2.1 WHAT IS BITCOIN

- Bitcoin is the first and most widely recognized form of decentralized digital currency. Bitcoin is a cryptocurrency that can be used to make online transactions and send it quickly to anyone on the platform, without any need of a middle man.
- A digital money ecosystem is built on the ideas and technologies contained in Bitcoin. Value is stored and transferred among participants in the bitcoin network using bitcoin units.
- The internet and other transport networks can also be used by Bitcoin users to communicate with one another using the bitcoin protocol.
- Open-source software that runs on a wide range of computing platforms, including smartphones and laptops, makes the bitcoin protocol stack accessible. Almost anything that can be done with conventional currencies, such as buying and selling goods, sending money to individuals or organizations, or providing credit, can be done by users who transfer bitcoin over the network. At specialized currency exchanges, Bitcoin can be purchased, sold, and exchanged for other currencies. Due to its speed, safety, and borderlessness, Bitcoin is, in a sense, the ideal form of internet currency.
- Bitcoin is a peer-to-peer, distributed system. As a result, there is neither a "central" server nor a control point.
- Bitcoins are created through a process known as "mining," in which participants compete to solve a mathematical problem while bitcoin transactions are processed.
- A miner is a member of the bitcoin network who uses their computer's processing power to verify and record transactions. This includes anyone with a device that runs the entire bitcoin protocol stack.
- A bitcoin miner is rewarded with brand-new bitcoin on average every 10 minutes if they are successful in validating the previous 10 minutes' transactions.
- In essence, bitcoin mining eliminates the need for any central bank and decentralizes the issuance and clearing of currency. Actually, this invention has only been put to use in the form of the bitcoin currency. Bitcoin is the result of decades of research into distributed systems and cryptography.
- Bitcoin comprises of :
  (1) A public transaction ledger (the blockchain)
  (2) A decentralized peer-to-peer network (the bitcoin protocol)
  (3) A set of rules for independent transaction validation and currency issuance (consensus rules)
  (4) A mechanism for reaching global decentralized consensus on the valid blockchain (Proof-of-Work algorithm)

- A blockchain is a public ledger of all the transactions. Miners are rewarded in cryptocurrency and can buy/sell on exchanges.
- Bitcoin was created in 2009 by an anonymous developer (known simply as Satoshi Nakamoto)
- Bitcoin is the most well-known cryptocurrency in the world, and it has inspired plenty of other coins.
- It may seem like a competitor might try to replace the system or use it as a security token, but stakeholders in the system have an interest in keeping it going and so they would be unlikely to do either of those things.
- Bitcoin is a cryptocurrency that is based on the blockchain technology. Users can make peer-to-peer transactions by connecting to an open and decentralized network.
- Bitcoin transactions are authenticated through a Proof-of-Work mechanism that rewards Bitcoin miners for validating them.

2.1.1 Bitcoin's Basic Features

(1) **Distributed :** Bitcoin transactions are recorded publicly i.e. in Blockchain and anyone can view that details. Bitcoin miners help the network by running this software which records all the information about a transaction.

(2) **Transparent:** New transactions are added to the Bitcoin blockchain at intervals of 10 minutes by consensus. Every transaction is verified according to the immutable, transparent protocol.

Bitcoin is the first and most popular cryptocurrency. It was created in 2009 as a response to the 2008 global financial crisis by an unknown individual who goes by the name of Satoshi Nakamoto. Bitcoin has no central authority, meaning that any transactions are verified according to the immutable and transparent protocol.

(3) **Peer-to-Peer :** When it comes to payments, blockchain does not rely on a third party. The transfers are direct so there is no room for anyone else.

The use of a third party has always been a problem in payments. Blockchain was developed to solve this issue. The transfers are direct, so there is no room for error. There is also no need for any kind of additional verification because the transactions are verified by all parties involved in the network.

With blockchain, there is less room for error and fraud which would lead to more security and reliability in payments.

(4) **Permissionless :** BTC is completely open access and ready to use for everyone - there are no complicated rules of entry. Any transactions that follow the set algorithm will be processed with certainty.

Bitcoin is a decentralized digital currency that is not regulated by any central bank and has no set rules on how transactions are processed.

Transactions are processed by a blockchain algorithm that provides certainty of processing. There's no fee to convert Bitcoin to dollars or vice versa, making it an inexpensive way to send money internationally.

(5) **Public :** All Bitcoin transactions are publicly accessible, reducing the likelihood of fraudulent activity. However, it is possible to connect certain Bitcoin addresses with individual identities. To achieve a higher level of privacy, a number of solutions have been proposed and are being explored in the context of Bitcoin's governance process.

2.1.2 History of Bitcoin

August 2008 marked the start of Bitcoin.org when its domain name was officially registered - although today, the original registrar's identity is protected by Whois Guard privacy. The domain was registered with the aim of providing information about the Bitcoin cryptocurrency and promoting its use. The website has since become a key resource for the Bitcoin community, providing news, information and resources about the Bitcoin network.

- In October 2008, someone using the pseudonym Satoshi Nakamoto posted a message on metzdowd.com, introducing what would become known as Bitcoin.
- With their paper "Bitcoin: A Peer-to-Peer Electronic Cash System," the groundwork was laid for what is now the foundation of Bitcoin's operations. This document can now be considered the Magna Carta of Bitcoin.
- The mining of the first Bitcoin block, called Block 0 or the "genesis block," occurred on Jan. 3, 2009. This block contains the text: "The Times 03/Jan/2009 Chancellor on brink of second bailout for banks," which may indicate that it was mined either on or after this date and potentially provides commentary about relevant political events of the time.
- The Bitcoin block reward is halved every 210,000 blocks. To illustrate this, when Bitcoin launched in 2009, the reward for a block discovery was 50 new coins.
- On May 11th 2020, the third 'halving' occurred which reduced the reward per block to 6.25 BTC.
- Bitcoin has the capability of divisibility up to eight decimal places (100 millionths of one Bitcoin), commonly referred to as a Satoshi.
- It may be possible for miners to accept a modification in the future, increasing the number of decimal places beyond eight.

2.2 BITCOIN TRANSACTION

- When you send or receive bitcoin, this is called a transaction. To send a bitcoin, you input the receiver's address into your wallet application, then enter your private key and agree to the transaction fee. After that, press the button that says "send."

- The receiver must then wait for the transaction to be verified by the mining network, which can take up to 30 minutes as transactions are queued in a mining pool called the mempool.
- The mempool is where transactions waiting to be verified are stored. The network typically confirms a block of transactions every ten minutes; however, not all new transactions are included in the newly created block.
- This is because blocks can only hold a limited amount of data, and each transaction comes with a mining fee. Consequently, not all new transactions are guaranteed to be included in the next block.
- To be processed, transactions must meet the minimum fee threshold. The transactions with the highest fees are processed first. Thus, the issue of increasing fees arises. With Bitcoin's popularity, demand for transactions has also grown, giving miners the opportunity to charge higher fees.
- When the fee is paid, the transaction is added to a block. Miners then validate the transaction information in the block and close it.
- All receivers can then collect their bitcoin. Afterward, both wallets display the updated balance and the next transactions are processed.

2.3 BITCOIN CONCEPTS

When it comes to addresses, keys, and wallets, there are a few key differences that are important to understand. Let's break down each one so that we can have a better understanding of each.

A bitcoin is simply data with ownership assigned. When transactions are made, the ownership of that data is transferred in a similar way to when you use your debit card to make an online purchase. You use a digital wallet, usually in the form of a mobile application, to send or receive bitcoin.

2.3.1 Key

- Bitcoin is transferred to an owner through a blockchain transaction. The owner then receives a number, which is their private key.
- You have a public address, called your public key, that is used when someone sends you bitcoin. It's similar to the way they would enter your email address when sending you an email.
- There are two types of keys in cryptocurrency: public and private. Public keys are similar to account numbers and can be shared with anyone.

- Private keys, as the name suggests, should be kept confidential. They work like a PIN or verification code that, together with the corresponding public key, provides access to funds on the blockchain.
- You should always keep your private key(s) safe and secure. The best way to do this is to store them in a paper wallet or hardware wallet.
- Remember, your keys are not stored on a blockchain. Instead, they are saved in an encrypted file. This file can be saved anywhere, offline.

2.3.2 Address

- An address is a randomly generated set of numbers and letters which represents a type of unique number similar to a bank account number. For example, the Bitcoin genesis address - the first Bitcoin address ever is 1A1zP1eP5QGefi2DMPTfTL5SLmv7DivfNa.
- The difference is that an address can be created for free by anyone and within a matter of seconds without needing a third party. You can create as many public addresses as you like or need.
- You can freely share your public address with others. That way, people can send cryptocurrencies to your address.

2.3.3 Wallets

- A wallet is a piece of software that allows you to send and receive bitcoins as well as monitor your balance. The wallet searches the blockchain network for your bitcoin on your behalf.
- Bitcoin is stored in segments on the blockchain, which functions as a ledger. Since bitcoin is made up of data inputs and outputs, it is dispersed across the blockchain in fragments from prior transactions. Your wallet app locates them all, adds up the value, and then shows it.
- There are two main types of wallets: custodial and noncustodial. A custodial wallet is one where a trusted entity, like an exchange, holds your keys for you. For example, when you sign up for a Coinbase exchange account, you can elect to have them store your keys for you as custodians.
- Noncustodial wallets are wallets where the user takes responsibility for securing the keys, such as in your wallet application on your mobile phone. Storing keys in an application connected to the internet is referred to as hot storage. However, hot storage is the vulnerability most often exploited.

2.3.4 Bitcoin Transaction

Let's use the following scenario to introduce bitcoin transaction procedures:

- Bob, a businessman who operates online, chooses to accept bitcoins as payment.
- Alice, a buyer, wants to buy things from Bob with bitcoins.

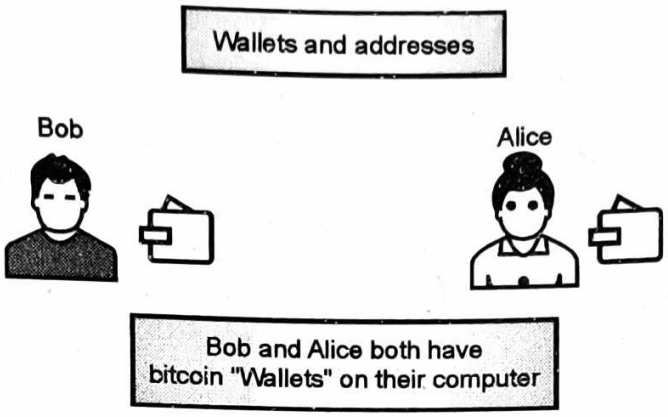

Fig. 2.3.1

- Multiple bitcoin addresses can be accessed through files known as wallets.
- Each bitcoin address has its own balance, which is a string of letters and numbers like 1HULMwZEPkjEPech43BeKJL1ybLCWrfDpN.
- The addresses function somewhat differently than bank accounts. Users of Bitcoin can establish as many addresses as they like.

(1) Creating a new address	(2) Submitting a payment
(3) Cryptographic Hashes	(4) Nonces
(5) Miner Awards	(6) Transaction Sequence

(1) Creating a new address

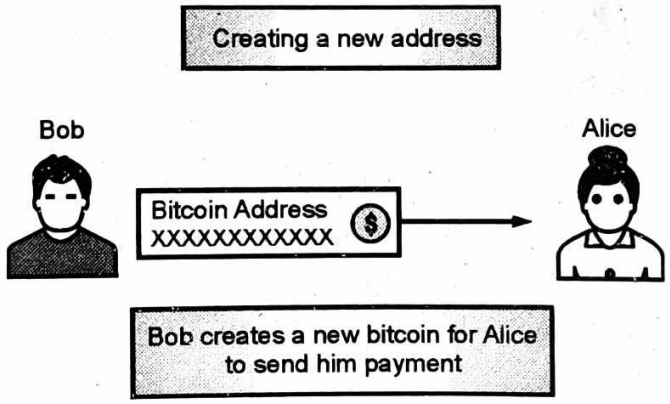

Fig. 2.3.2

- Users' bitcoin addresses are generated by bitcoin client software.
- Bob is actually creating a "cryptographic key pair" when he creates a new address. This "cryptographic key pair" consists of a private key that only you know and a public key that everyone knows.
- The matching public key can be used to verify a message signed with a private key. Bob's new bitcoin address is a one-of-a-kind public key, and his wallet contains the associated private key. Anyone can check the validity of a message signed with the private key using the public key.

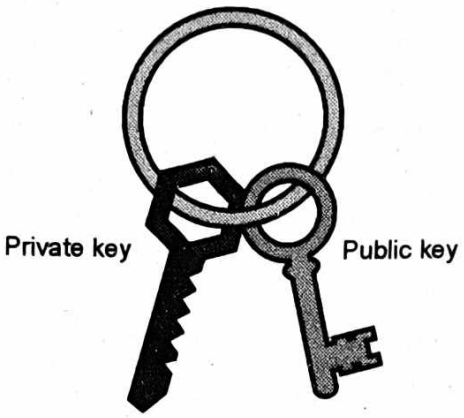

Fig. 2.3.3

(2) Submitting a payment

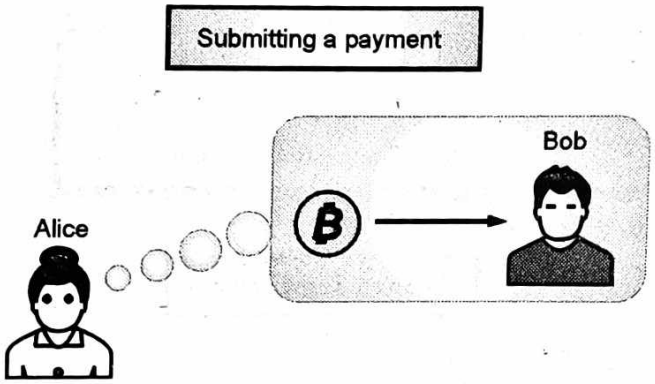

Fig. 2.3.4

- Alice informs her bitcoin client that she wants to send her bitcoins to Bob's address.
- The private keys for each of Alice's addresses are kept in her wallet. The bitcoin client signs her exchange demand with the confidential key of the location she's moving bitcoins from.
- The public key can now be used by anyone on the network to confirm that the transaction request is actually coming from the correct account owner.

- When receiving bitcoins, the public key is utilized.
- To spend those bitcoins, transactions are signed with the private key.
- In order to spend bitcoins, the current owner must provide their public key and digital signature in a bitcoin transaction.

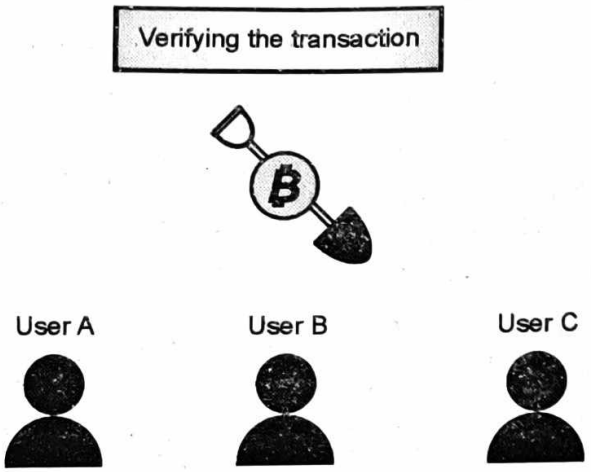

Fig. 2.3.5

- A private key that indicates authorization to spend the funds referred to in the transaction is used to digitally sign the transaction.
- With the public key and the computerized signature, everybody in the bitcoin organization can check and acknowledge the exchange as substantial, affirming that the individual moving the bitcoins claimed them at the hour of the exchange.

Say User A, User B and User C are Bitcoin miners. Their computers bundle the transactions of the past 10 minutes into a new "transaction block."

(3) Cryptographic Hashes

- A set of data is transformed by cryptographic hash functions into an alphanumeric string of a predetermined length, or hash value.

Input data	Hash value
Cryptocurrency	A81246023E3F6C6167A08-BA224409026F88BB8E98ED1431CD53CB63A328C6E84
Cryptocurrenci	B008011285CF8B6752819949F08466B77F-BA5B969DB2B0DB6E159541C7842A2C
Cryptocurrensy	9079F14392951359CFA41257499607606A8DC7828E0EE3CAC116AE4A9C93C932

Fig. 2.3.6

- The resulting hash value will be significantly altered by even the tiniest modifications to the initial data. Predicting which initial data set will produce a particular hash value is nearly impossible.

(4) Nonces

- Bitcoin makes use of nonces to generate distinct hash values from the same data. The hash value is drastically different when the nonce is changed.
- The mining PCs compute new hash values in view of a mix of the past hash esteem, the new exchange block, and a nonce.
- Although creating hashes is a simple computational task, the Bitcoin system stipulates that the new hash value must have a particular form, specifically a predetermined number of zeros.
- The Miners have no real way to foresee which nonce will deliver a hash esteem with the necessary number of driving zeros. As a result, they are compelled to generate numerous hashes using various numbers until they find one that works.

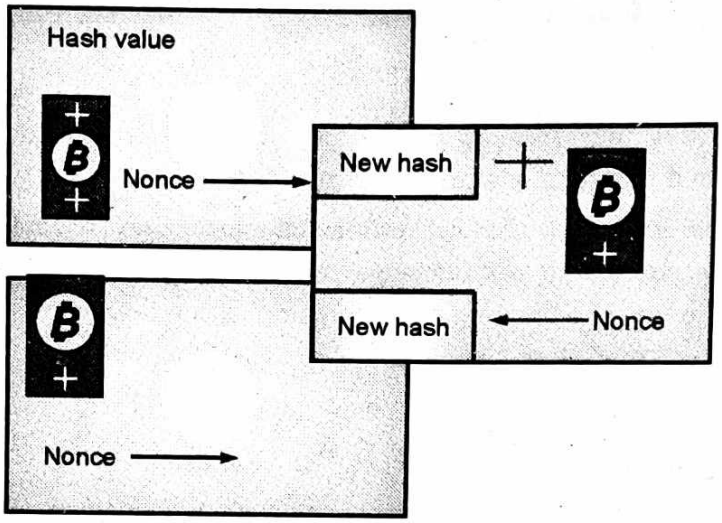

Fig. 2.3.7

(5) Miner Awards

- A "coinbase" transaction is included in each block, and the winner, Gary in this instance, receives 50 bitcoins. Gary's wallet is updated with a new address for the remaining bitcoins.
- Additionally, the miner receives payment for transactions sent by users. The expense is a motivation for the excavator to remember the exchange for their block.
- Alice's transfer to Bob gets lost in the shuffle of more recent transactions as time goes on. Because any changes require a completely different winning nonce and then redo the work of all the subsequent miners, he would have to redo Gary's work in order to change the details. A feat like this is almost impossible.

(6) Transaction Sequence

- The transaction is broadcast on the bitcoin network, where it is validated by each participant and spread to nearly every node in the network.
- A mining node verifies the transaction and records it in a block of transactions on the blockchain.
- The transaction becomes a permanent part of the bitcoin open distributed ledger and is accepted as valid by all participants once it is recorded on the blockchain and confirmed by enough subsequent blocks.
- After the transaction, the new owner can spend the bitcoin in a new transaction.

2.3.5 UTXO(Unspent Transaction Output)

Prior to presenting UTXO, we really want to know the information and result in bitcoin exchanges.

Input

- A reference to an output from a previous transaction is an input. The sum of all of the new transaction's input values (that is, the total coin value of the previous outputs that were referenced by the new transaction's inputs)is added up, and the new transaction's outputs use the entire sum(less any transaction fees)in their entirety.
- Numerous inputs are frequently included in a transaction.

Output

Instructions for sending bitcoins are contained in an output. The value that this output will be worth when claimed will be included in a transaction output.

UTXO (Unspent Transaction Output)

The outputs of all of the transactions that are included in the blockchain can be categorized as either Unspent Transaction Outputs (UTXOs) or spent transaction outputs due to the fact that each output of a particular transaction can only be spent once. A payment must only use UTXOs as inputs in order to be valid.

- A transaction output that can be used as input in a new transaction is referred to as an unspent transaction output (UTXO). In essence, UTXOs specify the starting and ending points of each blockchain transaction. A fundamental component of Bitcoin and numerous other cryptocurrencies is the UTXO model.
- To put it another way, inputs and outputs are the components of cryptocurrency transactions. A user uses one or more UTXOs as the inputs whenever a transaction is made.

- The user's digital signature is used to verify ownership of the inputs, which then produce outputs. The consumed UTXOs are now considered "spent" and cannot be utilized.
- In the meantime, the results from the exchange become new UTXOs - which can be spent in another exchange later.
- The main components of a bitcoin transaction are depicted in the figure above. At least one input and one output make up each transaction. The satoshis paid to a previous output are spent by each input.
- After that, each output acts as an Unspent Transaction Output (UTXO) until a subsequent input uses it. When your bitcoin wallet shows that you have a balance of 10,000 satoshis, it actually means that you have 10,000 satoshis in one or more UTXOs waiting for you.
- For instance, consider that you have 0.5 BTC and need to pay somebody 0.3 BTC. Your currency will be divided into two separate outputs during the transaction: the remaining 0.2 BTC in addition to the 0.3 BTC you paid. The 0.2 BTC is returned to you uninvested and converted into a UTXO that can be used as input in subsequent transactions.
- An example probably provides a better explanation for this. Alice's wallet contains 0.45 BTC. We might think of this as a fraction of a coin, but it isn't. Rather, it is a group of UTXOs. More specifically, two UTXOs with a value of 0.05 BTC and 0.4 BTC each—results of previous transactions. Now, let's suppose that Alice needs to pay Bob 0.3 BTC.
- Breaking up the 0.4 BTC unit and sending Bob 0.3 BTC and herself 0.1 BTC is her only option in this situation. Due to mining fees, she would typically reclaim less than 0.1 BTC, but let's keep things simple and exclude the miner.
- By means of a transaction, Alice basically communicates with the network: Break up my 0.4 BTC UTXO, send 0.3 BTC to Bob's address, and return the remaining 0.1 BTC to my address as an input. The 0.4 BTC can no longer be used because it is now a wasted output. In the meantime, two brand-new UTXOs(0.3 BTC and 0.1 BTC)have been produced.
- Although we broke up a UTXO in this example, Alice could have produced a UTXO with a value of 0.42 BTC by combining her 0.4 BTC with another 0.05 BTC and returning 0.03 BTC to herself.
- In conclusion, the protocol's mechanism for tracking where coins are at any given time is the UTXO model. In a sense, they function similarly to checks: they are addressed to particular users, or more accurately, their public addresses. UTXOs cannot be spent entirely; instead, new checks must be made from the existing ones and distributed accordingly.

2.4 VALIDATION OF TRANSACTION

When receiving transactions, every node ensures that they adhere to the network's rules. Transaction validation is the name of this procedure. Here are the primary things that a hub checks :

(1) Transaction validation is the method involved with deciding whether a transaction adjusts to explicit guidelines to consider it as substantial.

(2) As part of the validating process, validators check to see if transactions meet protocol requirements before adding them to the distributed ledger.

(3) Nodes that keep complete copies of the blockchain do this validation process. A transaction is added to the mempool once it has been approved by nodes (short for memory pool).

(4) In a proof of work network, transaction fees are used to encourage miners to confirm these transactions by adding them to a block in the blockchain.

(5) This creates a clear chronological record of when the transaction took place so that a subsequent transaction cannot use the same coins as the original.

(6) When the sender of a transaction has a starting value in their wallet that is equal to or higher than the amount being sent in the transaction, the transaction is deemed legitimate (including the transaction fee). This rule is typically relevant to all protocols, while other rules may exist based on the particular protocol in question.

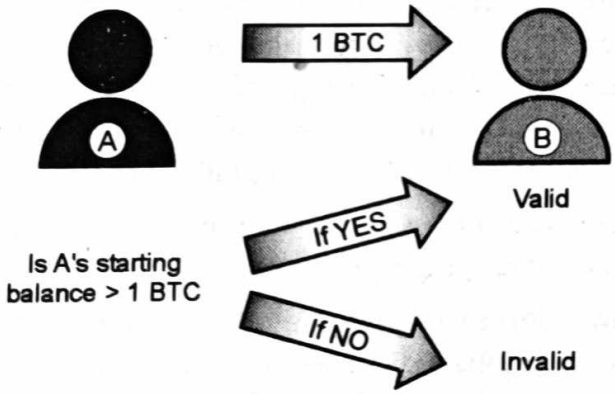

Fig. 2.4.1

(7) The so-called double-spend attack is an illustration of a situation in which a transaction should be rejected by a cryptocurrency system. In this situation, the first transaction is a legitimate transaction in which coins are spent from an address with sufficient funds to cover the transaction amount.

(8) The malevolent actor in this instance then attempts to block the inclusion of this legitimate transaction on the distributed ledger.

(9) They accomplish this by possessing the majority of the network's mining power, which enables them to mine the blockchain's longest iteration (i.e. the valid chain according to Nakamoto Consensus).

(10) On the off chance that fruitful, the malevolent entertainer will actually want to spend similar coins from the underlying exchange a second time in what might be an invalid exchange assuming the first legitimate exchange had been remembered for the record.

How does Bitcoin transaction confirmation work ?

(1) You will receive a "private key" to use when making a request for Bitcoin each time. This key is only accessible to you, and it is generated automatically and is unique for each transaction.

(2) The private key will be used to request the transaction, which will be broadcast on the Bitcoin network.

(3) A miner adds it to their own version of the blockchain ledger once it has been solved. The new block containing all of those transactions will then be added to the public blockchain after other miners and users known as nodes confirm the validity of the initial miner's proposal. Your transaction has been confirmed as a result of being included as a block in the blockchain.

2.5 BITCOIN KEYS

You may have heard that the mathematics of cryptography, which is extensively utilized in computer security, is the foundation of bitcoin. Greek for "secret writing," cryptography means "encryption," but the field of cryptography encompasses more than just secret writing. Cryptography can also be used to verify the authenticity of data or demonstrate knowledge of a secret without divulging it (digital signature).

The mathematical tools that are essential to bitcoin and extensively utilized in bitcoin applications are these kinds of cryptographic proofs.

(1) Digital keys, bitcoin addresses, and digital signatures are the means by which ownership of bitcoin is established. Users create and store the digital keys in a file or simple database called a wallet, which is not actually stored in the network.

(2) A user's wallet's digital keys can be generated and managed by the user's wallet software without referring to the blockchain or the internet because they are completely unrelated to the bitcoin protocol.

(3) The decentralized trust and control of bitcoin, ownership attestation, and the cryptographic-proof security model are all made possible by keys.

(4) To be included in the blockchain for the majority of bitcoin transactions, a valid digital signature that can only be generated with a secret key must be used; consequently, the bitcoin is in the hands of whoever has a copy of that key.

(5) In cryptography, the digital signature used to spend money is also called a witness. In a bitcoin transaction, the witness data shows who actually owns the money being spent.

(6) There are two types of keys: a public key and a private (secret) key. The public key is analogous to the number on a bank account, and the private key is analogous to the secret PIN, or signature on a check, that grants access to the account.

(7) The people who use bitcoin rarely see these digital keys. They are typically managed by the software for the bitcoin wallet and stored within the wallet file.

(8) A bitcoin address, which is a digital fingerprint of the recipient's public key, is used in the payment portion of a bitcoin transaction in the same way that the beneficiary name on a check is used (for example, "Pay to the order of"). As a rule, a bitcoin address is produced from and relates to a public key. But not every bitcoin address is a representation of public keys; as we will see in a later section of this chapter, they can also represent other beneficiaries, such as scripts.

(9) Similar to paper checks, bitcoin addresses abstract the recipient of funds, making transaction destinations flexible: a single payment method that can be used to transfer funds to cash, people's accounts, company accounts, or pay bills.

(10) Because this is the part of the keys that users are required to share with the rest of the world, the bitcoin address is the only representation of the keys that they will typically see.

2.5.1 Public Key Cryptography

- Public key cryptography was developed during the 1970s and is a numerical starting point for PC and data security.
- Since the innovation of public key cryptography, a few reasonable numerical capabilities, like indivisible number exponentiation and elliptic bend duplication, have been found. These numerical capabilities are basically irreversible, implying that they are not difficult to work out in one course and infeasible to compute the other way.
- In view of these numerical capabilities, cryptography empowers the production of computerized mysteries and unforgeable advanced marks. Bitcoin involves elliptic bend increase as the reason for its cryptography.
- In bitcoin, we utilize public key cryptography to make a key pair that controls admittance to bitcoin. The key pair comprises a confidential key and — got from it — an exceptional public key. The public key is utilized to get reserves, and the confidential key is utilized to sign exchanges to spend the assets.
- There is a numerical connection between people in general and the confidential key that permits the confidential key to be utilized to produce marks on messages. This mark can be approved against the public key without uncovering the confidential key.

- While spending bitcoin, the current bitcoin proprietor presents her public key and a mark (different each time, yet made from a similar confidential key) in an exchange to spend those bitcoins. Through the introduction of the public key and mark, everybody in the bitcoin organization can check and acknowledge the exchange as substantial, affirming that the individual moving the bitcoin possessed them at the hour of the exchange.

2.5.2 Private and Public Keys

Each key pair in a bitcoin wallet is made up of a private key and a public key. The private key (k) is usually a random number. We generate a public key (K) by employing the one-way cryptographic function elliptic curve multiplication from the private key. From the public key (K), we utilize a one-way cryptographic hash capability to produce a bitcoin address (A). In this section, we will first generate the private key, then examine the elliptic curve math used to convert the private key into a public key, and finally generate a bitcoin address using the public key. Figure depicts the connection between the bitcoin address, the public key, and the private key.

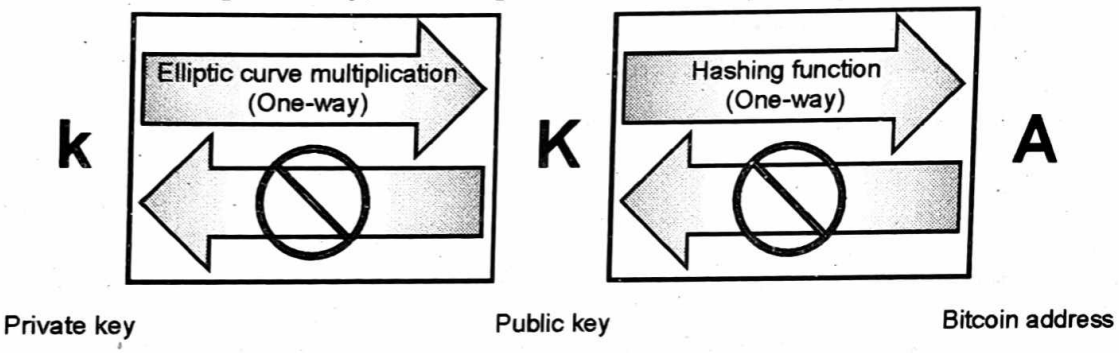

Fig. 2.5.1 : Private Key, public key, and bitcoin address

Private Key

(1) A private key is simply a number, picked at random. Ownership and control over the private key is the root of user control over all funds associated with the corresponding bitcoin address.

(2) The private key is used to create signatures that are required to spend bitcoin by proving ownership of funds used in a transaction.

(3) The private key must remain secret at all times, because revealing it to third parties is equivalent to giving them control over the bitcoin secured by that key.

(4) The private key must also be backed up and protected from accidental loss, because if it's lost it cannot be recovered and the funds secured by it are forever lost, too.

(5) The following is a private key (k) that was generated at random and is displayed in hexadecimal (256 bits, each 64-bit hexadecimal digit).

1E99423A4ED27608A15A2616A2B0E9E52CED330AC530EDCC32C8FFC6A526AEDD

Public Key

(1) An irreversible method called elliptic curve multiplication is used to derive the public key from the private key:

K = k * G, K is the generated public key, k is the private key, and G is a constant point known as the generator point.

(2) Calculating k if you know K - the reverse operation known as "finding the discrete logarithm" - is as challenging as performing a brute-force search on all k values.

(3) Let's take a closer look at elliptic curve cryptography before demonstrating how to create a public key from a private key.

2.6 ELLIPTIC CURVE CRYPTOGRAPHY

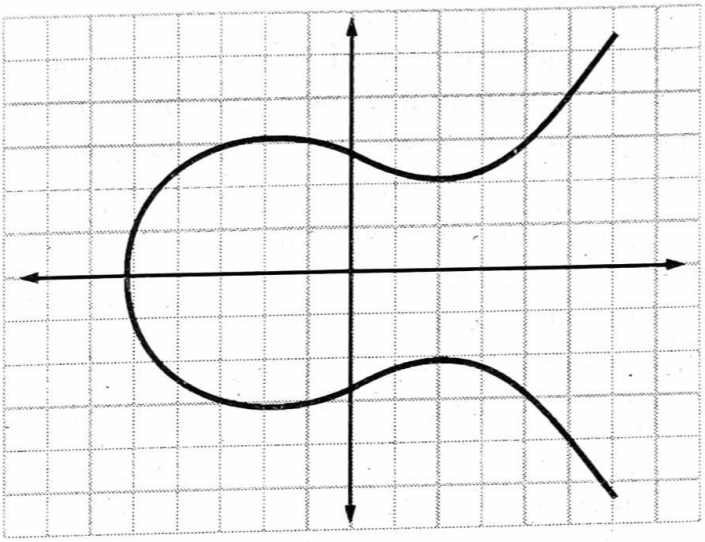

Fig. 2.6.1: An elliptic curve

- A type of asymmetric or public key cryptography known as elliptic curve cryptography is based on the discrete logarithm problem, which is expressed by adding and multiplying on the points of an elliptic curve.
- An example of an elliptic curve, like bitcoin's, can be seen in Fig. 2.6.1.
- The National Institute of Standards and Technology (NIST) established a standard that defines Bitcoin's elliptic curve and a set of mathematical constants. The secp256k1 bend is characterized by the accompanying capability, which delivers an elliptic bend :

$$y^2 = (x^3 + 7) \text{ over } (F_p)$$

or $$y^2 \bmod p = (x^3 + 7) \bmod p$$

- This curve crosses a finite field of prime order p, also known as F_p, where p =, a very large prime number.

- This curve looks like a pattern of scattered dots in two dimensions because it is defined over a finite field of prime order rather than real numbers. This makes it hard to see.
- On the other hand, the calculations are the same as for an elliptic curve over actual numbers.

2.7 BITCOIN ADDRESSES

A bitcoin address is a string of digits and characters that can be shared with anyone who wants to send you money. Addresses produced from public keys consist of a string of numbers and letters, beginning with the digit "1." Here's an example of a bitcoin address:

1J7mdg5rbQyUHENYdx39WVWK7fsLpEoXZy

- The bitcoin address is typically referred to as the "recipient" of the funds in a transaction. A bitcoin address can either represent the owner of a private/public key pair or something else, like a payment script. For the time being, let's focus on the straightforward example of a bitcoin address that is derived from a public key.
- Using one-way cryptographic hashing, the public key and the bitcoin address are derived. A one-way function that generates a fingerprint or "hash" of an input of any size is known as a "hashing algorithm" or simply "hash algorithm."
- In bitcoin, cryptographic hash functions are frequently used: in the mining Proof-of-Work algorithm, script addresses, and bitcoin addresses. The Secure Hash Algorithm (SHA) and the RACE Integrity Primitives Evaluation Message Digest (RIPEMD), specifically SHA256 and RIPEMD160, are the algorithms that are used to generate a bitcoin address from a public key.

We compute the SHA256 hash starting with the public key K and then the RIPEMD160 hash of the result, resulting in a 160-bit (20-byte) number:

A = RIPEMD160 (SHA256 (K))

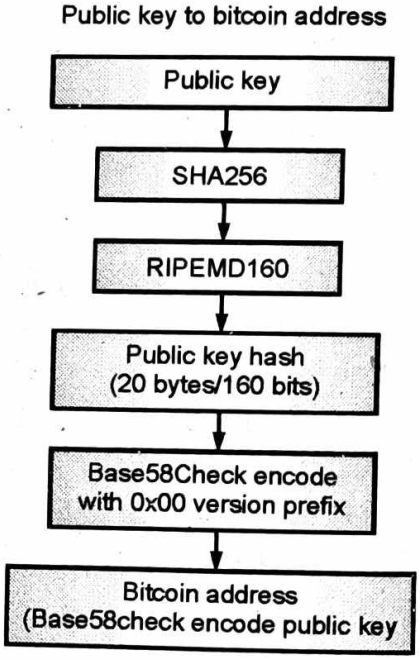

Fig. 2.7.1: Public key to bitcoin address: conversion of a public key into a bitcoin address

2.8 BASE58

- Most of the time, Bitcoin addresses are encoded as "Base58Check" (see "Base58 and Base58Check Encoding"), which uses 58 characters (from the Base58 number system) and a checksum to help humans read them, avoid ambiguity, and prevent mistakes when entering addresses.
- When a user needs to read and correctly transcribe a number, such as a bitcoin address, a private key, an encrypted key, or a script hash, Base58Check is also used in many other ways in bitcoin.
- Base58 is a text-based binary encoding format that is utilized in numerous other cryptocurrencies in addition to bitcoin. It strikes a balance between easy readability, concise representation, and error detection and prevention.
- The entire Base58 alphabet can be seen in Example

  123456789ABCDEFGHJKLMNPQRSTUVWXYZabcdefghijkmnopqrstuvwxyz
- Base58Check is a built-in error-checking code for the Base58 encoding format, which is frequently utilized in bitcoin, and provides additional security against typos or transcription errors.
- The data that is being encoded has an additional four bytes added to it for the checksum. Since the checksum is derived from the encoded data's hash, it can be used to find and avoid typing and transcription errors.
- The decoding software will compute the data's checksum and compare it to the checksum included in the code when presented with Base58Check code. An error has been introduced, and the Base58Check data is invalid if the two do not match.
- This forestalls a mistyped bitcoin address from being acknowledged by the wallet programming as a substantial objective, a blunder that would somehow bring about loss of assets.
- The first step in converting data (a number) into a Base58Check format is to add a prefix known as the "version byte," which makes it simple to identify the type of encoded data.
- For instance, the prefix for a bitcoin address is zero (hexadecimal: 0×00), whereas the prefix for encoding a private key is 128 (hexadecimal: 0×80). Table 2.8.1 provides a list of common version prefixes.

Table 2.8.1 : Base58Check version prefix and encoded result examples

Type	Version prefix (hex)	Base58 result prefix
Bitcoin Address	0 × 00	1
Pay-to-Script-Hash Address	0 × 05	3
Bitcoin Testnet Address	0 × 6F	m or n
Private Key WIF	0 × 80	5, K, or L
BIP-38 Encrypted Private Key	0 × 0142	6P
BIP-32 Extended Public Key	0 × 0488B21E	xpub

- The "double-SHA" checksum is the result of applying the SHA256 algorithm twice to the previous result(the prefix and the data)as follows :

  Checksum = SHA256(SHA256(prefix+data))

- We only take the first four bytes from the 32-byte hash (hash-of-a-hash) that was generated. The error-checking code, or checksum, consists of these four bytes. The checksum is added (concatenated) at the end.
- There are three components to the outcome: a prefix, the information, and a checksum. The previously mentioned Base58 alphabet is used to encode this result. The process of encoding Base58Check is depicted in Fig. 2.8.1.

2.8.1 Base58Check Encoding

- The majority of the data presented to users in bitcoin is encoded using Base58Check, making it compact, simple to read, and straightforward to spot errors. The version prefix in Base58Check encoding is used to create formats that are easy to tell apart.

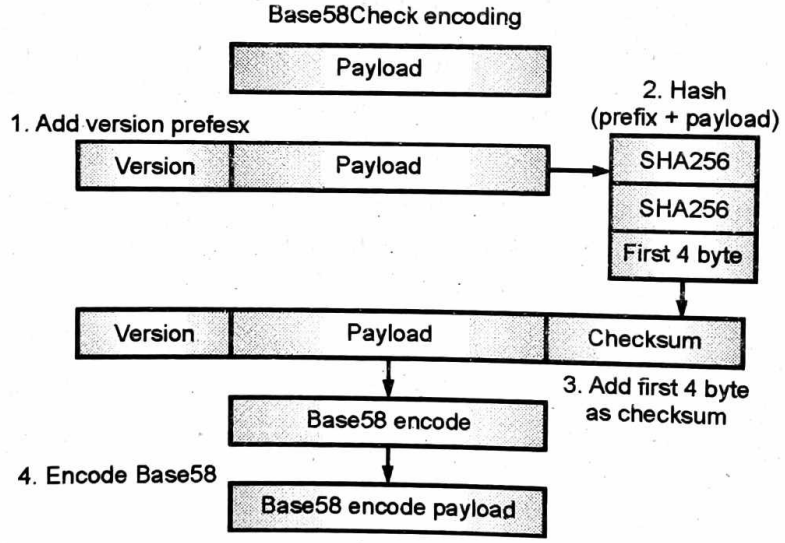

Fig. 2.8.1: Base58Check encoding: a Base58, versioned, and check summed format for unambiguously encoding bitcoin data

- When encoded in Base58, these formats contain particular characters at the payload's beginning. These characters make it simple for people to recognize the kind of information that is encoded and how to utilize it. This is what distinguishes, for instance, a bitcoin address encoded with Base58Check that begins with a 1 from a private key WIF encoded with Base58Check that begins with a 5.

2.8.2 Base58Check Decoding

- Shell scripts and command-line "pipes" that manipulate bitcoin keys, addresses, and transactions are made simple by the Bitcoin Explorer commands. On the command line, you can decode the Base58Check format with Bitcoin Explorer.
- To decode the uncompressed key, we make use of the base58check-decode command :

```
$ bx base58check-decode 5J3mBbAH58CpQ3Y5RNJpUKPE62SQ5tfcvU2JpbnkeyhfsYB1Jcn
wrapper
{
    checksum 4286807748
    payload 1e99423a4ed27608a15a2616a2b0e9e52ced330ac530edcc32c8ffc6a526aedd
    version 128
}
```

- A checksum, the WIF version prefix 128, and the key as payload are all contained in the outcome.
- The compressed key's "payload" includes the suffix 01, indicating that the derived public key must be compressed :

```
$ bx base58check-decode KxFC1jmwwCoACiCAWZ3eXa96mBM6tb3TYzGmf6YwgdGWZgawvrtJ
wrapper
{
    checksum 2339607926
    payload 1e99423a4ed27608a15a2616a2b0e9e52ced330ac530edcc32c8ffc6a526aedd01
    version 128
}
```

2.9 ENCRYPTED PRIVATE KEYS (BIP-38)

It is a well-known fact that the private keys need to be kept private, but it is very hard to do in practice because it conflicts with the availability goal, which is just as important for security. When you need to store backups of the private key to avoid losing it, keeping the key private becomes much more difficult.

Although a password-encrypted wallet containing a private key may be safe, a backup of that wallet is required. Users may need to transfer keys between wallets in order to upgrade or replace the software.

However, what happens if the backup is stolen or lost? BIP-38, a portable and convenient standard for encrypting private keys in a way that can be understood by many different wallets and bitcoin clients, was established as a result of these competing security objectives.

(1) BIP-38 proposes a common method for encoding private keys with Base58Check and encrypting them with a passphrase so that they can be safely transported between wallets, stored on backup media, or used in any other situation where the key could be exposed.

(2) The norm for encryption utilizes the Advance Encryption Standard (AES), a standard laid out by the NIST and utilized extensively in information encryption executions for business and military applications.

(3) A Base58Check string with the prefix "5" serves as the input for a BIP-38 encryption scheme. The bitcoin private key is typically encoded in the WIF.

(4) In addition, the BIP-38 encryption method requires a passphrase, also known as a lengthy password. Passphrases typically consist of a number of words or a complex string of alphanumeric characters.

(5) If you see a key with the prefix 6P, it is encrypted and requires a passphrase to be decrypted back into a WIF-formatted private key (prefix 5) that can be used in any wallet. The BIP-38 encryption scheme produces an encrypted private key that begins with the prefix 6P.

(6) To decrypt and import the key, many wallet applications now recognize BIP-38-encrypted private keys and will prompt the user for a passphrase. Outsider applications, for example, the extraordinarily valuable program based Digit Address (Wallet Subtleties tab), can be utilized to unscramble BIP-38 keys.

(7) Paper wallets that can be used to back up private keys on a piece of paper are the most common use case for BIP-38 encrypted keys.

(8) A paper wallet with BIP-38 encrypted private keys is extremely secure and a great way to create offline bitcoin storage (also known as "cold storage") as long as the user selects a strong passphrase.

Table 2.9.1: Example of BIP-38 encrypted private key

Private Key (WIF)	5J3mBbAH58CpQ3Y5RNJpUKPE62SQ5tfcvU2JpbnkeyhfsYB1Jcn
Passphrase	MyTestPassphrase
Encrypted Key (BIP-38)	6PRTHL6mWa48xSopbU1cKrVjpKbBZxcLRRCdctLJ3z5yxE87MobKoXdTsJ

2.10 PAY-TO-SCRIPT(P2S) ADDRESS

- We are aware that traditional bitcoin addresses are derived from the public key, which is derived from the private key, and begin with the number "1." Despite the fact that anybody can send bitcoin to a "1" address, that bitcoin must be spent by introducing the related private key signature and public key hash.
- Bitcoin tends to start with the number "3" are pay-to-prearrange hash (P2SH) addresses, in some cases mistakenly called multi signature or multisig addresses. Instead of naming the owner of a public key, they name the beneficiary of a bitcoin transaction as the hash of a script.
- The feature, which was made available with BIP-16 in January 2012, is getting a lot of use because it lets you add functionality to the address itself.
- Funds sent to "3" addresses, in contrast to pay-to-public-key-hash (P2PKH) transactions that "send" funds to traditional "1" bitcoin addresses, necessitate more than just the presentation of a single public key hash and private key signature as proof of ownership.
- All inputs to this address will be burdened with the same requirements, which are specified at the time the address is created within the script.
- A transaction script is used to create a P2SH address, which specifies who can spend transaction output. To encode a P2SH address, the double-hash function used to create a bitcoin address is applied to the script rather than the public key :

  script hash = RIPEMD160(SHA256(script))

- The "script hash" that is generated is encoded using Base58Check with a version prefix of 5 and an encoded address that begins with a 3.
- The Bitcoin Explorer commands script-encode, sha256, ripemd160, and base58check-encode can be used to derive an example of a P2SH address, which is 3F6i6kwkevjR7AsAd4te2YB2zZyASEm1HM.

```
$ echo \
'DUP HASH160 [89abcdefabbaabbaabbaabbaabbaabbaabbaabba] EQUALVERIFY
CHECKSIG' > script
$ bx script-encode < script | bx sha256 | bx ripemd160 \
| bx base58check-encode --version 5
3F6i6kwkevjR7AsAd4te2YB2zZyASEm1HM
```

2.11 MULTISIGNATURE ADDRESSES

- The multi-signature address script is currently the P2SH function's most common implementation.
- In order to demonstrate ownership and, as a result, spend funds, the underlying script necessitates multiple signatures, as the name suggests.
- The bitcoin multi-signature feature is designed to require M signatures (also known as the "threshold") from a total of N keys. This is called an M-of-N multisig, where M is equal to or less than N.
- It is similar to a "joint account," which allows both spouses to spend with a single signature, which is used in traditional banking.

2.12 VANITY ADDRESS

- Valid bitcoin addresses with messages that can be read by humans are known as vanity addresses.
- For instance, the address 1HaveBPzzD72PUXLzCkYAtGFYmK5vYNR33 is legal because it contains the first four Base-58 letters that make up the word "Have."
- Before a bitcoin address matching the desired pattern can be found, billions of candidate private keys must be generated and tested for vanity addresses.
- The vanity generation algorithm has some optimizations, but the process basically entails picking a private key at random, deriving the public key, deriving the bitcoin address, and comparing it to the desired vanity pattern billions of times until a match is found.
- The private key that was derived from the vanity address can be used by the owner to spend bitcoin in the same way as any other address once it is discovered and matches the desired pattern. Any address, vanity included, is just as safe as any other.
- Like any other address, they rely on SHA and Elliptic Curve Cryptography (ECC). You can no more effectively find the confidential key of a location beginning with a vanity design than you can some other location.

2.13 BITCOIN WALLETS

One of the bitcoin ecosystem's most actively developed applications is bitcoin wallets. There are a lot of wallets that are focused on particular platforms or uses. Some are better for beginners, while others have a lot of features for more advanced users. However, we are able to classify bitcoin wallets according to their platform and function, providing some clarity regarding the various wallet types that are available.

The following categories of bitcoin wallets can be found on the platform:

(1) Desktop wallet	(2) Mobile wallet	(3) Web wallet
(4) Hardware wallet	(5) Paper wallet	

(1) Desktop wallet

- Many people use desktop wallets for the features, autonomy, and control they provide. The desktop wallet was the first type of bitcoin wallet created as a reference implementation.
- However, because these platforms are frequently unsecure and poorly configured, running on general-purpose operating systems like Windows and Mac OS poses some security risks.

(2) Mobile wallet

- The most prevalent kind of bitcoin wallet is the mobile wallet. These wallets, which are compatible with smartphone operating systems like Android and iOS, are frequently an excellent option for novice users.
- While many mobile wallets are made to be simple and easy to use, there are also fully featured options for power users.

(3) Web wallet

- Web wallets store the user's wallet on a server that belongs to a third party and can be accessed through a web browser. In that it entirely depends on a server from a third party, this is similar to webmail.
- Some of these services use client-side code that runs in the browser of the user; giving the user control of the bitcoin keys.
- The majority, on the other hand, make a concession by giving users control of the bitcoin keys in exchange for making it easier to use. It is not a good idea to store a lot of bitcoin on systems owned by third parties.

(4) Hardware wallet

- A hardware wallet is a device that uses special-purpose hardware to run a secure, self-contained bitcoin wallet. They can be controlled via near-field communication (NFC) on a mobile device or USB with a desktop web browser.
- These wallets are thought to be very secure and suitable for storing large amounts of bitcoin because they handle all bitcoin-related operations on specialized hardware.

(5) Paper wallet

- The keys that control bitcoin can also be printed and stored for a long time. Despite the fact that other materials (such as wood, metal, etc.) can be utilized.
- A low-tech but highly secure option for long-term bitcoin storage are paper wallets. Cold storage is another common name for offline storage.

2.13.1 Technologies in Bitcoin HD Wallet from Seed

- In bitcoin, the term "wallet" refers to a number of different things.
- A wallet is an application that serves as the primary user interface at a high level. A user's money is managed, keys and addresses are managed, the balance is tracked, and transactions are created and signed by the wallet.
- From a programmer's point of view, the term "wallet" refers more specifically to the data structure used to store and manage a user's keys.

2.13.1(A) Creating an HD Wallet from the Seed

(1) **Seed :** The private keys are obtained by combining the seed, a number that is generated at random, with other data, such as an index number or "chain code."

(2) **HD Wallet :** Keys in HD wallets are derived in a tree structure, so a parent key can derive a sequence of keys for children, each of which can derive a sequence of keys for grandchildren, and so on and so forth to an infinite depth.

(3) HD wallets are a very effective tool for managing a large number of keys and addresses. A standardized method for creating seeds from a sequence of English words that is simple to transcribe, export, and import across wallets makes them even more useful. A mnemonic is a type of memory.

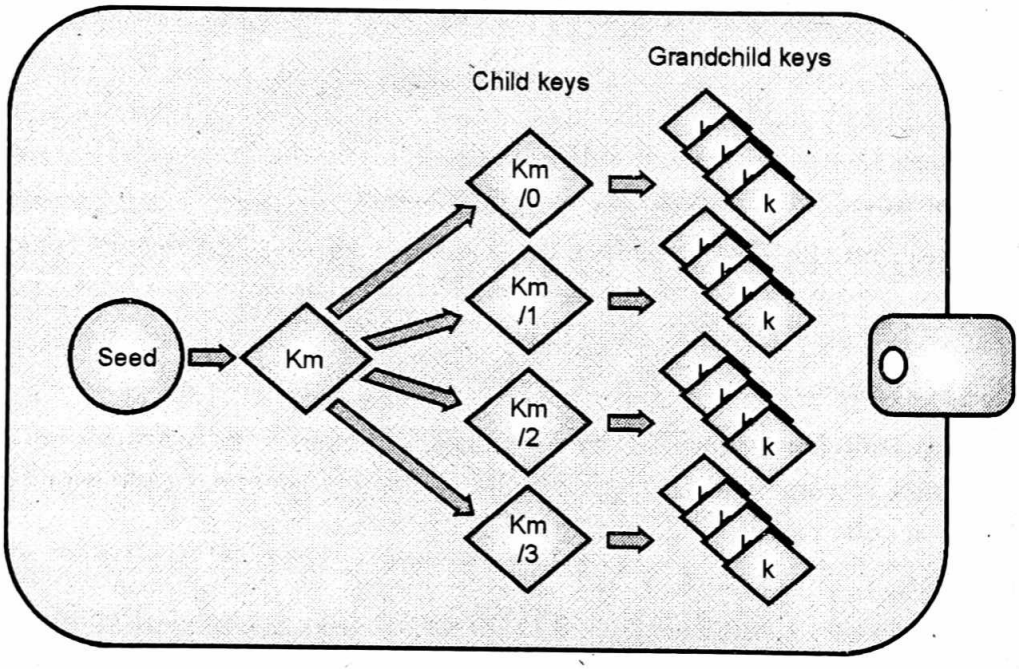

Fig. 2.13.1: HD wallet: a tree of keys generated from a single seed

(4) All keys are generated deterministically from the seed, as shown in the image above. With a seed and any HD wallet that is compatible, users can now re-create a new HD wallet. With a single root seed, it is now easier to backup, store, import, and export

HD wallets with millions of keys.

The steps required to create an HD wallet's master key and master chain code are outlined below:

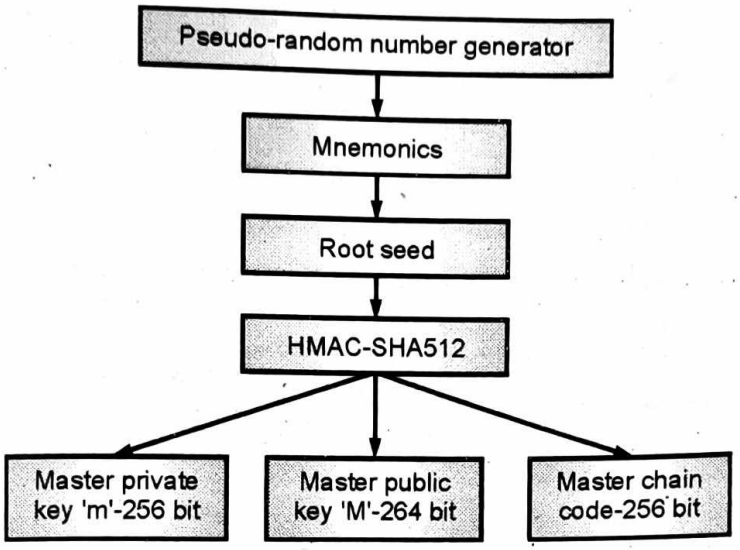

Fig. 2.13.2

- The HMAC-SHA512 hashing algorithm returns a hash above the root seed, which we will use to generate a private key and master chain code. The elliptic curve multiplication method is then used to generate a master public key that corresponds to the first.
- Entropy of 256 bits makes up the chain code. It adds randomness to the function that generates parental keys for children. Given a single child key, this makes it impossible for malicious parties to identify sibling keys. The seed of the wallet is the source of the chain code's root.
- The root seed was hashed, and the result is the master chain code. The extended public key is made by combining the master private key and the master chain code.
- The seed phrase is used to generate the 256-bit master private key. As can be seen in the first image, it makes it possible to create an infinite number of keys for the wallet. We derive the master public key from the master key, which is the foundation of the entire deterministic wallet.
- The master private key is the source of the master public key. In this scenario, the funds cannot be lost even if a public key is exposed. Because they do not have access to the private key, malicious individuals can only view key-related transactions.
- The master public key can be used to generate an infinite number of public keys, each of which can be used to create a unique wallet address. We are able to use them without having to back them up because of this. In this case, if a key is lost, we can recover a wallet in its entirety by starting from the beginning.

2.13.2 Deriving a Private Child Key

- To get child keys from parent keys, a Child key derivation (CKD) function is used. One-way functions (trapdoors) that combine the parent private or public key(1), a chain code(2), and an index number(3) serve as the foundation for these functions.
- Since the original chain code seed (the root seed) is generated from the original seed and subsequent child nodes are derivations of the parent chain code, the chain code introduces randomization into the process, making it impossible to derive other child keys with only the index and a child key. Information on the chain code is essential to have the option to find the youngster key.

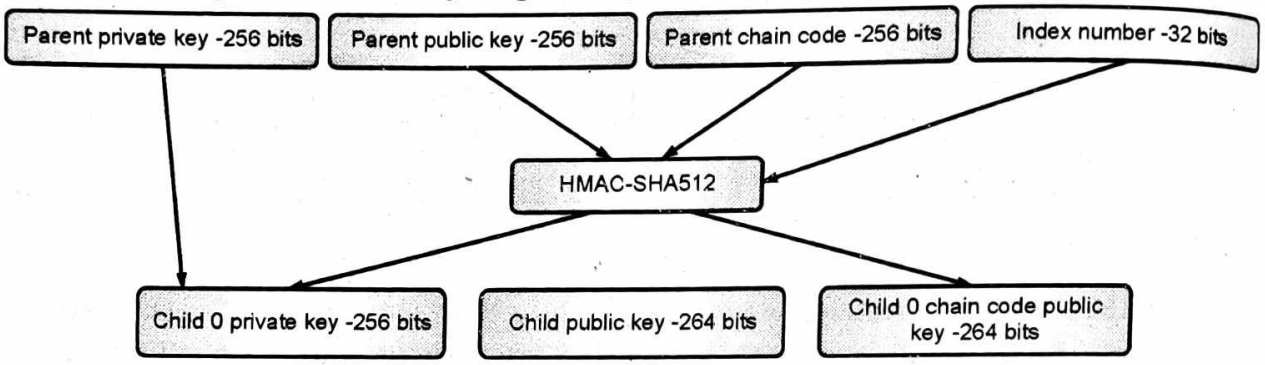

Fig. 2.13.3

2.13.3 Deriving a Public Child Key

- Without the need for private keys, HD wallets enable us to derive public child keys from public parent keys. Either the parent public key or the child private key can be used to derive a public key.

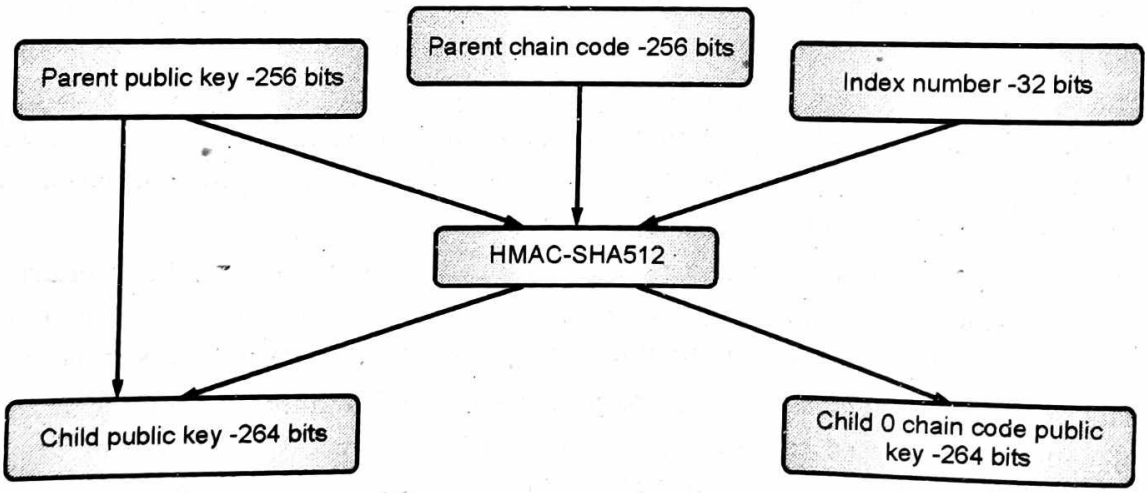

Fig. 2.13.4

- As a result, all public keys in a branch of the HD wallet structure are derived from extended public keys. A server can store a copy of the extended public key but no private keys for secure public-key deployments as a result of this.

2.14 TRANSACTION SCRIPT

- Transactions on the blockchain are carried out with the help of a scripting language in Bitcoin. A script is a straightforward stack-based list of instructions recorded with each transaction that explains how the recipient of bitcoins can access the coins.
- Theoretically and purposefully, Bitcoin scripts lack a looping mechanism and are theoretically complete. They are deterministic, and since there can be no infinite loops when using scripts, we know that the program will end at some point. DDoS attacks, code blocking, and infinite looping are just a few of the problems this feature protects the system from. Scripts likewise ensure that the computational power expected to handle exchanges is limited.
- Using the following methods, a bitcoin script prevents the sent bitcoins from being spent in the future :
  (1) **Hashed public key:** A public key that has been hashed and contains the script-integrated address of the destination wallet.
  (2) **Digital Signature:** A digital signature from the sender that shows that the sender owns the bitcoins by showing that they own the private key of the public key that corresponds to it.
- If the combined script does not result in a failure and the top stack item is True after the script has finished running, then the transaction is said to be valid.
- For B to be able to spend the bitcoins, B provides the inputs to the previous scripts, resulting in a combined script that terminates with a True value at the top of the stack. For example, if A sent B some bitcoins, B dictates the script operations that need to be performed for the sent amount to be used in another subsequent transaction by A.
- Lets learn about the bitcoin scripting language that is used for transactions on the blockchain and show how it is used to express conditional spending while protecting the blockchain's security.

(1) Turing Incompleteness

- A system is said to be complete in the theory of computation if it can be used to simulate an Alan Turing machine. Turing-complete programming languages like Python, Java, and C++ are the majority.
- A turing INCOMPLETE system, on the other hand, cannot replicate a turing machine. The Bitcoin Script programming language is one example of a turing incomplete system.

- The Content backings numerous functionalities of a customary significant level programming language like constants, the stack, bitwise tasks, held catchphrases, number-crunching, graft, and substantially more. Except for looping mechanisms, all of these.
- As a result, scripts can only run for a predetermined amount of time. By restricting scripts, network bandwidth is conserved and logic bombs and infinite loops, which can be embedded in bitcoin transactions, are prevented from crashing or blocking.

(2) Stack-Based

- A stack is a data structure in computer science that lets data be processed using LIFO (Last In, First Out) order. The operations of stacks can be compared to those of a "stack of trays or plates," with the first to be "popped" off from the top of the stack being the last to be "pushed" on top of the stack.
- For its operations with OP_Codes, which are the subject of the following section, Bitcoin script makes use of a stack LIFO mechanism.
- The Bitcoin script puts everything on a stack, in contrast to conventional high-level programming languages that have variables.
- The Bitcoin script stack mechanism enables the language to avoid introducing security issues on its own.

(3) Bitcoin Script OP_Codes

- A type of programming language known as opcode, or operational code, is very similar to machine code. Opcodes are used to specify the data operations that will be carried out with operands.
- Opcodes enable the programmability of blockchain transactions in Bitcoin. Keep in mind that cryptocurrencies are money that can be changed. Out of the 256 opcodes (from 0 to 255) that Bitcoin uses, 116 are the most frequently used.
- They incorporate activities for putting inputs on top of the stack (OP_TOTALSTACK), eliminating the best two stack items(OP_ADD), checking for copies on the stack(OP_IFDUP), popping-correlation pushing(OP_EQUAL), signature verification(OP_CHECKSIG) among others.

2.15 SCRIPT ADDRESS

The language that Bitcoin uses to determine who can spend a specific amount of bitcoin (a UTXO) is called script. To keep Bitcoin's simplicity and ensure that anyone can run a node and validate the blockchain, script is intentionally limited.

Pay-to-Script-Hash (P2SH) Address

(1) P2SH (Pay-To-Script Hash) makes it simple to represent a scriptPubKey as a straightforward BitcoinScriptAddress, regardless of how complicated its underlying m-of-n signature configuration is.

(2) Pay-to-Script-Hash (P2SH) is a type of ScriptPubKey that lets you spend bitcoin if the script meets the requirements of the hash that is specified in the transaction. A P2SH transaction is one in which a P2SH ScriptPubKey was used to lock the inputs.

(3) For instance, in a P2SH transaction where Alice sends Bob one bitcoin, she includes the hash of the script needed to spend the bitcoin.

(4) This script may necessitate Bob's private keys' signatures or a variety of other qualifications. Bob reconstructs the script that Alice used to send him the bitcoin and signs the transaction with any private keys required by the script when he wants to spend it.

(5) Because it lets users create any kind of script, P2SH is very adaptable. P2SH is used to make new transaction types, like SegWit, compatible with older versions of the system.

(6) In addition, the transaction sender need not be aware of the script type being sent to. In the model above, Sway can secretly develop his ideal content and just send Alice the hash of that content, saving more security for Bounce.

2.16 BITCOIN MINING

- Bitcoin mining is the process by which new bitcoin enters circulation and the process by which new transactions to the Bitcoin digital currency system are verified.
- The purpose of bitcoin mining is to store current bitcoin transactions in blocks, which are then added to a blockchain a record of transactions that have already taken place.
- The process of creating new bitcoins by solving extremely difficult math problems that verify bitcoin transactions is known as bitcoin mining. A predetermined amount of bitcoin is given to the miner whenever a bitcoin is successfully mined.

2.16.1 How Bitcoin Mining Works

- Bitcoin miners compete to solve extremely complex math problems that necessitate the use of expensive computers and enormous amounts of electricity in order to successfully add a block.
- Miners must be the first to arrive at the correct or closest answer to the question in order to complete the mining process. Proof of work is the procedure of guessing the correct number (hash).

- Miners use a lot of computing power to guess the target hash by guessing as many times as possible at random. As more miners join the network, the difficulty only gets harder.
- Application-specific integrated circuits, or ASICs, are the necessary pieces of computer hardware and can run upwards of $10,000. ASICs use a lot of electricity, which hurts miners' profitability and has been criticized by environmental groups.
- A miner will be rewarded with 6.25 bitcoins if they add a block to the blockchain successfully. Every 210,000 blocks, or about every four years, the reward is cut in half. Bitcoin was trading at around $20,000 in September 2022, making 6.25 bitcoins worth $125,000.

2.16.2 How is Bitcoin Mining Difficulty Calculated?

- The difficulty of mining Bitcoin is determined using a variety of formulas. The most typical, however, is: Difficulty Level = Current Target x Difficulty Target
- Take note that the target hash's hexadecimal notation, the Difficulty Target, has a mining difficulty of 1.
- The current target, on the other hand, is the target hash of the most recent transaction block. At the point when the two qualities are separated, it yields an entire number which is the trouble level of mining bitcoin.

2.17 STRUCTURE OF A BLOCK

A block is a container data structure that collects transactions to be added to the blockchain's public ledger. A header that contains metadata is followed by a long list of transactions that make up the majority of the block's size.

The average transaction is at least 250 bytes long, and the average block contains more than 500 transactions. The block header is 80 bytes long. As a result, the block header is 1,000 times smaller than the entire block, including all transactions.

The structure of a block is shown in Table 2.17.1.

Table 2.17.1: The structure of a block

Size	Field	Description
4 bytes	Block Size	The size of the block, in bytes, following this field
80 bytes	Block Header	Several fields from the block header
1-9 bytes (VarInt)	Transaction Counter	How many transactions follow
Variable	Transactions	The transactions recorded in this block

2.18 BLOCK HEADER

There are three sets of block metadata in the block header. In the first place, there is a reference to a past block hash, which associates this block to the past block in the blockchain. The difficulty, timestamp, and nonce of the second set of metadata are related to the mining competition. The merkle tree root, a data structure used to effectively summarize all of the transactions in the block, is the third piece of metadata.

Size	Field	Description
4 bytes	Version	A version number to track software/protocol upgrades
32 bytes	Previous Block Hash	A reference to the hash of the previous (parent) block in the chain
32 bytes	Merkle Root	A hash of the root of the merkle tree of this block's transactions
4 bytes	Timestamp	The approximate creation time of this block (seconds from Unix Epoch)
4 bytes	Difficulty Target	The proof-of-work algorithm difficulty target for this block
4 bytes	Nonce	A counter used for the proof-of-work algorithm

Constituents of Block Header are:

(1) Timestamp	(2) Version	(3) Merkle Root
(4) Difficulty Target	(5) Nonce	(6) Previous Hash

(1) Timestamp

- In the blockchain, the timestamp is used as a parameter to verify the authenticity of any block and serves as evidence that a particular block was used at what point in time.

(2) Version

- There are three kinds of Blockchain versions, and this indicates which version is being used by the particular block.
- Blockchain (cryptocurrency) Version 1.0 used a public ledger to store data, such as Bitcoin.
- Blockchain Version 2.0 (smart Contract): Smart contracts are programs that run on their own, like Ethereum. To create a decentralized structure like Tor Browser,
- Blockchain Version 3.0 (DAPPS) is used. Blockchain Version 4.0 (Blockchain for Industry) is used to make a blockchain network that is scalable and affordable so that more people can use it.

(3) Merkle Root

- To determine whether or not the data has been tampered with, hacked, or corrupted, a Merkle root employs mathematical formulas.
- For instance, Assume one block has 10 exchanges, then to distinguish that block we want 10 exchanges to join and shape one Hash Worth, so it utilizes the idea of the double tree to make the hash of the block and that worth is known as the Merkle Root

(4) Difficulty Target

- It tells how complicated the network is and how much power is needed to mine it. If the target is difficult, it means that we need a machine that is more expensive to use for computation. For instance, target algorithms like SHA-2 and SHA-3 should be made harder. It uses RIPEMD, MD5, and BLAKE2.

(5) Nonce

- It is a number that blockchain miners are finding. On average, it takes almost ten times to find the correct nonce, which is abbreviated as "number only used once."
- Because a nonce is a 32-bit number with a maximum value of 2(32), the task of the bitcoin miners is to determine the correct integer value, which is a random integer between 0 and 32, which requires a lot of computational power.

(6) Previous Hash

- The previous hash stores the hashed value of the previous node's address because a blockchain is a collection of several interconnected nodes called blocks.
- The Genesis Block, which is the first block in the blockchain, does not have a previous block hash value.

2.19 THE GENESIS BLOCK

- The genesis block, which was created in 2009, is the first block in the blockchain. Because it is the common ancestor of all blockchain blocks, the genesis block will eventually be reached if you start at any block and follow the chain backward in time.
- Due to the fact that the genesis block is statically encoded within the bitcoin client software and cannot be altered, every node always starts with a blockchain consisting of at least one block. Every node always "knows" the hash and structure of the genesis block, as well as the specific time at which it was created and even the single transaction within.
- As a result, every node has the secure "root" from which to construct a trusted blockchain, which serves as the blockchain's starting point.

2.20 LINKING OF BLOCKS

- Starting with the genesis block, Bitcoin full nodes maintain a local copy of the blockchain. As new blocks are discovered and used to extend the chain, the local copy of the blockchain is constantly updated.
- A node will validate the blocks it receives from the network and then link them to the existing blockchain. A node will look for the "previous block hash" in the incoming block header in order to establish a link.

Bitcoin Network

2.21 BITCOIN CORE NODE AND API

Nodes play a pivotal role in the Bitcoin network. You can think of them as "guardians," who are always keeping an eye on the Bitcoin blockchain to tell the difference between legitimate and fraudulent Bitcoin transactions. Their primary responsibility is to stop people from spending bitcoins that have already been spent elsewhere twice.

What does it mean to be a "full node"?

- Although the terms "node" and "full node" are frequently used interchangeably, there is actually a difference between the two.
- "A full node is a program that fully validates transactions and blocks," the Bitcoin Core documentation states. Additionally, nearly all full nodes contribute to the network by accepting transactions and blocks from other full nodes, validating those transactions and blocks, and then relaying them to additional full nodes.

2.21.1 Types of Full Nodes

- The **pruned full node**, which downloads blocks from the beginning of the chain until it reaches a certain limit and then deletes the oldest blocks, is one type of full node. Because the subtree of the decision tree has been removed, the pruned node takes up less space on the hard drive, which is why it is referred to as a "pruned" node.
- **Archival full nodes**, then again, have the whole blockchain, taking up much more hard-drive space than the pruned full hub. Additional subcategories are used to organize archive nodes.

2.21.2 Bitcoin Core Application Programming Interface (API)

What is the API?

- Application Programming Interface (API) aims to simplify the underlying implementation by providing an abstract layer for software program interaction.

- Simply put, an API would be useful for gaining access to the application of another developer without requiring the entire code. Access would be my primary focus here.

How do I utilize the Bitcoin API?

- You will need an API key to authenticate and provide any data in order to use the endpoints. Make sure to use the Crypto APIs dashboard to generate an API Key. To begin, select "Create New API Key."
- This is applicable to all three: production, testing, and development.

(1) Content-Type: application/json (2) X-API-Key: my-api-key

General Information Endpoint

- Second, let's say you need general information about the node or the most recent block from my-api-key's endpoint. That can be retrieved from the base resource.
- The object that is returned will include a plethora of data about the blockchain, such as its height, the time and hash of the most recent block, and other details.

Block Endpoint

- Thirdly, if you need to query a particular block, you can use a hash string or integer form (height) as parameters to request the block.
- The block's height, the number of transactions within it, the transaction hashes listed in the canonical order in which they appear in the block, and other information will be represented by the returned object in JSON format.

Addresses Endpoint

- What's more, Programming interface is likewise carried out for explicit recovery of data in: omni layer, webhook notifications, addresses, wallets, transactions, and payment forwarding Simply use the default Address Endpoint to GET the address's information, including its bitcoin balance and the number of transactions associated with it.
- CryptoAPIs provides the Multisig Address Endpoint if that address is involved in multisignature addresses

Create Wallets Endpoints

- Create Endpoints for Wallets Later, the Address API can use both normal and HD wallets by using their $NAME instead of their $ADDRESS. Typically, this will result in batching the wallet's collection of addresses.
- As a result, using the appropriate endpoint and a partially filled-out Wallet or HDWallet object, create a new wallet. The WALLET_NAME attribute and at least one public address in the ADDRESSES array are required for normal wallets.

{ "walletName" : ${WALLET_NAME}, "addresses" : ${ADDRESSES} }

For HD wallets, you must include WALLET_NAME, the ADDRESS_COUNT and the PASSWORD attributes

{ "walletName" : ${WALLET_NAME}, "addressCount" : ${ADDRESS_COUNT}, "password" : ${PASSWORD} }

it will return a Wallet or HDWallet object containing the requested data.

2.22 PEER-TO-PEER NETWORK ARCHITECTURE

Bitcoin is built on top of the Internet as a peer-to-peer network architecture. Peer-to-peer, or P2P, refers to a network in which all participating computers are peers to one another, are treated equally, and share responsibility for providing network services with no "special" nodes. In a mesh network with a "flat" topology, the network nodes connect to each other. The network does not have a server, a centralized service, or a hierarchy.

- In a peer-to-peer network, nodes simultaneously provide and consume services, and participation is rewarded with reciprocity. Decentralized, open, and resilient by design, peer-to-peer networks The early Internet was the best example of a P2P network architecture because all IP network nodes were equal.
- While the Internet Protocol still possesses its flat-topology essence, the architecture of the Internet of today is more hierarchical. File sharing is the largest and most successful use of P2P technologies outside of bitcoin, with Napster being the pioneer and BitTorrent being the most recent architecture evolution.
- The P2P network architecture of Bitcoin is much more than just a choice of topology. By design, Bitcoin is a peer-to-peer digital cash system, and the network architecture reflects and is based on that fundamental feature.
- A flat, decentralized P2P consensus network is the only way to achieve and maintain decentralization of control, which is a fundamental design principle.
- The collection of nodes that implement the bitcoin P2P protocol is referred to as the "bitcoin network." Other protocols, such as Stratum, are utilized for mining and lightweight or mobile wallets in addition to the bitcoin P2P protocol.
- Gateway routing servers access the bitcoin network via the bitcoin P2P protocol and then extend that network to nodes running other protocols to provide these additional protocols. Stratum servers, for instance, bridge the Stratum protocol to the bitcoin P2P protocol and connect Stratum mining nodes to the main bitcoin network via the Stratum protocol.
- The entire network that includes the bitcoin P2P protocol, pool-mining protocols, the Stratum protocol, and any other related protocols that connect the components of the bitcoin system is referred to as the "extended bitcoin network" by us.

2.23 NODES TYPES AND ROLES

Nodes in the bitcoin peer-to-peer network are all equal, but depending on the functionality they support, they may play different roles. A collection of functions is called a bitcoin node. Mining, wallet services, the blockchain database, routing, and mining Fig. 2.23.1 depicts a complete node with all four of these functions.

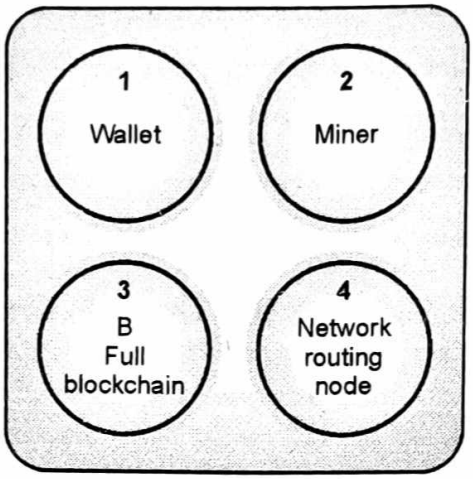

Fig. 2.23.1: A bitcoin network node with all four functions: wallet, miner, full blockchain database, and network routing

- To participate in the network, each node includes the routing function and may include additional functionality. All nodes discover and maintain connections to peers, validate transactions and blocks, and propagate them. An 4th circle labeled "Network Routing Node" denotes the routing function in the full-node example in Fig. 2.23.
- Full nodes are nodes that also keep a complete and current copy of the blockchain. Without any external reference, full nodes are able to independently and authoritatively verify any transaction.
- Simplified payment verification, or SPV, is a method by which some nodes verify transactions while only maintaining a portion of the blockchain. SPV or lightweight nodes are the names given to these nodes.
- A 3rd circle labeled "Full Blockchain" denotes the full-node blockchain database function in the figure's full-node example.
- Mining nodes use specialized hardware to solve the proof-of-work algorithm in order to compete for the creation of new blocks.
- Some mining hubs are likewise full hubs, keeping a full duplicate of the blockchain, while others are lightweight hubs taking part in pool mining and contingent upon a pool server to keep a full hub. The mining capability is displayed in the full hub as a 2nd circle named "Miners".

- As is typically the case with desktop bitcoin clients, user wallets may be a component of a complete node. SPV nodes are becoming increasingly common in user wallets, particularly those running on resource-constrained devices like smartphones. Fig. 2.23 depicts the wallet function as a 1st circle labeled "Wallet."
- There are servers and nodes running different protocols, such as lightweight client-access protocols and specialized mining pool protocols, in addition to the primary types of nodes on the bitcoin P2P protocol.

2.24 INCENTIVE BASED ENGINEERING

- At a specific time, the miner has an enormous number of exchanges that should be handled.
- Additionally, the system has already established the maximum block size, so a sender can only send the information that is absolutely necessary for the transaction. Consequently, to expedite the entire procedure and simplify Miner's work, send only pertinent data.
- The sender typically sends incentives (rewards) in the form of the cryptocurrency used in the transaction to encourage the Miner to make good decisions during the process. In most cases, it accounts for a relatively insignificant portion of the overall transaction.
- For instance, the sender might send the Miner an incentive worth approximately $15 in Bitcoin for a $250 Bitcoin transaction.
- Because the incentive is entirely dependent on the cryptocurrency's transaction fees, it has been steadily decreasing over the past few years. When Blockchain transaction volumes decrease, transaction fees also decrease, making it challenging to compensate miners for their labor and computational resources.
- Block Rewards A Bitcoin Miner receives a Block Reward, which is essentially a reward, for each Block for which he solves the intricate mathematical algorithm and adds transaction records to the Blockchain.
- The Block Reward simply halves itself once every 2016 blocks that are mined, or every four years, according to the halving process that is associated with Bitcoin.

Bitcoin every 4 years -

2012 - 25.00 BTC.

2016 - 12.50 BTC.

2020 - 6.25 BTC.

2.25 THE EXTENDED BITCOIN NETWORK

- There are between 7,000 and 10,000 listening nodes running various versions of the bitcoin reference client (Bitcoin Core) and a few hundred nodes running various other implementations of the bitcoin P2P protocol, such as BitcoinJ, Libbitcoin, and btcd, on the main bitcoin network, which runs the bitcoin P2P protocol.
- On the bitcoin P2P network, a small number of nodes compete with one another as mining nodes, validating transactions, and creating new blocks. Full-node clients based on the Bitcoin Core client are used by a number of large businesses to connect to the bitcoin network.
- These clients have full copies of the blockchain and a network node but no mining or wallet functions. Other services like exchanges, wallets, block explorers, and merchant payment processing can be built on top of these nodes, which act as network edge routers. The lengthy bitcoin network incorporates the organization running the bitcoin P2P convention, depicted prior, as well as hubs running particular conventions. There are a number of pool servers and protocol gateways that connect nodes running other protocols to the main bitcoin P2P network.

2.26 BITCOIN RELAY NETWORK

- While the bitcoin P2P network meets the needs of many different types of nodes, it is too slow for the specific needs of bitcoin mining nodes.
- In an effort to extend the blockchain and resolve the Proof-of-Work issue, Bitcoin miners are competing against one another in a race against time. Bitcoin miners participating in this competition must shorten the time it takes for a winning block to propagate before the next round begins. Network latency has a direct impact on profit margins in mining.
- A Bitcoin Relay Network is a network that tries to reduce the amount of time it takes for blocks to get from one miner to the next. Matt Corallo, the main developer of Bitcoin, came up with the original Bitcoin Relay Network in 2015 to make it possible for miners to quickly synchronize blocks with very low latency.
- The majority of miners and mining pools were connected by the network, which was made up of several specialized nodes worldwide hosted on the infrastructure of Amazon Web Services.
- The Fast Internet Bitcoin Relay Engine, or FIBRE, which was also developed by core developer Matt Corallo, took the place of the original Bitcoin Relay Network in 2016. Blocks are relayed within a network of nodes using the UDP-based relay network known as FIBRE. To further reduce the amount of data transmitted and the network latency, FIBRE uses compact block optimization.

- Bitcoin's peer-to-peer network cannot be replaced by relay networks. Instead, they are overlay networks that connect additional nodes with specialized requirements to one another.
- You still need small roads to connect to the freeways, just as freeways aren't replacements for rural roads but rather shortcuts between two points with a lot of traffic.

2.27 NETWORK DISCOVERY

- In order to participate, a new node must locate other bitcoin nodes on the network when it boots up. A new node must connect to at least one existing node on the network before this process can begin.
- It doesn't matter where other nodes are geographically; There is no geographical boundary to the topology of the bitcoin network. As a result, any and all Bitcoin nodes can be chosen at random.
- Nodes establish a TCP connection, typically to port 8333 (the bitcoin-specific port), or an alternative port if one is provided, in order to connect to a known peer.
- The node will initiate a "handshake" after establishing a connection by sending a version message containing basic identifying information, such as :

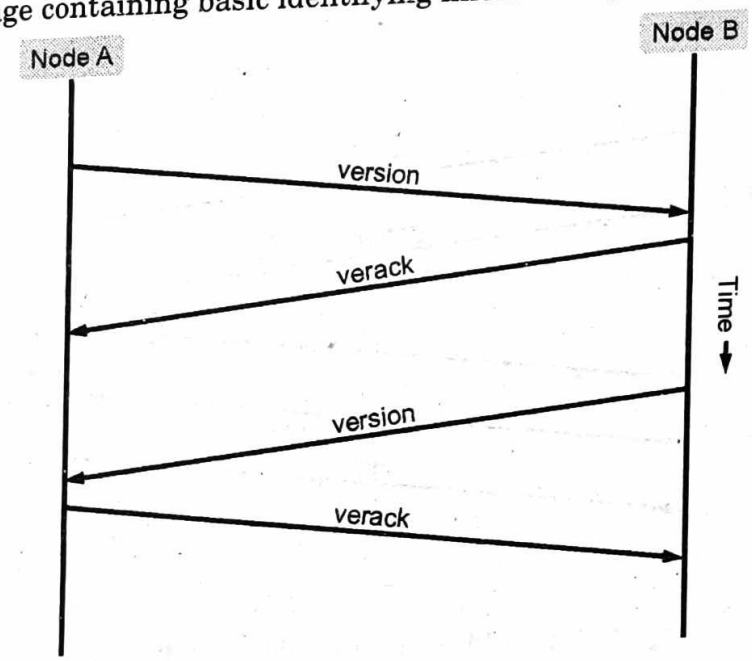

Fig. 2.27.1: The initial handshake between peers

- To acknowledge and establish a connection, the peer node responds with verack and optionally sends its own version message if it wishes to connect back as a peer.

- A new node finds peers in what way? Utilizing a number of "DNS seeds," which are DNS servers that provide a list of bitcoin node IP addresses, is the first approach. A static list of IP addresses for stable bitcoin listening nodes is provided by some of these DNS seeds.
- Customized BIND (Berkeley Internet Name Daemon) implementations that return a random subset from a list of bitcoin node addresses gathered by a crawler or a long-running bitcoin node make up some of the DNS seeds.
- Five distinct DNS seeds are listed in the Bitcoin Core client. For the initial bootstrapping process, the diversity of ownership and implementation of the various DNS seeds provides a high level of reliability. The option switch -dnsseed (set to 1 by default to use the DNS seed) controls the Bitcoin Core client's ability to utilize DNS seeds.
- The new node will send its neighbors an addr message containing its own IP address once one or more connections are made. The neighbors will, in turn, forward the addr message to their neighbors, ensuring that the newly connected node is better known and connected.
- Additionally, the newly connected node can request a list of peers' IP addresses from its neighbors by sending getaddr to them. A node can then advertise its existence on the network so that other nodes can locate it and connect to it. The address discovery protocol is shown in Fig. 2.27.2.

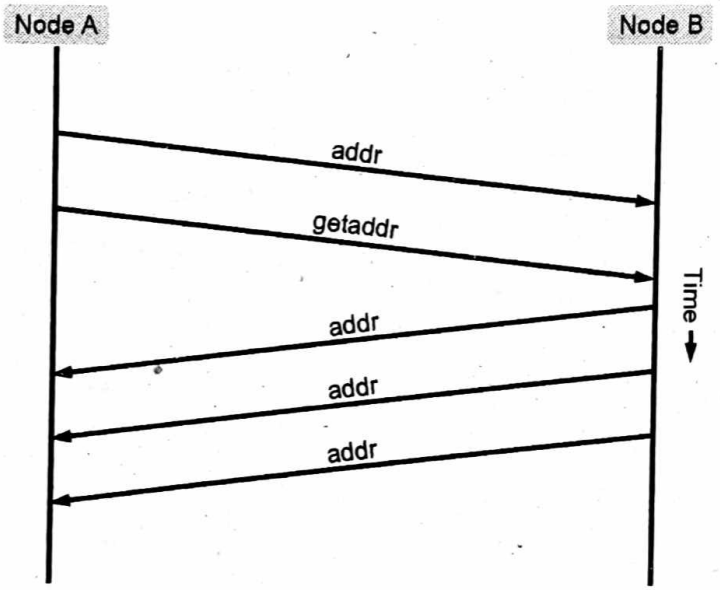

Fig. 2.27.2 : Address propagation and discovery

- In order for a node to establish distinct entry points into the bitcoin network, it must connect to a few distinct peers. Because paths are not reliable, nodes come and go, so the node has to keep finding new nodes as it loses connections and helps other nodes bootstrap.

- Bootstrapping only requires one connection because the first node can introduce itself to its peers and those peers can introduce themselves to others. Connecting to more than a few nodes is also wasteful of network resources and unnecessary.
- A node will remember its most recent successful peer connections after bootstrapping, allowing it to quickly reconnect to its previous peer network when rebooted. In the event that none of the previous companions answer its association demand, the hub can utilize the seed hubs to bootstrap once more.

2.28 FULL NODE

- Nodes that keep a full blockchain with all transactions are called full nodes. They probably ought to be referred to as "full blockchain nodes" to be more accurate. All bitcoin nodes were full nodes in the early days, and the Bitcoin Core client is now a full blockchain node.
- New bitcoin clients, on the other hand, have emerged in the last two years that do not operate as full blockchain clients but rather as lightweight clients.
- From the very first block (the genesis block) all the way up to the most recent known block in the network, full blockchain nodes independently build and verify a complete and up-to-date copy of the bitcoin blockchain that contains all of the transactions.
- Any transaction can be independently verified by a full blockchain node without the need for any other nodes or information sources. The full blockchain node depends on the network for information about new transaction blocks, which it then verifies and adds to its local copy of the blockchain.
- You get the full bitcoin experience when you run a full blockchain node: without the need to rely on or trust any other systems, independent verification of all transactions. Because the full blockchain requires more than 20 gigabytes of persistent storage (disk space), it is simple to determine whether you are running a full node.
- You are operating a full node if you require a lot of disk space and sync to the network takes two to three days. Complete independence and freedom from central authority come at a cost.
- Full blockchain bitcoin clients can be implemented using a variety of programming languages and software architectures in a few different ways. However, the reference client Bitcoin Core, also known as the Satoshi client, is the most widely used implementation.
- Different versions of Bitcoin Core are used by more than 90% of the nodes that make up the Bitcoin network. As previously mentioned, it is shown by the command getpeerinfo and identified as "Satoshi" in the sub-version string sent in the version message; such as /Satoshi:0.8.6/.

2.29 EXCHANGING "INVENTORY"

A full node will attempt to construct a complete blockchain as soon as it connects to peers. It only knows one block, the genesis block, which is statically embedded in the client software, if it is a brand-new node with no blockchain at all.

In order to synchronize with the network and reestablish the entire blockchain, the new node will need to download hundreds of thousands of blocks beginning with block #0 (the genesis block).

The version message, which contains BestHeight, a node's current blockchain height (number of blocks), is where the process of syncing the blockchain begins. A node will be able to compare the number of blocks in its own blockchain to the number of blocks in version messages sent by its peers.

A getblocks message containing the hash (or fingerprint) of the top block on their local blockchain will be exchanged between peer nodes. One of the peers will be able to determine that its own local blockchain is longer than its peer's by identifying the received hash as belonging to a block that is not at the top but rather an older block.

The peer with the longest blockchain has more blocks than the other node and is able to determine which blocks it needs to "catch up" with. Using an inv (inventory) message, it will identify the first 500 blocks and share and transmit their hashes.

The hub missing these blocks will then recover them, by giving a progression of getdata messages mentioning the full block information and recognizing the mentioned blocks utilizing the hashes from the inv message.

For instance, let's say a node only has the genesis block. The hashes of the next 500 blocks in the chain will then be sent to it by its peers in an inv message. All it will begin mentioning blocks from its associated peers, spreading the heap and guaranteeing that it overpowers no companion with demands.

The node verifies that it does not exceed a limit (MAX_BLOCKS_IN_TRANSIT_PER_PEER) by keeping track of the number of blocks that are "in transit" for each peer connection—that is, blocks that it has requested but has not yet received.

This way, if it needs a lot of blocks, it will only ask for more as previous requests are fulfilled, giving the peers control over how quickly updates happen and keeping the network from getting overwhelmed. The blockchain is updated with each new block that is received.

More blocks are requested and received as the local blockchain builds up, and the process continues until the node catches up to the rest of the network.

- Any time a node goes offline for any length of time, this process of comparing the local blockchain to the peers and retrieving any missing blocks takes place. A node starts by sending getblocks, receives an inv response, and begins downloading the blocks that are missing, regardless of how long it has been offline a few minutes and missing a few blocks, or a month and missing a few thousand blocks.

2.30 SIMPLIFIED PAYMENT VERIFICATION (SPV) NODES

- Not all hubs can store the full blockchain. Many bitcoin clients are made to run on devices that are limited in space and power, like smartphones, tablets, or embedded systems.
- Simplified Payment Verification (SPV) is used to enable these devices to function without storing the entire blockchain. SPV clients or lightweight clients are two names for these kinds of clients.
- The SPV node is becoming the most common type of bitcoin node, particularly for bitcoin wallets, as bitcoin adoption grows.

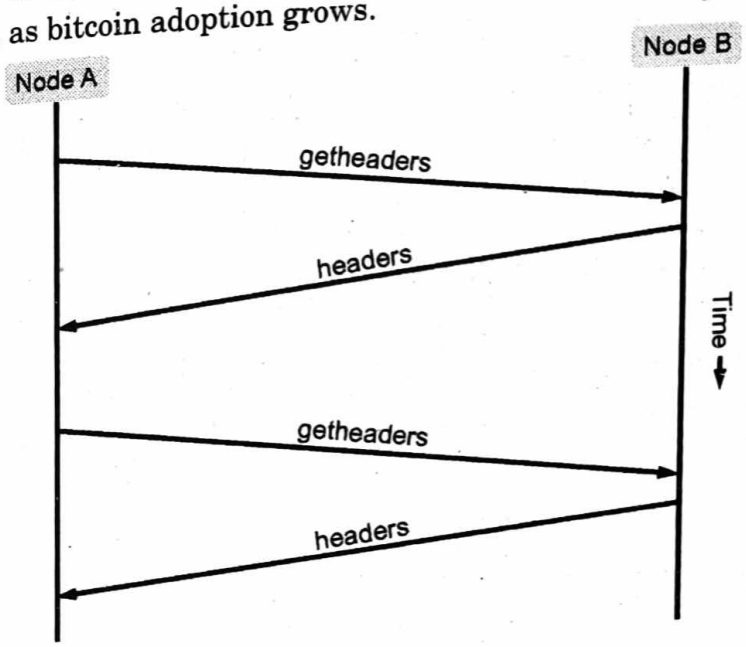

Fig. 2.30.1: SPV node synchronizing the block headers

- SPV nodes only download the block headers, not the transactions that are part of each block. Without transactions, the resulting chain of blocks is 1,000 times smaller than the entire blockchain.
- Because they do not know about all of the transactions that take place on the network, SPV nodes are unable to create a comprehensive picture of all of the UTXOs that are available for spending. SPV nodes use a slightly different method to verify transactions, relying on peers to provide partial views of relevant blockchain parts on demand

- Instead of looking at a transaction's height, simplified payment verification looks at its depth in the blockchain. An SPV node will verify the chain of all blocks (but not all transactions) and link that chain to the transaction of interest, whereas a full blockchain node will construct a fully verified chain that reaches down the blockchain (back in time) all the way to the genesis block.
- A full node, for instance, links all 300,000 blocks down to the genesis block to examine a transaction in block 300,000 and creates a full database of UTXO, verifying that the UTXO has not been spent.
- The UTXO cannot be verified by an SPV node as being unspent. Instead, the SPV node will use a merkle path to connect the transaction to the block that holds it (see Merkle Trees).
- The SPV node then verifies the transaction by determining its depth under blocks 300,006 to 300,001 after waiting until it sees the six blocks 300,001 through 300,006 stacked on top of the block containing the transaction.
- By proxy, the transaction was not a double-spend because other network nodes accepted block 300,000 and then completed the necessary work to generate six additional blocks on top of it.

2.31 SPV NODES AND PRIVACY

- An SPV node is able to specify a search pattern for transactions that can be tuned toward precision or privacy using Bloom filters, which serve this purpose.
- Accurate results can be obtained with a more specific bloom filter, but it will reveal the user's wallet addresses. While the node will be able to maintain better privacy, a bloom filter that is less specific will produce more data about more transactions, many of which are irrelevant to it.
- A bit field, or array of N binary digits, and a variable number of M hash functions are used to create bloom filters. The hash functions are made to always give an output that is between 1 and N, which is the number of binary digits in the array.
- Since the hash functions are generated deterministically, any node that implements a bloom filter will always use the same hash functions and obtain the same output for a particular input.
- The bloom filter's accuracy and privacy can be altered by selecting different lengths (N) of bloom filters and different numbers (M) of hash functions.

2.32 TRANSACTION POOLS

- The memory pool, mempool, or transaction pool is a temporary list of unconfirmed transactions that is kept by nearly every bitcoin network node. This pool is used by nodes to keep track of transactions that the network knows about but haven't added to the blockchain yet.
- A node that holds a user's wallet, for instance, will make use of the transaction pool to keep track of incoming payments that have been received on the network but have not yet been confirmed to the user's wallet.
- After being received and checked, transactions are added to the transaction pool and sent to nearby nodes to spread throughout the network.
- A separate pool of abandoned transactions is also kept up by some node implementations. The orphan transaction will be temporarily stored in the orphan pool until the parent transaction arrives if a transaction's inputs refer to a transaction that is not yet known, such as a missing parent.
- The orphan pool is checked for any orphans that refer to this transaction's outputs (its children) when a transaction is added to the pool. After that, any orphans that match are checked. They are added to the transaction pool and taken out of the orphan pool if they are legitimate, completing the chain that began with the parent transaction.
- The procedure is repeated in a recursive search for any additional descendants in light of the newly added transaction, which is no longer an orphan, until there are no more descendants.
- By reuniting orphans with their parents further down the chain, the arrival of a parent transaction initiates a cascade reconstruction of an entire chain of interdependent transactions.
- When implemented, neither the transaction pool nor the orphan pool are saved on persistent storage; instead, they are saved in local memory. rather, they are dynamically filled with messages from the network that come in. Both pools are empty when a node starts and are gradually filled with new network transactions.

2.33 BLOCKCHAIN FORK

- Participants on the network must be able to agree on the shared state of the blockchain (shared public ledger, blocks, and the blockchain protocol) due to the decentralized nature of public blockchains like Bitcoin and Ethereum.
- A single blockchain with verified data (transactions) that the network claims to be accurate is created when all network nodes reach a unanimous consensus. However, there are a lot of times when the network's nodes cannot agree on the blockchain's future state.

- Forks are the point at which the ideal "single" chain of blocks is divided into two or more chains that are all valid, like a tuning fork used in experimental science.
- Based on the blockchain protocol's backwards compatibility and the time instant at which a new block is mined, there are three types of forks that can occur as a result.
- The following are examples :

(1) **Soft Forks :** when the blockchain protocol is modified in a way that is compatible with previous versions,

(2) **Hard Fork :** when the blockchain protocol is modified in a way that is not compatible with previous versions

(3) **Temporary Fork :** when two miners simultaneously mine a new block.

(1) Soft Fork

At the point when there is an adjustment of the product that sudden spikes in demand for the hubs (better called as 'full hubs') to work as an organization member, the change is with the end goal that the new blocks mined based on new standards (in the Blockchain convention) are likewise viewed as substantial by the old rendition of the product. Backwards compatibility is another name for this capability.

Example

The Bitcoin organization's SegWit update added another class of addresses (Bech32). However, the existing P2SH addresses were not invalidated by this. With a node with a Bech32 address, a full node could carry out a valid transaction.

(2) Hard Fork

- The term "hard fork" refers to a situation in which the software that runs on all nodes in order to participate in the network is altered to the point where the old version of the software no longer considers the new blocks that are mined in accordance with the new rules in the Blockchain protocol to be valid.
- When a hard fork occurs, a new currency (with a valid original currency) emerges, like Ethereum (original: New Ethereum: Bitcoin (original: Ethereum Classic) and Ethereum New Bitcoin: Cash in Bitcoin). To ensure that there is no material loss, an equivalent amount of currency is distributed to all nodes that choose to upgrade their software. These hard forks frequently provoke controversy (community disputes).
- The full node makes the final decision about whether or not to join a particular chain. If a node decides to join the new chain, its software must be updated to allow for more recent transactions to be accepted, whereas nodes that do not upgrade their software continue to function as before.

Example

- The consensus protocol will shift from Proof of Work (PoW) to Proof of Stake (PoS) with the new Casper update to the Ethereum Blockchain.
- The brand-new consensus protocol will be utilized by the nodes that install the Casper update. Full nodes that choose not to install the Casper update will become incompatible with full nodes that choose to do so.

(3) Temporary Fork / Accidental Fork

- At the point when various Miners mine another block at almost a similar time, the whole organization may not settle on the decision of the new block. Some people are willing to accept the block that was mined by one party, which will start a new chain of blocks from that point on, while others are able to agree on the other blocks that are available.
- Because it takes a certain amount of time for the information to spread throughout the entire blockchain network, this circumstance can arise. As a result, there may be divergent opinions regarding the order in which the events occurred. The block height of two or more blocks in this fork is the same.
- The majority of full nodes choose the other chain to add new blocks to and sync with, so temporary forks eventually resolve themselves when one chain dies out (gets orphaned).

Example

Mining a block by multiple parties at nearly the same time is a common cause of temporary forks, which occur more frequently than not.

The events that led to a fork in the blockchain :

(1) **Adding new functionality :** The Blockchain code is regularly updated. People from all over the world work on developing the majority of public blockchains because they are open source. When the time is right, new versions are made, issues are fixed, and improvements are made.

(2) **Improve security :** When compared to conventional forms of currency (notes, coins, and checks), blockchain (and cryptocurrency on top of it) is a relatively new technology, and research to fully comprehend it is still ongoing. As a result, security issues that arise are fixed by bumping versions and releasing updates.

(3) **Reverse transactions :** If it turns out that the community was hacked and malicious, they can actually cancel all of the transactions for a given time period.

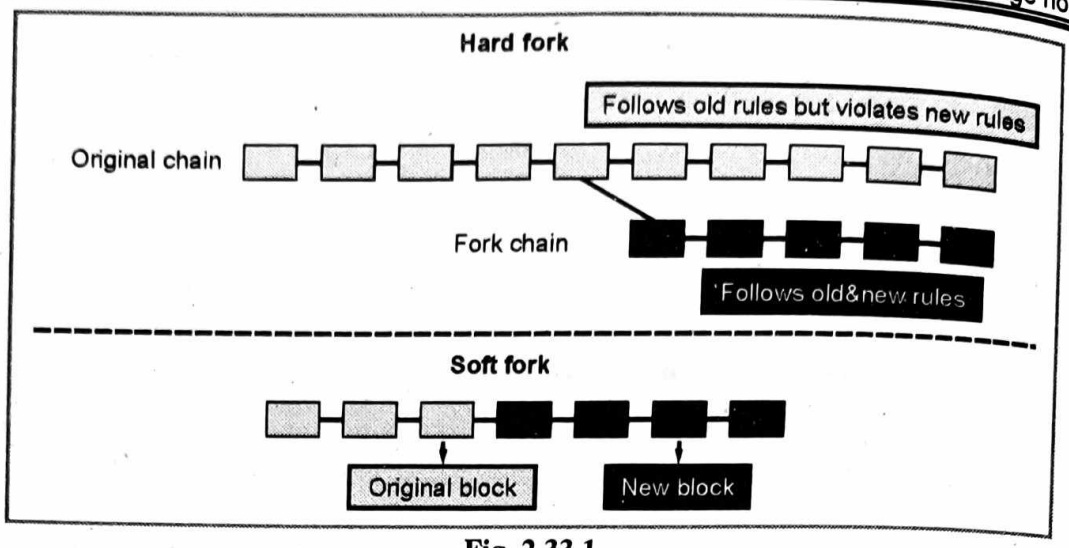

Fig. 2.33.1

2.34 BITCOIN TESTNET

- The test blockchain, network, and currency that is utilized for testing is known as the testnet. A live P2P network with wallets, test bitcoins (testnet coins), mining, and all other mainnet features is the testnet.
- In reality, there are only two differences: Since testnet coins are intended to be worthless, their mining difficulty should be low enough that anyone can mine them without difficulty.
- Test any software that will be used in production on Bitcoin's mainnet first on the testnet with test coins. This safeguards the network from bugs' unintended behavior and the developers from monetary losses.
- It's not easy, though, to keep the coins worthless and make mining simple. Despite developers' pleas, some individuals mine on the testnet with cutting-edge mining hardware like GPUs and ASICs.
- Because of this, mining with a CPU becomes impossible, making it more difficult to obtain test coins, and eventually people begin to value them to ensure that they are not worthless. Consequently, the testnet must occasionally be scrapped and restarted from a brand-new genesis block, resetting the difficulty.
- Using testnet Bitcoin Core, like nearly all other bitcoin software, allows you to mine testnet coins and run a full testnet node in addition to having full support for operation on testnet rather than the mainnet.
- The testnet switch is used to start Bitcoin Core on testnet rather than mainnet:

  $ bitcoind -testnet

- You should be able to see in the logs that a new blockchain is being created by bitcoind in the testnet3 subdirectory of the default bitcoind directory:

  Bitcoind : Using data directory /home/username/.bitcoin/testnet3

- The bitcoin-cli command-line tool is used to connect to bitcoind; however, you must also change it to testnet mode :

```
$ bitcoin-cli -testnet getinfo
{
  "version": 130200,
  "protocolversion": 70015,
  "walletversion": 130000,
  "balance": 0.00000000,
  "blocks": 416,
  "timeoffset": 0,
  "connections": 3,
  "proxy": "",
  "difficulty": 1,
  "testnet": true,
  "keypoololdest": 1484801486,
  "keypoolsize": 100,
  "paytxfee": 0.00000000,
  "relayfee": 0.00001000,
  "errors": ""
}
```

2.35 BASICS OF BITCOIN FORENSICS : ANALYSIS OF ADDRESS AND WALLET

- Once you find an address, you can use it to learn a lot about its history and infer additional data by looking at the metadata that goes along with it.
- Simply browsing to one of the primary blockchain viewers for the currency the address refers to yields access to a wealth of information.
- You are free to use the recovered Bitcoin Core wallet file in any way that the original users were able to. You have full admittance to their private and public keys and the entirety of their past exchanges.

2.36 CLUSTERING OF ADDRESSES FOLLOWING MONEY

- Due to the pseudo anonymity offered by decentralized cryptocurrencies like Bitcoin, it has become extremely challenging for law enforcement to identify users, identify suspicious activities, and obtain transaction records for criminals.
- By linking addresses controlled by the same user based on information from the blockchain, such as transaction graphs, address clustering aims to end this pseudo anonymity. For the purpose of clustering Bitcoin addresses, there are already two widely used heuristics. One is based on transactions' multiple input addresses.

Chapter Ends...

❑❑❑