JSON file with the metadata will appear in the file explorer where the compiled file is located. This JSON file contains the link to the libraries, the bytecode, the deployed bytecode, the gas estimation, the method identifiers, and the ABI. You can write scripts that can access this file.

There is a second approach to debugging, which can be done directly from the Debugger. Click the Debug icon (Figure B.17) to get to the debugger in the side panel. If you do not see the Debug icon, go to the plugin manager and activate the debugger. You can start a debug session by providing a transaction hash. In order to find the transaction hash, go through the following steps:

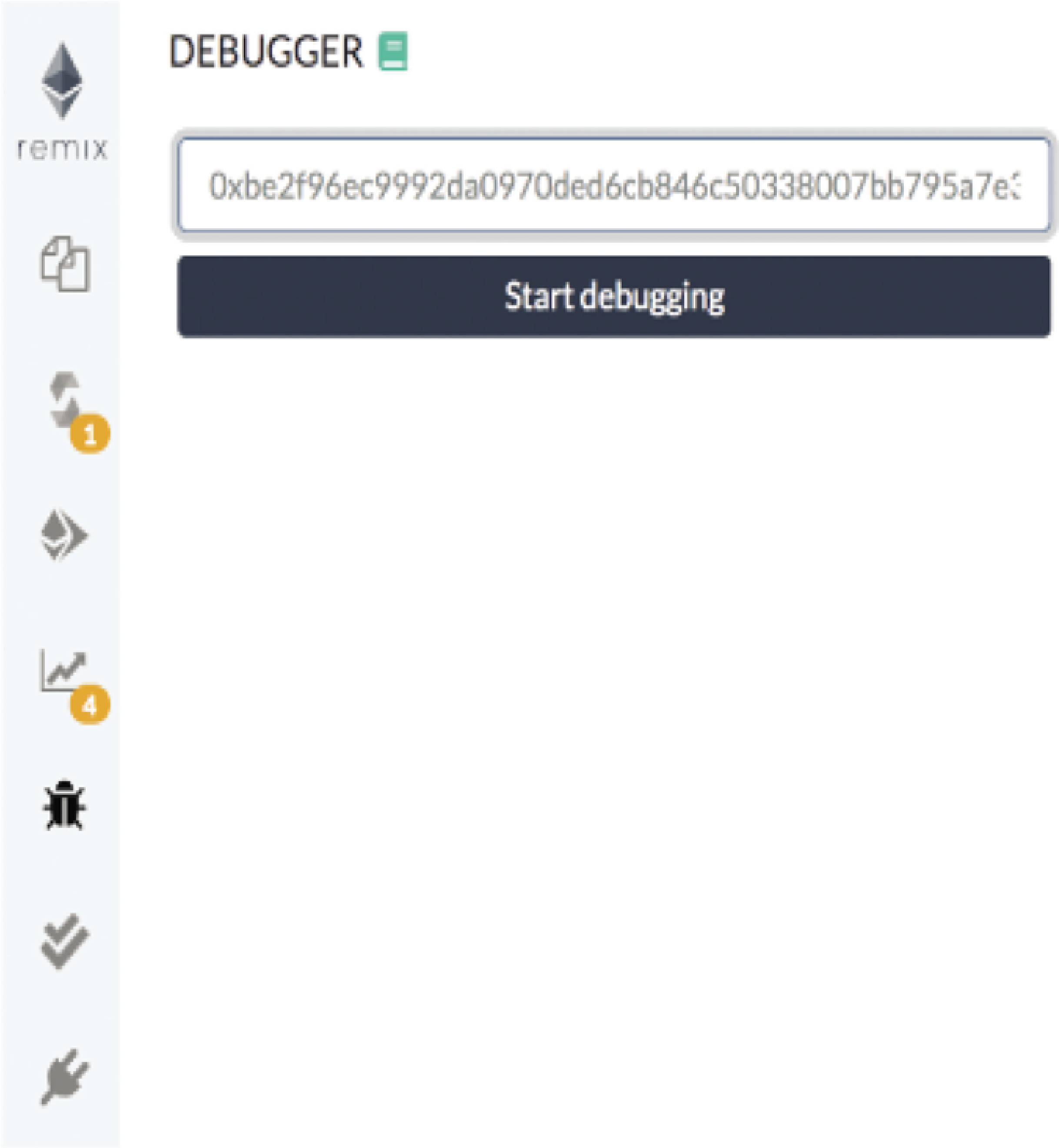


```
[vm] from:0xca3...a733c to:Donation.donate() 0x038...072ba value:2 wei data:0xed8...8c68e logs:0
hash:0x727...a27b6
status              0x1 Transaction mined and execution succeed
transaction hash    0x727b8818122f6cdc300905075386b1c877ce1c674b6bce77272b44b59a1a27b6
from                [illegible]
to                  Donation.donate() 0x038f160ad632409bfb18582241d9fd88c1a072ba
gas                 3000000 gas
transaction cost    102342 gas
execution cost      81070 gas
hash                0x727b8818122f6cdc300905075386b1c877ce1c674b6bce77272b44b59a1a27b6
input               0xed8...8c68e
decoded input       {}
decoded output      {}
logs                []
value               2 wei
```

Figure B.17 Debugging.

Step 1 – Go to a transaction in the terminal.
Step 2 – Click a line with a transaction to expand the log.
Step 3 – Select and copy the transaction hash.

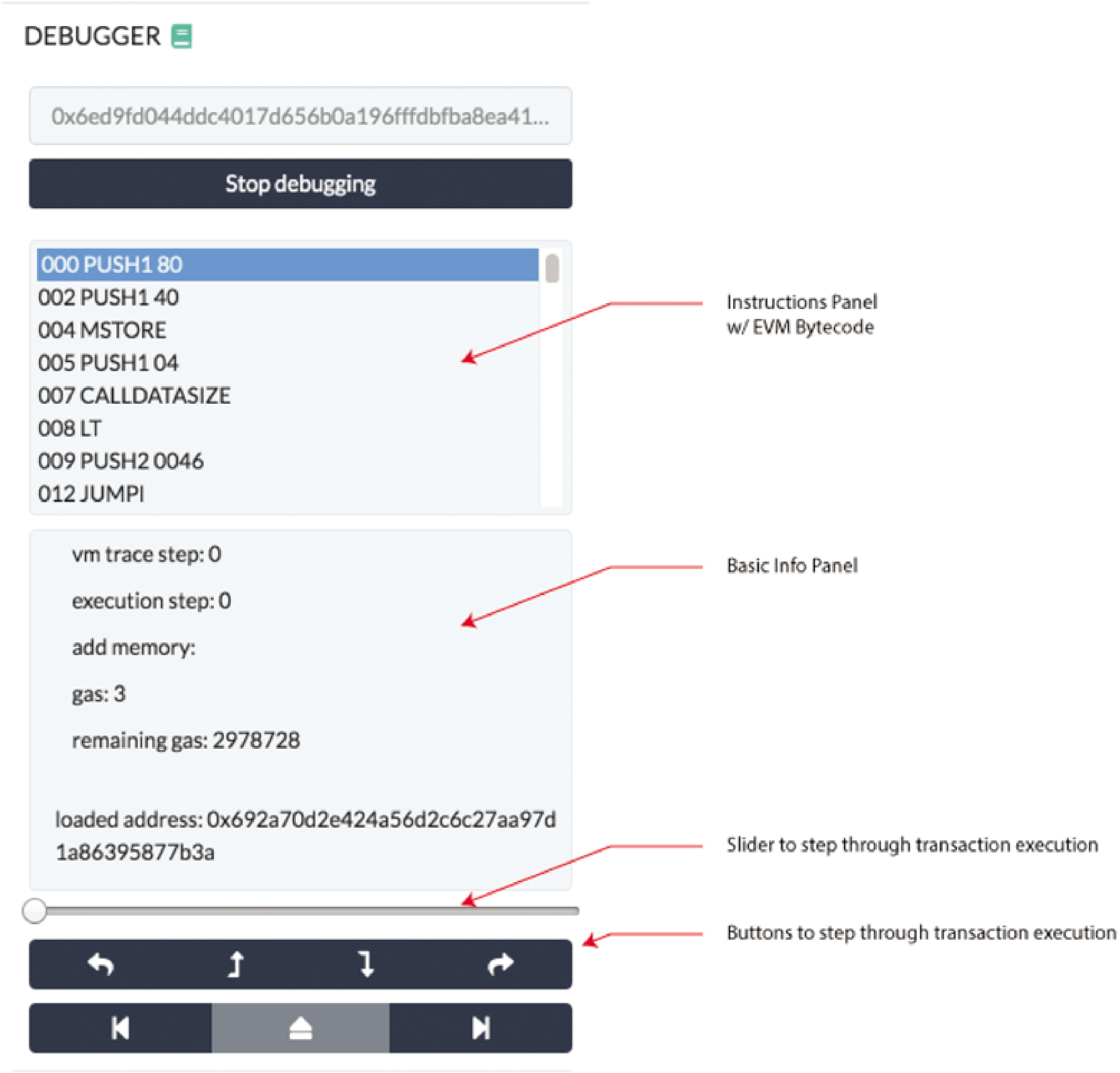
Then click in the debugger, paste the hash, and click on the **Start debugging** button (Figure B.18).



0xbe2f96ec9992da0970ded6cb846c50338007bb795a7e3

Figure B.18 Start debugging.

Figure B.19 shows how to use the debugger.



DEBUGGER

0x6ed9fd044ddc4017d656b0a196fffdbfba8ea41...

Stop debugging

```
000 PUSH1 80
002 PUSH1 40
004 MSTORE
005 PUSH1 04
007 CALLDATASIZE
008 LT
009 PUSH2 0046
012 JUMPI
```

vm trace step: 0

execution step: 0

add memory:

gas: 3

remaining gas: 2978728

loaded address: 0x692a70d2e424a56d2c6c27aa97d1a86395877b3a

Instructions Panel
w/ EVM Bytecode

Basic Info Panel

Slider to step through transaction execution

Buttons to step through transaction execution

Figure B.19 How to use the debugger.

The debugger allows you to see detailed information about the transaction's execution. It uses the editor to display the location in the source code of the current execution. The navigation part contains a slider and buttons that can be used to step through the transaction execution.

B.4 Remix Commands

In the console, you can run the following commands. Once you start to type a command, there is *auto completion*. These commands use the following libraries:

Ethers – This library is a compact and complete JavaScript library for Ethereum.

Remix – Ethereum IDE and tools for the web.

web3 – This library is a collection of modules that contain specific functionality for Ethereum ecosystem.

Swarmgw – This library can be used to upload/download files to Swarm via https://swarm-gateways.net/

The following are the list of commands:

`remix.debug(hash)`**:** Starts debugging a transaction.

`remix.debugHelp()`**:** Displays help message for debugging

`remix.execute(filepath)`**:** Runs the script specified by the file path. If the path is empty, the script currently displayed in the editor is executed.

`remix.exeCurrent()`**:** Runs the script currently displayed in the editor.

`remix.getFile(path)`**:** Returns the content of the file located at the given path.

`remix.help()`**:** Displays help message.

`remix.loadgist(id)`**:** Loads a gist in the file explorer.

`remix.loadurl(url)`**:** Loads the given URL in the file explorer. The URL can be of type GitHub, Swarm, or IPFS.

`remix.setFile(path, content)`**:** Sets the content of the file located at the given path.

`remix.setproviderurl(url)`**:** Changes the current provider to Web3 provider and sets the URL endpoint.

`swarmgw.get(url, cb)`: Downloads files from Swarm via https**://swarm-gateways.net/

`swarmgw.put(content, cb)`: Uploads files to Swarm via https**://swarm-gateways.net/

`ethers.Contract`: Provides a graceful connection to a contract deployed on the blockchain, simplifying calling and querying its functions and handling all the binary protocol and conversion as necessary.

`ethers.HDNode`: A Hierarchical Deterministic Wallet represents a large tree of private keys that can reliably be reproduced from an initial seed.

`ethers.Interface`: The Interface Object is a metaclass that accepts a Solidity (or compatible) ABI and populates functions to deal with encoding and decoding the parameters to pass in and results returned.

`ethers.providers`: A Provider abstracts a connection to the Ethereum blockchain for issuing queries and sending state changing transactions.

`ethers.SigningKey`: This provides an abstraction around the secp256k1 elliptic curve cryptography library.

`ethers.utils`: The utility functions exposed in both the ethers umbrella package and the ethers-utils.

`ethers.utils.AbiCoder`: Creates a new ABI Coder object

`ethers.utils.RLP`: This encoding method is used internally for several aspects of Ethereum such as encoding transactions and determining contract addresses.

`ethers.Wallet`: A wallet manages a private–public key pair that is used to cryptographically sign transactions and prove ownership on the Ethereum network.

`ethers.version`: Contains the version of the ethers container object.

`web3.bzz`: The bzz module is for interacting with Swarm network.

`web3.eth`: The eth module is for interacting with the Ethereum network.

`web3.eth.accounts`: This contains functions to generate Ethereum accounts and sign transactions and data.

`web3.eth.abi`: This function lets you decode and encode parameters to ABI for function calls to the Ethereum Virtual Machine (EVM).

`web3.eth.ens`: This function lets you interact with ENS.

`web3.eth.Iban`: This function converts Ethereum addresses from and to IBAN and BBAN.

`web3.eth.net`: This module is for interacting with network properties.

`web3.eth.personal`: This module is for interacting with Ethereum accounts.

`web3.eth.subscribe`: This function lets you subscribe to specific events in the blockchain.

`web3.givenProvider`: When using web3.js in an Ethereum compatible browser, it will set with the current native provider by that browser. It will return the given provider by the (browser) environment, otherwise null.

`web3.modules`: Contains the version of the web3 container object.

`web3.providers`: Contains the current available providers.

`web3.shh`: This module is for interacting with the Whisper protocol.

`web3.utils`: This package provides utility functions for Ethereum Dapps and other **web3.js packages.

`web3.version`: Contains the version of the web3 container object.

`web3.eth.clearSubscriptions();`: Resets subscriptions.

`web3.eth.Contract(jsonInterface[, address] [, options])`: The **web3.eth.Contract object makes it easy to interact with smart contracts on Ethereum blockchain.

`web3.eth.accounts.create([entropy]);`: The web3.eth.accounts contains functions to generate Ethereum accounts and sign transactions and data.

B.5 Setting up Ethereum Project

Infrastructure

The intent of Ethereum is to create an alternative protocol for building decentralized applications, with particular emphasis on situations where rapid development time, security for small and rarely used applications, and the ability of different applications to very efficiently interact are important. Ethereum does this by building the abstract foundational layer of a blockchain with a builtin programming language, allowing anyone to write smart contracts and decentralized applications. It thus provides a platform to create your own arbitrary rules for ownership, transaction formats, and state transition functions. A bare-bones version of Namecoin can be written in two lines of code, and other protocols like currencies and reputation systems can be built within twenty lines of code. Smart contracts, cryptographic "boxes" that contain value and only unlock if certain conditions are met, can also be built on top of the platform. As we know, transactions are at the heart of the Ethereum blockchain (or any blockchain for that matter).

When you interact (Figure B.20) with the Ethereum blockchain, you are executing transactions and updating its state. In this section, we present what exactly happens when you execute a transaction in Ethereum. These transactions run on the EVM and are commonly referred to as *smart contracts*. The most popular languages for writing smart contracts on Ethereum are Solidity and Vyper. Solidity is inspired by C++, Python, and JavaScript.

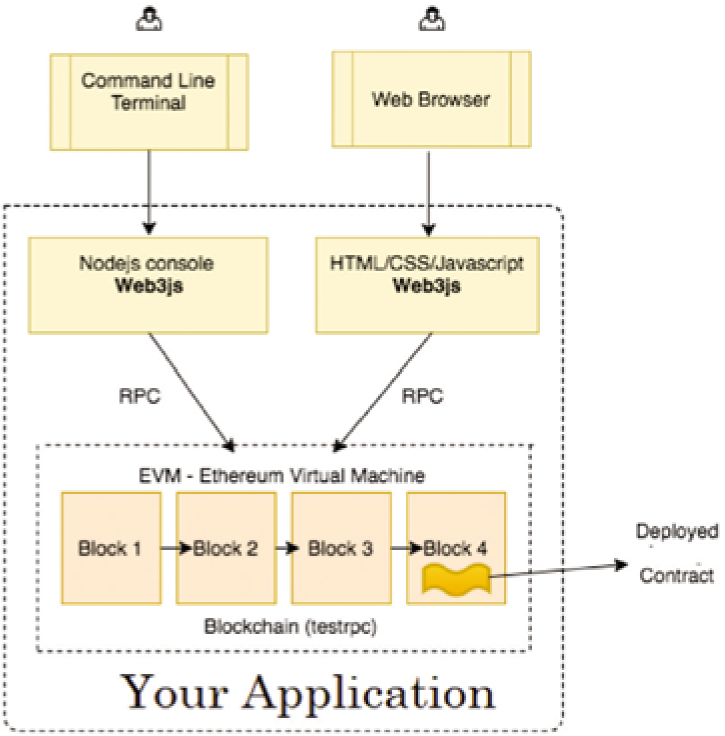


Figure B.20 Interacting with Ethreuem blockchain.

Installing the Solidity Compiler

Solidity versions follow semantic versioning and in addition to releases, nightly development builds are also made available. The nightly builds are not guaranteed to work and despite best efforts they might contain undocumented and/or broken changes, therefore we recommend you use the latest release. Choose a command-line compiler if you are working on a larger contract or if you require more compilation options, shown as follows:

npm/Node.js

Use `npm` for a convenient and portable way to install `solcjs`, a Solidity compiler.

```
npm install -g solc
```

The command-line executable is named `solcjs`. The command-line options of `solcjs` are not compatible with `solc` and tools (such as Geth).

Docker

Docker images of Solidity builds are available using the `solc` image from the Ethereum organization. Use the stable tag for the latest released version, and nightly for potentially unstable changes in the develop branch.

The Docker image runs the compiler executable, so you can pass all compiler arguments to it. For example, the command below pulls the stable version of `solc` image (if you do not have it already), and runs it in a new container, passing the `--help` argument.

```
docker run ethereum/solc:stable --help
```

You can also specify release build versions in the tag, for example, for the 0.5.4 release.

```
docker run ethereum/solc:0.5.4 --help
```

To use the Docker image to compile Solidity files on the host machine, mount a local folder for input and output and specify the contract to compile. For example,

```
docker run -v /local/path:/sources ethereum/solc:stable -o /sources/output --abi --bin /sources/Contract.sol
```

You can also use the standard JSON interface (recommended when using the compiler with tooling). When using this interface it is not necessary to mount any directories.

```
docker run ethereum/solc:stable --standard-json <
input.json > output.json
```

Binary Packages

Binary packages of Solidity are available at solidity/releases for Ubuntu, you can get the latest stable version using the following commands:

```
sudo add-apt-repository ppa:ethereum/ethereum
sudo apt-get update
sudo apt-get install solc
```

The nightly version can be installed using the following commands:

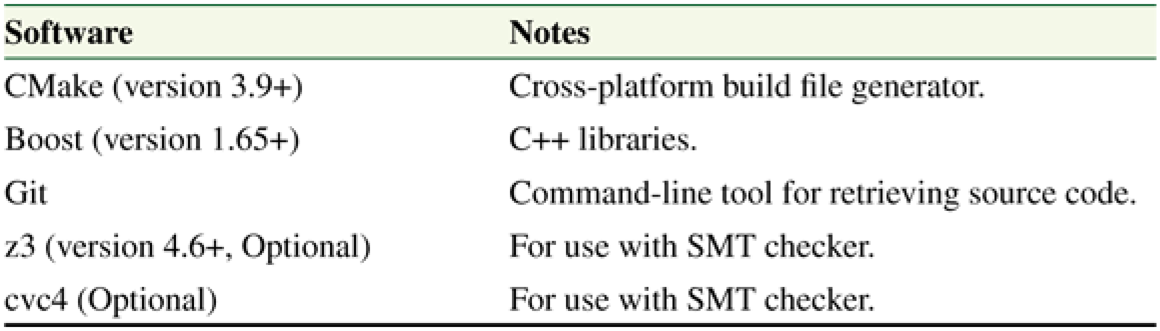
```
sudo add-apt-repository ppa:ethereum/ethereum
sudo add-apt-repository ppa:ethereum/ethereum-dev
sudo apt-get update
sudo apt-get install solc
```

If you want to help testing the latest development version of Solidity with the most recent changes, use the following:

```
sudo snap install solc --edge
```

Building from Source – Prerequisites for all Operating Systems

The following table shows the dependencies for all builds of Solidity:



Software	Notes
CMake (version 3.9+)	Cross-platform build file generator.
Boost (version 1.65+)	C++ libraries.
Git	Command-line tool for retrieving source code.
z3 (version 4.6+, Optional)	For use with SMT checker.
cvc4 (Optional)	For use with SMT checker.

Minimum compiler versions

The following C++ compilers and their minimum versions can build the Solidity codebase:

1. GCC, version 5+
2. Clang, version 3.4+
3. MSVC, version 2017+

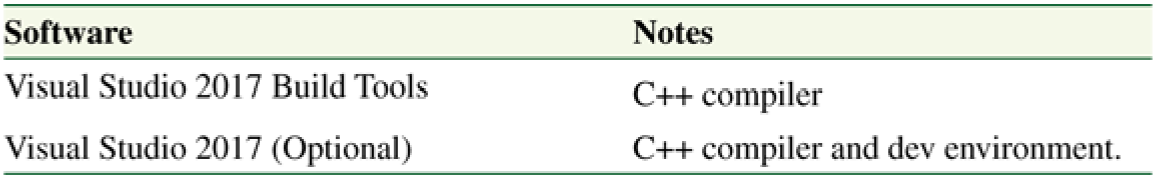
Prerequisites – macOS

For macOS builds, ensure that you have the latest version of Xcode installed. This contains the Clang C++ compiler, the Xcode IDE and other Apple development tools which are required for building C++ applications on OS X. If you are installing Xcode for the first time, or have just installed a new version then you will need to agree to the license before you can do command-line builds:

```
sudo xcodebuild -license accept
```

Prerequisites – Windows

You need to install the following dependencies for Windows builds of Solidity:



Software	Notes
Visual Studio 2017 Build Tools	C++ compiler
Visual Studio 2017 (Optional)	C++ compiler and dev environment.

If you already have one IDE and only need the compiler and libraries, you could install Visual Studio 2017 Build Tools.

Visual Studio 2017 provides both IDE and necessary compiler and libraries. So if you do not have an IDE and prefer to develop solidity, Visual Studio 2017 may be a choice for you to get everything set up easily.

Following is the list of components that should be installed in Visual Studio 2017 Build Tools or Visual Studio 2017:

1. Visual Studio C++ core features
2. VC++ 2017 v141 toolset (x86, x64)
3. Windows Universal CRT SDK
4. Windows 8.1 SDK
5. C++/CLI support

Command-Line Build

As mentioned above, you have to install External Dependencies before build. Solidity project uses CMake to configure the build. You might want to install ccache to speed up repeated builds. CMake will pick it up automatically. Building Solidity is quite similar on Linux, macOS and other Unices:

```
mkdir build
cd build
cmake .. && make
```

or even easier on Linux and macOS, you can run:

#note: this will install binaries solc and soltest at usr/local/bin

```
./scripts/build.sh
```

For Windows, you can run:

```
mkdir build
cd build
cmake -G ”Visual Studio 15 2017 Win64”..
```

This latter set of instructions should result in the creation of **solidity.sln** in the build directory. Double-clicking on that file should result in Visual Studio firing up. Alternatively, you can build for Windows on the command-line, as follows:

```
cmake --build. --config Release
```

CMake options

If you are interested what CMake options are available run

```
cmake.. -LH.
```

This completes the installation of the Solidity compiler.

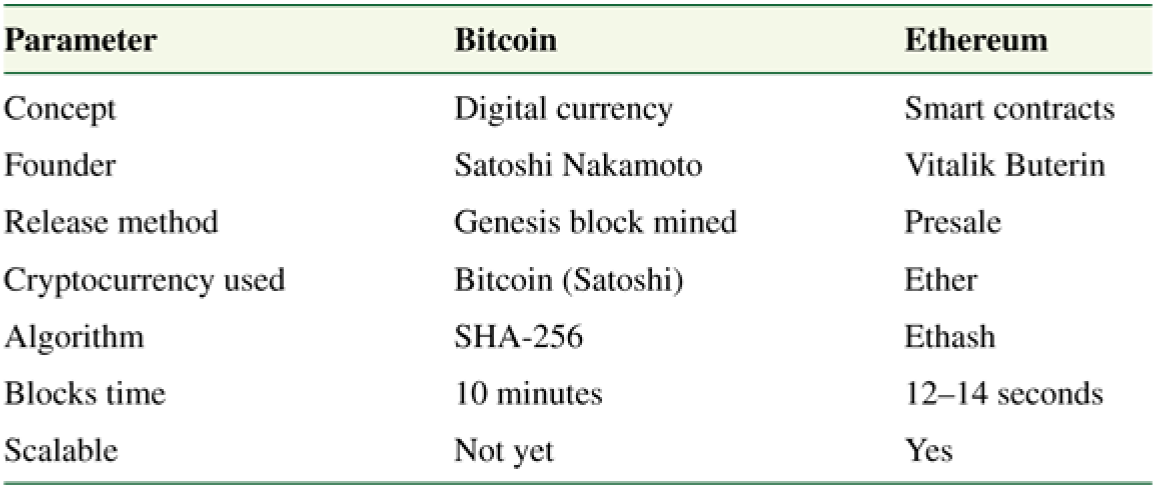
Interview Questions and Answers

C ANNEXURE

In today's changing skill landscape, no job profile is considered to be safe. Professionals from all walks of industries and experience levels are upgrading their skill sets to stay relevant in the competitive environment. Here, we present a collection of the most frequently asked questions in the field of blockchain and related technologies by interviewers at various levels.

1. What do you know about blockchain? What is the difference between two prominent blockchain approaches – Bitcoin and Ethereum?

Answer: Blockchain is a decentralized distributed database of immutable records. The technology was discovered with the invention of Bitcoins. The table below gives a snapshot of the two prominent blockchain approaches.



Parameter	Bitcoin	Ethereum
Concept	Digital currency	Smart contracts
Founder	Satoshi Nakamoto	Vitalik Buterin
Release method	Genesis block mined	Presale
Cryptocurrency used	Bitcoin (Satoshi)	Ether
Algorithm	SHA-256	Ethash
Blocks time	10 minutes	12–14 seconds
Scalable	Not yet	Yes

2. What is the core idea on which blockchain technology is based on?

Answer: It enables information to be stored in a distributed fashion among users, with tamper-proofing.

3. What are the different types of blockchains?

Answer: Blockchains are of three types – public, private, and consortium

4. Why is blockchain considered as a trusted approach?

Answer: Blockchain is considered to be trusted for the following reasons:

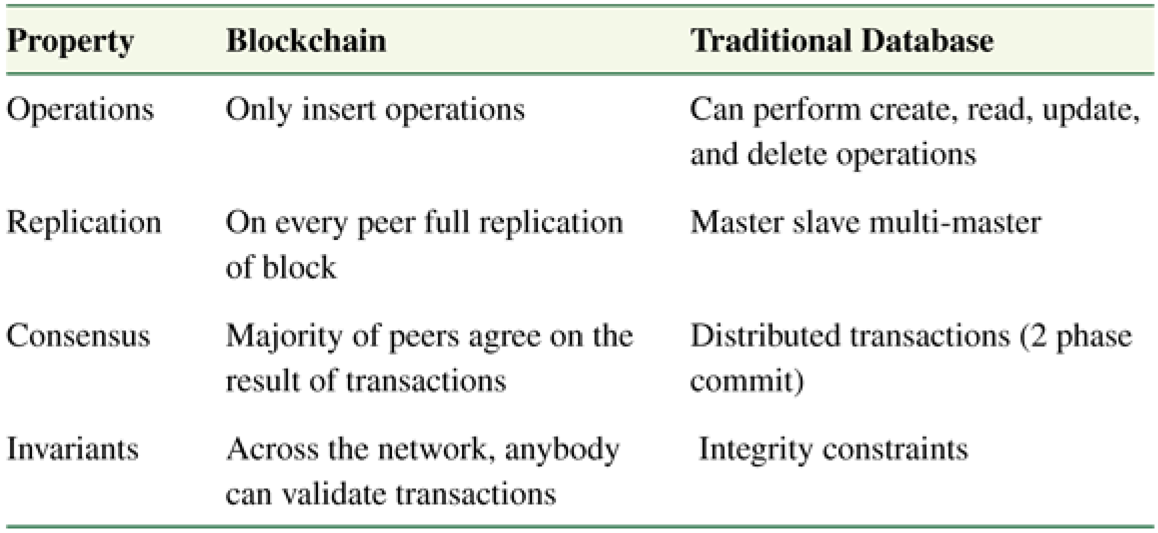
- It is compatibility with other business applications due to its open-source nature.
- It is secure due to the strong cryptography being used to implement it.

5. What are the two types of records present in blockchain?

Answer: The records that are present in blockchain database are block records and transactional records. Both these records can be accessed easily, and it is possible to integrate them with each other without following any complex algorithms.

6. Blockchain is a distributed database. How does it differ from traditional databases?

Answer:



Property	Blockchain	Traditional Database
Operations	Only insert operations	Can perform create, read, update, and delete operations
Replication	On every peer full replication of block	Master slave multi-master
Consensus	Majority of peers agree on the result of transactions	Distributed transactions (2 phase commit)
Invariants	Across the network, anybody can validate transactions	Integrity constraints

7. What are the properties of blockchain?

Answer: There are four key features of blockchain:

- **(a)** Decentralized systems
- **(b)** Distributed ledger
- **(c)** Secure ecosystem
- **(d)** Immutable

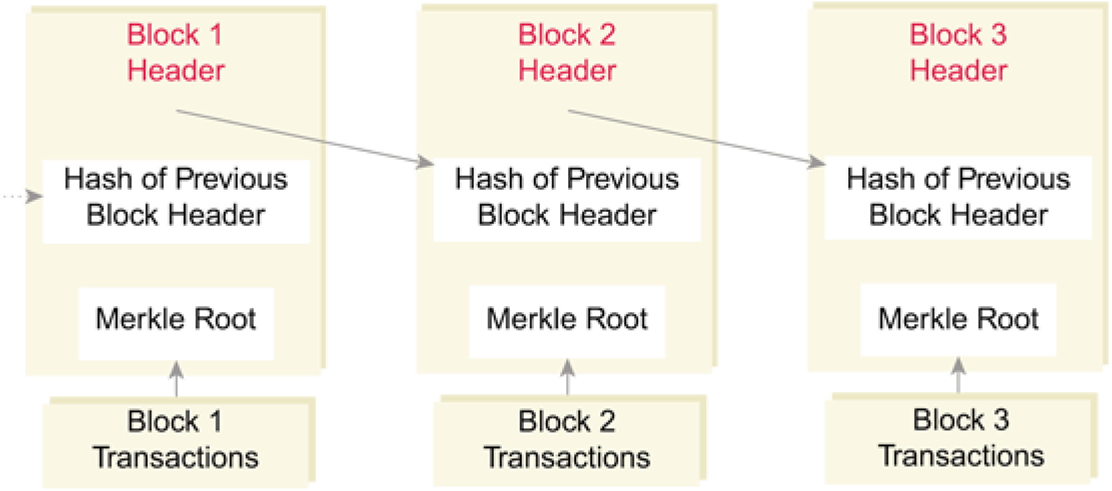
8. What is an encryption? What is its role in blockchain?

Answer: In the today's world, data security matters immensely. Encryption is one of the technical approaches that help organizations keep their data secure.

While carrying out the encryption, the plain data is encoded or changed up to some extent before it is sent out of a network by the sender. Only authorized parties can access this information. In blockchain, this approach is useful because it adds to the overall security and authenticity of blocks and helps to keep them secure.

9. What are blocks in the blockchain technology?

Answer: Blockchain consists of a list of records. These records are stored in blocks, which is in turn linked with other blocks and hence constitutes a chain called blockchain.



10. How is a block recognized in the blockchain?

Answer: Every block in the online ledger basically consists of a hash pointer that acts as a link to the block, which is prior to it containing the transaction data.

11. After writing the data, is it possible to modify it?

Answer: No, it is not possible to modify the data once it is written in a block.

12. What are block identifiers in blockchain?

Answer: Blocks in blockchain can be identified by the block header hash and the block height.

13. Is blockchain different from banking ledgers?

Answer: Accounting systems in banks use ledgers to track transactions, while blockchain is a completely decentralized distributed ledger. This implies that people do not have to rely on or trust the central bank to keep track of the transactions. The peer-to-peer network technology in blockchain can keep track of all the transactions without fear of losing them. Further, since blockchain is open-source in nature, it is more versatile and programmable than central banking ledgers. If users need new functionality on a blockchain, they can simply innovate and program on top of already existing software platforms through consensus mechanism. However, this is difficult for central banks because of all of their regulations and one point of failure.

14. Explain what you know about the security of a block.

Answer: A block or the entire blockchain is protected by a strong cryptographic public key and hash algorithm. Each block has a unique hash pointer. Any modification in the block data will influence the change in the hash identifier of the block, which in turn offers good security.

15. What is a Merkle tree? How important is the Merkle tree in blockchain?

Answer: Merkle tree also known as *hash tree*. It is a data structure in cryptography in which each leaf node is a hash of a block of data, and each non-leaf node is a hash of its child nodes.

The advantage of using the Merkle tree in blockchain is that instead of downloading every transaction and every block, it facilitates so that a “light client” can only download the chain of block headers.

Further, if someone needs to verify the existence of a specific transaction in a block, then he does not have to download the entire blockchain. Rather, downloading a set of a branch of this tree which contains this transaction is sufficient in order to verify the transaction.

16. Does blockchain qualify to be an incorruptible ledger?

Answer: Blockchain is considered incorruptible. Any ill-intentioned individual acting alone is powerless. To take over the network, an attacker would have to control more than 50% of the total computing power of the blockchain. We should note that the energy required to power the computers for blockchain system to work will be very high in this scenario and thus infeasible.

17. What are the common types of ledgers that can be considered in blockchain?

Answer: The common types of ledgers are public, private, and hybrid.

18. How is a blockchain ledger different from an ordinary ledger?

Answer: A blockchain ledger is a digital ledger that can be decentralized very easily. The probability of error to occur in this ledger is far less than that in an ordinary ledger. An ordinary ledger is manually prepared by human efforts, whereas blockchain performs almost all its tasks automatically.

19. Which type of records can be stored on a blockchain?

Answer: Industries are using blockchain for securing all types of records, including the following:

- **(a)** Records of medical transactions
- **(b)** Identity management
- **(c)** Transaction processing
- **(d)** Business transactions
- **(e)** Management activities
- **(f)** Documentation

20. A distributed digital ledger is the core idea for recording transaction in blockchain. What does the system rely on?

Answer: The system relies on the network service protocol and the connected nodes of the network.

21. In a typical business, what are the core requirements to use blockchain?

Answer: Shared ledger, smart contract functionality, privacy and trust are the core business blockchain requirements.

22. What is a 51% attack?

Answer: This attack refers to a situation where a group of miners that hold more than 50% of the Network Hash Rate implied that they could manipulate the new transactions or stop the transactions to proceed or that they are able to reverse the transactions that were recently confirmed. In a nutshell, it is a type of doing double spending.

23. Name some popular platforms for developing blockchain applications.

Answer: After the introduction of Bitcoin, several blockchain platforms started coming up. Ethereum came right after the evolution of Bitcoin. It is one of the popular public platforms for building blockchain-based applications. Further, there is a Hyperledger community for building enterprise-based solutions. Qtum, IOTA, and EOS are some of the widely used platforms for building blockchain.

24. What is double spending? Is it possible to double spend in a Bitcoin blockchain system?

Answer: It is a situation where one digital token is spent multiple times because the token generally consists of a digital file that can easily be cloned, which leads to inflation. Confirming a transaction by multiple parties before the actual transaction is written to the ledger, the blockchain prevents double spending. It is no exaggeration to say that the entirety of Bitcoin's system of blockchain, mining, proof-of-work, etc., exist to produce this history of transactions that is computationally impractical to modify.

25. In which organization can one use blockchain technology?

Answer: There are no strict guidelines defined for the category of business which can consider the use of blockchain. Big financial institutions, corporations, private businesses, government departments, and even defence organizations can trust this technology.

26. What are the key principles in that need to be followed in blockchain, which are helpful in eliminating security threats?

Answer: There are a few principles that need to be followed with respect to time. They are as follows:

(a) Auditing

(b) Securing applications

(c) Secure testing and similar approaches

(d) Database security

(e) Continuity planning

(f) Digital workforce training

27. What are some of the popular consensus algorithms?

Answer: Some of the popular consensus algorithms are:

(a) Practical Byzantine Fault Tolerance

(b) Proof-of-work

(c) Proof-of-stake

(d) Delegated proof-of-stake

(e) Proof-of-elapsed time

Multiple-Choice Questions and Answers

D ANNEXURE

1. A blockchain consists of which of the following?
   - **(a)** A hash pointer to the previous block
   - **(b)** Timestamp
   - **(c)** List of transactions
   - **(d)** All of the above

2. Out of the following, which is the first distributed blockchain implementation?
   - **(a)** Bitcoin
   - **(b)** Ethereum
   - **(c)** Hyperledger
   - **(d)** None of the above

3. Bitcoin is based on which type of blockchain?
   - **(a)** Private
   - **(b)** Public
   - **(c)** Public permissioned
   - **(d)** Permissioned

4. A blockchain can be stored as which of the following?
   - **(a)** A flat file

(b) A database

(c) Both (a) and (b)

(d) None of the above

5. In blockchain, blocks are linked to which of the following?

(a) Backward to previous block

(b) Forward to next block

(c) Not linked with each other

(d) Forward to previous block

6. The primary benefit of immutability is ________________.

(a) Scalability

(b) Improved security

(c) Tamper-proof

(d) Increased efficiency

7. Blockchain is generated using which of the following hash algorithm?

(a) SHA128

(b) SHA256

(c) SHA652

(d) SHA821

8. A block in the blockchain can have how many parent blocks?

(a) More than one parent block

(b) Only one parent block

(c) Either (a) or (b)

(d) Only two parent blocks

9. Forks in a blockchain can result in which of the following?

(a) Multiple parent blocks

- **(b)** Multiple children blocks
- **(c)** Either (a) or (b)
- **(d)** None of the above

10. Which of the following is asymmetric encryption algorithm?

- **(a)** Blowfish
- **(b)** Twofish
- **(c)** RSA
- **(d)** Triple DEA

11. Which of the following are properties of blockchain?

- **(a)** Distributed ledger
- **(b)** Integrity and safety
- **(c)** Decentralized system
- **(d)** All of the above

12. A block in blockchain is pointed using which of the following?

- **(a)** Hash pointer
- **(b)** User ID
- **(c)** Transaction ID
- **(d)** Timestamp

13. The property of consistency is preserved in blockchain by maintaining which of the following?

- **(a)** Global copy of the total Information
- **(b)** Local copy of the global information
- **(c)** Global list of transactions
- **(d)** None of the above

14. What is the other name for authentication codes for messages?

(a) Keyed hash function

(b) Hash code

(c) Message hash function

(d) None of the above

15. What is the current cryptographic hash algorithm used in Bitcoin?

(a) RSA

(b) MD5

(c) Double SHA128

(d) Double SHA256

16. Which of the following is used to ensure consensus in Bitcoin framework?

(a) Proof-of-work

(b) Proof-of-stake

(c) Proof-of-concept

(d) None of the above

17. After padding, what is the length of a message in SHA512?

(a) 992

(b) 960

(c) 768

(d) 896

18. In an RSA algorithm the difficulty lies on which of the following?

(a) Discrete logarithm problem

(b) Integer factorization problem

(c) Traveling salesman problem

(d) Matrix inversion problem

19. In public key cryptosystem, the message digest is signed by which of the following?

(a) Sender's public key

(b) Sender's private key

(c) Receiver's public key

(d) Receiver's private key

20. Consider a system with three users X, Y, and Z. The available Bitcoins for each user are 30, 20, and 30, respectively. Which of the following is an example of double spending?

(a) X->Y:30, Z->Y:30

(b) X->Y:30, Z->X:30

(c) X->Y:30, X->Z:30

(d) All of the above

21. A user can access Bitcoin wallet using which of the following?

(a) His/her email address

(b) His/her public key based address

(c) His/her private key based address

(d) His/her public and private key based address

22. What is the digital signature algorithm used in Bitcoin?

(a) Elliptic Curve Digital Signature Algorithm

(b) Digital Signature Algorithm

(c) RSA algorithm

(d) All of the above

23. Suppose a user X wants to transfer 20 bitcoins to his friend Y through Bitcoin wallet. What are the essential details that X must send to Y to validate the transfer?

(a) Transaction, X’s signature, X’s private key

(b) Transaction, X’s signature, Y’s public key

(c) Transaction, X’s signature, X’s public key

(d) Transaction, Y’s public key, X’s public key

24. Computing resources are highly essential for which of the following?

(a) Mining

(b) Block creation

(c) Scripting

(d) Coding

25. In distributed consensus, all the correct individuals either reach a value or null. What is this property called?

(a) Termination

(b) Validity

(c) Integrity

(d) Agreement

26. What is the distributed consensus mechanism that works in an open Internet grade computing system?

(a) Shared memory

(b) Message passing

(c) PBFT

(d) None of the above

27. While using the traditional distributed consensus algorithms to blockchain-based systems over the Internet, which of the following challenges make the application difficult?

(a) Large number of users

(b) Global accessibility of the Internet

(c) Asynchronous nature of the Internet

(d) All of the above

28. Which of the following variables among the components of the block header is computed to achieve the difficulty posed by the blockchain network?

(a) Merkle tree root

(b) Nonce

(c) Timestamp

(d) Previous block hash

29. Which of the following consensus algorithms is considered for virtual resources or digital coins for participating in mining activity?

(a) Proof-of-stake

(b) Raft consensus

(c) Proof-of-burn

(d) Proof-of-work

30. Which of the following strategies is used by Bitcoin network to control the inflow of bitcoins?

(a) Increasing the nonce

(b) Lower the mining reward

(c) Increasing the block size

(d) All of the above

31. In a permissioned blockchain, which of the following will serve as the primary assumption?

(a) Closed network

(b) Chosen miners

(c) No malicious miners

(d) All of the above

32. The consensus protocol that supports message authentication is __________________.

(a) PAXOS

(b) RAFT

(c) Byzantine General model

(d) Practical Byzantine General model

33. The data structure used for storing recent transaction output is __________________.

(a) Blockchain

(b) Merkle tree

(c) World state

(d) Wallet

34. Which of the following allows modification of data element?

(a) Blockchain

(b) World state

(c) Both (a) and (b)

(d) None of the above

35. Which of the following components of Hyperledger v1 architecture check and invalidate conflicting transactions?

(a) External-CA

(b) Endorser

(c) Ordering services

(d) Committers

36. After the delivery of a block by the ordering service, which step is followed to accept transactions into the blockchain?

(a) Validation

(b) Endorsement

(c) Proposal

(d) Notification

37. Which of the following helps in giving multitenancy by providing privacy to individual ledgers?

(a) Ordering services

(b) Peers

(c) Channels

(d) Endorsement policies

38. In a blockchain solution, which of the following roles is the prime focus for the development of Hyperledger Composer?

(a) Network Service Provider

(b) Business Service Provider

(c) Network Service Consumer

(d) None of the above

39. Which of the following data formats is used by the Hyperledger Composer to serialize data?

(a) Extensible Markup Language (XML)

(b) JavaScript Object Notation (JSON)

(c) Extensible Stylesheet Language (XSL)

(d) None of the above

40. In which way does the backend Node.js application connect to the peers in Hyperledger Fabric?

(a) Google RPC connection

(b) Http connection

(c) WebSocket connection

(d) Any of the above

41. Who (in terms of roles) defines the legitimate participants in the business network in Hyperledger Composer?

(a) Business service consumer

(b) Business service provider

(c) End user

(d) Administrator

42. The relationship between assets and participants in Hyperledger Composer is defined in which of the following?

(a) Access control file

(b) Transaction file

(c) Model file

(d) Query file

43. Business network card in Hyperledger Composer consists of __________________.

(a) Transaction

(b) User's private key

(c) Smart Contract logic

(d) All of the above

44. Where are the mapping details of the participants and Hyperledger Fabric identity stored?

(a) Participant registry

(b) Identity registry

(c) Application

(d) An external data store

45. Your organization decides to work together with other similar organizations in a blockchain network. The suitable network for your organization is ________________.

(a) Consortium based network

(b) Founder directed network

(c) Community based network

(d) Privately funded network

ANSWER KEY

1. **(d)** All of the above

2. **(a)** Bitcoin

3. **(b)** Public

4. **(c)** Both (a) and (b)

5. **(a)** Backward to the previous block

6. **(d)** Increased efficiency

7. **(b)** SHA256

8. **(a)** More than one parent block

9. **(b)** Multiple children blocks

10. **(c)** RSA

11. **(d)** All of the above

12. **(a)** Hash Pointer

13. **(b)** Local copy of the global information

14. **(a)** Keyed Hash Function

15. **(d)** Double SHA256

16. **(a)** Proof-of-work

17. **(d)** 896

18. **(b)** Integer Factorization Problem

19. **(b)** Sender's private key

20. **(c)** X->Y:30,X->Z:30

21. **(b)** His/her public key based address

22. **(a)** Elliptic Curve Digital Signature Algorithm

23. **(c)** Transaction, X's signature, X's public key

24. **(a)** Mining

25. **(c)** Integrity

26. **(c)** PBFT

27. **(c)** Asynchronous nature of the Internet

28. **(b)** Nonce

29. **(c)** Proof-of-burn

30. **(b)** Lower the mining reward

31. **(a)** Closed network

32. **(d)** Practical Byzantine General model

33. **(c)** World state

34. **(b)** World state

35. **(d)** Committers

36. **(a)** Validation

37. **(c)** Channels

38. **(b)** Business Service Provider

39. **(b)** JavaScript Object Notation (JSON)

40. **(a)** Google RPC connection

41. **(a)** Business Service Consumer

42. **(c)** Model file

43. **(b)** User's private key

44. **(b)** Identity registry

45. **(a)** Consortium based network

BIBLIOGRAPHY

Adrian, D., Bhargavan, K., Durumeric, Z., Gaudry, P., Green, M., Halderman, A. J. Heninger, N., Springall, D., Thomé, E., Valenta, L., VanderSloot, B., Wustrow, E., Zanella-Béguelin, S., Zimmermann, P. 2015. Imperfect forward secrecy: How Diffie-Hellman fails in practice. CCS'15, 12–16 October 2015, Denver. [DOI: http://dx.doi.org/10.1145/2810103.2813707]

Back, A. 1997. A partial hash collision based postage scheme. [Available at http://www.hashcash.org/papers/announce.txt]

Bertoni, G., Daemen, J., Peeters, M., and Van Assche, G. 2011. Cryptographic Sponge Functions. [Available at http://sponge.noekeon.org/CSF-0.1.pdf]

Birkyukov, A., Shamir, A., and Wagner, D. 2000. Real time cryptanalysis on A5/1 on a PC. At the Fast Software Encryption Workshop, 10–12 April 2000, New York City. [Available at https://cryptome.org/a51-bsw.htm]

Bressoud, D., and Wagon, S. 2000. A Course in Computational Number Theory. Springer: Berlin.

Bunz, B., Agrawal, S., Zamani, M., and Boneh. D. 2018. Zether: towards privacy in a smart contract world. [Available at https://crypto.stanford.edu/~buenz/papers/zether.pdf]

Chandra T. D., Griesemer, R., and Redstone, J. 2007. Paxos made live: An engineering perspective. In Proc. 26th ACM Symposium on Principles of Distributed Computing (PODC), pp. 398–407.

Chaum, D. 1983. Blind signatures for untraceable payments. Advances in Cryptology: Proceedings of Crypto, 82(3): 199–203.

Chaum, D., and Brands, S. 1997. "Minting" electronic cash. IEEE Spectrum [Special issue on electronic money]

Decker, C. and Wattenhofer, R. 2013. Information propagation in the bitcoin network. In 13th IEEE International Conference on Peer-to-Peer Computing, pp. 1–10. IEEE.

den Boer, B., and Bosselaers, A. 1993. Collisions for the compression function of MD5. In Advances in Cryptology – EUROCRYPT 1993, pp. 293–304. Springer: Berlin.

Dobbertin, H. 1996. Cryptanalysis of MD5 compress. In Advances in Cryptology – EUROCRYPT 1996. Springer: Berlin.

Dwork, C., and Naor, M. 1993. Pricing via Processing or Combatting Junk Mail. In Advances in Cryptology — CRYPTO' 92, E. F. Brickell (ed). CRYPTO 1992. Lecture Notes in Computer Science, vol 740. Springer: Berlin.

Finck, M. 2019. Blockchain Regulation and Governance in Europe. Cambridge University Press: Cambridge.

Kiayias, A., Russell, A., David, B., et al. 2017. PPCoin: Peer-to-Peer Cryptocurrency with Proof-of-Stake. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security – CCS'16. 1919: 1–27. [DOI:10.1017/ CBO 9781107415324.004.]

Klima, V. 2006. Tunnels in Hash Functions: MD5 collisions within a minute. [Available at http://eprint.iacr.org/2006/105]

Kleinjung, T., Aoki, K., Franke, J., Lenstra, A. K., Thomé, E., Bos W. J., Gaudry, P., Kruppa, A., Montgomery, P. L., Osvik, D. A., te Riele, H., Timofeev, A., and Zimmermann, P. 2010. Factorization of a 768-bit RSA modulus., version 1.4. [Available at https://eprint.iacr.org/2010/006.pdf]

Lamport, L. 1998. The part-time parliament. ACM Transactions on Computer Systems, 16(2):133–169.

Lamport, L. 2001. Paxos made simple. SIGACT News, 32(4):51–58.

Lampson, B. 2001. The ABCD's of Paxos. In Proc. 20th ACM Symposium on Principles of Distributed Computing (PODC).

Leurent, G. 2007. Message freedom in MD4 and MD5 collisions: Application to APOP. In Proceedings of FSE 2007. [Lecture Notes in Computer Science 4715]. Springer: Berlin.

Liskov, B. 2010. From viewstamped replication to Byzantine fault tolerance. In B. Charron-Bost, F. Pedone, and A. Schiper (eds). [Lecture Notes in Computer Science, pp. 121–149.] Springer: Berlin.

Liskov, B., and Cowling, J. 2012. Viewstamped replication revisited. [MIT-CSAIL-TR-2012–021.]

Luu, L., Velner, Y., Teutsch, J., and Saxena, P. 2017. Smart pool: Practical decentralized pooled mining. IACR Cryptology ePrint Archive, 2017: 19.

Merkle, R. C. 1998. A digital signature based on a conventional encryption function. In Advances in Cryptology – CRYPTO'87. [DOI: 10.1007/3-540-48184-2_32]

Myers, J. and Rose, M. 1996. Post office protocol – Version 3. [STD 53, RFC 1939]

Nakamoto, S. 2008. Bitcoin: A peer-to-peer electronic cash system. [Available at https://bitcoin.org/bitcoin.pdf]

Nystrom, M. 2005. Identifiers and test vectors for HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512. [RFC 4231]

Oki, B. M., and Liskov, B. 1998. Viewstamped replication: A new primary copy method to support highly available distributed systems. In Proceedings of 7th ACM Symposium on Principles of Distributed Computing (PODC), pp. 8–17.

Popper, N. 2016. Hacker may have taken $50 million from cybercurrency project. The New York Times. [Retrieved 23 October 19 from https://web.archive.org/web/20170620012726/https://www.nytimes.com/2016/06/18/business/dealbook/hacker-may-have-removed-more-than-50-million-from-experimental-cybercurrency-project.html]

Price, R. 2016. Digital currency Ethereum is cratering amid claims of a $50 million hack. Business Insider. [Retrieved 23 October 19 from https://web.archive.org/web/20170611195628/http://uk.businessinsider.com/dao-hacked-ethereum-crashing-in-value-tens-of-millions-allegedly-stolen-2016–6]

Sari, L., and Sipos, M. 2109. FileTribe: Blockchain-based Secure File Sharing on IPFS. In European Wireless 2019; 25th European Wireless Conference, pp. 1–6.

Shannon, C. 1949. Communication theory of secrecy system. Bell Labs Technical Journal 28: 656–715.

Sompolinsky, Y., and Zohar, A. 2013. Accelerating bitcoin's transaction processing. fast money grows on trees, not chains. IACR Cryptology ePrint Archive, 2013: 881.

Song, J. H., Poovendran, R., Lee, J., and Iwata, T. 2006. The AES-CMAC algorithm. [RFC 4493]

Stevens, M. 2006. On collisions for MD5. [Master's Thesis, Eindhoven University of Technology. Available at http://www.win.tue.nl/hashclash/On%20Collisions%20for%20MD5%20-%20M.M.J.%20Stevens.pdf]

Sumartono, I., Siahaan, A. P. U., and Mayasari, N. 2016. An overview of the RC4 algorithm. IOSR Journal of Computer Engineering 18(6): 67–73.

Szabo, N. 2013. Bit Gold. Unenumerated. Blogspot. [Archived from the original on 22 September 2011. Retrieved 5 December 2013]

Wang, X., and Yu, H. 2005. How to break MD5 and other hash functions. In Advances in Cryptology – EUROCRYPT 2005. Springer: Berlin.

Wang, X., Feng, D., Lai, X., Yu, H. 2004. Collisions for hash functions MD4, MD5, HAVAL-128 and RIPEMD. [Available at http://eprint.iacr.org/2004/199.pdf]

Zamfir, V. 2015. Introducing Casper "the Friendly Ghost". [Available at https://blog.ethereum.org/2015/08/01/introducing-casper-friendly-ghost]

Zhang, F., Eyal, I., Escriva, R., Juels, A. and Renesse, R. V. 2017. REM: Resource-efficient mining for blockchains. In 26th USENIX Security, Symposium (USENIX Security 17), pp. 1427–1444. USENIX Association.

GLOSSARY OF TERMS

Bitcoin A type of digital currency in which a record of transactions is maintained and new units of currency are generated by the computational solution of mathematical problems and which operates independent from a central bank.

Blockchain network A chain of blocks that are linked using cryptography and typically managed by a peer-to-peer network, collectively adhering to a protocol for inter-node communication and validating new blocks.

Chaincode A computer program written in language Go, node.js, or Java that implements a prescribed interface. It runs in a secured Docker container isolated from the endorsing peer process by initializing and managing ledger state through transactions submitted by applications.

Consensus A general agreement and group discussion where everyone's opinions are heard and understood, and a solution is created that respects those opinions. Consensus is not what everyone agrees to, nor is it the preference of the majority. It results in the best solution that the group can achieve at the time.

Content-based addressing Used to address content using names, typically consisting of a provider name, content name, and version.

Cryptographic token Programmable assets or access rights that are managed by a smart contract and an underlying distributed ledger.

Cryptography A process of converting ordinary plain text into unintelligible text and vice-versa. It is a method of storing and transmitting data in a particular form so that only those for whom it is intended can read and process it.

Decentralized application A computer application that runs on a distributed computing system.

Decentralized system A system in which lower-level components operate on local information to accomplish global goals by cumulating their capabilities.

Digital signature A mathematical technique used to validate the authenticity and integrity of a message, software or digital document.

Ethereum An open source, public, blockchain-based distributed computing platform and operating system featuring smart contract functionality.

Hash function A function used to map data of arbitrary size to fixed-size values.

Internet of Things An interconnection of interrelated computing devices via the Internet, embedded in everyday objects, enabling them to send and receive data without requiring human-to-human or human-to-computer interaction.

Proof-of-stake A consensus algorithm by which a cryptocurrency blockchain network aims to achieve distributed consensus.

Proof-of-work A method that requires a not-insignificant but feasible amount of effort in order to deter frivolous or malicious uses of computing power, such as sending spam emails or launching denial of service attacks.

Smart contract A self-executing contract with the terms of agreement between buyer and seller being directly written into lines of code.

INDEX

Career Options in Blockchain

- Developer
- Crypto Journalists
- Accountant
- Analyst
- Public Relations
- Marketers
- Crypto Brokers
- Consultants
- Legal Advisors
- ICO Advisors

Confidentiality
Integrity
Cryptography Primitives
Authentication
Nonrepudiation

Cryptography Primitives

Key Generation

Select p, q	p and q both prime
Calculate $n = p{\times}q$	
Calculate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\phi(n), e) = 1; 1 < e < \phi(n)$
Calculate d	$d = e^{-1} \bmod \phi(n)$
Public key	$KU = \{e, n\}$
Private key	$KR = \{d, n\}$

Encryption

Plaintext	$M < n$
Ciphertext	$C = M^e \pmod{n}$

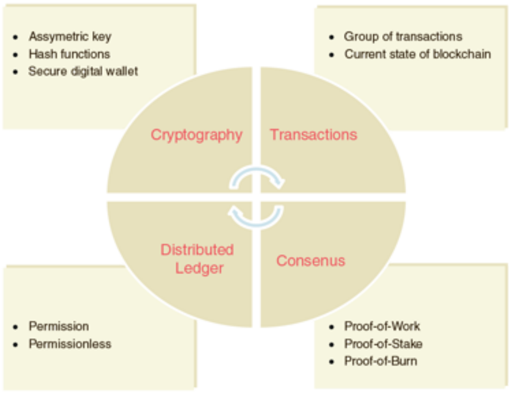
Decryption

Ciphertext	C
Plaintext	$M = C^d \pmod{n}$

RSA Algorithm

Global blockchain devices market generated a revenue of $300 million and is projected to reach $23.5 billion by 2030, advancing at a 48.7% (CAGR Report)

Blockchain will generate an annual business value of more than US $3 trillion by 2030. (Gartner)



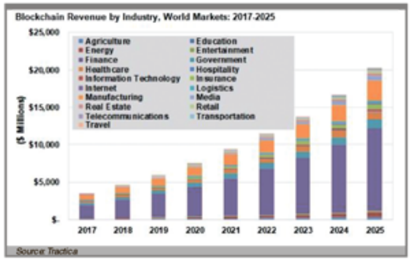
- Assymetric key
- Hash functions
- Secure digital wallet

- Group of transactions
- Current state of blockchain

- Permission
- Permissionless

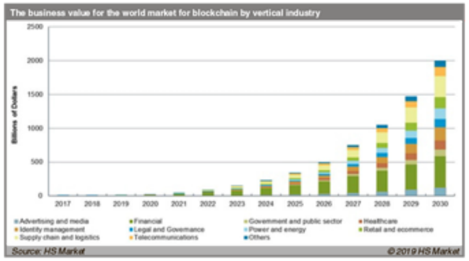
- Proof-of-Work
- Proof-of-Stake
- Proof-of-Burn

Components of Blockchain



Blockchain Revenue by Industry, World Markets: 2017-2025

Source: Tractica



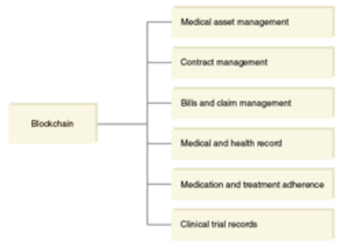
The business value for the world market for blockchain by vertical industry

Source: HS Market

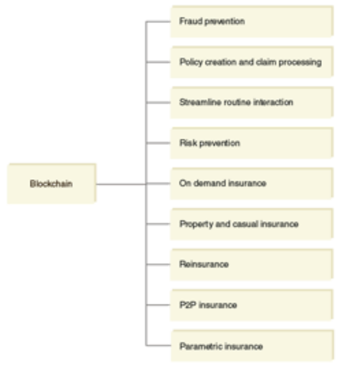
© 2019 HS Market

Blockchain Technology Market to Reach US $21.07 billion by 2025 (Fortune Business Insights)

$11.7 billion projected Investments in Blockchain Solutions by 2022 (International Data Corporation)


Blockchain
Medical asset management
Contract management
Bills and claim management
Medical and health record
Medication and treatment adherence
Clinical trial records

Blockchain in Healthcare



Blockchain in Insurance

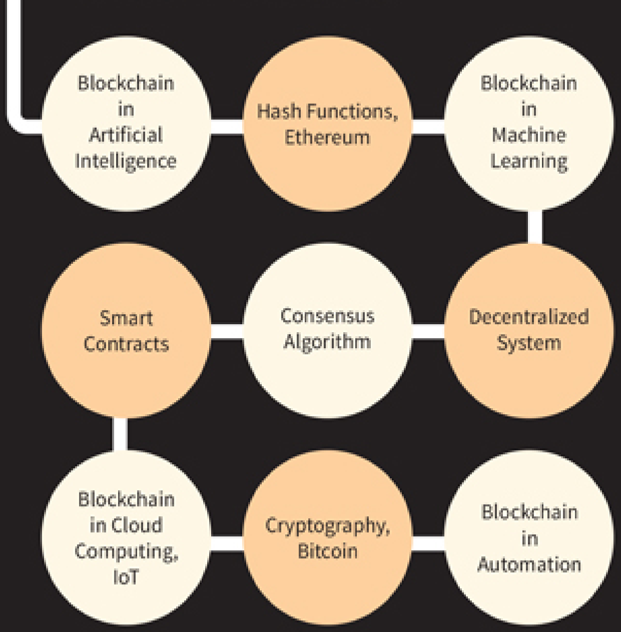
- Payments
- Smart Contracts
- Distributed Storage
- Digital Identity
- Notary
- IoT Systems
- Track Responsibility
- Gift Vouchers
- Supply Chain
- Automation
- Trade Policies
- Audit

Blockchain Business Applications

Blockchain is a digital, decentralized, distributed public ledger database where blocks are linked cryptographically, and transactions are digitally signed and managed using consensus model. It is one of the fastest growing skills in the IT sector as it provides safe and secured online transactions. Many organizations have started adopting blockchain as it offers several benefits to the industry.

Blockchain Technology: Concepts and Applications not only covers the core concepts of blockchain technology but also goes into details of all the concepts required for its understanding. It discusses the features and benefits of blockchain via examples at distinct situations. It imparts knowledge regarding blockchain and defines new ways to manage IT resources with mutual trust, immutability, decentralization, and auditability. This book will be useful not only to beginners but also to academicians, developers, system architects, solution architects, and technical managers of blockchain as it gives the specific solutions to real-world problems.

SALIENT FEATURES



Blockchain in Artificial Intelligence

Hash Functions, Ethereum

Blockchain in Machine Learning

Decentralized System

Consensus Algorithm

Smart Contracts

Blockchain in Cloud Computing, IoT

Cryptography, Bitcoin

Blockchain in Automation

TECHNICAL STUFF addresses technical practices

FLASH QUIZ tests your level of understanding

FACT ALERT points out an important truth

QUICK TIP gives practical advice

ACTIVITY essential exercise to carry out

Instructor Resources available at https://www.wileyindia.com

Code to Access Videos

follow us on
facebook.com/wileyindia twitter.com/wileyindiapl linkedin.com/in/wileyindia

READER LEVEL
Beginner to Advanced

SHELVING CATEGORY
Engineering

WILEY

Wiley India Pvt. Ltd.
Customer Care +91 120 6291100
csupport@wiley.com
www.wileyindia.com
www.wiley.com

eISBN: 978-81-265-9012-4
9788126590124