Blockchain Engineering Playbook 2025

Rohas Nagpal

Version 1.7 dated 2 January 2025

This book is part of the Official Courseware of the
Blockchain Engineering Program
conducted by Rohas Nagpal.

How Blockchain Engineering skills will help you

1. **Software Developers:** To integrate blockchain into existing systems.

2. **Network Engineers:** To understand and implement decentralized networks.

3. **Cloud Architects:** To design blockchain-as-a-service (BaaS) solutions.

4. **Cybersecurity Analysts:** To leverage blockchain for enhanced security features.

5. **IT Project Managers:** To manage blockchain projects effectively.

6. **System Administrators:** To oversee blockchain-based systems and networks.

7. **Data Scientists:** To analyze blockchain data for insights.

8. **DevOps Engineers:** To deploy and manage blockchain applications.

Contents

About the Author

Rohas Nagpal is the **Chief Blockchain Architect** of Hybrid Finance Blockchain (HYFI).

In 2016, he co-founded **BankChain**, a community of **37 banks** + IBM, Microsoft, and Intel. He has also been a consultant for the **Reserve Bank Innovation Hub** for preparing a Whitepaper on Central Bank Digital Currency (CBDC). He authors the Crypto Profits premium newsletter and conducts the Tokenization Expert Program, and the Blockchain Engineering Program.

This book is part of the courseware of the free Blockchain Engineering Program conducted by Rohas Nagpal.

The information in my documents, social media networks, websites, and videos is for general information only and should not be taken as constituting professional advice from me.

I am not liable for any loss caused, whether due to negligence or otherwise arising from the use of, or reliance on, the information provided directly or indirectly.

I link to external resources for your convenience. I am selective about them but I don't endorse them.

No investigation has been made of common-law trademark rights in any word. Words that are known to have current trademark registrations are shown with an initial capital and are also identified as trademarks.

Companion Spreadsheet is here

89 Blockchain Tools & Resources

Something missing? Email me on rohasnagpal@gmail.com

#	Category	Name	Brief Description	Website
1	Developer Tools	Alchemy Account Kit	Framework to enable email / social login, gas sponsorship, and batched transactions in web3 apps.	https://www.alchemy.com/account-kit
2	Developer Tools	Alchemy Supernode	Web3 development platform for building and scaling dApps.	https://www.alchemy.com/supernode
3	Developer Tools	Alchemy Token API	Complete token data on all EVM-supported chains.	https://www.alchemy.com/token-api
4	Developer Tools	Alchemy Transaction Simulation	Preview how transactions will behave on-chain.	https://www.alchemy.com/transaction-simulation
5	Developer Tools	Alchemy Transfers API	Get all historical transfers for an address or a contract, capturing internal, external & token transfers.	https://www.alchemy.com/transfers-api
6	Developer Tools	Alchemy Webhooks	Real-time notifications	https://www.alchemy.com/webhooks
7	Developer Tools	API3	Blockchain-native, decentralized APIs.	https://api3.org/
8	Developer Tools	Aragon App	No-code platform to launch and manage DAOs.	https://aragon.org/aragon-app
9	Developer Tools	Aragon OSx	Smart contract framework to build custom DAOs.	https://aragon.org/aragonosx
10	Developer Tools	Drizzle	Front-end libraries for creating dApp user interfaces.	https://trufflesuite.com/drizzle/
11	Developer Tools	Ganache	1-click Ethereum Blockchain for testing & development.	https://trufflesuite.com/ganache/
12	Developer Tools	Hardhat	Compile & run contracts on this Ethereum development environmentand to get Solidity stack traces, cor	https://hardhat.org/
13	Developer Tools	Hyperledger Bevel	Accelerator for deploying production-ready distributed networks across cloud providers.	https://www.hyperledger.org/projects/bevel
14	Developer Tools	Hyperledger Cacti	Tool enabling the integration of different blockchains.	https://www.hyperledger.org/projects/cacti
15	Developer Tools	Hyperledger Caliper	Tool for measuring blockchain performance with a set of predefined use cases.	https://www.hyperledger.org/projects/caliper
16	Developer Tools	Hyperledger Cello	Console for managing blockchains running on bare-metal, virtual machine & container platforms.	https://www.hyperledger.org/projects/cello
17	Developer Tools	Hyperledger Firefly	Complete stack for building & scaling secure Web3 applications.	https://www.hyperledger.org/projects/firefly
18	Developer Tools	Hyperledger Solang	Compiler, written in rust, for compiling Solidity for Solana and Substrate.	https://www.hyperledger.org/projects/solang
19	Developer Tools	Moralis Blockchain API	Block data, transactions, logs, raw & decoded data.	https://moralis.io/api/blockchain/
20	Developer Tools	Moralis DeFi API	Liquidity reserves and pair data across multiple blockchains.	https://moralis.io/api/defi/
21	Developer Tools	Moralis NFT API	Cross chain NFT Transfers, Prices & Metadata	https://moralis.io/api/nft/
22	Developer Tools	Moralis Price API	Crypto Price API.	https://moralis.io/api/price/
23	Developer Tools	Moralis ERC20 Token API	Real-time token prices, wallet balances, transfers, and liquidity.	https://moralis.io/api/token/
24	Developer Tools	Moralis Wallet	API for integrating wallet functionalities into Web3 dapps.	https://moralis.io/api/wallet/
25	Developer Tools	MythX	Detect security vulnerabilities in your Ethereum smart contracts.	https://mythx.io/
26	Developer Tools	OpenZeppelin Defender	Secure operations platform for smart contracts.	https://www.openzeppelin.com/defender
27	Developer Tools	OpenZeppelin Contracts Wizard	Interactive smart contract generator for Solidity and Cairo.	https://wizard.openzeppelin.com/
28	Developer Tools	Push Protocol	Cross-chain notifications & messaging for dapps & wallets.	https://push.org/
29	Developer Tools	Remix IDE	No-setup tool with a GUI for developing, deploying, debugging, and testing EVM-compatible smart cont	https://remix-project.org/
30	Developer Tools	Tatum Blockchain API	Indexed Block Data.	https://tatum.io/blockchain-api
31	Developer Tools	Tatum Metamask	Integrate MetaMask into a website or mobile app in seconds.	https://tatum.io/metamask-integration
32	Developer Tools	Tatum NFT	Tools for building NFT apps.	https://tatum.io/nfts
33	Developer Tools	Tatum Notifications	Fast Web3 notifications.	https://tatum.io/notification
34	Developer Tools	Tatum Wallets	Back-end for crypto wallet applications.	https://tatum.io/wallets
35	Developer Tools	The Graph	Indexing and querying data from blockchains.	https://thegraph.com/en/
36	Developer Tools	Truffle	Suite of tools for smart contract development.	https://trufflesuite.com/
37	Data & News	Coin Market Cal	Upcoming current events on different coins.	https://coinmarketcal.com/en/
38	Data & News	Crypto ATM Map	Find Crypto ATMs across the world.	https://coinatmradar.com/
39	Data & News	Crypto Miso	Github commit history of 200+ cryptocurrencies.	https://www.cryptomiso.com/
40	Data & News	CryptoPanic	Crypto news aggregator.	https://cryptopanic.com/
41	Data & News	REKT Database	Database of the Top Crypto Hacks.	https://de.fi/rekt-database
42	Explorers	Atlas	Blockchain search engine.	https://coinmetrics.io/atlas/
43	Explorers	Blockchair	Explore data stored on 40 blockchains.	https://blockchair.com/
44	Explorers	Etherscan	Block Explorer and Analytics Platform for Ethereum.	https://etherscan.io/
45	Frameworks	Hyperledger Besu	Ethereum client, optimized for enterprise environments, for permissioned networks.	https://www.hyperledger.org/projects/besu

01

Blockchain Basics

A blockchain:

- is a linear, chronological structure
- consists of blocks of data (transactions) that are chained together.

Blockchains are **Internets of Value.**

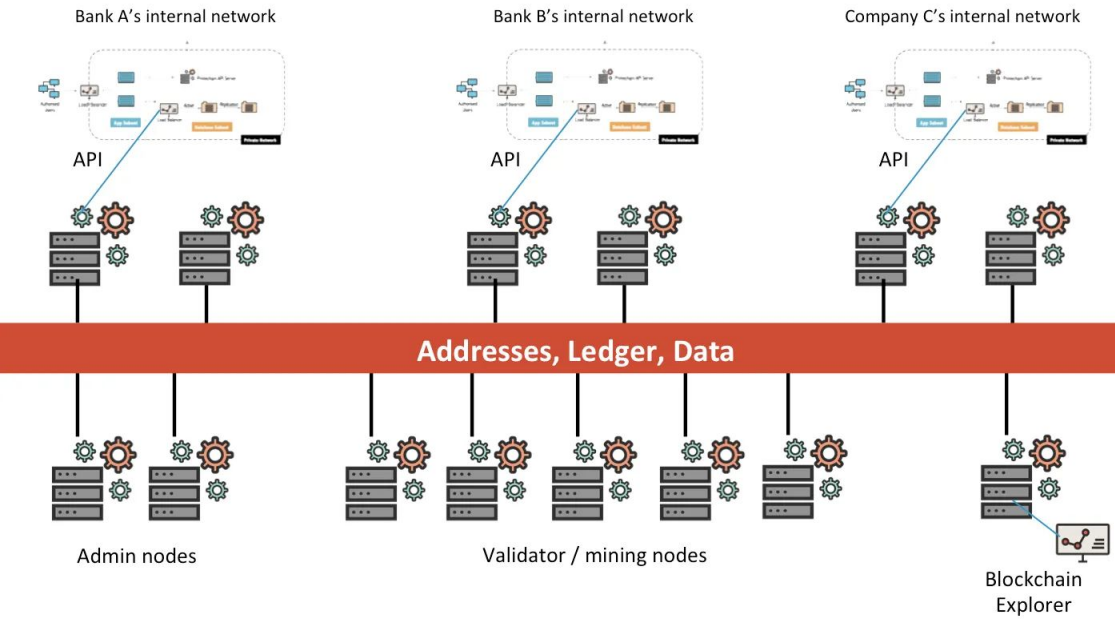
The conventional internet uses protocols like TCP/IP and SMTP to move data across the globe in seconds (email video pdf, text, etc).

Blockchains enable the **movement of value** across the world in seconds.

This value can be crypto assets like Bitcoin & Monero or tokenized versions of real-world assets like airplanes, carbon credits, real estate, etc.

Tokenization of real-world assets is the most important Blockchain use case. It is explained in depth later in this book.

Blockchains increase transparency and reduce the need for intermediaries in financial use cases.



Conceptual Image of a Blockchain Network

While Blockchains have a linear chronological structure, Distributed Ledger Technologies (DLTs) can have different structures, such as:

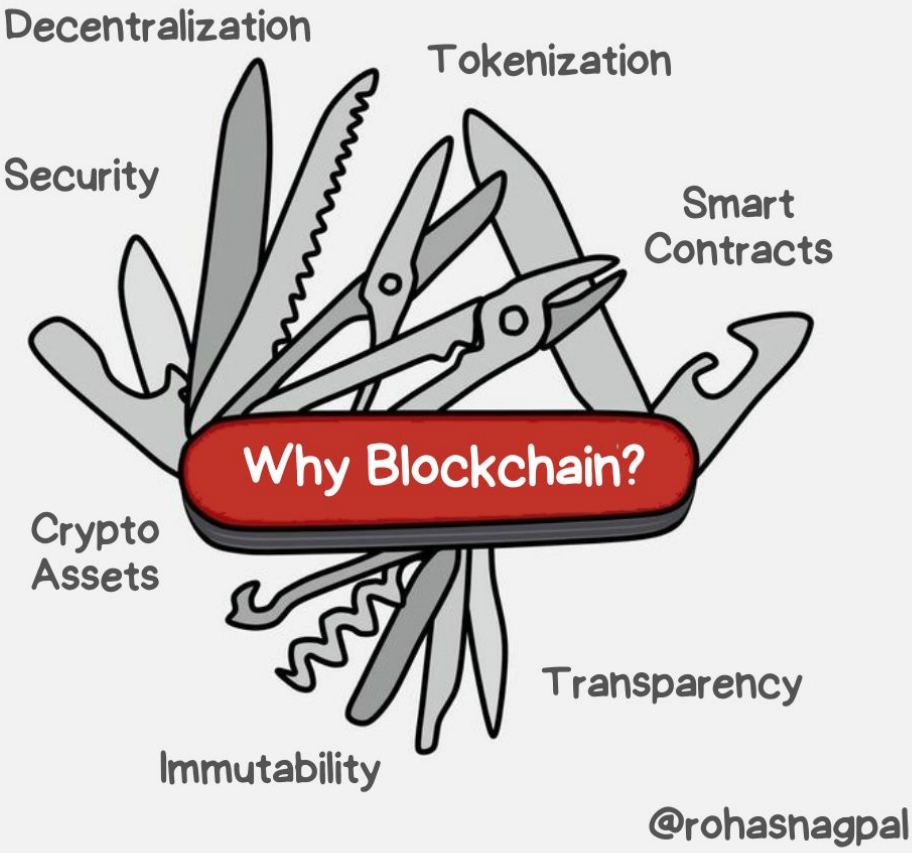
- Graph e.g. IOTA
- Mesh e.g. Holochain
- Tree e.g. Corda

DLTs are suited for multiple use cases such as:

- Supply Chain Management
- Identity and Verification
- Healthcare
- Government and Public services

All Blockchains are DLTs,
but all DLTs are not Blockchains.


Blockchain Mindmap by Rohas Nagpal
Use cases
Cross-border remittance
Cryptocurrencies
Decentralized Finance
Decentralized Autonomous Organizations
Decentralized Identity
Supply chain
Tokenization of real-world assets
Blockchain Consensus Mechanisms
Chain-based Proof of Work
Chain-based Proof of Stake
Chain-based Proof of Capacity / Space
Chain-based Hybrid models
Chain-based Proof of Burn
Chain-based Trusted computing algorithms
Chain-based DAG
Blockchain Forks
Hard Fork
Soft Fork
Blockchain Frameworks
Multichain
Hyperledger
Blockchain Bridges
Trusted Bridge
Trustless Bridge
Blockchain Metrics
Fault tolerance threshold
Finality
Scalability
Throughput
Layers of a Blockchain Network
Infrastructure or Physical Layer
Logical Layer
Application Layer
User Interface Layer
Record-keeping models
Account/Balance Model
Hybrid
UTXO Model (Unspent Transaction Output)
Blockchain Nodes
Archival Full Nodes
Pruned Full Node
Masternodes
Light nodes
Cold nodes
Lightning nodes
Types of Blockchains (Governance)
Permission-less blockchains
Permissioned blockchains
Federated blockchains
Blockchain Wallets
Paper Wallets
Hardware Wallets
Software Wallets
Exchange Wallets
Hot Wallets
Cold wallets
Types of Blockchains (Tech)
Layer-0 blockchains
Layer-1 blockchains
Layer-2 blockchains
EVM-compatible chains


Decentralization
Tokenization
Security
Smart Contracts
Why Blockchain?
Crypto Assets
Transparency
Immutability
@rohasnagpal

1.1 Blockchain Nodes

A blockchain is a **transaction database** shared by all nodes participating in a network.

A node is a computer that runs the blockchain's software to validate & store the complete history of transactions.

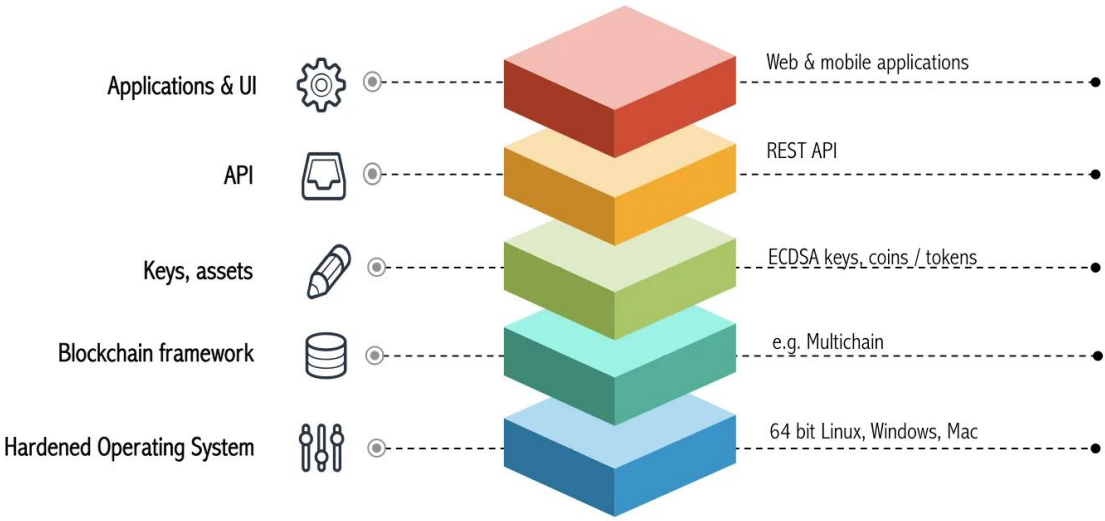
A full copy of a blockchain contains every transaction ever executed in the native coin (e.g. ETH) and tokens (e.g. BAT) created on the blockchain.

Transactions are bundled in **blocks**.

Each block contains a record of some / all recent transactions. Each block also contains a reference to the block that came immediately before it.

A blockchain node is a **computer** that runs the blockchain's software to validate & store the complete history of transactions on the network.

Blockchain nodes **constantly exchange information** on new transactions & blocks.



A conceptual overview of a Blockchain Node

Archival Full Nodes

They are the most important. They host the entire blockchain, validate blocks & maintain consensus.

Pruned Full Node

A pruned full node first downloads the entire blockchain and then deletes blocks beginning with the oldest block till it holds only the most recent transactions up to the size limit set by the operator.

Master nodes

Users run masternodes to earn network rewards. Some amount of native tokens have to be "locked" by masternode operators.

Light nodes

A light node does not hold the full copy of the blockchain. It saves download time & storage space by only downloading block headers.

Cold nodes

They are used for signing transactions offline and storing private keys away from the network.

Lightning nodes

They reduce the load on the network by enabling off-chain transactions. These nodes enable faster and cheaper transactions.

1.2 Mining

In a Proof of Work blockchain like Bitcoin, each block also contains an answer to a difficult-to-solve **mathematical puzzle**.

Mining is the process of competing to be the next to find the answer that "solves" the current block.

New blocks cannot be submitted to the network without the correct answer.

The mathematical problem in each block is very difficult to solve, but once a valid solution is found, it is very easy to verify.

Every block contains a **hash of the previous block**.

This creates a chain of blocks from the first (genesis) block to the current block.

It is extremely difficult to modify a block once it has been in the blockchain for some time because every block after it would also have to be generated again.

1.3 Layers of a Blockchain Network

Layer 0: Infrastructure or Physical Layer

This consists of the hardware, software, and networks that enable the functioning of the blockchain.

Layer 1: Logical Layer

This consists of the protocols, rules, and standards that define the functioning of the blockchain.

Layer 2: Application Layer

This consists of the protocols, applications, and services that build on top of the basic building blocks provided by layer 1.

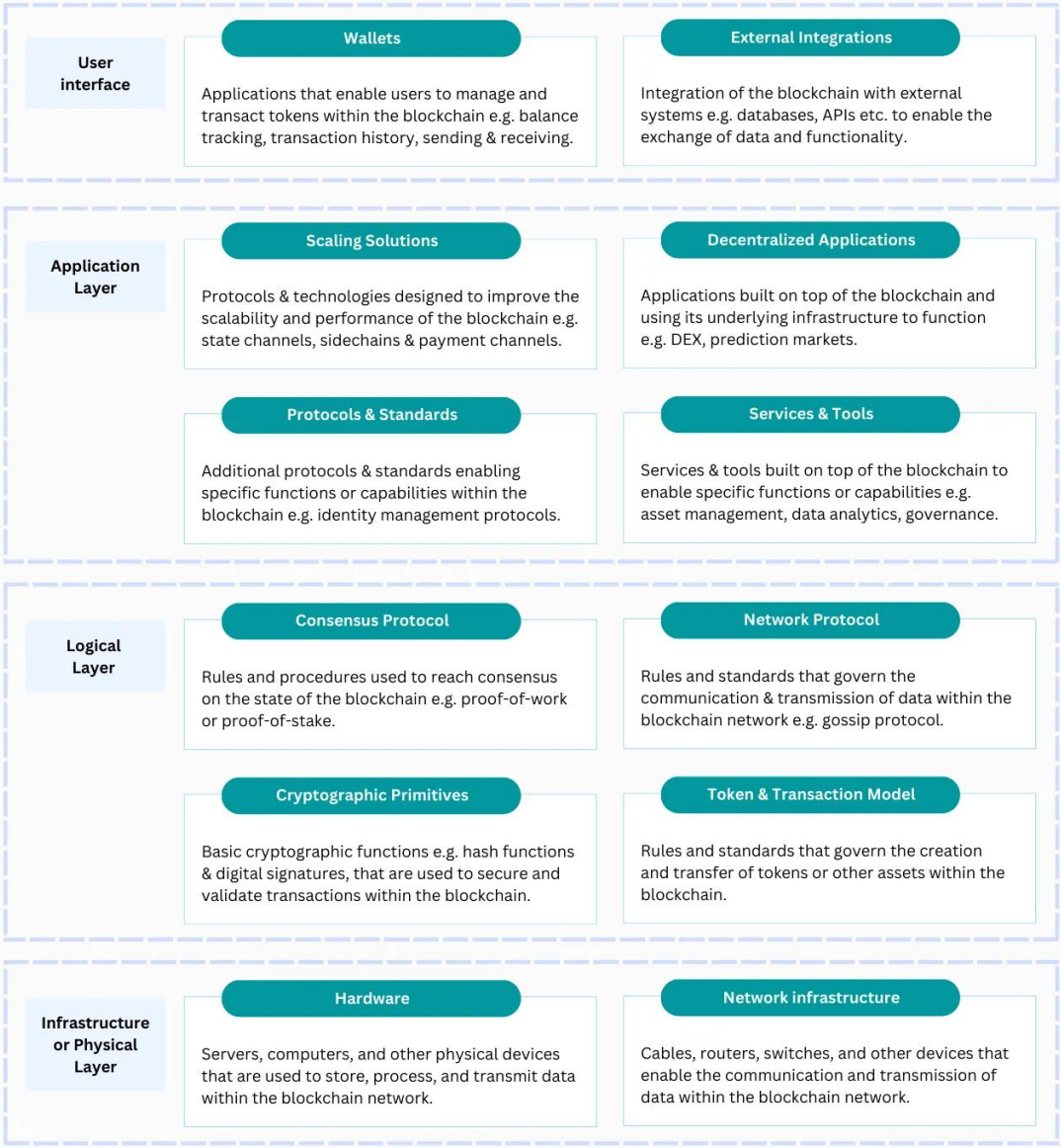
Layer 3: User Interface Layer

This consists of the interfaces, applications, and services that enable users to interact with the blockchain and access its underlying functionality.

The infographic below depicts the 4 layers of a Blockchain Network.

Layers of a Blockchain Network

Something missing? Spotted an error? Email me: rohasnagpal@gmail.com



Connect with me @rohasnagpal on LinkedIn, Instagram, YouTube, Twitter, or Gmail.

1.4 Types of Blockchains

Layer-0 Blockchains

Layer-0 blockchains enable developers to create custom blockchains.

Example: Polkadot enables the creation of sovereign blockchains with their own tokens.

These parachains plug into a single base platform called the Relay Chain which is responsible for shared security, consensus & cross-chain interoperability.

Other examples of L0 are Cosmos and Horizen.

L0 blockchains are different from Blockchain Frameworks like Multichain and Hyperledger Fabric.

Frameworks are open-source software that enable developers to create custom blockchains.

Remember: Layer-0 blockchains are networks while Frameworks are software programs.

Layer-1 Blockchains

Layer-1 blockchains validate & execute transactions without the need for any external network. *Examples:* Bitcoin, and Ethereum Mainnet.

Layer-2 Blockchains

Layer-2 blockchains are "sidechains" built on top of Layer-1 blockchains. The underlying Layer-1 (e.g. Ethereum Mainnet) provides decentralization & security, while the Layer-2 (e.g. Polygon PoS) provides scalability.

Permission-less blockchains

Anyone can participate on public / permission-less blockchains without restrictions. *Examples:* Bitcoin, Litecoin, Ethereum.

Permissioned blockchains

Various controls can be set in a private / permissioned blockchain.

Example: Hybrid Finance Blockchain (HYFI) where permissions such as connect, send, receive, issue, create, mine, activate, and admin can be set.

Federated blockchains

In a federated / consortium blockchain network, the validation process is controlled by a select set of nodes.

EVM-compatible chains

Ethereum Virtual Machine (EVM) is "the environment in which all 'Ethereum' accounts and smart contracts live".

Smart contracts are programs that run automatically when some pre-defined conditions are met.

EVM compatibility is the ability to write & deploy smart contract code that are:

1. compatible with EVM, and
2. can be recognized by Ethereum nodes.

Examples of EVM-compatible blockchains are Avalanche, Binance Smart Chain, Fantom, and Polygon.

1.5 Blockchain Consensus Mechanisms

Consensus algorithms are the **heart of blockchains.**

They enable network participants to agree on the contents of a blockchain in a distributed and trustless manner.

Today there are over 75 consensus algorithms including:

Proof-of-Work (PoW)
This was the world's first consensus algorithm. Miners "solve" mathematical puzzles by investing in electricity and computational power e.g. Bitcoin.

Proof of Stake (PoS)
Holders "lock" a number of coins as a "stake" and are randomly assigned validation rights for a new block e.g. Algorand.

Hybrid PoW / PoS
This brings together the security of Proof of work and the governance & energy efficiency of Proof-of-Stake e.g. Decred.

Proof-of-Work-Time (PoWT)

This features a variable blocktime that scales with mining power. The blockchain speeds up with power increases. This mechanism scales the blockchain well and increases transaction speed with power.

Proof of Meaningful Work (PoMW)

The energy invested by miners' is used for calculations that benefit public scientific research projects (e.g. medical research for cures, chemical research, and astrophysical simulations).

Proof of Elapsed Time (PoET)

Each node goes to sleep for a random "wait time" generated by the network. The node with the shortest wait time wakes up first and commits a new block e.g. Hyperledger Sawtooth.

Proof-of-replication (PoRep)

Storage miners prove 2 things:

- that they are using space to store replicas of data,
- that the data can easily be accessed.

They get rewards in exchange for their storage space e.g. Filecoin.

Delegated Proof of Stake (DPoS)

Holders "lock" a number of coins as a "stake" but outsource validation to "delegates" selected based on reputation and trustworthiness e.g. Bitshares.

Proof-of-Spacetime (PoSt)

Randomly selected miners prove that they have been physically storing data for a certain period of time e.g. Filecoin.

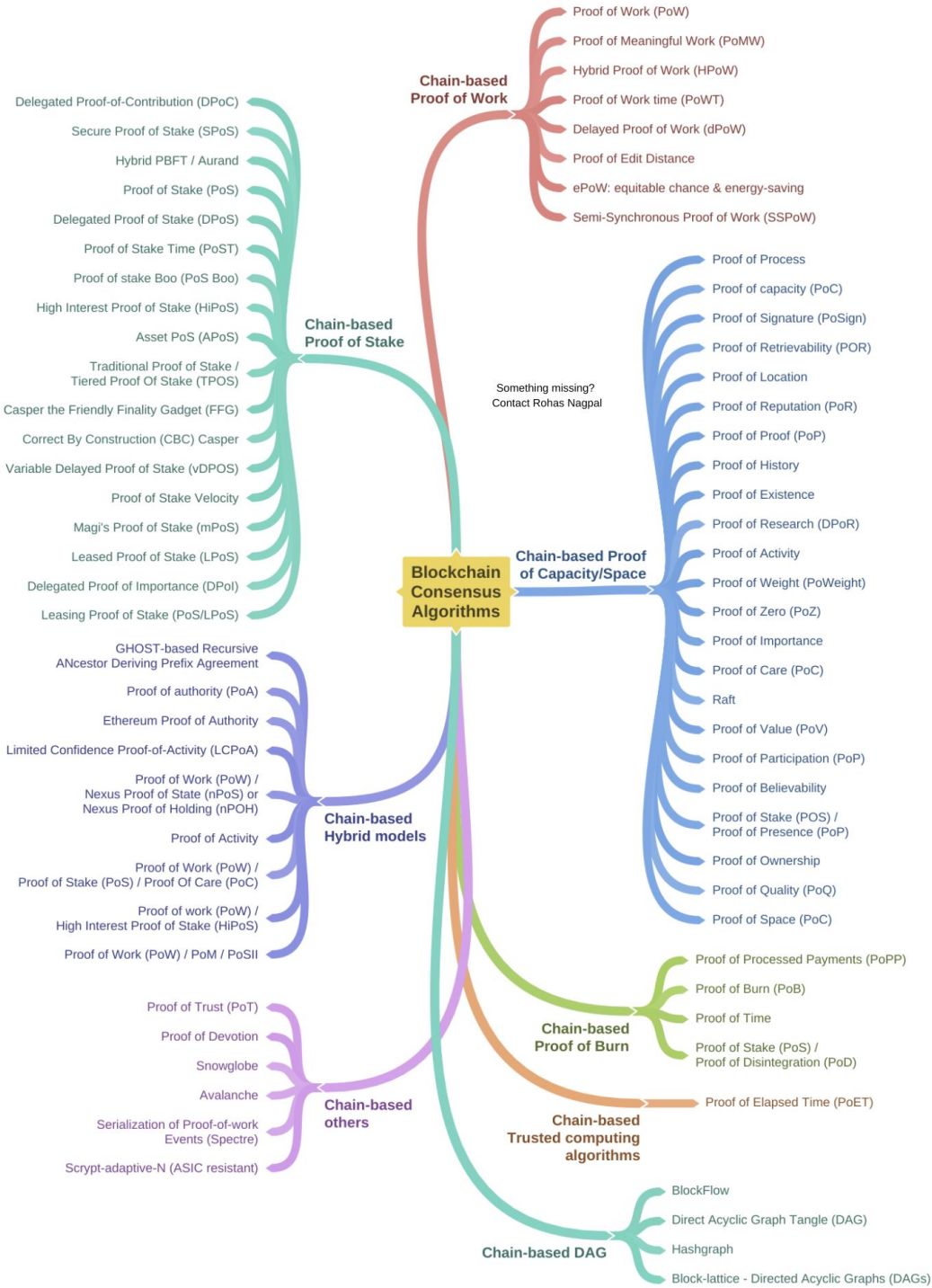
Proof-of-burn (PoB)

Miners reach a consensus by sending coins to an "eater" or "burn" address. This permanently eliminates coins from circulation, reduces inflation, and validates transactions e.g. Slimcoin.

Proof-of-Authority (PoA)

Identified, known, and credible validators produce blocks in this system. It is meant for private & enterprise blockchains.

This mindmap shows 82 blockchain consensus mechanisms divided into 9 categories.



1.6 Blockchain Bridges

Let's say you live in India and are traveling to Singapore. In India, we use Indian Rupees (INR) while in Singapore, they use Singapore Dollars (SGD).

So you could convert some money from INR to SGD using a bank. Or you could use an international card that enables you to pay in SGD when you are in Singapore.

This is simple in the world of banks. But a lot more difficult in the world of Blockchains.

Ethereum Mainnet and Terra are two independent Blockchains. They have different rules and consensus mechanisms. The native token of Ethereum Mainnet is ETH while that of Terra is LUNA.

Now suppose that you have a bunch of LUNA on Terra, but want to use it on the Ethereum Mainnet. How do you do that?

That's where a Blockchain Bridge comes into the picture. You could use the Terra Bridge to buy Wrapped Luna (WLUNA) which is an ERC-20 token native to the Ethereum Mainnet.

In simple terms, here's how a typical bridge works:

1. It receives one type of crypto e.g. LUNA.

2. It locks this LUNA as a deposit.

3. It "mints" an equal amount of another crypto e.g. Wrapped LUNA and releases it on another blockchain e.g. Ethereum Mainnet.

Hacking Blockchain Bridges

Because bridges hold a ton of crypto, they are juicy targets for hackers.

And because writing the code for bridges is insanely complex, hackers are having a field day plundering them.

When a bridge gets attacked, the hacker withdraws crypto from one side of the bridge, e.g. Wrapped LUNA, without depositing anything on the other side e.g. LUNA.

One of the biggest crypto attacks took place in March 2022 when the Ronin bridge, built for Axie Infinity, was hacked for $600+ million in ETH and USDC.

Ronin worked "off-chain" - it interfaced with the blockchain but existed on external servers that were not a part of the blockchain.

Ronin relied on 9 validator nodes, of which 5 nodes were needed to validate transactions.

The hackers exploited code vulnerabilities and also used social engineering.

Another major Bridge hack targeted Wormhole which supports 6 blockchains - Terra, Solana, Ethereum, Binance Smart Chain, Avalanche, and Polygon.

The cost of the hack was $325 million.

Bridges can be **Trusted or Trustless.**

A **Trusted Bridge** depends upon a central entity and you lose control of your cryptos when you use it.

A **Trustless Bridge** operates using smart contracts and you remain in control of your cryptos.

1.7 Blockchain Metrics

Throughput

Throughput is the number of transactions per second that a Blockchain consensus algorithm can process.

Fault tolerance threshold

Fault tolerance threshold is the upper limit of faulty nodes that directly impacts the performance of a blockchain consensus algorithm.

Finality

Finality (also called Latency) represents the time it takes for a transaction to be settled in the "ledger" of a blockchain.

Scalability

Scalability is the ability of a blockchain to expand without degrading performance.

Bitcoin

- Throughput: 7 tps
- Threshold: 51%
- Finality: 60 min

Ethereum Mainnet

- Throughput: 14 tps
- Threshold: 51%
- Finality: 6 min

Polkadot

- Throughput: 1500 tps
- Threshold: 33%
- Finality: 60 secs

1.8 Merged Mining

Merged mining is the process of **mining two or more blockchains at the same time.**

Essentially the same proof of work can be used on multiple chains.

Merged mining is technically called **Auxiliary Proof-of-Work** (AuxPoW).

The concept is that the work done on one blockchain can be leveraged as valid work on another blockchain.

The blockchain that provides the proof of work is called the parent blockchain.

The blockchain that accepts the proof of work as valid is the auxiliary blockchain.

To perform merged mining, all the involved blockchains should be using the same algorithm.

Since Bitcoin uses SHA-256, any other blockchain that uses SHA-256 can be mined along with Bitcoin subject to some technical implementations.

Merged Mining increases the profitability and performance of mining and is hugely beneficial for miners.

Some examples of merge mining:

- Dogecoin and Litecoin
- Bitcoin and RSK

1.9 Blockchain Forks

A "**fork**" is when a single blockchain splits into two.

Forks can be categorized as **hard forks** or **soft forks**.

Hard Fork

A hard fork is the result of an extensive network change. Each node has to upgrade its software to be compatible with the new processes.

If a node does not upgrade its software, it will be on a different blockchain.

Soft Fork

In a soft fork, nodes running the old software are still able to validate transactions. Software forks are backward-compatible.

Ethereum Fork

In 2016, the world's first decentralized autonomous organization (DAO) raised about US$ 150 million worth of ETH through a token sale.

But a hacker exploited a bug in the "smart contract" and siphoned out all the money!

A hard fork rolled back Ethereum's history to before the hack.

This reallocated the hacked ether to a different "smart contract" and allowed investors to withdraw their funds.

The purists hated this, which led to Ethereum splitting into 2 blockchains: Ethereum and Ethereum Classic.

Bitcoin Cash Fork

In July 2017, Bitcoin (BTC) miners representing more than 80% of the Bitcoin computing power voted to incorporate the SegWit2x (segregated witness) technology to improve Bitcoin.

Many miners and developers, who did not want SegWit2x to be introduced, initiated a hard fork and created a new blockchain and cryptocurrency - Bitcoin Cash (BCH).

1.10 Blockchain Addresses

When you buy gold or real estate, you actually get physical possession.

Blockchain assets (e.g. cryptocurrencies) are very different.

On a blockchain, we all need a key pair - public key & private key.

This key pair is generated using an algorithm such as **ECDSA** (Elliptic Curve Digital Signature Algorithm).

Sample Private Key

b5ba96aae89dc703c27ec5b3d478a8b176b874f248c83e533d9edc18e6356d44

The private key is what you would need to digitally "sign" transactions i.e. to send crypto to someone else.

If someone gets hold of your private key, they can transfer all your crypto to another address.

This is what happens in most crypto ‘hacks’.

That's why we say that "**Not your keys, not your coins**".

Sample Public Key

02b90ecf13d0ef14cd777f3a427b712ddd46703375f458092714c1c72106803ca2

The public key is used to verify the digital signature. It proves ownership of the private key.

Sample WIF

L3Jy6k2KCJ6uDNEj1hirw49sPWghT7Cg77rSDvfpAkAC7F63PhGe

A "Wallet Import Format" (wif) is a shorter version of a private key.

Sample Address

1AAE1EDcCAUmyBfi46G3vpik8oCaKVoabT

The address is similar to your bank account or UPI ID. Anyone can send crypto to your address. The address is derived from the public key.

If you send crypto to the "wrong" address, it’s gone forever!

1.11 Blockchain Wallets

A blockchain wallet is designed to:

- store your public and private keys,
- send and receive cryptos,
- monitor ‘balances’, and
- interact with supported blockchains.

There are **4 main types of wallets**:

1. Paper Wallets
2. Hardware Wallets
3. Software Wallets
4. Exchange Wallets

A hot wallet is one that is connected to the Internet and is considered the most vulnerable to hacking.

Examples: Software Wallets, Exchange Wallets.

A cold wallet is not connected to the Internet and is considered more secure.

Examples: Paper Wallets, Hardware Wallets,

Paper wallets are inconvenient to use but are the safest option. Consider using them if you have a large number of crypto assets to keep for a long period of time.

BitAddress is a JavaScript Client-Side Bitcoin Wallet Generator.

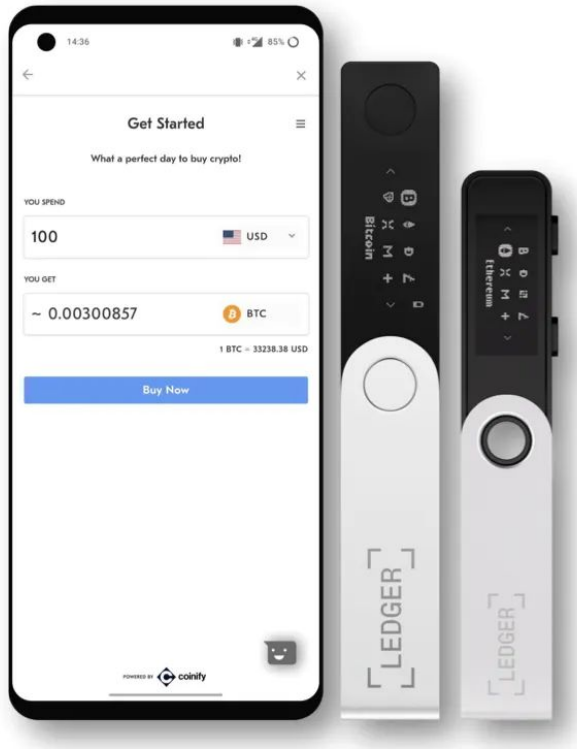
You can use it to generate Bitcoin addresses and their corresponding private keys in a web browser.

Site: https://www.bitaddress.org

To generate a Bitcoin address and its corresponding private key, simply move your mouse randomly on the bitaddress page.

A wallet will be generated in your web browser. It will look something like this:

Hardware wallets are a little pricey and there's always the risk of losing or breaking them.



Software wallets are free & easy to use. But if you accidentally delete them, your crypto is gone forever.

Examples: Trust Wallet, Metamask

So remember to back up the **seed phrase** - a bunch of words that you can write down. Example:

history
lumber
quote
board
young
dove
robust
kit
invite
plastic
regular
skull

Wallets provided by Crypto Exchanges are called "custodial" because the private keys are under the control of the Exchange.

If they go bankrupt, you lose all your crypto.

Remember: Not your keys, not your coins.

1.12 Blockchain Record-keeping Models

Accounting or **record-keeping** methods are used by blockchains to determine the provenance and ownership of coins in the network.

3 types of record-keeping models are used in blockchain networks:

1. UTXO (Unspent Transaction Output) Model e.g. Bitcoin

2. Account/Balance Model e.g. Ethereum

3. Hybrid e.g. Cardano

Each Bitcoin transaction spends **output** from prior transactions.

Each Bitcoin transaction also generates new outputs that can be spent by transactions in the future.

A Bitcoin user's wallet keeps track of all unspent transactions associated with all addresses owned by the user.

The balance of the wallet is the sum of those unspent transactions.

UTXO can be compared to spending banknotes and getting back change.

The account / balance model can be compared to spending using a debit or credit card.

The benefits of the UTXO Model are scalability and privacy (if the user uses new addresses for each transaction).

The benefits of the account / balance model are simplicity and efficiency.

Bitcoin derivatives such as Bitcoin Cash, Zcash, Litecoin, Dogecoin, Dash, etc. are UTXO chains.

Ethereum, EOS, Tron, and Ethereum Classic are account-based chains.

Cardano, Komodo, Tron, and Qtum use a hybrid model.

1.13 Asset Tokenization

Imagine that you are relaxing on your favorite beach. You whip out your smartphone and within minutes you've...

...bought shares in an innovative startup halfway across the world

...traded a fraction of a Picasso painting for a fantastic pair of collectible sneakers

...invested in the copyright license of your favorite movie

...swapped your gold & platinum holdings for fractional ownership of a private plane and a cruise liner...

...invested in fractional ownership of an office building in an upcoming high-rent location...

That's the **power of tokenization.**

A Boston Consulting Group (BCG) report predicts that the total size of illiquid asset tokenization globally would be $16 trillion by 2030.

Considering that the total market capitalization of Crypto Assets is currently less than 1.5 trillion, that's an amazing 10x opportunity!

1. What are Illiquid Assets?

From a financial perspective, Assets can be either Liquid or Illiquid.

Liquid Assets are those that can be easily converted to cash e.g. Stocks, Bonds, Gold, Agricultural commodities, etc.

Illiquid Assets are those that cannot easily be converted to cash.

Examples:

1. Art: Paintings, Sculptures, and other forms of Fine Art.

2. Carbon Credits: Tradable certificates representing the right to emit a certain amount of carbon dioxide or other greenhouse gases.

3. Copyright Licenses: Legal rights granted to the creators of original works, such as music, movies, books, and software.

4. Private Equity: Investments in startups and companies that are not publicly traded on a stock exchange.

5. Rare Collectibles: Unique items like vintage cars, rare coins & stamps, luxury watches & jewellery, and memorabilia.

6. Real Estate: Properties including residential, commercial, industrial, and agricultural.

7. Structured Products: Complex financial instruments such as Market-Linked Investments, Structured Notes, Derivative Products, Insurance-Linked Securities, and Collateralized Debt Obligations.

8. Luxury Liquor: High-end Whisky Casks and Wines.

2. What is Tokenization?

Tokenization is the process of converting the economic rights of an asset into digital tokens on a blockchain. These digital tokens represent a share or fraction of the underlying asset.

Effectively this brings the asset to the Crypto world where it can be fractionalized and traded 24x7 by a global audience.

The biggest advantages of Tokenization are:

1. Democratization of investments by lowering entry barriers.

2. Enabling small & individual investors to participate in markets that were previously accessible only to wealthy individuals or institutional investors.

3. Increased Liquidity and Market Efficiency

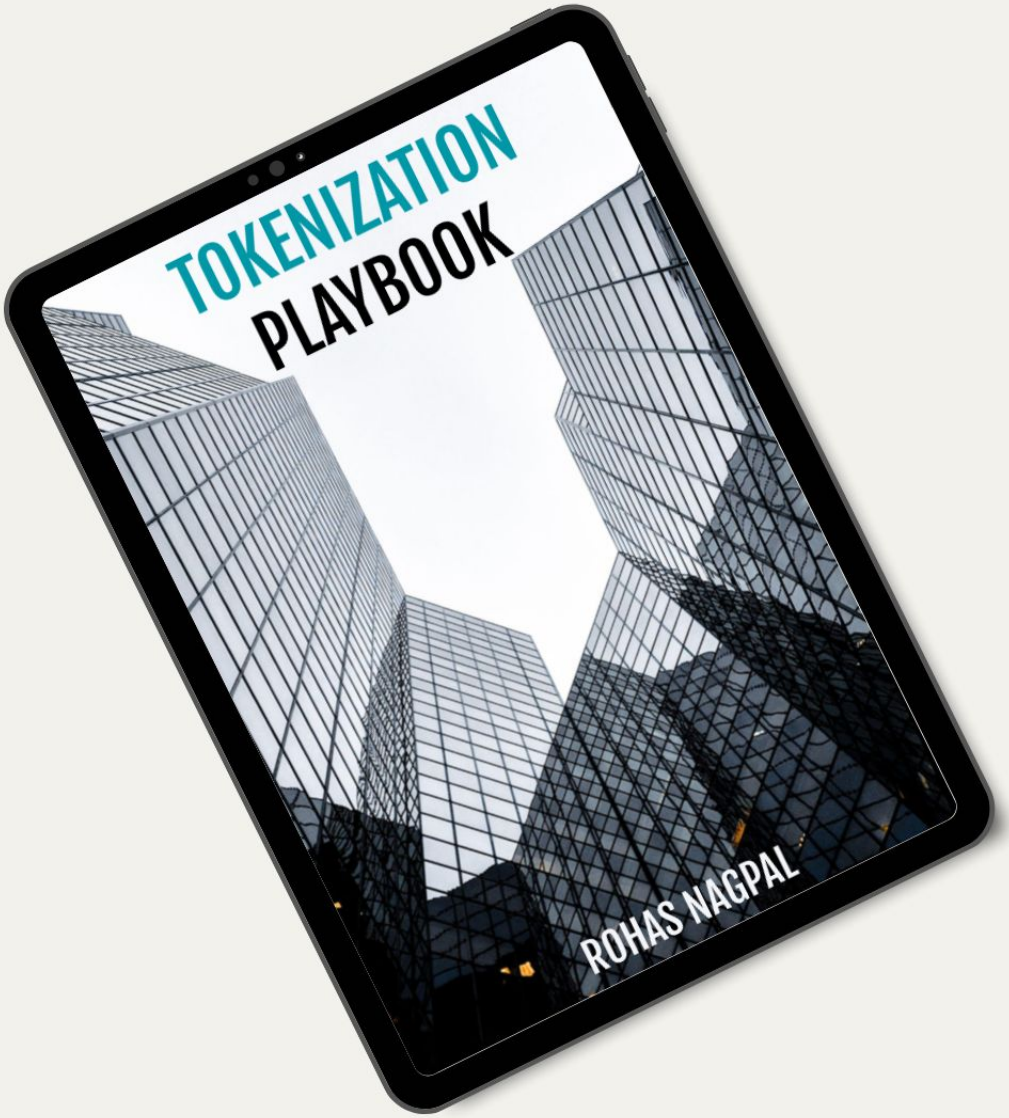
4. Bringing liquidity to markets that are traditionally illiquid.

5. Bringing greater market efficiency and price discovery.

3. How does Tokenization Create Wealth?

Tokenization offers new opportunities for investing in various asset classes. Here are some statistics on the market value of tokenizable assets:

- Art: US$ 580 Billion
- Carbon Credits: US$ 25 Billion
- Copyright Licenses: US$ 173 Billion
- Private Equity: US$ 11.7 Trillion
- Rare Collectibles: US$ 370 Billion
- Real Estate: US$ 326 Trillion
- Structured Products: US$ 7 Trillion
- Whisky Casks: US$ 300 Billion (5 years)



Download the
Tokenization Playbook by Rohas Nagpal
rohasnagpal.com/docs/Tokenization_Playbook.pdf

1.14 Smart Contracts

A smart contract is a **computer program** that automatically executes tasks when specified conditions are met.

Smart contracts:

- enable the automation of digital assets,
- enable trustless, transparent & verifiable transactions,
- streamline processes,
- reduce the need for intermediaries and their associated costs.

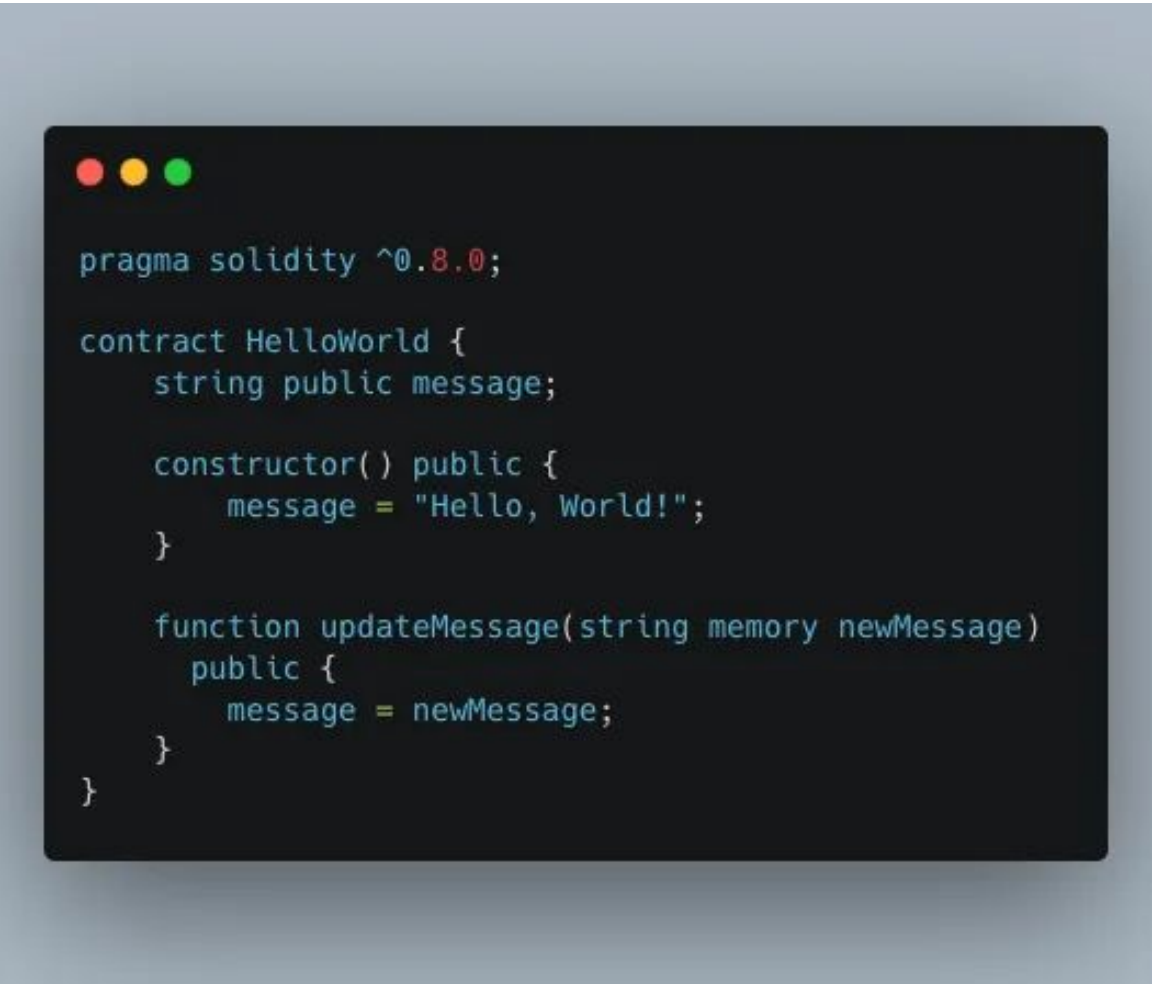
Smart contracts can be used for:

- financial transactions,
- supply chain management,
- digital identity,
- legal agreements.

The most important smart contract languages are Solidity, Vyper, Cairo, and Rust.

Ethereum is the most widely used smart contract blockchain.

Here's the code for a very basic smart contract called "**HelloWorld**".



```
pragma solidity ^0.8.0;

contract HelloWorld {
    string public message;

    constructor() public {
        message = "Hello, World!";
    }

    function updateMessage(string memory newMessage)
      public {
        message = newMessage;
    }
}
```

- It has one string variable called "message" which is public and can be accessed by anyone.
- The contract has a constructor and an "updateMessage" function.

- The constructor function executes automatically when the contract is first deployed. In this case, it sets the initial value of the "message" variable to "Hello, World!".

- The "updateMessage" function allows anyone to change the value of the "message" variable. It takes in a single input, "newMessage", which is the new value that the message variable should be set to.

- This contract can be deployed on Ethereum. Other contracts & external accounts can interact with it via function calls.

To learn more, see these:

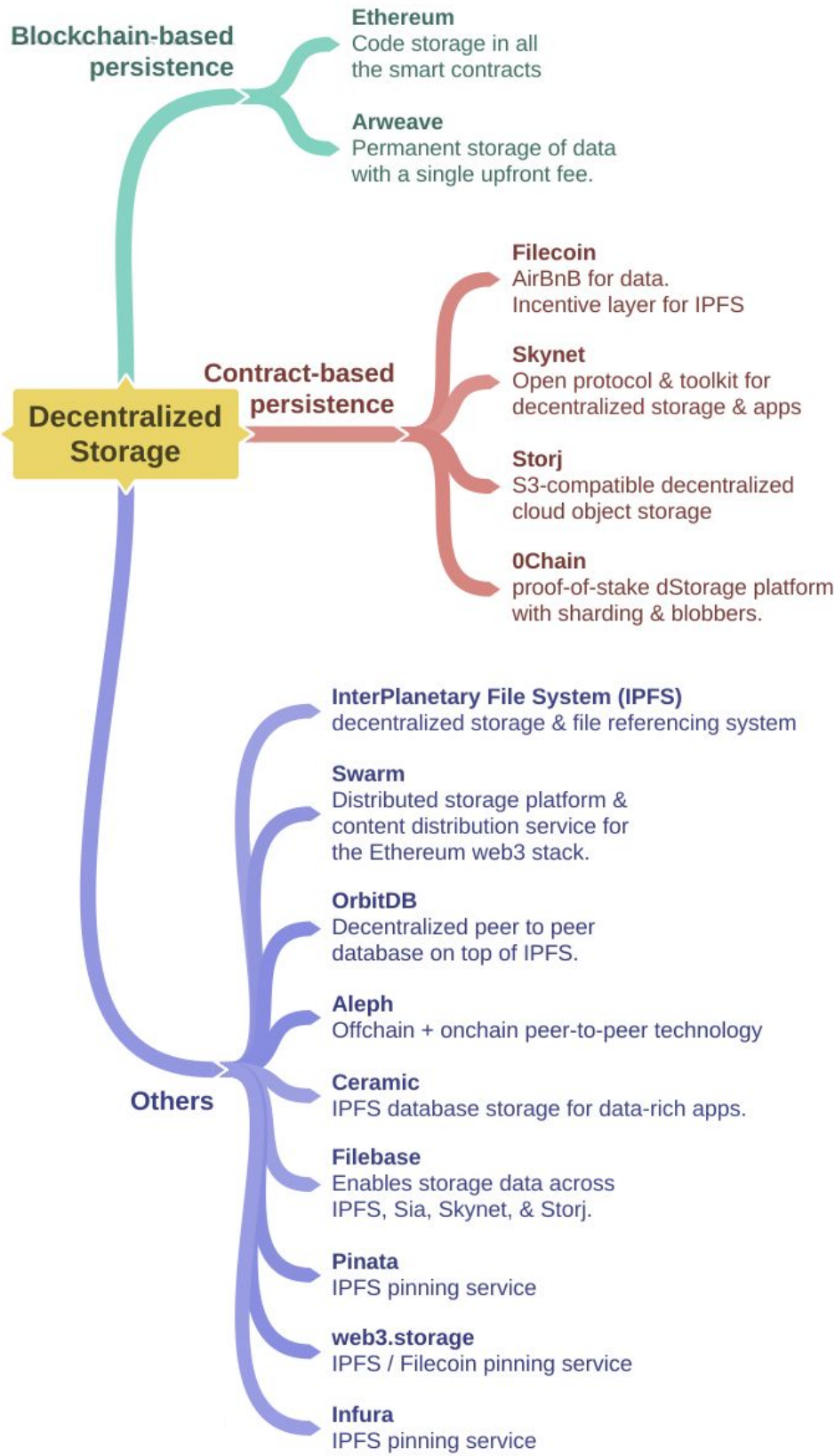
- Getting started with Smart Contracts
- How to write, compile & deploy a simple Solidity smart contract
- How to create an ERC 20 token
- How to create an ERC721 token
- dApp Development frameworks
- How to build smart contracts with Marlowe

1.15 Blockchain use-cases

- Transfer of land records / property
- Digital certificates management
- Pharmaceutical supply chain
- e-Notary service
- Blockchain-enabled e-Sign solution
- Farm Insurance
- Identity management
- Duty payments
- Automated customs enforcement and compliance
- Agriculture / farm produce supply chains
- E-Voting
- Smart Grid applications include energy transmission, distribution, trading, and marketing of energy
- Authorized access to relaying in the substation protection system
- Government crypto wallet platform for selling, buying, and trading
- Multiple layer and multiple level access Blockchain-based cloud storage of health test records
- Validation of Bill of Lading in cross-border transport
- Ease of validation of documents at customs at the ports of entry and exit

- Electronic health record management
- Digital evidence management system
- Public service delivery
- Blockchain for social good use cases (charity, donations)
- Metering and settlement
- Payment security mechanism
- Authentication and authorization services
- Automated control of decentralized power
- Smart grid application and grid management
- Microfinance for Self-Help Groups (SHG)
- Customs and trade finance
- Cross border trade
- Renewable energy trading and management
- Insurance underwriting and claims management
- Aggrotech environment
- Micro-financing, financing small businesses or individuals
- Secured logistics document exchange (SLDE)
- Cold chain for supply chain
- National and state highways, toll collection, tracking of public infrastructure
- Blockchain for urban development tracking through Public Private Partnership
- Tracking the progress on climate agreement through Blockchain

- Asset transfer across different government departments
- Digital identities, and verifiable credentials to secure privacy and enable new use cases
- Safe and secure vaccine distribution and administration
- IoT device management and security
- Vehicle lifecycle management
- Chit fund operation and administration

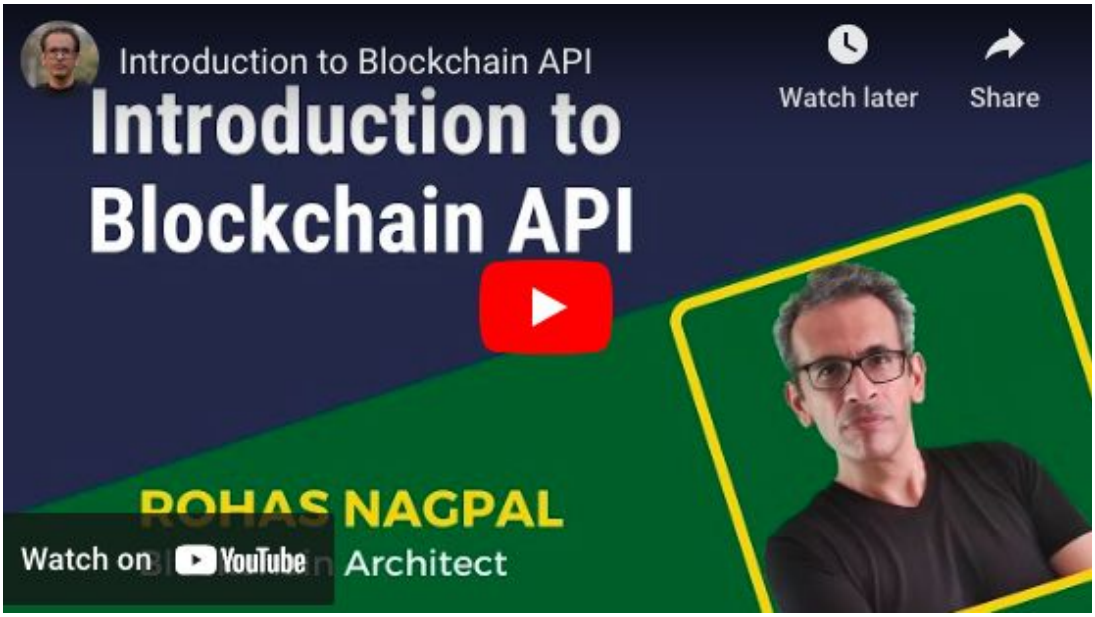

Decentralized Storage
Blockchain-based persistence
Ethereum
Code storage in all the smart contracts
Arweave
Permanent storage of data with a single upfront fee.
Contract-based persistence
Filecoin
AirBnB for data.
Incentive layer for IPFS
Skynet
Open protocol & toolkit for decentralized storage & apps
Storj
S3-compatible decentralized cloud object storage
0Chain
proof-of-stake dStorage platform with sharding & blobbers.
Others
InterPlanetary File System (IPFS)
decentralized storage & file referencing system
Swarm
Distributed storage platform & content distribution service for the Ethereum web3 stack.
OrbitDB
Decentralized peer to peer database on top of IPFS.
Aleph
Offchain + onchain peer-to-peer technology
Ceramic
IPFS database storage for data-rich apps.
Filebase
Enables storage data across IPFS, Sia, Skynet, & Storj.
Pinata
IPFS pinning service
web3.storage
IPFS / Filecoin pinning service
Infura
IPFS pinning service

02

Blockchain API

2. Blockchain API

Start with this video:



https://www.youtube.com/live/mX-WNX3C6VQ?feature=share&t=52

1. Private Key

A private key is created by picking a random 256-bit number using a method that is not predictable or repeatable e.g. bitaddress.org asks you to randomly move your mouse around for a few seconds.

This should only be done using a cryptographically secure pseudo-random number generator (CSPRNG).

A private key can be represented in multiple formats:

- Hex - 64 hexadecimal digits
- WIF - Wallet Import Format with the prefix 5
- WIF-compressed with the prefix K or L

Example of a Bitcoin private key in hexadecimal format:
f7d662f28e1e6cbf6ac5a7129f36b2871e72c3f1d31036509ae87962803b53df

Example of a Bitcoin private key in WIF-compressed format:
L5XUUSuoF1SyrZW3kPqyFNWhDS52DsrGh2rQEw6bs4kERcgiJU45

2. Public Key

A Public Key is calculated from the private key using irreversible Elliptic Curve Cryptography.

Public Key = Private Key * Generator point (a constant)

Example of a Bitcoin public key in hexadecimal format:
021511f3e82638681e19c2ad9bf8550e82abfcf981449404288d81a84445dbf27a

Calculating the Private Key from the Public Key is computationally infeasible as of today's computing technology.

3. Address

The address is calculated from the public key through a one-way cryptographic hash function.

Example: To calculate a bitcoin address, we compute the SHA256 hash of the Public Key and then compute the RIPEMD160 hash of the result.

Address = RIPEMD160(SHA256(Public Key))

Example of a Bitcoin Address:
1AThkzMLY37PYYJMUDydmjzS46xBr9ZMcm

4. Non-deterministic (Random) Wallets

Blockchain Wallets are of two main types:

- Non-deterministic (Random) Wallets
- Deterministic (Seeded) Wallets

Non-deterministic (Random) Wallets are just a collection of randomly generated private keys. The biggest problem with them is that they are difficult to manage, back up, and import.

The basic code to generate a non-deterministic wallet using BlockCypher API services is *https://api.blockcypher.com/v1/{BLOCKCHAIN}/main/addrs*

You will need to replace {BLOCKCHAIN} with the blockchain name e.g. btc for Bitcoin, eth for Ethereum Mainnet, etc.

5. Deterministic (Seeded) Wallets

Deterministic (Seeded) Wallets contain private keys that are all derived from a common seed using a one-way hash function.

A seed is a randomly generated number combined with other data e.g. an index number to derive the private keys.

A mnemonic code is a sequence of words that represent a seed e.g.
urge pulp usage sister evidence arrest palm math please chief egg abuse

The mnemonic code is used to create and re-create all the keys. That's why you only need to back up these words.

This process is defined in BIP0039:

- Create a random sequence of 128 to 256 bits.
- Create a checksum of the random sequence by taking the first few bits of its SHA256 hash.
- Add the checksum to the end of the random sequence.
- Divide the sequence into sections of 11 bits, using those to index a dictionary of 2048 predefined words.
- Produce 12 to 24 words representing the mnemonic code.

6. Hierarchical Deterministic Wallets

Hierarchical deterministic (HD) wallets contain keys in a tree structure. Parent keys can produce children keys, which in turn can produce grandchildren keys. This can go on infinitely.

Step 1: Generating Mnemonic Code and Extended Public Key

The code for generating a BIP44 HD Wallet for Bitcoin is here.
https://gist.github.com/rohasnagpal/dcec046406da515c4fbd85f895ebd0e9#file-bip44_hd_wallet-php

Note: You will need to generate a free Tatum API to use this.

This is a sample response:
{
"mnemonic": "urge pulp usage sister evidence arrest palm math please chief egg abuse",
"xpub":
"xpub6EsCk1uU6cJzqvP9CdsTiJwT2rF748YkPnhv5Qo8q44DG7nn2vbyt48YRsNSUYS44jFCW9gwvD9kLQu9AuqXpTpM1c5hgg9PsuBLdeNncid"
}

SLIP-0044 : Registered coin types for BIP-0044
BIP-0044 defines a logical hierarchy for deterministic wallets. Level 2 of the hierarchy describes a coin type in use. See details here:
https://github.com/satoshilabs/slips/blob/master/slip-0044.md

Step 2: Generating the address

This code generates the address from the extended public key and derivation path index: https://gist.github.com/rohasnagpal/117b459e9a8459b20ce95dcda904ae8e#file-generating-the-address

Note: You will need to generate a free Tatum API to use this.

This is a sample response.

{
"address":
"0xa7673161CbfE0116A4De9E341f8465940c2211d4"
}

Step 3: Generating the private key

This code generates a private key for an address from a mnemonic for a given derivation path index:

https://gist.github.com/rohasnagpal/c791fd4573e84088437688a4859ae673#file-generate-private-key

Note: You will need to generate a free Tatum API to use this.

This is a sample response.

{"key":"cMotAJwwC3hruht3gYKBBLm9kUhEWvfovDTLGPy4biyNbR2VBXLG"}

Sample BTC HD Wallet:
https://www.rohasnagpal.com/web3/files/hd_wallet_btc.php

Sample ETH HD Wallet:
https://www.rohasnagpal.com/web3/files/hd_wallet_eth.php

03

Network Security & Privacy

3.1 Blockchain Network Attacks & Vulnerabilities

51% Attack

If a single miner or mining pool controls more than 50% of the network's mining hashrate, they can manipulate the blockchain's consensus mechanism, allowing them to reverse transactions and double-spend coins. This attack is mostly a concern for smaller, less well-established blockchains like Bitcoin SV.

Sybil Attack

In a Sybil attack, an attacker subverts the reputation system of a peer-to-peer network by creating a large number of pseudonymous identities, using them to gain a disproportionately massive influence.

Eclipse Attack

An Eclipse attack is when an attacker controls all of the victim's connections to the network, isolating the victim from the rest of the blockchain network, and feeding them false information.

Routing Attack

By manipulating the routing of information over the internet, an attacker can partition the blockchain network or delay block propagation, increasing the risk of double-spending attacks.

Selfish Mining Attack

In a selfish mining attack, a miner mines a new block and keeps it secret from other miners. If the selfish miner can find a second block before the others find any, they can publish their chain and invalidate the work of the other miners.

Double Spending Attack

This type of attack involves a situation where a user manages to spend their cryptocurrencies more than once by creating multiple copies of the transaction.

Replay Attack

A replay attack involves re-transmitting or delaying a valid data transmission that has been fraudulently repeated or delayed. This might occur during a hard fork if proper security measures are not in place.

Timejacking Attack

This attack involves manipulating the timestamp of a node, which can affect the blockchain's functionality, as block generation relies on accurate timekeeping.

Smart Contract Vulnerabilities

Smart contracts are prone to various types of bugs and vulnerabilities, including re-entrancy attacks, where external contract calls can be hijacked. They are also vulnerable to issues with underflows and overflows.

Phishing Attacks

Similar to traditional phishing attacks, blockchain phishing attempts involve tricking users into giving away their private keys, often through fake websites or social engineering tactics.

Note: Each blockchain protocol will have its own specific potential vulnerabilities depending on its design and architecture. It's important for blockchain engineers to be aware of these possible attacks.

3.2 Node Security Best Practices

Securing a blockchain node is essential to maintain the integrity of the overall network and to protect the information stored within the node.

Here is a comprehensive checklist for best practices in node security:

- ❏ **Regular Software Updates:** Always ensure that the node software and all of its dependencies are up-to-date. This includes the operating system, blockchain software, and any other installed software.

- ❏ **Firewall Configuration:** Use a firewall to limit and control the traffic that your node accepts. The firewall should be configured to allow necessary P2P traffic while blocking unnecessary or potentially harmful traffic.

- ❏ **DDoS Protection:** Protect your node against Distributed Denial-of-Service (DDoS) attacks. This could include rate limiting, IP whitelisting, or using a third-party DDoS protection service.

- ❏ **Secure Communication:** Use secure communication protocols to protect data in transit. For instance, Secure Socket Layer (SSL) / Transport Layer Security (TLS) can be used to encrypt communication.

- ❏ **Role-Based Access Control:** Use Role-Based Access Control (RBAC) to limit who can access what resources. It should be ensured that users and applications only have the permissions that they need and nothing more.

- ❏ **Secure Key Management:** Private keys should be securely stored and never exposed. They should also be backed up in a secure location. Hardware Security Modules (HSMs) or secure key management services can be used for enhanced security.

- ❏ **Regular Auditing:** Regularly audit your node for any signs of irregular or suspicious activity. This includes monitoring log files, checking for unauthorized access, and inspecting outgoing traffic.

- ❏ **Physical Security:** Physical security is as important as digital security. The hardware running the node should be secured against physical tampering.

- ❏ **Redundancy and Backups:** Ensure data redundancy and regular backups. If the node data gets corrupted or lost, having a backup allows for a quick recovery.

- ❏ **Secure APIs:** If your node provides an API, ensure that it's secure. This might include requiring API keys, rate limiting, and regularly auditing API usage.

- ❏ **Patch Management:** Regularly apply security patches to all software. Unpatched software can be a security vulnerability.

- ❏ **Network Segmentation:** Use network segmentation to limit the potential impact of a compromise. Sensitive systems, like a node, should be isolated from less secure parts of the network.

- ❏ **Incident Response Plan:** Have a clear incident response plan in place in case of a security breach. The quicker you can respond to a security incident, the less damage it's likely to cause.

3.3 Network Monitoring Tools

Network monitoring is an essential part of maintaining a healthy and secure blockchain node.

Here are some popular tools used for network monitoring:

Wireshark

This is an open-source protocol analyzer that network professionals use for troubleshooting, analysis, software and protocol development, and education. It allows you to examine data from a live network or from a capture file on disk.

Nagios

Nagios is a powerful, enterprise-grade open-source monitoring system that allows organizations to identify and resolve IT infrastructure problems before they affect critical business processes.

Zabbix

Zabbix is an open-source monitoring tool for networks and applications. It is designed to monitor and track the status of network services, servers, and other network hardware.

Prometheus

This open-source systems monitoring and alerting toolkit can handle multi-dimensional data collection as well as querying and alerting. It's often used in combination with a visualization tool, like Grafana.

Netdata

Netdata is a free, open-source and real-time performance and health monitoring tool with a web front-end. It's useful for tracking performance anomalies.

Ethereum Network Intelligence API

This is a backend-only part of a tool that provides a complete overview of the current state of the Ethereum network, including information about blocks, transactions, and other network activity.

Bitcoin CLI (Command-Line Interface)

For Bitcoin nodes, the built-in command-line interface can provide a wealth of information about the node's operation and the network's status.

IFTOP

It listens to network traffic on named interfaces and shows a table of current bandwidth usage by pairs of hosts.

TCPdump

This command-line utility is a powerful tool for analyzing network behavior and troubleshooting problems.

Elastic Stack (ELK - Elasticsearch, Logstash, Kibana):

This is a set of open-source tools that work together to help you monitor your system by collecting logs from various sources, storing, indexing and visualizing them in a centralized location.

3.4 Privacy Enhancing Technologies

Privacy Enhancing Technologies (PETs) are designed to uphold data minimization principles and allow individuals to retain control and ownership of their personal data.

In the context of blockchain, several technologies can enhance privacy:

Zero-Knowledge Proofs (ZKPs)

ZKPs are cryptographic methods that allow one party (the prover) to prove to another party (the verifier) that they know a value of a specific piece of information without conveying any other information apart from the fact that they know this value.

Zcash, a cryptocurrency, uses ZKPs to maintain the privacy of transactions.

Ring Signatures

A type of digital signature that can be performed by any member of a group of users that each have keys. It's impossible to determine which of the group members' keys was used to produce the signature.

Monero, a privacy-focused cryptocurrency, uses ring signatures to provide transaction privacy.

Coin Mixing or CoinJoin

Services like CoinJoin allow multiple users to combine their transactions into a single transaction, which makes it difficult to determine which inputs (coins sent) correspond to which output (coins received).

Mimblewimble

This blockchain protocol uses confidential transactions and a unique transaction structure to dramatically improve both scalability and privacy. It's used by cryptocurrencies like Grin and Beam.

Homomorphic Encryption

This form of encryption allows computations to be done on encrypted data without decrypting it first. The result of the computation is encrypted and can be decrypted to get the actual result.

Private Information Retrieval (PIR)

PIR protocols allow a user to retrieve an item from a server in possession of a database without revealing which item is retrieved. This is useful for light clients in blockchain networks, which need to retrieve certain parts of the blockchain without revealing their interests.

Secure Multi-Party Computation (sMPC)

sMPC allows a set of parties to compute a function over their inputs while keeping those inputs private. This could be used in a blockchain setting to enable private transactions and smart contracts.

Tor/I2P Networks

While not specific to blockchain, using privacy networks like Tor or I2P can help hide a user's IP address when making transactions or operating a node, thus enhancing privacy.

These technologies can be used individually or combined to provide multiple layers of privacy protection.

Download the Blockchain Security Controls from:

https://www.rohasnagpal.com/docs/blockchain-security/blockchain_security_controls.pdf

04

Node Maintenance & Performance Optimization

4.1 Blockchain Data Storage and Management

In a blockchain network, data storage and management is a crucial aspect of maintaining efficient operations and reliability.

Here's a breakdown of key concepts and strategies involved in blockchain data storage and management:

Blockchain Structure

A blockchain is essentially a linked list where each block contains a group of validated transactions and a reference to the previous block via a hash.

Each transaction in a block is made up of metadata and inputs and outputs.

On-Chain Storage

Storing data directly on the blockchain is known as on-chain storage. This means that the data is stored in the transactions themselves and becomes an immutable part of the blockchain.

This is generally the most secure form of storage but is also the most expensive and least scalable due to the size limitations of a block and the cost of transactions.

Off-Chain Storage

Large scale data storage in blockchain is not feasible due to cost and size restrictions. Hence, only the hash of the data or a reference to the data is stored on-chain, while the actual data is stored off-chain, i.e., outside of the blockchain.

Off-chain data storage could be centralized servers, distributed file systems (like IPFS), or data clouds.

Decentralized Storage Systems

Decentralized storage systems like the InterPlanetary File System (IPFS), Filecoin, or Swarm are used for more efficient and distributed data storage and retrieval.

This type of system is often used in conjunction with a blockchain to create decentralized applications that require file storage.

Sharding

Sharding is a scalability technique where the state of the blockchain is divided into partitions or "shards", each capable of processing its own transactions and smart contracts.

Sharding can potentially allow a blockchain to process many transactions in parallel, improving scalability.

Pruning

Blockchain pruning is a method of limiting a node's storage requirements by discarding older blocks from storage after they've been verified and their transactions have been accounted for.

Pruned nodes keep only the set of transactions that are necessary to maintain a fully validating node (i.e., the blockchain's UTXO set).

State Channels

These are two-way pathways opened between two users that want to make multiple transactions without committing all transactions to the blockchain.

This reduces the strain on the network and reduces fees for the two participants.

Database Management

Depending on the specific implementation, blockchains may utilize traditional database systems (like LevelDB or RocksDB used in Bitcoin and Ethereum) to store and retrieve data efficiently.

Backups

Regular backups of the blockchain data (i.e., the entire ledger or just the latest state) are essential to ensure data availability in case of any faults or data corruption.

A combination of these techniques can be used depending on the specific use case and requirements.

Each technique comes with its own trade-offs in terms of security, cost, scalability, and decentralization.

4.2 Node Performance Metrics

Node performance metrics help in understanding the overall health and efficiency of a blockchain node.

They enable you to identify potential issues and optimize the node for better performance.

Here are some key performance metrics for a blockchain node:

Block Propagation Time

The amount of time it takes for a block to propagate through the network. Faster propagation times generally indicate a healthier network.

Memory Usage

The amount of RAM being used by the node. Excessive memory usage could indicate a memory leak or inefficient memory management.

CPU Usage

The amount of CPU resources being used by the node. Similar to memory usage, high CPU usage could indicate inefficiencies or bugs in the node's software.

Disk Usage

The amount of disk space being used by the node. Blockchain data can take up a significant amount of storage space, especially for full nodes.

Network Latency

The time it takes for a packet of data to get from one designated point to another in the network. Lower latency is better for overall network performance.

Number of Connected Peers

The number of other nodes that your node is connected to. A healthy number of peers is crucial for the robustness and redundancy of the network.

Number of Transactions in the Mempool

The mempool (or memory pool) is a holding area for transactions waiting to be confirmed by the network. A growing mempool could indicate network congestion.

Transaction Verification Time

The time it takes to verify a transaction. Faster verification times improve the overall user experience.

Block Verification Time

The time it takes to verify a block of transactions. Faster block verification times improve the overall throughput of the blockchain network.

Uptime

The amount of time the node has been running without interruption. High uptime is generally desirable, as interruptions can disrupt the functioning of the network and the applications running on it.

Each of these metrics can be monitored using various tools and techniques, including command-line tools, APIs, log analysis, and dedicated monitoring software.

Regular monitoring and optimization based on these metrics can help ensure the smooth operation of your blockchain node.

4.3 Performance Tuning & Optimization

Performance tuning and optimization for a blockchain node involves identifying and resolving issues that hinder its efficiency and reliability. Here are some strategies:

Hardware Upgrade

Enhancing the node's hardware can significantly improve performance. This could mean adding more memory (RAM), using faster storage (SSD instead of HDD), or upgrading the CPU.

A fast and stable internet connection is also essential.

OS Optimization

The operating system hosting the node can be optimized to better handle the node's demands. This can include tuning network settings, optimizing disk I/O, adjusting CPU scheduling, and ensuring adequate system resources are available.

Database Optimization

The underlying database that the node uses to store blockchain data can often be optimized. This could involve tuning database parameters, indexing, or optimizing database queries.

Node Configuration

Most blockchain nodes have several configuration options that can be tuned for better performance.

These options can control things like memory usage, the number of peer connections, or the amount of data to keep in cache.

Sharding

Sharding is a technique where the state of the blockchain is divided into partitions or "shards", each capable of processing its own transactions and smart contracts.

Sharding can potentially allow a blockchain to process many transactions in parallel, improving scalability.

Off-Chain Transactions

Implementing off-chain transactions or state channels can help scale the blockchain by handling transactions off the blockchain and settling the net result on-chain.

Pruning

Pruning is a method of reducing storage requirements by discarding older blocks from storage after they've been verified and their transactions have been accounted for.

Network Optimization

This involves optimizing the node's connections to its peers to ensure efficient communication. This could involve managing the number of connections, optimizing data transfer protocols, or selecting peers based on their network latency or reliability.

Threading and Parallel Processing

If the node software and the hardware support it, enabling multi-threading or parallel processing can significantly improve the performance of a node.

Regular Monitoring and Updates

Regularly monitor performance metrics to identify potential bottlenecks or issues. Keep the node software up-to-date, as updates often include performance improvements and bug fixes.

Note: Optimization strategies can vary depending on the specific blockchain technology, the role of the node (full node, miner, validator, etc.), and the underlying hardware and network conditions.

Always test changes in a controlled environment before applying them to a production node to understand their impact.

4.4 Backup & Disaster Recovery

Backup and disaster recovery are crucial aspects of maintaining a reliable and resilient blockchain node.

Here's how you might approach this:

Regular Backups

Regularly back up your node's data. This includes the blockchain data, configuration files, and any private keys or wallets associated with the node.

The frequency of the backups will depend on your node's activity and your tolerance for data loss.

Offsite Storage

Store backups in a different location from your primary node. This protects against physical disasters like fires, floods, or hardware failures.

Secure Storage

Protect your backups from unauthorized access. This might involve encryption, access controls, or physically secure storage locations.

Redundancy

Consider setting up redundant nodes that can take over if your primary node fails. This can be part of a load balancing setup, which can also improve performance.

Testing

Regularly test your backup and recovery procedures to make sure they work as expected. This might involve setting up a test node and trying to restore it from a backup.

Disaster Recovery Plan

Have a disaster recovery plan that outlines what to do in various disaster scenarios. This plan should be reviewed and updated regularly.

Failover Mechanisms

Consider setting up automatic failover mechanisms that can detect when your primary node is down and switch over to a backup node.

Backup Software

Use reliable backup software that can automate the backup process, handle encryption, and ensure data integrity.

Documentation

Document your backup and recovery procedures, and make sure that the relevant people are trained and familiar with these procedures.

Node Software Backups

Besides the blockchain data, keep copies of the node software or docker images. In case of software corruption, having these backups will be useful.

Note: Remember, the goal of backup and disaster recovery is not just to protect against data loss, but also to minimize downtime and ensure the smooth operation of your node and the services that depend on it.

The specific strategies you choose will depend on the specific requirements of your node, your network, and your organization.

05

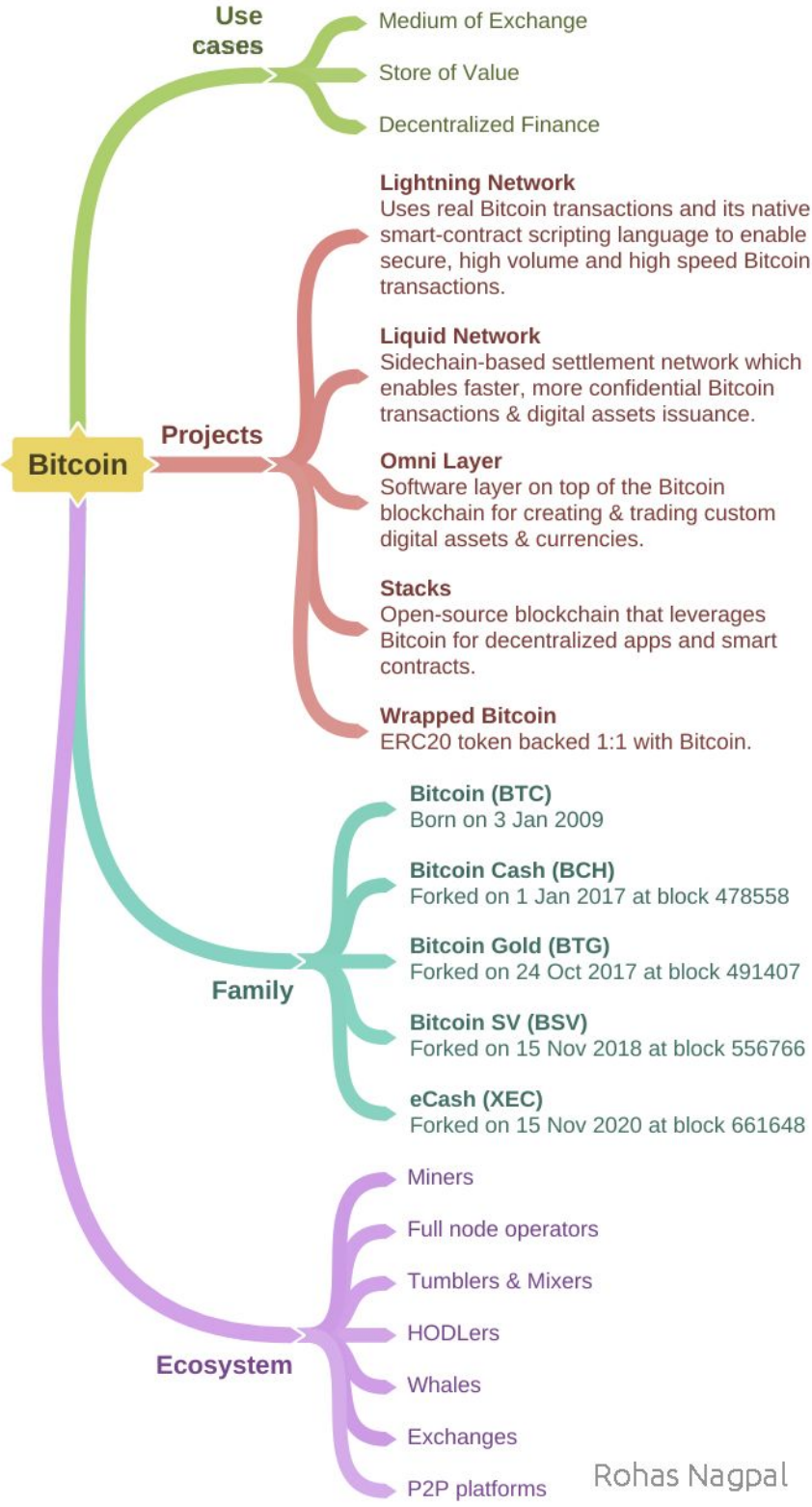
Bitcoin

If blockchain technology were a religion, the Bitcoin whitepaper would be its Bible.

The Bitcoin whitepaper is the **foundational document** that started the Blockchain revolution. As a Blockchain Engineer, you MUST have a deep understanding of it.

The video below covers Digital Signatures, Hash-based Proof-of-Work, Double-spending, Transactions, Simplified Payment Verification, Combining & Splitting Value, and Privacy.

https://www.youtube.com/watch?v=3BBRF1XgjTE&t=75s


Bitcoin
Use cases
Medium of Exchange
Store of Value
Decentralized Finance
Projects
Lightning Network
Uses real Bitcoin transactions and its native smart-contract scripting language to enable secure, high volume and high speed Bitcoin transactions.
Liquid Network
Sidechain-based settlement network which enables faster, more confidential Bitcoin transactions & digital assets issuance.
Omni Layer
Software layer on top of the Bitcoin blockchain for creating & trading custom digital assets & currencies.
Stacks
Open-source blockchain that leverages Bitcoin for decentralized apps and smart contracts.
Wrapped Bitcoin
ERC20 token backed 1:1 with Bitcoin.
Family
Bitcoin (BTC)
Born on 3 Jan 2009
Bitcoin Cash (BCH)
Forked on 1 Jan 2017 at block 478558
Bitcoin Gold (BTG)
Forked on 24 Oct 2017 at block 491407
Bitcoin SV (BSV)
Forked on 15 Nov 2018 at block 556766
eCash (XEC)
Forked on 15 Nov 2020 at block 661648
Ecosystem
Miners
Full node operators
Tumblers & Mixers
HODLers
Whales
Exchanges
P2P platforms
Rohas Nagpal

Setting up a Bitcoin node

Here's a Step-by-Step Guide to setting up a Bitcoin node.

1. Download Bitcoin Core

Visit the official Bitcoin Core website (https://bitcoincore.org/en/download/).

Download the latest version of Bitcoin Core for your operating system (Windows, macOS, Linux).

2. Install Bitcoin Core

Follow the installation instructions for your specific operating system. It's a straightforward process.

3. Initial Blockchain Sync

Launch Bitcoin Core.

Allow it to sync with the Bitcoin blockchain. This process can take several hours, as it involves downloading and verifying the entire blockchain history. Make sure you have enough storage space.

4. Configure Bitcoin Core

Once the initial sync is complete, you can configure your node by creating a bitcoin.conf file.

This file allows you to set various parameters, such as enabling RPC (Remote Procedure Call) for interacting with your node.

5. Let Your Node Run

Keep your Bitcoin Core software running 24/7 if possible. This helps contribute to the Bitcoin network by relaying transactions and blocks.

Your node will automatically join the Bitcoin network and start helping validate transactions and blocks.

06

Ethereum

Ethereum is NOT a blockchain. It's NOT a cryptocurrency either! It's actually a protocol (a set of rules or procedures) like "HTTP" or "HTTPS".

Multiple independent blockchains run on the Ethereum protocol:

- **Ethereum Mainnet**: This is the production network.
- **Görli**: This is a Proof of Authority testnet.
- **Private networks**: Development networks, and Consortium networks.
- **Layer 2 testnets**: Arbitrum Rinkeby, and Optimistic Kovan.

When most people talk about Ethereum, they are talking about **Mainnet** - the primary public Ethereum production blockchain.

This is where actual-value transactions occur on the blockchain. The native crypto of this Ethereum is **Ether** (ETH).

Ethereum Virtual Machine (EVM)

One of the greatest blockchain innovations is the Ethereum Virtual Machine (EVM).

EVM is "the environment in which all 'Ethereum' accounts and smart contracts live". Smart contracts are programs that run automatically when some predefined conditions are met.

The sole purpose of the Ethereum protocol is to keep "the continuous, uninterrupted, and immutable operation" of the EVM. At any given block, Ethereum has only one "canonical" or unique state.

EVM defines the rules for computing new valid states from one block to another.

EVM exists as a single entity maintained by a large number of connected computers (nodes) running an Ethereum client e.g. Geth or OpenEthereum.

A client is a software that enables nodes to read blocks on the blockchain and smart contracts.

For details, see:
https://ethereum.org/en/developers/docs/evm

Ethereum standards

For a comprehensive list, see:
https://eips.ethereum.org/erc

The most popular Ethereum standards are:

ERC-20: Token Standard
A standard for fungible tokens on Ethereum, enabling consistent interaction with different tokens across various applications.

ERC-223: Token with transaction handling model
Addresses ERC-20,'s problem of accidental token transfers to contracts that cannot handle them.

ERC-721: Non-Fungible Token Standard
A standard for unique, indivisible assets with distinct properties on the Ethereum blockchain.

ERC-777: Token Standard
An advanced token standard with features like operator functions & hooks for more flexible and secure token interactions.

ERC-1155: Multi Token Standard
A multi-token standard that supports both fungible and non-fungible tokens in a single contract, optimizing batch transfers and reducing transaction costs.

ERC-2981: NFT Royalty Standard
A standard for handling royalty payments for NFTs, allowing creators to receive a percentage of sales whenever their NFT is sold on a secondary market.

ERC-3156: Flash Loans
A standard interface for flash loans, which are uncollateralized loans that must be repaid in the same transaction they are borrowed, enabling unique financial operations in DeFi.

ERC-3475: Abstract Storage Bonds
A standard for decentralized creating, managing, and trading of bond tokens with varying terms & conditions.

ERC-3525: Semi-Fungible Token
A standard for for tokens with both fungible and non-fungible characteristics for representing a wide range of assets with varying attributes.

ERC-3643: Token for Regulated EXchanges
Standard for management and legally compliant transfer of security tokens.

ERC-4626: Tokenized Vaults
A standard for tokenized vaults, defining a common interface for yield-bearing tokens and enabling consistent interactions with DeFi protocols.

ERC-4907: Rental NFT
Enables NFT rentals without transferring by enabling a temporary “user” role separate from the owner,

ERC-4910: Royalty Bearing NFTs
Extends the ERC-721 standard to preclude central authorities from manipulating royalty payments.

ERC-5192: Minimal Soulbound NFTs
Standard for non-transferrable, non-fungible, socially priced tokens bound to a single account.

ERC-5489: NFT Hyperlink Extension
Enables embedding NFTs with hyperlinks and allows owners to authorize a URL slot to a specific address.

ERC-5507: Refundable Tokens
Enables refunds for ERC-20, ERC-721 & ERC-1155 tokens purchased in an initial token offering.

ERC-5528: Refundable Fungible Token
Extends EIP-20 to facilitate an escrow service where sellers and buyers exchange tokens through a smart contract, allowing withdrawals upon success or refunds if the escrow fails or is incomplete.

ERC-5570: Digital Receipt Non-Fungible Tokens
Enables digital receipts of transactions with transaction details for record keeping.

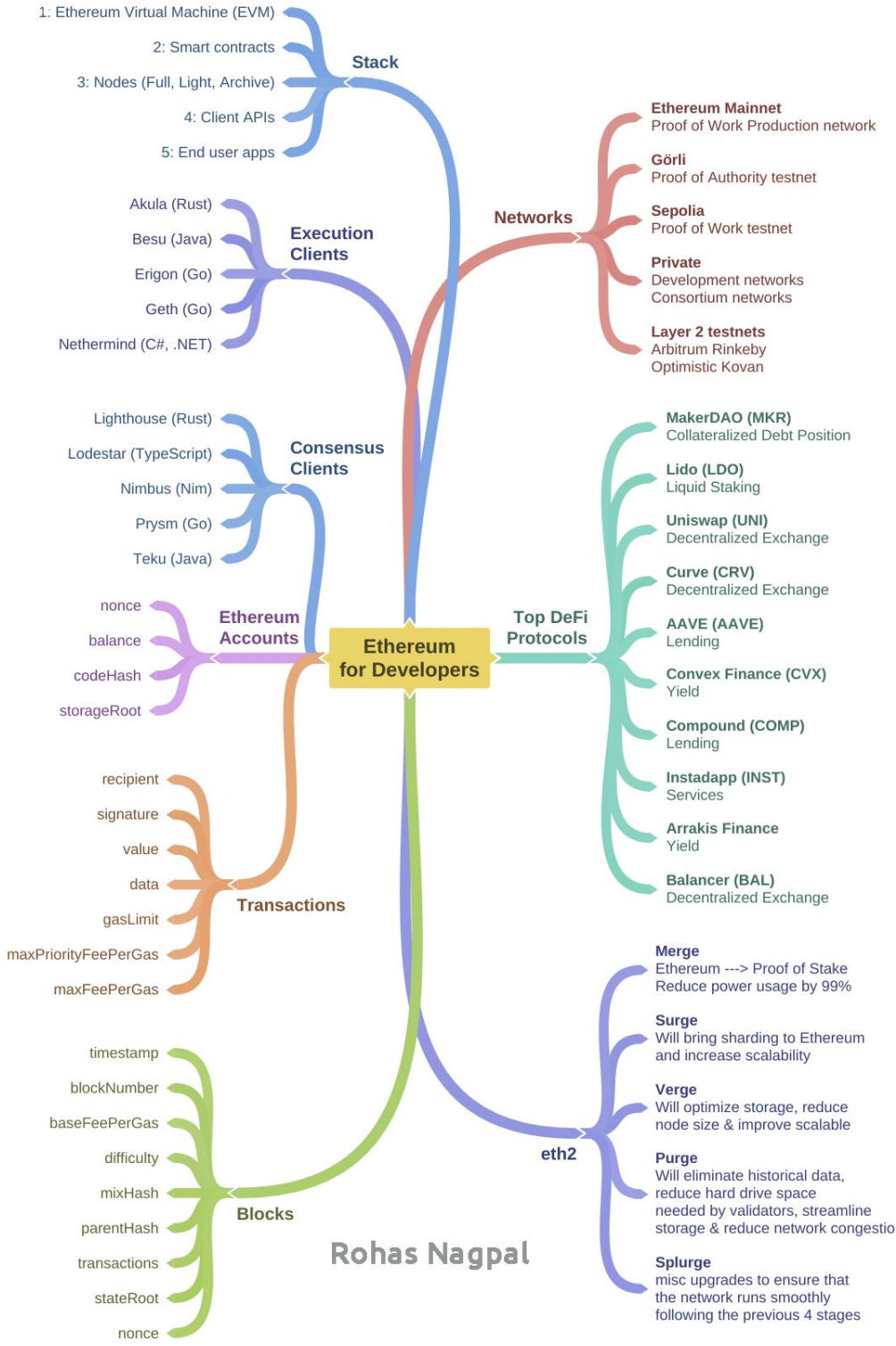
ERC-5585: ERC-721 NFT Authorization
Allows independent management of rights by separating ERC-721 NFT's commercial usage rights from its ownership.

ERC-5606: Multiverse NFTs
Indexes digital assets like wearables and in-game items across metaverses & games, extending ERC-721 and ERC-1155 to allow for bundled trading.

ERC-5725: Transferable Vesting NFT
Extends ERC-721 to represent vested tokens with functionalities for creating, claiming, and managing vesting properties.

ERC-6672: Multi-redeemable NFTs
Extends ERC-721 to enable NFTs to be redeemed in multiple scenarios for physical or digital objects.

ERC-7439: Prevent ticket touting
Extends ERC-721 to enable ticketing agents and event organizers to prevent ticket scalping and authorize secure ticket resale.


Ethereum for Developers
Stack
1: Ethereum Virtual Machine (EVM)
2: Smart contracts
3: Nodes (Full, Light, Archive)
4: Client APIs
5: End user apps
Execution Clients
Akula (Rust)
Besu (Java)
Erigon (Go)
Geth (Go)
Nethermind (C#, .NET)
Consensus Clients
Lighthouse (Rust)
Lodestar (TypeScript)
Nimbus (Nim)
Prysm (Go)
Teku (Java)
Ethereum Accounts
nonce
balance
codeHash
storageRoot
Transactions
recipient
signature
value
data
gasLimit
maxPriorityFeePerGas
maxFeePerGas
Blocks
timestamp
blockNumber
baseFeePerGas
difficulty
mixHash
parentHash
transactions
stateRoot
nonce
Networks
Ethereum Mainnet
Proof of Work Production network
Görli
Proof of Authority testnet
Sepolia
Proof of Work testnet
Private
Development networks
Consortium networks
Layer 2 testnets
Arbitrum Rinkeby
Optimistic Kovan
Top DeFi Protocols
MakerDAO (MKR)
Collateralized Debt Position
Lido (LDO)
Liquid Staking
Uniswap (UNI)
Decentralized Exchange
Curve (CRV)
Decentralized Exchange
AAVE (AAVE)
Lending
Convex Finance (CVX)
Yield
Compound (COMP)
Lending
Instadapp (INST)
Services
Arrakis Finance
Yield
Balancer (BAL)
Decentralized Exchange
eth2
Merge
Ethereum ---> Proof of Stake
Reduce power usage by 99%
Surge
Will bring sharding to Ethereum
and increase scalability
Verge
Will optimize storage, reduce
node size & improve scalable
Purge
Will eliminate historical data,
reduce hard drive space
needed by validators, streamline
storage & reduce network congestion
Splurge
misc upgrades to ensure that
the network runs smoothly
following the previous 4 stages
Rohas Nagpal

Setting up an Ethereum node

There are two main types of Ethereum nodes: "full nodes" and "light nodes."

Full nodes store the entire Ethereum blockchain, while light nodes rely on others to provide them with blockchain data.

This section provides a simplified guide to setting up an Ethereum full node.

Note: Running a full Ethereum node requires significant disk space, processing power, and a stable internet connection. Make sure your computer meets the system requirements before proceeding.

1. Install Ethereum Software

The most commonly used Ethereum client software is called "Geth."

You can download it from the official Ethereum website: https://geth.ethereum.org/downloads/

Follow the installation instructions for your specific operating system (Windows, macOS, Linux).

2. Syncing the Blockchain

After installation, open a terminal or command prompt.

After installation, open a terminal or command prompt.

Start syncing the Ethereum blockchain by running the following command:

geth --syncmode full

This command will download and verify the entire Ethereum blockchain. It can take a significant amount of time (several hours to days) depending on your internet speed and hardware.

3. Creating an Account

Once the blockchain is fully synced, you can create an Ethereum account with the following command:

geth account new

Follow the prompts to set a password for your account. Make sure to keep this password safe, as it's required to access your account.

4. Interacting with Your Node

You can interact with your Ethereum node using various tools and libraries. For example, you can use web3.js or web3.py for programming interactions.

You can also connect your Ethereum wallet (like MetaMask) to your local node by specifying its RPC address.

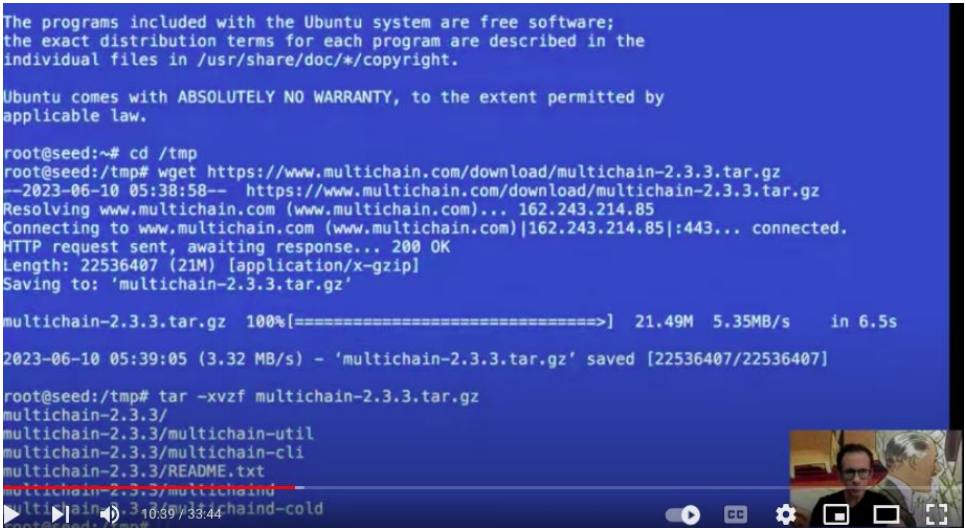
07

Multichain

Multichain is a framework for creating permissioned blockchains.

Practical 1: Setting up a permissioned blockchain

Watch the recording here: https://www.youtube.com/live/WX_LjGQdf5o?feature=share&t=89

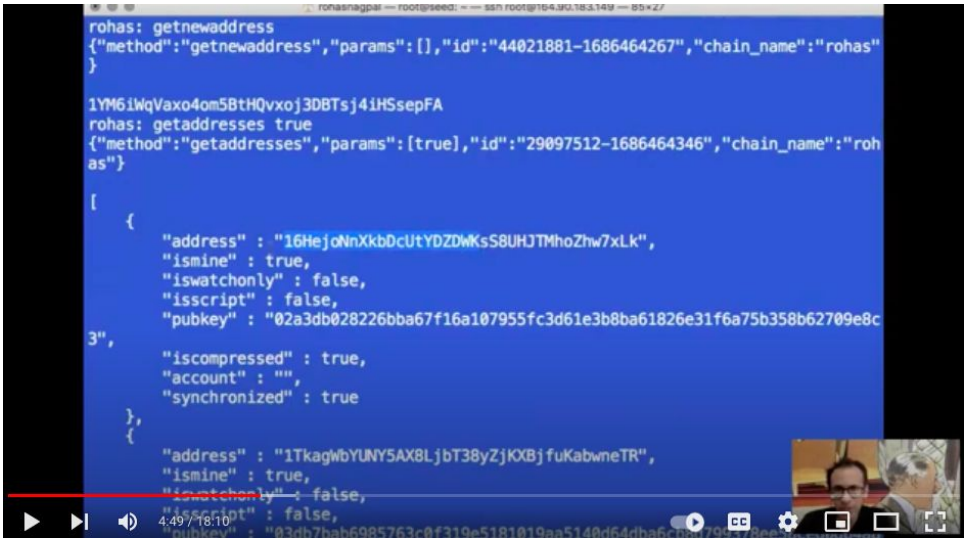


Use these instructions to set up the blockchain:

- https://www.multichain.com/download-community
- https://www.multichain.com/developers/creating-connecting

Practical 2: Addresses & Permissions

1. Creating custodial addresses
2. Creating non-custodial addresses
3. Managing permissions



https://www.youtube.com/live/mZk1K0NEUyw?feature=share&t=102

Address related commands covered in this video

getnewaddress
Returns a new custodial address whose private key is added to the wallet.

getaddresses true
Returns a list of addresses in this node's wallet with more information about each address,

getaddresses false
Returns a list of addresses in this node's wallet.

listaddresses
Returns information about the addresses in the wallet.

createkeypairs

Generates one or more non-custodial public / private key pairs, which are not stored in the wallet or drawn from the node's key pool, ready for external key management.

For each key pair, the address, pubkey (as embedded in transaction inputs) and privkey (used for signatures) is provided.

To generate multiple keypairs, use:
createkeypairs 2
createkeypairs 3
etc.

validateaddress

Returns information about address, or the address corresponding to the specified private key or public key, including whether this node has the address's private key in its wallet.

importaddress

Adds address (or an array of addresses) to the wallet, without an associated private key.

This creates one or more watch-only addresses, whose activity and balance can be retrieved via various APIs (e.g. with the includeWatchOnly parameter), but whose funds cannot be spent by this node.

Permissions related commands covered in this video

In MultiChain, there are many global permissions that can be granted to addresses:

connect
to connect to other nodes and see the blockchain's contents

send
to send funds

receive
to receive funds

issue
to create new native assets

create
to create data streams

mine
to create blocks

activate
to change connect, send and receive permissions for other users

admin
to change all permissions for other users, including issue, mine, activate and admin

For more on permissions, see:
https://www.multichain.com/developers/permissions-management

grant addresses permissions
Example:
grant 1bCrPbynVnRuqra8E1M1XdY5HFarKK2zuh2puf send,receive

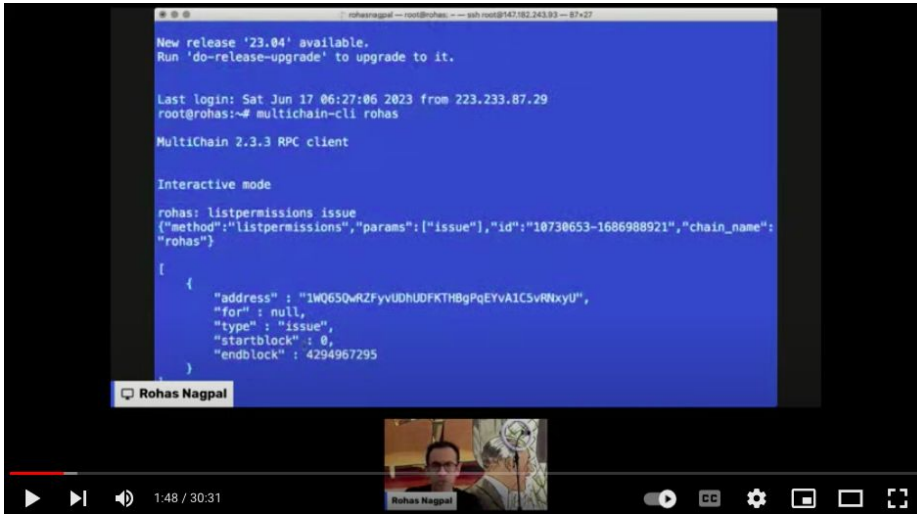
listpermissions
Returns a list of all permissions which have been explicitly granted to addresses.

revoke addresses permissions
Revokes permissions from addresses
Example:
revoke 1bCrPbynVnRuqra8E1M1XdY5HFarKK2zuh2puf receive

For more, see:
https://www.multichain.com/developers/json-rpc-api

Practical 3: Smart Asset Management

Watch the recording here:
https://www.youtube.com/live/u5N4fqMPn-c?feature=share&t=77



Step 1: Address with issue permission

We start by finding an address that can create assets. For this, run these 2 commands:

listpermissions issue
listaddresses

An address is suitable if it appears in the output of the first command, and also in the second command together with "ismine" : true.

If there is no address with issue permissions, you can give an address this permission using this:

grant <address> issue

Step 2: Issues the asset

Before issuing the asset, we need the following information:

- **Issue from:** The address from which the asset is to be created. This address must have *issue* permission.

- **Issue to:** The address to which the newly created asset should be sent. This address must have *receive* permission.

Depending upon the use case, the issue from and issue to addresses can be the same.

Once the asset is issued, it can be sent to other addresses as and when required.

- **Name:** Name of the asset - max 32 characters.

- **Open:** If more units can be issued in the future, set as *true*. Otherwise set as *false.*

- **Can open** and **Can close:** Whether to allow the asset to be closed and/or (re)opened for future issuance using the update command.

- **Restrict:** Whether per-asset send and receive permissions are to be enabled.

- **Total Limit:** The maximum total number of units which can be issued.

- **Issue Limit:** The maximum number of units per issuance.

- **Quantity being issued:** Number of tokens being issued in this transaction.

- **Asset description:** Details of the underlying asset Example:.
    - Category: Tokenized RWA
    - Subcategory: Aviation Asset
    - Type: Aircraft
    - AssetName: Cessna Skyhawk 172
    - Identifier: N123SP
    - TokenIssuer: HYFI Aviation
    - Custodian: ABC Custodian
    - ProspectusURL: www.example.com
    - RealTimeDataURL: www.example.com
    - Starts: 17-July-2023
    - Ends: NA

See the sample code for issuing this asset:
https://gist.github.com/rohasnagpal/610a906ff2954fe72c5f14b6959d8b11

Step 3: View asset details

listassets
displays details of all assets

listassets <identifier>
Displays details of asset having the specified:

- Name e.g. *Cessna-Skyhawk-172.8*
- Issuance transaction ID e.g. *269-266-32631*
- Asset reference number e.g. *777feac622d9af4c5ae91f8cdb62f3575e31530930feecf839e9febaa8567f63*

Step 4: Update asset params (Optional)

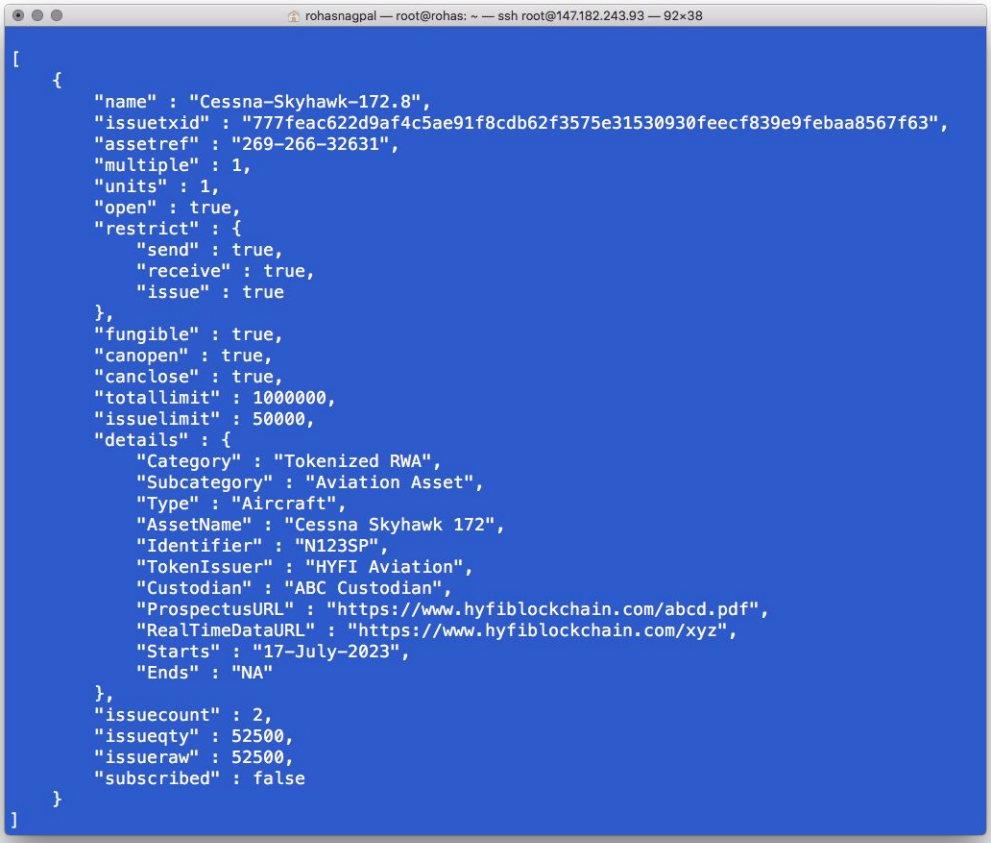
update Cessna-Skyhawk-172.8 '{"open":true}'

or

update 269-266-32631 '{"open":true}'

or

update 777feac622d9af4c5ae91f8cdb62f3575e31530930feecf839e9febaa8567f63 '{"open":true}'


rohasnagpal — root@rohas: ~ — ssh root@147.182.243.93 — 92×38
[
{
"name" : "Cessna-Skyhawk-172.8",
"issuetxid" : "777feac622d9af4c5ae91f8cdb62f3575e31530930feecf839e9febaa8567f63",
"assetref" : "269-266-32631",
"multiple" : 1,
"units" : 1,
"open" : true,
"restrict" : {
"send" : true,
"receive" : true,
"issue" : true
},
"fungible" : true,
"canopen" : true,
"canclose" : true,
"totallimit" : 1000000,
"issuelimit" : 50000,
"details" : {
"Category" : "Tokenized RWA",
"Subcategory" : "Aviation Asset",
"Type" : "Aircraft",
"AssetName" : "Cessna Skyhawk 172",
"Identifier" : "N123SP",
"TokenIssuer" : "HYFI Aviation",
"Custodian" : "ABC Custodian",
"ProspectusURL" : "https://www.hyfiblockchain.com/abcd.pdf",
"RealTimeDataURL" : "https://www.hyfiblockchain.com/xyz",
"Starts" : "17-July-2023",
"Ends" : "NA"
},
"issuecount" : 2,
"issueqty" : 52500,
"issueraw" : 52500,
"subscribed" : false
}
]

Step 5: Issue more tokens (Optional)

issuemorefrom 1WQ65QwRZFyvUDhUDFKTHBgPqEYvA1C5vRNxyU 1WQ65QwRZFyvUDhUDFKTHBgPqEYvA1C5vRNxyU Cessna-Skyhawk-172.8 2500 0 '{"Metadata":"Reasons for issuing more tokens e.g. token split"}'

Step 6: Per-asset permissions (Optional)

Examples:

grant <address> Cessna-Skyhawk-172.8.send,receive

revoke <address> Cessna-Skyhawk-172.8.send,receive

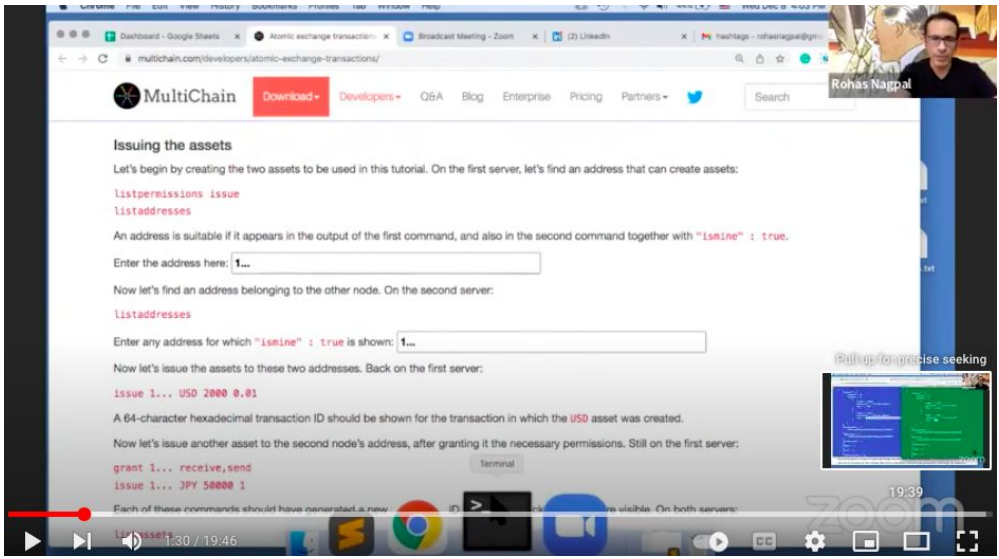
Step 7: Send assets

sendassetfrom <from-address> <to-address> <asset> <quantity>

All the commands are here: https://www.multichain.com/developers/json-rpc-api

Practical 4: Atomic Exchange Transactions

Watch the recording here:
https://www.youtube.com/live/li9guOpF9oI



The commands are here:

https://www.multichain.com/developers/atomic-exchange-transactions

Production blockchain parameters

These are the recommended settings for long-term deployment:
https://www.multichain.com/developers/blockchain-parameters/#production

Tips for performance optimization

These are the recommended server specifications and usage patterns:
https://www.multichain.com/developers/performance-optimization/

Upgrading nodes and chains

Keeping up to date with new MultiChain features and maintenance releases:
https://www.multichain.com/developers/upgrading-nodes-chains/

Backing up and restoring nodes

Being prepared for catastrophic system failures:
https://www.multichain.com/developers/backing-up-restoring-nodes/

Cold nodes and wallets

To keep private keys away from nodes connected to the network:
https://www.multichain.com/developers/cold-nodes-wallets/

Clustering for high availability

Minimizing downtime due to power cuts, system crashes or loss of connectivity:
https://www.multichain.com/developers/clustering-high-availability/

08

HYFI Blockchain

Hybrid Finance Blockchain (HYFI) is a Legally-compliant Permissioned Layer-1 Blockchain for the Tokenization of Real World Assets (Authentication, Provenance, Fractional Ownership, and Trading).

HYFI is tokenizing these assets:

1. Art: US$ 579.52 Billion
2. Carbon Credits: US$ 25.35 Billion
3. Copyright Licenses: US$ 172.5 Billion
4. Private Equity: US$ 11.7 Trillion
5. Rare Collectibles: US$ 370 Billion
6. Real Estate: US$ 326 Trillion
7. Structured Products: US$ 7 Trillion
8. Whisky Casks: US$ 300 Billion (5 years)

Website: https://www.hyfiblockchain.com/

8.1 HYFI is a permissioned blockchain

1. How HYFI provides Better Security:
As a permissioned blockchain, the HYFI network is limited to verified participants. This reduces the threat of malicious actors and hacking attempts.

2. How HYFI provides Improved Scalability:
As a permissioned blockchain, HYFI can handle more transactions per second than many public blockchains. This makes HYFI suitable for high-volume applications.

3. How HYFI provides Enhanced Privacy:
As a permissioned blockchain, HYFI members have control over who can access the network. This helps to keep sensitive data confidential.

4. How HYFI provides Faster Consensus:
HYFI uses a distributed consensus between identified block validators. This is similar to PBFT (Practical Byzantine Fault Tolerance). There is one validator per block, working in a round-robin type of fashion.

5. How HYFI provides Efficient Resource Management:
As a permissioned blockchain, HYFI optimizes the use of network resources, reducing costs and improving performance.

6. How HYFI provides Customizable Rules:
As a permissioned blockchain, HYFI tailors the network's rules and governance/ This enables greater flexibility and control.

7. How HYFI provides Legal Compliance:
As a permissioned blockchain, HYFI is designed to comply with regulatory requirements - KYC, AML, CFT, Consumer Protection, Data Privacy,
Right-to-be-forgotten Regulations, and Freezing of assets.

8.2 Legal & regulatory compliance

Hybrid Finance (HYFI) Blockchain supports regulatory compliance in 7 ways:

1. How HYFI supports KYC (Know Your Customer)

HYFI nodes are operated by verified entities from FATF-compliant jurisdictions.

Regulators nodes are available for real-time monitoring of all activities on the HYFI network.

HYFI is a permissioned blockchain and each address is given permissions based on the level of KYC.

2. How HYFI supports AML (Anti-Money Laundering)

HYFI nodes are operated by verified entities from FATF-compliant jurisdictions.

These entities have robust policies for customer identification & verification, risk assessment, and monitoring & reporting of suspicious transactions.

Regulators nodes are available for real-time monitoring of all activities on the HYFI network.

3. How HYFI supports CFT (Countering the Financing of Terrorism)

HYFI nodes are operated by verified entities from FATF-compliant jurisdictions. These entities have robust policies for due diligence, transaction monitoring & suspicious activity reporting.

Regulators nodes are available for real-time monitoring of all activities on the HYFI network.

4. How HYFI supports Freezing of Assets

Assets can be frozen on the HYFI network, through the use of specific permissions, based on orders from courts & enforcement agencies.

This enables added security, fraud minimization, and control over the movement & usage of assets within the network.

5. How HYFI supports Consumer Protection

Consumer protection is provided through responsible address allocation & permission management.

The allocation of custodial addresses and management of permissions reduces the risk of unauthorized access & loss of funds, providing a safer experience for users.

Role-based asset controls allow for business, compliance, and regulatory oversight.

6. How HYFI supports Right-to-be-forgotten Regulations

Off-Chain Data Purging allows for the selective removal of off-chain data from local storage to meet right-to-be-forgotten regulations.

7. How HYFI supports Data Privacy

Stream Read Restrictions enable publishing and retrieving items only visible to nodes with appropriate permissions.

Fully encrypted peer-to-peer connections prevent intermediate routers from seeing any private data.

8.3 HYFI Security Features

Hybrid Finance (HYFI) Blockchain has several security features to protect users and their data:

1. Only authorized users and nodes can participate in the network and access its data, making it secure from unauthorized access.

2. The P2P connections in HYFI Blockchain are fully encrypted, preventing intermediate routers from seeing any private data.

3. HYFI Blockchain offers flexible private key management options, including support for external private keys and multi-signatures for all transactions.

4. In the HYFI blockchain, cold nodes play a crucial role in enhancing the system's overall security. By keeping private keys offline, users can minimize the risk of theft or unauthorized access to their assets.

5. HYFI supports full multi-signature support and external key management, which enables users to securely manage their assets using hardware security modules (HSMs). This allows users to store their private keys in a secure and encrypted manner, which provides an additional layer of protection against potential threats.

8.4 HYFI Scalability Features

Some of the scalability features of Hybrid Finance (HYFI) Blockchain are:

1. HYFI supports millions of addresses, assets, streams, and unlimited transactions / stream items, and can handle unlimited nodes in the network.

2. With the capability to process more than 1000 transactions per second, HYFI can handle large volumes of transactions.

3. HYFI blockchain's block time is as low as 2 seconds, which helps to reduce latency & improve overall efficiency.

4. The HYFI Blockchain provides ample storage space with each transaction being able to store up to 64 MB of data.

5. Streams in the system support various data structures such as key-value, identity, and time series, making it easy to store and search for information.

6. The scalability of the HYFI Blockchain is improved through Selective Stream Indexing and Selective Data Retrieval.

7. With Selective Stream Indexing, the indexing of streams can be controlled for enhanced performance & reduced disk usage.

8. Selective Data Retrieval allows for the specific selection of off-chain items to be retrieved from the network, conserving bandwidth and disk space.

8.5 HYFI Cold Nodes

A blockchain cold node is a node that is offline or disconnected from the network. It does not actively participate in the validation of transactions or blocks.

Cold nodes are used for the storage of sensitive information, such as private keys, which are crucial for accessing and managing blockchain assets.

In the Hybrid Finance (HYFI) Blockchain, cold nodes play a crucial role in enhancing the system's overall security. By keeping private keys offline, users can minimize the risk of theft or unauthorized access to their assets.

8.6 Integrating HYFI with other applications

Hybrid Finance (HYFI) Blockchain can be integrated with other applications through its unified JSON-RPC API.

The API for customer applications in HYFI works by providing a secure and convenient interface for developers to interact with the blockchain and build custom applications.

The API is designed to cleanly separate the application from the node and is compatible with any API library developed for Bitcoin Core.

This makes it easier for developers to build applications that interact with the HYFI blockchain and add new functionality to the platform.

The API allows developers to easily access and utilize the various functions and services available in HYFI, such as issuing and transferring assets, managing permissions, and retrieving information about the blockchain and its transactions.

By providing a simple and flexible API, HYFI enables developers to create a wide range of custom applications, from simple web interfaces to complex decentralized applications (dApps), that can interact with the blockchain in a secure and efficient manner.

8.7 Freezing of funds on HYFI

Funds can be frozen in Hybrid Finance (HYFI) Blockchain through the use of specific permissions and conditions set by the address owner or administrators of the blockchain network.

This allows for added security, fraud minimization, and control over the movement & usage of assets within the network.

8.8 HYFI Data Streams

Data streams are a way to securely publish & retrieve items in the blockchain.

These items can be visible only to nodes with the appropriate permissions, and they allow for the creation of channels where specific groups of participants can access and interact with certain data.

Data streams provide a mechanism for organizing & managing data within the blockchain in a secure and scalable manner, making it possible for a wide range of use cases, including but not limited to, content distribution, copyright licensing, and secure data sharing.

Technically speaking, data streams enable a blockchain to be used as a general-purpose append-only database, with the blockchain providing time stamping, notarization & immutability.

The Hybrid Finance (HYFI) Blockchain provides ample storage space with each transaction being able to store up to 64 MB of data.

Streams in the system support various data structures such as key-value, identity, and time series, making it easy to store and search for information.

8.9 HYFI Smart Filters

A Smart Filter is a piece of code that is embedded in the blockchain, and which allows custom rules to be defined regarding the validity of transactions or stream items.

Smart Filters are written in JavaScript and run within a deterministic version of Google's V8 JavaScript engine, which is embedded directly within MultiChain 2.

This is the same JavaScript engine used in Chrome, Node.js, and many other platforms. It offers excellent performance by compiling JavaScript to machine code and optimizing that code as it runs.

See the HYFI Documentation for:

1. HYFI Nodes
2. Addresses
3. Tokenizing Assets on HYFI Blockchain
4. Transactions

09

Hyperledger

Hyperledger is an open-source initiative for growing enterprise use of blockchain technologies.

Hyperledger projects include:
- 5 Distributed Ledgers
- 4 Libraries
- 6 Tools

Distributed Ledgers

Distributed Ledgers are **multiparty databases** with no central trusted authorities.

Hyperledger has **5 Distributed Ledgers**:
1. Besu
2. Fabric
3. Indy
4. Iroha
5. Sawtooth.

Libraries

Libraries are code bases that add functionality to enterprise blockchain use cases e.g. digital credentials, smart contracts & cryptographic code.

Hyperledger has **4 Libraries**:
1. AnonCreds
2. Aries
3. Transact
4. Ursa

Tools

Tools improve interoperability, performance & security in enterprise blockchains.

Hyperledger has **6 Tools** -

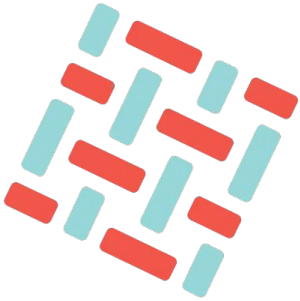
1. Bevel
2. Cacti
3. Caliper
4. Cello
5. Firefly
6. Solang

HYPERLEDGER
BESU

Hyperledger Besu

Hyperledger Besu is an Ethereum client that is optimized for use in enterprise environments and can be used for both public and private permissioned networks.

- It includes several consensus algorithms & permissioning schemes.
- It can run on various test networks.
- It is flexible & modular, with an extractable EVM implementation.
- It is suitable for consortium environments.



HYPERLEDGER
FABRIC

Hyperledger Fabric

Hyperledger Fabric is a modular blockchain platform with plug-and-play components such as consensus & membership services.

It can be used to develop applications for a broad range of enterprise use cases.

It enables high performance & preserves privacy.

Hyperledger Indy

Hyperledger Indy is a set of tools, libraries & reusable components for creating interoperable digital identities.

Indy enables the decentralization of identity.

Indy can be used standalone or integrated with other blockchain systems.

HYPERLEDGER
IROHA

Hyperledger Iroha

Hyperledger Iroha is a blockchain platform designed for simplicity & ease of integration into infrastructural or IoT projects.

It features a simple construction, modular, domain-driven C++ design.

Its focus is on client application development.

It has a crash fault-tolerant consensus algorithm called YAC.

Hyperledger Sawtooth

Hyperledger Sawtooth is a flexible & modular blockchain platform that separates the core system from the application domain.

It allows smart contracts to define business rules for applications without needing knowledge of the underlying core system design.

It supports multiple consensus algorithms such as Practical Byzantine Fault Tolerance (PBFT) and Proof of Elapsed Time (PoET).

AnonCreds

Hyperledger AnonCreds is a type of verifiable credential that utilizes zero-knowledge proof cryptography to support advanced privacy-protecting capabilities.

Aries

Hyperledger Aries is a toolkit for creating, transmitting & storing verifiable digital credentials in a p2p blockchain-rooted environment.

Transact

Hyperledger Transact provides a standard interface for executing smart contracts, separate from the underlying ledger implementation. It supports multiple smart contract engines.

Ursa

Hyperledger Ursa is a shared cryptographic library that provides a centralized repository for cryptographic code & interfaces.

Bevel

Hyperledger Bevel is an accelerator for deploying production-ready distributed networks across public & private cloud providers.

Cacti

Hyperledger Cacti is a blockchain integration tool enabling the integration of different blockchains.

Caliper

Hyperledger Caliper is a blockchain benchmark tool for measuring the performance of a blockchain implementation with a set of predefined use cases.

Cello

Hyperledger Cello is an operational console for managing blockchains running on bare-metal, virtual machine & container platforms. It can also be used for creating Blockchain as a Service.

HYPERLEDGER
FIREFLY

Firefly

Hyperledger FireFly is a complete stack for building & scaling secure Web3 applications.

Solang

Hyperledger Solang compiles Solidity for Solana & Substrate.

10

Blockchain & Web3 Tech Stack

1. Akash Network

Akash Network is called the "Airbnb for Cloud Compute". It leverages "85% of underutilized cloud capacity in 8.4 million global data centers, enabling anyone to buy and sell cloud computing".

The platform claims to provide cloud computing at 1/3 the cost of providers like Amazon, Google, and Microsoft.

Get started: https://akash.network

2. Alchemy

- Alchemy Supernode is a blockchain engine that ensures scalability, reliability and data accuracy.
- Alchemy Web3 APIs simplify and optimize common request patterns.
- Alchemy enables the sending of real-time notifications to users for critical events like mined and dropped transactions.
- Alchemy Explorer enables searching through millions of historical requests to identify error patterns and optimize performance.
- Alchemy Composer enables prototyping and fixing failing requests or exploring the behavior of new methods, directly from the dashboard.

- Mempool Visualizer enables troubleshooting transactions that may be delayed, stuck or dropped in the mempool.

- You can track requests by method and geo-location, without sacrificing privacy or security.

Get started: https://www.alchemy.com

3. API3

API3 provides secure and cost-efficient blockchain-native, decentralized APIs (dAPIs).

Beacons enable connecting Web3 applications to continuously updated streams of off-chain data e.g. latest cryptocurrency, stock and commodities prices.

Managed Oracle Services - dAPIs are data feeds built entirely on-chain from first-party, API provider-operated oracles with decentralized governance.

API3 provides Quantum Random Numbers for Smart Contracts for free. You need to pay for the gas.

Get started: https://api3.org

4. Aragon
Aragon App is a simple, modular and adaptable DAO platform.

Aragon Client is a DAO platform for communities to raise funds, pay contributors, and govern together. Aragon Voice is a gasless and universally verifiable voting solution for web3.

Vocdoni is a secure, end-to-end verifiable, and censorship-resistant voting solution for organizations.

Get started: https://aragon.org

5. Arweave

Arweave is "a collectively owned hard drive that never forgets. It allows the permanent storage of data with a single upfront fee.

Arweave is a decentralized storage network for the indefinite storage of data. At its core is "permaweb" - a "permanent, decentralized web with applications and platforms like UI hosting, database writes & queries, and smart contracts.

Miners are paid in Arweave's native cryptocurrency, AR, to indefinitely store information.

Get started: https://www.arweave.org

6. Audius

Audius is a decentralized music streaming protocol that aims to give "everyone the freedom to distribute, monetize, and stream any audio content".

Audius is built on both Ethereum and Solana.

$AUDIO is the native token that enables network security, exclusive feature access, and community-owned governance.

$AUDIO can be staked to run discovery or content nodes. Artists can also stake $AUDIO to unlock artist tokens and badges, and to "receive voting power from fans who want to share in their success".

Get started: https://audius.co

7. Band Protocol

Band Protocol is a decentralized cross-chain data oracle platform that aggregates and connects real-world data and APIs to smart contracts.

The BandChain Oracle solution is a middle layer operating between dApps and multiple data providers.

Get started: https://bandprotocol.com

8. Basic Attention Token

Basic Attention Token (BAT) is disrupting the $330 billion digital advertising industry.

Here's how the BAT ecosystem works:

- Users can earn BAT for viewing ads while maintaining privacy.

- Content creators earn ad revenue, user contributions, and tips.

- Advertisers get a better return on investment and know their ads' effectiveness without violating the privacy of users.

Brave Browser is at the heart of the BAT ecosystem. Brave blocks ads and trackers. This makes your browsing 3 times faster and very private.

Brave Wallet is the first crypto wallet that is built directly into a browser. This is unlike Metamask which is a browser extension. Because of this, Brave wallet is less vulnerable to faked versions and phishing.

Get started: https://basicattentiontoken.org

9. Chainlink

Chainlink is a decentralized network of independent oracle node operators. It provides:

- price feeds of financial market data,

- verifiable randomness that is needed for on-chain gaming,

- proof of reserve for asset-backed cryptos such as stablecoins.

How can a smart contract get data from the outside world? That's the problem that Oracles solve. They act as middleware between smart contracts and external sources of data.

Software Oracles handle data that originates from online sources e.g. temperature, prices of commodities and goods, flight delays.

Hardware Oracles get data from the physical world (e.g. from IoT devices) and are popular in the supply chain industry.

Inbound Oracles provide data from the external world to the blockchain while Outbound Oracles enable smart contracts to send data to the outside world.

Consensus-based Oracles get their data from human consensus and prediction markets e.g. Augur, Gnosis, etc.

Chainlink is NOT a blockchain.

Each Chainlink oracle network comprises multiple independent oracle nodes. These nodes fetch data from multiple independent data providers.

This data is then aggregated into a single data point and delivered "on-chain" for consumption by smart contracts.

LINK is the crypto token that is used for paying Chainlink node operators for providing oracle services.

Get started: https://chain.link

10. Civic Pass

Civic Pass is an integrated "permissioning tool" that helps businesses control access to their dApps.

It is used by crypto-native communities, NFT platforms & marketplaces, and DAOs.

Get started: https://www.civic.com

11. DeFi Llama

DefiLlama is a DeFi TVL (Total Value Locked) aggregator.

Get started: https://defillama.com

12. Drizzle

Drizzle is a collection of front-end libraries that make creation of dApp user interfaces easier.

Get started: https://trufflesuite.com/drizzle

13. Ethereum Name Service

Ethereum Name Service (ENS) is a decentralized naming system based on the Ethereum blockchain. It converts human-readable Ethereum addresses like *sanya.eth* into crypto addresses like *1Mk13r5uu51F5jQ6yGuBPxkuZw91nM4MeY* and vice-versa.

Get started: https://ens.domains

14. Filecoin

Filecoin is the Airbnb for data.

It is a decentralized data storage network where excess storage can be bought and sold.

It is also the incentive & security layer for InterPlanetary File System (IPFS). In simple words, it is a marketplace for unused storage - in consumer hardware and data centers.

In the conventional world, cloud service providers like Amazon Web Services (AWS) and Microsoft Azure provide centralized servers and IP addresses for user data.

In the blockchain world, Filecoin uses hash-addressed content structures to reduce redundancy and increase efficiency.

Filecoin is integrated with Ethereum. This enables developers to access Ethereum blockchain data and interact with Ethereum smart contracts.

Filecoin (FIL) is the native crypto of the Filecoin network. It can be used to pay miners to store/distribute data and to retrieve information. Storage providers guarantee a minimum service level by providing FIL as collateral.

Its consensus mechanism is Proof-of-replication (PoRep) and Proof-of-spacetime (PoSt).

PoRep enables storage miners to prove that they are physically storing a unique copy of client data. PoSt proves that storage miners are continuing to dedicate storage space to client data over time.

Get started: https://filecoin.io

15. Ganache

Ganache is a personal blockchain for rapid Ethereum and Corda dApp development. Ganache can be used across the entire development cycle.

Ganache comes in two flavors - a desktop application supporting Ethereum & Corda, and a command-line tool for Ethereum development.

Get started: https://trufflesuite.com/docs/ganache

16. Gnosis Safe

An externally owned account (EOA) is a single key wallet which is commonly used by crypto holders using MetaMask,Trustwallet, etc. An EOA is secured with a "seed phrase" which generates private keys. If the key is compromised, the crypto can be stolen.

Gnosis Safe is a multisig smart contract wallet that requires a minimum number of people (m-of-n) to approve a transaction before it can take place e.g. 3 out of 5 people.

Gnosis Safe supports Ether (ETH), ERC20 tokens and ERC721 NFTs. You can sign transactions using mobile wallets, browser extensions, and hardware wallets.

Gnosis Safe can be accessed in a browser, on the desktop, and on mobile.

Get started: https://gnosis-safe.io

17. Helium network

Internet of Things (IoT) is a multi-trillion dollar industry, with billions of connected devices. For proper functioning, most IoT devices need to be connected to the Internet.

The conventional technologies for this (cellular, WiFi, and Bluetooth) are expensive, power-hungry, or have a limited range.

The Helium network is a decentralized wireless network. It enables IoT devices to wirelessly connect to the Internet and geolocate themselves without satellite location hardware or cellular plans.

Get started: https://www.helium.com

18. Infura

Infura provides tools & infrastructure for dApp testing and scaling. It provides simple access to Ethereum and IPFS.

Infura supports Ethereum mainnet and testnets (Rinkeby, Ropsten, Kovan, Görli), IPFS, Filecoin (Beta), Eth2 Beacon Chain (Beta), Polygon PoS (Beta), Optimism Ethereum, and Arbitrum Rollup.

Infura is natively supported in the Azure Blockchain Development Kit extension for VS Code.

Infura uses the following clients: Geth (Go Ethereum), OpenEthereum, Hyperledger Besu, Teku, Lotus, Go-IPFS, Bor/Heimdall, l2geth, and Arbitrum.

Get started: https://infura.io

19. IPFS

InterPlanetary File System (IPFS) is a distributed system for storing and accessing files, websites, applications, and data.

Traditional URLs & file paths identify a file by where it's located.

Examples:
https://www.rohasnagpal.com/web3/index.html
file:///Users/sanyanagpal/Documents/hyfi.pdf

IPFS addresses a file by its content. The content identifier is a cryptographic hash of the content at that address. e.g.
bafkreih2jxpb4zewzqug3uk6oak5nzsrc6oz6sjhzonn7bktjde6xks7fm

The hash is unique to the content and enables verification of the data.

- To try it out, create a free account at: https://nft.storage

- Upload a file and copy the ipfsHash or CID (content identifier for the file) e.g. *bafkreih2jxpb4zewzqug3uk6oak5nzsrc6oz6sjhzonn7bktjde6xks7fm*

- To download the file, add https:// to the beginning and .ipfs.w3s.link to the end of the CID or ipfsHash. Example: *https://bafkreih2jxpb4zewzqug3uk6oak5nzsrc6oz6sjhzonn7bktjde6xks7fm.ipfs.w3s.link*

Get started: https://nft.storage

20. Livepeer

Video streaming, especially live streaming, is very expensive because video has to be transcoded first. That's the process of reformatting a raw video file so that it can be viewed well on all devices and bandwidths.

Livepeer is a protocol for reducing transcoding costs up to 50x. This is achieved by peer-to-peer infrastructure which interacts through an Ethereum-based marketplace.

Orchestrators can earn fees by contributing CPU, GPU, and bandwidth for transcoding and distributing video. The Livepeer token (LPT) is required for this. The more LPT you hold, the more work you can perform and the more fees you earn.

LPT holders can also earn fees by staking their tokens towards orchestrators.

Get started: https://livepeer.org

21. Moralis

Moralis Web3 API enables quick fetching block info, transaction info, NFT metadata, token prices, user balances, owner list of a particular NFT and other EVM blockchain data.

Get started: https://moralis.io

22. NuCypher

NuCypher is a decentralized encryption, access control, and key management system for public blockchains. NuCypher offers end-to-end encrypted data sharing on public blockchains and decentralized storage solutions.

Get started: https://www.nucypher.com

23. Ocean Protocol

Ocean Protocol enables individuals and businesses to monetize their data through ERC-20 based datatokens. Datasets can be discovered and traded on the Ocean Market.

Get started: https://oceanprotocol.com

24. Ontology

Ontology is a high-speed, low-cost public blockchain for bringing decentralized identity and data solutions to Web3.

Get started: https://ont.io

25. OpenZeppelin

OpenZeppelin Contracts minimize risk by using battle-tested libraries of smart contracts for most used implementations of ERC standards.

OpenZeppelin Defender manages smart contract administration including access controls, upgrades, and pausing. It Works with popular multi-sigs including Gnosis Safe.

OpenZeppelin Contracts Wizard is an interactive generator to bootstrap smart contracts.

Get started: https://www.openzeppelin.com

26. Push Protocol

Push Protocol is a decentralized blockchain-agnostic communication protocol providing notifications for on-chain & off-chain activity.

Push notifications can be triggered via smart contract, backend, and dApps. These come from an open communication network (push nodes) and are tied to wallet addresses..

Get started: https://push.org

27. Render Network

Render Network is a distributed graphics processing unit (GPU) rendering network built on top of the Ethereum blockchain. It connects artists and studios in need of GPU compute power with mining partners willing to rent their GPU capabilities.

Get started: https://rendertoken.com

28. Sia

Sia is a decentralized cloud storage platform and a data storage marketplace.

It encrypts and distributes files across a decentralized network. Users control their private encryption keys and own their data. This is unlike traditional cloud storage providers.

Sia's storage can cost upto 90% less than leading traditional cloud storage providers. These charges are paid using the native token Siacoin (SC).

According to the project website, storing 1TB of files on Sia costs about $1-2 per month, compared with $23 on Amazon S3.

Here's how it works:

Step 1: Files are divided into 30 segments - The Sia software divides each file into 30 segments using "Reed-Solomon erasure coding" which is also used in CDs and DVDs. The data can be recovered fully using just 10 of the 30 segments. This means that even if 20 of the 30 hosts go down, the data can be recovered.

Step 2: Each file segment is encrypted and then distributed - Each file segment is encrypted using the open-source Threefish algorithm before it is distributed to hosts across the world.

Step 3: Files are sent to hosts using smart contracts - Storage renters enter into file contracts (smart contracts) with hosts for pricing, uptime commitments, etc. Service Level Agreements (SLAs) are stored on the Sia blockchain and automatically enforced by the network.

Renters and hosts use Siacoin (SC) for payments. File contracts are usually for 90 days and hosts provide proof of storage using Merkle trees.

Get started: https://sia.tech

29. Steemit

Steemit is a blockchain-based blogging and social media platform. Users earn crypto for publishing and curating content.

Get started: https://steemit.com

30. The Graph

The Graph is a decentralized protocol for indexing and querying data from blockchains. Developers can build and publish open APIs (subgraphs) that make data easily accessible.

The Graph is currently used by many projects including Uniswap, Synthetix, and Decentraland.

Get started: https://thegraph.com/en

31. Theta

Theta is like Airbnb for video streaming - viewers earn rewards for sharing excess bandwidth and computing resources.

Theta's benefits are:

- Users earn rewards for sharing excess bandwidth & computing resources.
- Viewers get better quality streaming services.
- Content creators improve their earnings.
- Video platforms don't have to build expensive infrastructure.
- Video platforms can increase revenues.
- Theta also has smart contract capability for fully digitized item ownership, payment-consumption models, transparent royalty distributions, etc. Theta Enterprise Validators include Google, Binance, Sony Europe, and Samsung.

The Theta blockchain has two native tokens:

- Theta (THETA) for governance, and
- Theta Fuel (TFUEL) for powering transactions.

Get started: https://www.thetatoken.org

32. TokenTerminal

Token Terminal is a platform that aggregates financial data on the leading blockchains and dApps.

Get started: https://tokenterminal.com

33. Truffle

Truffle is a development environment, testing framework and asset pipeline for EVM blockchains.

Get started: https://trufflesuite.com/truffle

This video covers:

1. Web Storage
2. Web Workers
3. Server-Sent Events
4. Hash calculations
5. Password Security using Salt & Pepper
6. Cryptographically strong pseudorandom number generator (CSPRNG)
7. Encryption & Decryption

https://www.youtube.com/live/68u-GVvdbds?feature=share&t=74

11

ChatGPT Super Prompt Templates

ChatGPT Super Prompt Templates

Here are some ChatGPT super prompt templates that you can customize and use:

1. Troubleshooting smart contract code

I am currently developing a smart contract and require your expertise in troubleshooting some issues I have encountered. The purpose of the smart contract is [briefly describe the purpose or functionality], and it is built on the [specify the platform, e.g., Ethereum, Binance Smart Chain] platform. Here is the relevant code snippet where I am facing difficulties: [provide the code snippet(s)]. My concerns encompass functionality, where I have observed [describe problems or unexpected behavior]; security, with suspected vulnerabilities such as [mention security concerns]; gas consumption, particularly in relation to [explain concerns about excessive gas usage or optimization needs]; and code readability, specifically in areas like [highlight areas where the code is difficult to understand or maintain]. Given this information, identify the root cause of these issues and propose suitable solutions or optimizations. Additionally, provide any general best practices or recommendations for enhancing my smart contract.

2. Building scalable decentralized applications (dApps)

In need of guidance for building scalable decentralized applications (dApps), the following information is provided: the dApp's purpose [briefly describe the purpose or functionality], the target platform [specify the platform, e.g., Ethereum, Binance Smart Chain], and the intended user base [describe the target users or market]. Additionally, the dApp must address challenges such as [list specific scalability concerns or constraints]. Considering this context, please offer expert insights into the architecture, technologies, and best practices for developing a scalable dApp, addressing aspects like efficient smart contract design, off-chain data management, layer 2 scaling solutions, and other pertinent factors that contribute to the dApp's performance, security, and user experience.

3. Advising on token design & implementation

Deliver strategic and well-structured advice on token design and implementation for a [specific blockchain platform or project], tailored to [target audience, e.g., developers, project managers, or stakeholders], incorporating the following key elements: clear token utility and purpose, [specific token design principles or best practices], relevant and context-appropriate token distribution and governance models, and effective evaluation methods to assess the impact and benefits of the token design on the overall ecosystem and user experience.

4. Suggesting improvements to existing blockchain systems

Provide a comprehensive and innovative improvement proposal for an existing blockchain system [specify the blockchain], designed for [target audience, e.g., developers, stakeholders, or users], incorporating the following key elements: clear problem identification, [specific improvement technique or strategy], relevant and context-appropriate use cases, and effective evaluation methods to assess the impact and benefits of the proposed improvements.

5. Providing recommendations for tooling and development frameworks

Offer insightful and practical recommendations for tooling and development frameworks in the context of [specific technology or programming language], designed for [target audience, e.g., developers, project managers, or stakeholders], incorporating the following key elements: clear requirements analysis, [specific evaluation criteria or comparison methodology], relevant and industry-appropriate tools and frameworks, and effective implementation strategies to ensure seamless integration and adoption within the development process.

6. Choosing blockchain consensus mechanisms

Present a comprehensive and well-reasoned guide for choosing the most suitable blockchain consensus mechanism for a [specific blockchain project or use case], designed for [target audience, e.g., developers, project managers, or stakeholders], incorporating the following key elements: clear understanding of different consensus mechanisms and their trade-offs, [specific evaluation criteria or decision-making framework], relevant and context-appropriate comparisons of existing consensus mechanisms, and effective implementation strategies to ensure seamless integration and optimal performance of the chosen consensus mechanism within the blockchain system.

7. Best practices for building secure smart contracts

Deliver an exhaustive and actionable guide on best practices for building secure smart contracts on [specific blockchain platform or programming language], tailored for [target audience, e.g., developers, project managers, or stakeholders], incorporating the following key elements: clear understanding of common security vulnerabilities and risks, [specific secure coding techniques or principles], relevant and context-appropriate tools and frameworks for smart contract security analysis, and effective testing and auditing methodologies to ensure the robustness and reliability of the smart contract code against potential attacks and exploits.

8. Implementation of blockchain-based payment systems

Provide a detailed and practical blueprint for implementing a blockchain-based payment system within [specific business or industry context], designed for [target audience, e.g., developers, project managers, or stakeholders], incorporating the following key elements: clear understanding of payment system requirements and regulations, [specific blockchain platform or technology], relevant and context-appropriate transaction processing and settlement mechanisms, and effective security and scalability measures to ensure a robust, efficient, and user-friendly blockchain-based payment system.

9. Building decentralized finance (DeFi) protocols

Provide an in-depth and comprehensive guide for building decentralized finance (DeFi) protocols on [specific blockchain platform], designed for [target audience, e.g., developers, project managers, or stakeholders], incorporating the following key elements: clear understanding of DeFi concepts and financial primitives, [specific DeFi protocol design principles or best practices], relevant and context-appropriate smart contract templates and governance models, and effective risk management and security measures to ensure the stability, interoperability, and user trust in the developed DeFi protocol.

10. Design & implementation of blockchain-based gaming systems

Present a detailed and engaging blueprint for the design and implementation of blockchain-based gaming systems within [specific gaming genre or platform], tailored for [target audience, e.g., developers, project managers, or stakeholders], incorporating the following key elements: clear understanding of gaming mechanics and player incentives, [specific blockchain technology or tokenization strategy], relevant and context-appropriate integration of decentralized assets and game economy, and effective performance optimization and security measures to ensure an enjoyable, fair, and immersive gaming experience for users.

11. Securing an Ethereum Node

Prepare a detailed checklist with relevant commands and code for securing an Ethereum node. Include physical security measures, operating system security, application security, system administrator security, account security, monitoring procedures, and encryption methods. The checklist must be comprehensive and based on the best practices, tools, and techniques to secure each area to protect the server from unauthorized access, data breaches, and other security threats.

12. Simulating a Bitcoin node

I want you to act as a Bitcoin full node using Bitcoin Core. I will type commands and you will reply with what the node should show. I want you to only reply with the terminal output inside one unique code block, and nothing else. Do not write explanations. Do not type commands unless I instruct you to do so. When I need to tell you something in English I will do so by putting text inside curly brackets {like this}. My first command is bitcoin-cli getnewaddress.

12

Interview Questions

12.1 Blockchain Basics

1. How would you explain blockchain technology to someone without a technical background?

2. What are the key features that differentiate blockchain from traditional database systems?

3. Can you explain the basic structure and operation of a blockchain node?

4. What are some common consensus mechanisms used in blockchain systems? Can you discuss the pros and cons of each?

5. What is a blockchain bridge and what is its role in interoperability between different blockchains?

6. Could you explain the concept of merged mining?

7. Can you differentiate between hard and soft forks in a blockchain?

8. What are the different types of blockchain (public, private, consortium, etc.) and in what scenarios might each type be used?

9. What are blockchain wallets and addresses? How do they work?

10. Can you discuss some common use cases for blockchain technology?

12.2 Blockchain APIs

1. Can you explain what a Blockchain API is and how it differs from traditional APIs?

2. What are some common use cases for Blockchain APIs? Can you provide a specific example where a Blockchain API significantly improves a process or system?

3. What are the key security considerations when using Blockchain APIs? How do they address common security challenges in data transactions?

4. What challenges might a developer face when integrating a Blockchain API into an existing system? How would you approach these challenges?

5. Are there differences in APIs provided by various blockchain platforms like Ethereum, Hyperledger, or others? Can you give an example?

6. How do Blockchain APIs interact with smart contracts? Can you walk me through the process of deploying a smart contract using an API?

7. What metrics would you use to evaluate the performance of a Blockchain API? How do these metrics impact the choice of a particular API?

8. How do Blockchain APIs handle scalability and efficiency issues, especially in systems with high transaction volumes?

9. How does a Blockchain API handle the concepts of decentralization and consensus? Can you explain this with an example of a transaction process?

10. Where do you see the future of Blockchain APIs heading, and what new developments or trends should we be aware of?

12.3 Network Security & Privacy

1. Can you discuss some common vulnerabilities in blockchain networks and how to protect against them?

2. What are some best practices for maintaining the security of a blockchain node?

3. What tools would you use to monitor the performance and security of a blockchain network?

4. Can you name and describe some privacy-enhancing technologies used in blockchain?

12.4 Node Maintenance & Optimization

1. What are some key considerations for blockchain data storage and management?

2. What metrics would you use to assess the performance of a blockchain node?

3. How would you approach performance tuning and optimization for a blockchain node?

4. What are some best practices for backing up blockchain data and ensuring disaster recovery?

12.5 Bitcoin

1. Can you explain what Bitcoin is and how it differs from traditional fiat currencies?

2. How does the underlying blockchain technology of Bitcoin work, and why is it important for Bitcoin's functionality?

3. What is Bitcoin mining and how does it contribute to the Bitcoin network?

4. How does Bitcoin's decentralized nature contribute to its security and integrity?

5. Can you explain the process of a Bitcoin transaction from one wallet to another? What role do private and public keys play in this?

6. What factors influence the value of Bitcoin, and how does its price volatility impact its use as a currency?

7. How do various global regulations affect Bitcoin, and what challenges do these regulations pose for its widespread adoption?

8. What are the steps involved in setting up and managing a Bitcoin node?

9. What are the major risks associated with using and investing in Bitcoin?

10. Where do you see the future of Bitcoin in the next 5-10 years, both as a technology and a financial asset?

12.6 Ethereum

1. Can you explain what Ethereum is and how it differs from Bitcoin?

2. What are smart contracts in the context of Ethereum, and how do they work?

3. Can you describe the role of the Ethereum Virtual Machine (EVM) in the Ethereum ecosystem?

4. How does Ethereum's consensus mechanism work?

5. What is the difference between Ether and Gas in the Ethereum network?

6. Can you give examples of decentralized applications (DApps) that are built on Ethereum and explain their significance?

7. What languages and tools are commonly used for developing smart contracts on Ethereum?

8. What are some key security concerns when developing on Ethereum, particularly related to smart contracts?

9. How does setting up and managing an Ethereum node differ from managing a Bitcoin node?

10. What upcoming developments or upgrades in the Ethereum ecosystem are you most excited about, and why?

12.7 Multichain

1. Can you explain what MultiChain is and how it differs from Ethereum and Hyperledger?

2. How does MultiChain's approach to permissioned blockchains compare to permissionless blockchains?

3. What are some key features of MultiChain that make it suitable for enterprise blockchain applications?

4. How does MultiChain handle the creation and management of assets within its blockchain network?

5. What consensus mechanism does MultiChain use, and how does it ensure network integrity and security?

6. Can you discuss how smart contracts are implemented in MultiChain and their potential use cases?

7. What is a 'stream' in MultiChain, and how is it used for data sharing and management?

8. What are some important considerations when setting up and deploying a blockchain using MultiChain?

9. How does MultiChain support interoperability and integration with existing systems and other blockchain networks?

10. What are some challenges or limitations of using MultiChain for blockchain solutions, and how can they be addressed?

11. What is the process of setting up a private blockchain using Multichain?

12.8 HYFI Blockchain

1. Explain the legal & regulatory compliance features of HYFI Blockchain.

2. Explain the Security Features of HYFI Blockchain.

3. Explain the Scalability Features of HYFI Blockchain.

4. What are HYFI Cold Nodes?

5. Explain fow HYFI Blockchain can be integrated with other applications.

6. Explain freezing of funds on HYFI Blockchain.

7. What are HYFI Data Streams?

8. What are HYFI Smart Filters.

9. Explain address permissions in the HYFI Blockchain.

10. Explain the process of tokenizing assets on HYFI Blockchain.

11. Explain atomic exchange transactions on HYFI Blockchain.

12.9 Hyperledger

1. Can you explain what Hyperledger is and how it differs from other blockchain platforms like Ethereum?

2. Can you discuss a few key projects under the Hyperledger umbrella and their distinct features?

3. What makes Hyperledger particularly suited for enterprise solutions compared to public blockchain platforms?

4. How does Hyperledger handle consensus mechanisms across its different frameworks, and how do they differ from traditional mechanisms like Proof of Work?

5. What is the role of smart contracts in Hyperledger, and how are they implemented and executed?

6. In Hyperledger Fabric, what are Channels, and how do they contribute to privacy and scalability?

7. How does the permissioned nature of Hyperledger networks affect their security and governance?

8. How does Hyperledger facilitate interoperability and integration with existing systems and other blockchain networks?

9. Can you explain the modular architecture of Hyperledger frameworks and its benefits for custom blockchain solutions?

10. What are some of the challenges or limitations associated with using Hyperledger for blockchain solutions, and how can they be addressed?

12.10 Blockchain & Web3 Tech Stack

1. Explain the core differences between Akash Network, Filecoin, and Sia.

2. Explain how Arweave works.

3. Explain how InterPlanetary File System (IPFS) works.

4. Explain the core differences between Audius and Theta.

5. Explain how Chainlink works.

6. Explain how Livepeer works.

7. Explain how OpenZeppelin works.

13

Quiz Questions

BEP-1. Which of these Blockchain performance indicators represents the transactions per second that a consensus algorithm can process?

A. Fault tolerance threshold
B. Latency / Finality
C. Scalability
D. Throughput

BEP-2. Which of these represents an upper bound of faulty nodes that directly impacts the performance of the consensus algorithm?

A. Fault tolerance threshold
B. Latency / Finality
C. Scalability
D. Throughput

BEP-3. Which of these Blockchain performance indicators represents the time it takes for a transaction to become settled in the ledger?

A. Fault tolerance threshold
B. Latency / Finality
C. Scalability
D. Throughput

BEP-4. Which of these Blockchain performance indicators represents the ability for a network to expand without degrading performance?

A. Fault tolerance threshold
B. Latency / Finality
C. Scalability
D. Throughput

BEP-5. Which of these enables developers to create custom blockchains?

A. Layer-0 Blockchains
B. Layer-1 Blockchains
C. Layer-2 Blockchains

BEP-6. Which of these are open-source software that enable developers to create custom blockchains?

A. Blockchain Frameworks
B. Cosmos & Horizen

BEP-7. Which of these validate & execute transactions without the need for any external network?

A. Layer-0 Blockchains
B. Layer-1 Blockchains
C. Layer-2 Blockchains

BEP-8. Which type of blockchain nodes enable faster and cheaper transactions?

A. Cold nodes
B. Light nodes
C. Lightning nodes

BEP-9. Smart contracts are the environment in which the Ethereum Virtual Machine (EVM) lives.

A. True
B. False

BEP-10. Which of these runs the blockchain's software to validate & store the complete history of transactions on the network?

A. Nodes
B. Smart contracts

BEP-11. Which of these nodes hosts the entire blockchain, validates blocks & maintains consensus?

A. Archival Full Nodes
B. Pruned Full Node

BEP-12. In which type of Blockchain fork does each node need to upgrade its software to be compatible with the new processes?

A. Hard Fork
B. Soft Fork

BEP-13. Which of these blockchain nodes saves download time & storage space by only downloading block headers?

A. Light nodes
B. Lightning nodes

BEP-14. Which of these blockchain nodes are used for signing transactions offline and storing private keys away from the network?

A. Cold nodes
B. Lightning nodes

BEP-15. Which of these are "sidechains" built on top of Layer-1 blockchains?

A. Blockchain Frameworks
B. Layer-2 Blockchains

BEP-16. Which of these consists of the hardware, software, and networks that enable the functioning of a blockchain?

A. Infrastructure Layer
B. User Interface Layer

BEP-17. Which of these consists of the interfaces, apps & services that enable users to interact with the blockchain and access its functionality?

A. Application Layer
B. User Interface Layer

BEP-18. In which type of Blockchain Bridge do you remain in control of your cryptos?

A. Trusted
B. Trustless

BEP-19. Which of these Blockchain nodes holds only the most recent transactions up to the size limit set by the operator?

A. Archival Full Nodes
B. Pruned Full Node

BEP-20. Which of these enables fungible Ethereum tokens to be re-used by other applications such as wallets and decentralized exchanges?

A. ERC-20
B. ERC-721

BEP-21. Which of these describes a method for initially locking tokens within a token contract and slowly dispensing them using a Proof of Work algorithm?

A. ERC-918
B. ERC-1178

BEP-22. Ether (ETH) is NOT an ERC20 token.

A. True
B. False

BEP-23. Which standard improves on ERC-20 and enables operators to send tokens on behalf of other address—contracts?

A. ERC-10
B. ERC-777

BEP-24. Which standard outlines a smart contract interface that can represent any number of fungible and non-fungible token types?

A. ERC-1203
B. ERC-1155

BEP-25. Which standard enables multi-class fungible tokens within smart contracts?

A. ERC-1178
B. ERC-1203

BEP-26. Ordinals enable data to be inscribed into individual satoshis on the Bitcoin Blockchain.

A. True
B. False

BEP-27. Which of these is a Proof of Stake algorithm that implements stake slashing for bad behavior like chain halts & censorship?

A. Casper the Friendly Finality Gadget
B. Leasing Proof of Stake
C. Variable Delayed Proof of Stake

BEP-28. Which is a structure where every address gets its own chain that only it can write to, and everyone holds a copy of all of the chains?

A. Block-lattice
B. Proof of Zero
C. VeriBlock

BEP-29. In which do nodes share their known transactions at random so that eventually all the transactions are gossiped to all the nodes?

A. Hashgraph
B. Magi's POS
C. Raft

BEP-30. In transaction censoring, block miners choose not to mine a block containing certain addresses. Which of these best solves it?

A. Proof of Process
B. Proof of Stake Boo
C. Proof of Work

BEP-31. Which of these considers 3 components - stake amount, financial transfer activity, and social activity?

A. Delegated Proof of Importance
B. Proof of Believability
C. Proof of Care

BEP-32. The goal of which of these blockchain attacks is to perform a double spend?

A. 51% Attack
B. Sybil Attack

BEP-33. Penalizing delayed block submissions is a solution for which type of Blockchain attack?

A. 51% Attack
B. Sybil Attack

BEP-34. In which of these types of Blockchain attacks is a P2P network manipulated by creating multiple fake identities?

A. 51% Attack
B. DDOS Attack
C. Sybil Attack

BEP-35. In which of these does the attacker slow down or halt the Blockchain network by spamming it with a large number of transactions?

A. 51% Attack
B. DDOS Attack
C. Sybil Attack

BEP-36. In which of these can a number of Sybil nodes surround your node and prevent it from connecting to other, honest nodes on the network?

A. 51% Attack
B. Eclipse Attack

14

Quiz Answers

BEP-1: D	BEP-2: A	BEP-3: B
BEP-4: C	BEP-5: A	BEP-6: A
BEP-7: B	BEP-8: C	BEP-9: B
BEP-10: A	BEP-11: A	BEP-12: A
BEP-13: A	BEP-14: A	BEP-15: B
BEP-16: A	BEP-17: B	BEP-18: B
BEP-19: B	BEP-20: A	BEP-21: A
BEP-22: A	BEP-23: B	BEP-24: B
BEP-25: A	BEP-26: A	BEP-27: A
BEP-28: A	BEP-29: A	BEP-30: B
BEP-31: A	BEP-32: A	BEP-33: A
BEP-34: C	BEP-35: B	BEP-36: B

Credits

- Cover image: https://www.freepik.com
- https://blockchainblog.substack.com
- https://ethereum.org
- https://bitcoin.org
- https://blog.coinbase.com/a-simple-guide-to-the-web3-developer-stack-8364b612d69c
- https://alchemy.com/blog/web3-stack
- https://en.bitcoin.it
- https://www.multichain.com