The Blockchain Recipe: Building a blockchain from scratch for businesses

Taylor K. Adams

Department of MIS, The University of Alabama

MIS 598

December 9th, 2021

Dr. Gregory Bott

Abstract

Blockchain is rapidly becoming a key business tool for managing data. Public blockchains such as Bitcoin's and Ethereum's continue to gain significant traction with investors. Private blockchains, while lesser-known, can fill the place of a database when certain criteria are met. But when should a blockchain be utilized in a business environment? How does one go about understanding the details of how blockchains operate? This paper attempts to provide the reader with a generalized and slightly technical understanding of blockchain and the business tools currently available to implement a blockchain in a business environment, such as Hyperledger Fabric.

Keywords: blockchain, cryptocurrency, business use cases of blockchains, blockchain implementation, Demand Chain Management (DCM), Supply Chain Management (SCM)

A brief introduction to the history of Blockchain

Blockchain technology was first conceived by Stuart Haber and W. Scott Stornetta in 1990. However, a successful use case was not conceived until 2009. The first real-world application of blockchain technology was discussed by Satoshi Nakamoto in his whitepaper titled, "*Bitcoin A Peer-to-Peer Electronic Cash System*". Nakamoto lays the foundation for a decentralized way to transfer funds between untrusted sources, Bitcoin. While Bitcoin is a transaction system; The concept of Bitcoin serves as a proof for blockchain technology, or the ability to verify the authenticity of data (transactions) on a network. Initially, blockchain was tied directly to cryptocurrency, but in 2014 blockchain 2.0 was released adding the ability for smart contracts. A smart contract can be defined as: "autonomous computer programs that, once started, execute automatically and mandatorily the conditions defined beforehand, such as the

facilitation, verification or enforcement of the negotiation or performance of a contract." (Kehrli, 2016 p. 1). Smart contracts have added a new layer of complexity to traditional blockchain technology: "If blockchains give us distributed trustworthy storage, then smart contracts give us distributed trustworthy computations." (Kehrli, 2016, p. 1). In summary, smart contracts are essentially code that is guaranteed to run if the condition for the contract to execute is met. With the addition of smart contracts, following blockchain 2.0, came the advent of the Ethereum network and Ripple. These cryptocurrencies spurred the rise of altcoins, coins closely based on a cryptocurrencies network but operate for different use cases or concepts.

The general foundations of a blockchain

Blockchain in its most basic form is a database. It stores data and keeps records. Blockchain differentiates itself from traditional databases in it is a distributed ledger. While a distributed ledger might sound complex it is just two parts: a ledger (a book or other collection of financial accounts of a particular type) and that ledger is distributed among many different computers. This is the general idea of what a blockchain is: a ledger of records copied across many different computers. Having multiple copies of a ledger is important for data availability. This is where the core of blockchain's benefits lie: validating the authenticity of a transaction. Having a ledger distributed among many computers, allows the ledger to validate transactions and ensure they are real when an entry is placed on a blockchain. For example, how do you validate someone owns a particular bitcoin on the bitcoin blockchain? Every specific coin has a unique identifier stored on the blockchain and every person who owns a Bitcoin has a Bitcoin wallet (Each wallet has a unique address). Every time a transaction occurs on the Bitcoin blockchain the transaction is stored, and a record of the transfer is kept on the Bitcoin

blockchain. This allows you to verify ownership of a Bitcoin throughout its lifetime, from its creation to its most recent holder.

The Role of Cryptography in blockchain

To understand blockchain one also must understand the role of Cryptography within it. The word Cryptography is broken down to crypto (hidden) and graphy (graph) or hidden graph. It is the use of mathematical operations to protect messages. One of the primary cryptographic hashes used in cryptocurrency is the SHA-256 hashing algorithm. The goal of this algorithm (in terms of blockchain) is to ensure blocks on the blockchain can't be modified or artificially replicated. Every block has a unique hash value used as its identifier. This hash is derived from the data located inside the block. Due to the complexity of cryptography and the design goals of the SHA-256 algorithm it would be very difficult, if not near nearly impossible, to override the data within the block and still produce the same hash value.

An attacker could try a different approach by changing the block data but would need to override every single block after the target block to make the 'hacked' entry valid. The attacker would essentially have to break the blockchain's 'chain'. This feat is very difficult to accomplish because of the property known as a cryptographic link. Essentially this cryptographic link means as more blocks are added to the blockchain it becomes increasingly more difficult to modify an existing block on the blockchain. Cryptography, therefore, adds a layer of security to blockchain protocols to ensure immutable data.

Distributed Peer-To-Peer Network

While not a requirement, a distributed peer-to-peer network is another key feature of blockchain technology. A peer-to-peer network is a group of computers housing an exact copy of the same blockchain. When a block is added to the blockchain it is updated across that

blockchain's network. Every computer hosting the blockchain is updated with the new values (housed within a block) which are appended to the end of the blockchain. The primary benefit of a distributed peer-to-peer network is security and protecting the authenticity of the blockchain. The previous example discussed that if a hacker was able to modify a block and every subsequent block on a blockchain they could alter the network. A distributed peer-to-peer network takes on this problem. In this case, if the attacker can successfully modify a blockchain node, they will not be successful in altering the blockchain since it exists on a distributed network. In the case of a distributed network, the other nodes will notice a discrepancy on the blockchain copy located on the hacked node. The nodes will come to a consensus on what the correct blockchain is and the node which was attacked will automatically copy the correct blockchain over its existing compromised copy. To alter a blockchain as a whole, an attacker will need to control at least 51 percent of the nodes on a blockchain.

The Consensus Problem and Consensus Algorithms

A consensus algorithm within a blockchain is a type of algorithm which is chosen by the creators of a blockchain to validate blocks requesting to be added to a blockchain. A blockchain will typically have many validators and these validators need a way to ensure the blocks on a blockchain are valid.

> *"The consensus is a problem in distributed computing wherein nodes within the system much reach an agreement given the presence of faulty processes or deceptive nodes."* (Bach et al., 2018, p. 1545)

Aside:

Consensus algorithms cannot be spoken about without mentioning the Byzantine General's problem and the Byzantine fault tolerance.

The Byzantine General's problem is one where there are four generals each of whom can attack or retreat. The generals must all come to a consensus on whether to attack or retreat. This is complicated in two ways; the generals can only communicate by messengers who may or may not be reliable and some of the generals could be traitors. The traitorous general would be an example of a rogue node on a blockchain.

The Byzantine fault tolerance is: *"*a category of replication algorithms that aim to solve the problem of reaching consensus when nodes can generate arbitrary data.*"* (Bach et al., 2018, p. 1545). This "arbitrary data" could refer to the rogue nodes trying to take over a blockchain by providing invalid data for the next block. The Byzantine Fault Tolerance guarantees a level of safety in a consensus algorithm where up to 33 percent of the nodes on a blockchain can be corrupt and the blockchain should continue unaffected by the "bad" actor nodes.

There are many consensus algorithms being utilized now on different blockchains. The following table shows the top ten cryptocurrencies and the consensus algorithm being used. While this table shows only cryptocurrencies, consensus algorithms must be utilized by all blockchains regardless of purpose.

Currency Name	Consensus Algorithm	Market Cap
Bitcoin	Proof of Work	$ 157.3 B
Ethereum	Proof of Work[1]	$ 95.7 B
Ripple	Ripple Protocol Consensus Algorithm	$ 37.1 B
Bitcoin Cash	Proof of Work	$ 21.4 B
Cardano	Proof of Stake	$ 11.8 B
Stellar	Stellar Consensus Protocol	$ 8.3 B
NEO[2]	Delegated Byzantine Fault Tolerance	$ 8.2 B
Litecoin	Proof of Work	$ 8.1 B
EOS	Delegated Proof of Stake	$ 6.5 B
NEM	Proof of Importance	$ 5.7 B

1. Planned switch to Proof of Stake sometime in 2018
2. Formerly known as Antshares

Figure 1.3 – Consensus Algorithms by cryptocurrency market cap

(Bach et al., 2018, p. 1545)

Mining (Proof of work)

The goal of mining on a blockchain is to provide a "proof of work". This proof of work is used to create fair competition among miners on a blockchain. Every new block to be added on a blockchain will be given a target. Each miner must vary the nonce of the block and hash it until they hit a value less than or equal to the target value. When the miner gets a hash value below the target it is allowed to add a block to the blockchain, and the miner who got the hash value below the target first receives the block reward. This block reward incentivizes individuals to utilize their computer's processing power for the blockchain, establishing a distributed network. See figure 1.1 for a visual representation of a block on a blockchain.

Mining does however have drawbacks. Bitcoin was the first cryptocurrency to implement mining. Initially, mining was viewed by Satoshi Nakamoto (2008) as "one-CPU-one-vote" referring to utilizing CPU's over IP addresses since IP's can be spoofed. The keyword in his thought process was CPU. The power consumption of CPUs around 2008, when Bitcoin was conceived, pales in comparison to the power-hungry ASIC[1] mining farms of today. In 2018, "the annual electricity consumption of Bitcoin was 63.99 TWh." (Li et al., 2019 p.161). This power

usage has led to questioning the long-term environmental impact proof-of-work blockchains have on our planet. While several alternatives to proof of work have been envisioned, the most prominent is proof of stake.

[1]ASIC Miner – ASIC (Application-Specific Integrated Circuit) – a microprocessor designed for a specific computation task like mining nodes on a specific blockchain.

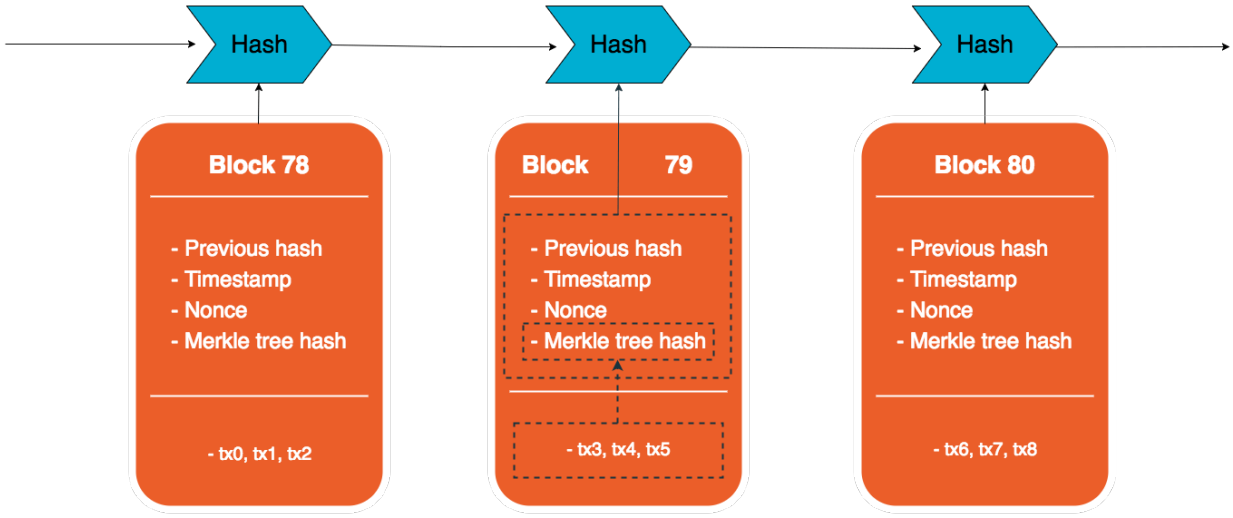


Figure 1.1: Blocks on a blockchain (*Blockchain, the next Big Thing?*, 2017)

Smart Contracts

Smart Contracts were first introduced by Nick Szabo in 1998. He described smart contracts as a kind of vending machine: "…machines take in coins, and via a simple mechanism (e.g., finite automata), dispense change and product according to the displayed price. Smart contracts go beyond the vending machine by proposing to embed contracts in all sorts of properties by digital means." (Wang et al., 2019, p. 2266). In this way, smart contracts are similar to vending machines in there is a condition that must be met in order for the contract to execute. Once met the contract will execute without any intervention from an intermediary. In its most basic form, a smart contract is extremely similar to a regular contractual agreement between two parties. However, this contract does not require any intermediary to ensure the

terms of the contract are met. A smart contract is code that will execute if the specified events of execution occur. This code is stored on a blockchain to ensure the contract is unmodifiable.

How does the smart contract know when an event has occurred? In Ethereum the validation of smart contracts is completed through oracles. Oracles bridge the real world with blockchain networks. An oracle is any device that connects to deterministic blockchains to provide the blockchain with real-world data. Enter the oracle problem. The oracle problem is: when all oracles are based on a centralized network, the main benefits of blockchain are stripped away from smart contracts because there becomes a single point of failure. Attacking the oracle can allow for the behavior of smart contracts to be artificially altered. To get around this issue a smart contract cannot use a single centralized oracle and remain secure.

To learn more about smart contracts and oracles visit the Ethereum documentation on Ethereum.org.

Permissioned vs Permissionless Blockchains

Permissioned blockchains are only authorized to be read by and written to by an authorized set of readers and writers. A technology called Hyperledger Fabric is one of the key players for permissioned/private blockchains. (Note: Permissioned blockchains can still be public if desired.) Hyperledger fabric's website describes it as: "a foundation for developing applications or solutions with a modular architecture." (Hyperledger Foundation, n.d.). Permissioned blockchains can also be centrally managed.

Permissionless blockchains are typically used in most if not all cryptocurrencies. Permissionless blockchains are defined by several things mentioned above such as utilizing peer-to-peer networks, decentralization, and a high number of untrusted writers.

	Permissionless Blockchain	Permissioned Blockchain	Central Database
Throughput	Low	High	Very High
Latency	Slow	Medium	Fast
Number of readers	High	High	High
Number of writers	High	Low	High
Number of untrusted writers	High	Low	0
Consensus mechanism	Mainly PoW, some PoS	BFT protocols (e.g. PBFT [5])	None
Centrally managed	No	Yes	Yes

Figure 1.2: Permissionless Blockchain vs Permissioned Blockchain vs Central Database (Wüst & Gervais, 2018, p. 48)

Blockchains vs Traditional Databases – When should each be deployed

There are many use cases for blockchain technology in the world today, and surely many use cases that have yet to be conceived. Obviously, cryptocurrency is a major use case for blockchains. This extends to the NFT market (Non-Fungible Tokens), smart contracts, etc. However, blockchains cannot replace traditional databases. There are still many use cases for traditional databases where the implementation of a blockchain could decrease performance.

When to use a blockchain

Provided are some examples of how blockchains can be used for business:

- Secure data sharing
- Monitoring of supply chain and logistics
- Distributed IOT management system (Oscar Nova)
- When collateral is being utilized
- Secure Loans via smart contracts

In their whitepaper, Wüst and Gervais provide a clear outline on when/if a blockchain should be used:

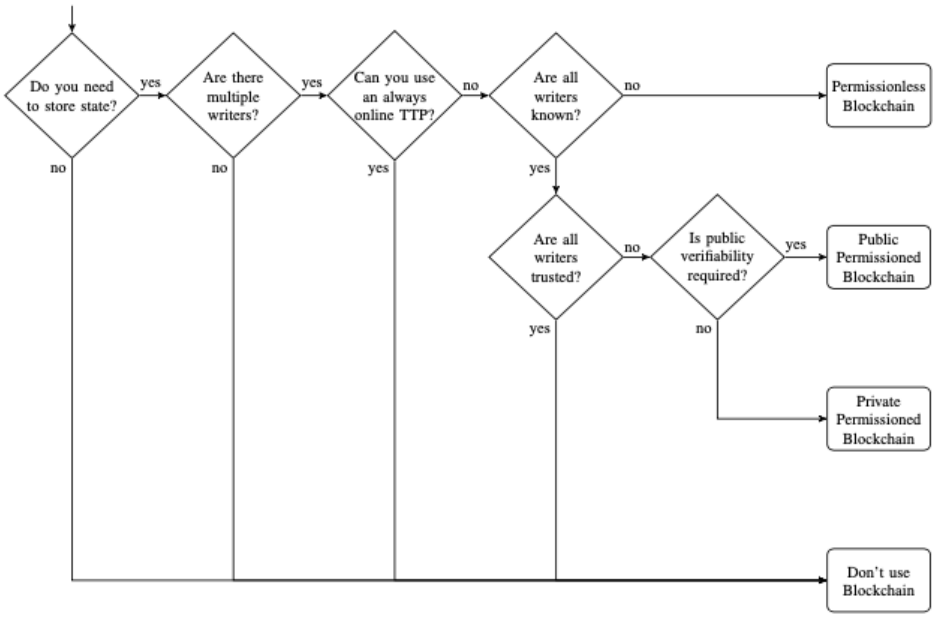


Figure 1.3: When should you use a blockchain

(Wüst & Gervais, 2018, p. 47)

Aside:

Many different altcoins employ new adaptations to block and could be utilized in a business as they are in their permissionless blockchain form. However, businesses could utilize the ideas behind these blockchains to create permissioned or private blockchains solely for internal use.

When to avoid using blockchain

While blockchain technology has incredible advantages for applicable use cases, it may not always be the best option. The primary setbacks of blockchain are scalability, capacity, and latency.

Scalability – surprisingly blockchain, in its current form, does exhibit scalability issues in terms of the data's rate of change. Utilizing a blockchain for data that is frequently changing can lead to network slowdowns as the nodes attempt to verify the authenticity of all the changes made. This slowdown would increase exponentially as more and more blocks are attempted to be added to the blockchain.

Capacity – The capacity of a blockchain is "defined as the average number of transactions successfully recorded per second in a final accepted blockchain, is restricted by the consistency requirement of blockchains, such as eventual consistency (across all chains maintained at different miners) ." (Wang et al., 2019, p. 2). The capacity of a blockchain closely ties in with its scalability. This boils down to the processing power allotted to the blockchain.

Latency – Block creation on a blockchain of any meaningful size takes time. For data sources needing live updates, a blockchain is likely not the way to go.

Figure 1.4, adds to Figure 1.3 by providing a more detailed step-by-step approach on when and when not to use a blockchain.

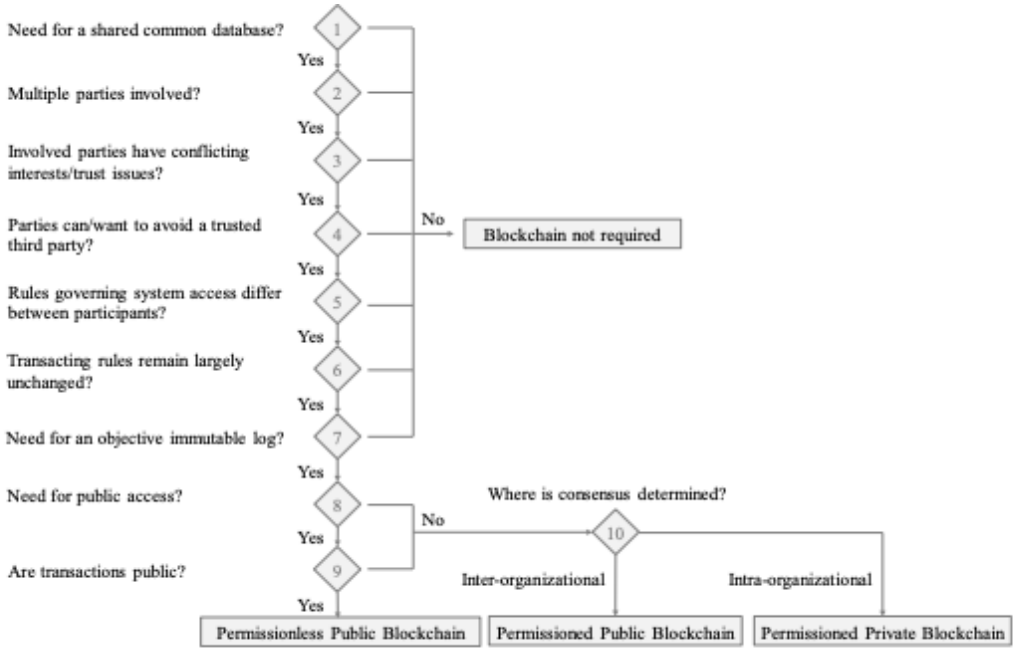


Figure 1.4: Overview of a blockchain decision path

(Pedersen et al., 2019, p.102)

For a deeper dive into a step-by-step approach to decide if a blockchain is a good solution for a data set see: "*A Ten-Step Decision Path to Determine When to Use Blockchain Technologies*" by Pedersen et al.

Example problems blockchain can solve

Below are several scenarios in which a blockchain can be utilized effectively:

Supply Chain Management

Global commerce is a complex, fragile, and high-cost endeavor. It requires planning and investment into supply chain management. Notably, supply chains were greatly affected in 2020 and beyond by the global COVID-19 pandemic. Wüst and Gervais state, "Some (Companies) even claim that blockchain technology paves the way to demand instead of supply chains, where businesses will benefit from a greater flexibility in interacting with different markets and balancing the price risk." (Wüst & Gervais, 2018, p. 47-48). In their paper, they introduce the

topic of Demand Chain Management (DCM) as opposed to supply chain management. In DCM, "customer's interest is at the core of the chain" (Wüst & Gervais, 2018, p.48) this, in turn, reduces the cost associated with supply chains creating a more efficient market structure. DCM, however, requires companies to have a complete view of the market they are in, from consumer demand to production. Deloitte, list the primary benefits of blockchain in supply chain management as:

- "Increase traceability of material supply chain to ensure corporate standards are met"
- "Lower losses from counterfeit/gray market trading"
- "Improve visibility and compliance over outsourced contract manufacturing"
- "Reduce paperwork and administrative costs"

  (*Using Blockchain to Drive Supply Chain Transparency and Innovation*, n.d.)

There are currently several blockchains being utilized to create a better solution for supply and demand chains utilizing blockchain including Skuchain.

IOT Access Management:

Internet of Things (IoT) has been growing rapidly over the past decade. From smart doorbells such as Ring to smart refrigerators to personal health monitors such as Fitbit, the world is becoming more connected by the day. Oscar Novo in his whitepaper "*Scalable Access Management in IoT Using Blockchain: A performance Evaluation*" states, "…given the ever increasing array of IoT devices, there is a requirement for global, yet highly secure and reliable IoT access management systems."(Novo, 2019, p.1) Essentially the growing number of IoT devices is creating a problem on how to manage them and their data. One could use a traditional centralized server approach, but this has a critical drawback which is a single point of failure that

prevents scaling. On top of this, centralized solutions can become inaccurate with IoT devices which frequently move locations. Oscar Novo proposes a solution to this problem by utilizing a blockchain to store and distribute policies and manage access control permission for IoT devices on a blockchain. The benefits blockchain can provide over traditional IoT management approaches are:

- "First, the architecture can be used in isolated managerial domains. Thereby, every managerial domain has the freedom to manage the IoT devices while the access control policies are still enforced by the rules in the blockchain."(Novo, 2019, p.4694)
- "Second, our proposal can manage simultaneous managers, and they can access or modify the access control policies of an IoT device concurrently."(Novo, 2019, p.4694)
- "Third, constrained managers can easily administer multiple IoT devices due to the fact that the IoT devices do not access the management information directly from the managers. At the same time, constrained managers do not need to be constantly connected to the system, which saves its energy."(Novo, 2019, p.4694)
- "Fourth, the managers and IoT devices are interacting through the blockchain network enabling cross-platform communication."(Novo, 2019, p.4694)

Conclusion

Where few solutions existed previously, blockchain technology can be used to implement business solutions to complex problems. When blockchain is utilized appropriately, it can bring tremendous value to an organization and its users. Blockchain in the proper context can bring about improvements in data security principles such as data availability and data integrity, over

the use of traditional databases. Blockchain technology is poised to grow as a powerful data management tool in organizations, much like the effect the Internet has had on global commerce. However, blockchain is not without drawbacks. It is not the "end-all-be-all" solution. Organizations should invest time and resources in understanding where blockchain can provide a viable solution. If a valid use case is found, early adopters can gain a completive advantage from implementing a blockchain solution.

Appendix:

Components of a Blockchain

Block – generally a Block on a blockchain essentially contains 3 main components: *Data*, *Previous Hash*, and the *Hash*.

- The Data contains the information which is stored within the block.
- The Previous Hash is the hash of the block in the blockchain directly before the current block.
- The Hash is essentially a cryptographic representation of the block that is unique. This Hash is commonly referred to as the 'fingerprint' of the block.
- Additional Block components:
    - Nonce – stands for "number only used once"

Hash – is essentially a number which can represent a file.

- There are certain algorithms called hashing algorithms which take an array of data(i.e. a file) and converts it into a series of numbers based on the length specified by the hash. Hashing algorithms can be used to verify file authenticity, so long as an appropriate algorithm is chosen which avoids collisions.

Blockchain – is a system of recording information which provides greater data integrity than traditional database/data storage methods.

Distributed Peer-To-Peer Network – the partitioning or distribution of a network to peers of equal say in the network.

Building a blockchain

The remainder of this paper will provide a step-by-step guide on the basics of building a blockchain. To create a blockchain in this way, some coding experience is necessary to understand what is happening. This code represents a very simple blockchain. It provides the general foundations of how to build and implement a blockchain.

Required Resources:

To build our basic blockchain we will utilize:

- Anaconda (Python)
- Flask
- Postman

Code:

```
# To be installed:
# Flask==0.12.2: pip install Flask==0.12.2
# Postman HTTP Client: https://www.getpostman.com/

# Importing the libraries
import datetime
import hashlib
import json
from flask import Flask, jsonify

# Part 1 - Building a Blockchain

class Blockchain:

    def __init__(self):
        self.chain = []
        self.create_block(proof = 1, previous_hash = '0')

    def create_block(self, proof, previous_hash):
        block = {'index': len(self.chain) + 1,
                 'timestamp': str(datetime.datetime.now()),
                 'proof': proof,
                 'previous_hash': previous_hash}
        self.chain.append(block)
        return block

    def get_previous_block(self):
        return self.chain[-1]

    def proof_of_work(self, previous_proof):
        new_proof = 1
        check_proof = False
        while check_proof is False:
            hash_operation = hashlib.sha256(str(new_proof**2 -
previous_proof**2).encode()).hexdigest()
            if hash_operation[:4] == '0000':
                check_proof = True
```

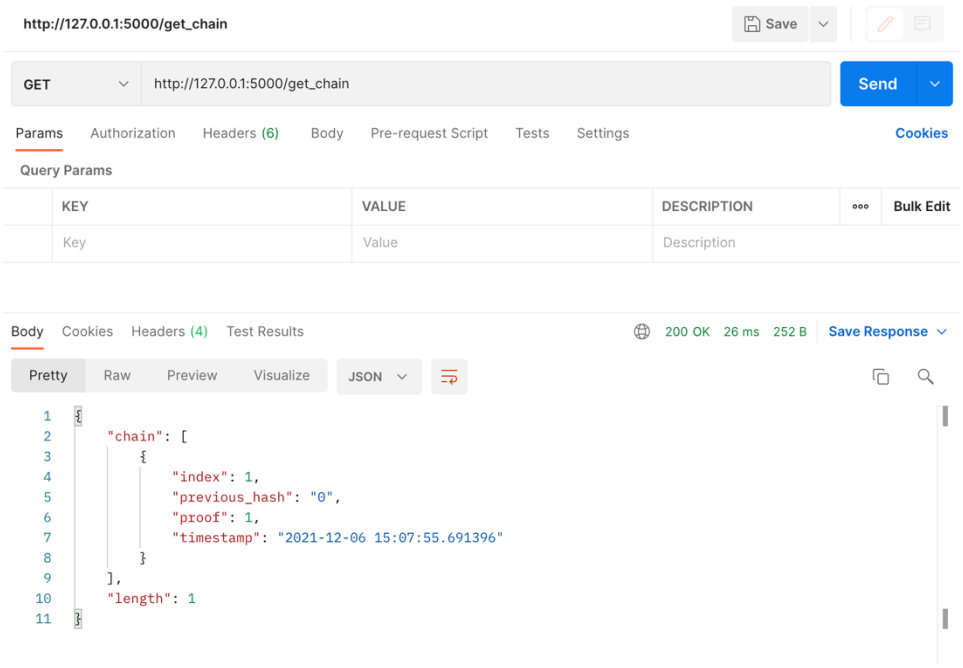
```
            else:
                new_proof += 1
        return new_proof

    def hash(self, block):
        encoded_block = json.dumps(block, sort_keys =
True).encode()
        return hashlib.sha256(encoded_block).hexdigest()

    def is_chain_valid(self, chain):
        previous_block = chain[0]
        block_index = 1
        while block_index < len(chain):
            block = chain[block_index]
            if block['previous_hash'] !=
self.hash(previous_block):
                return False
            previous_proof = previous_block['proof']
            proof = block['proof']
            hash_operation = hashlib.sha256(str(proof**2 -
previous_proof**2).encode()).hexdigest()
            if hash_operation[:4] != '0000':
                return False
            previous_block = block
            block_index += 1
        return True

# Part 2 - Mining our Blockchain

# Creating a Web App
app = Flask(__name__)
app.config['JSONIFY_PRETTYPRINT_REGULAR'] = False
# Creating a Blockchain
blockchain = Blockchain()

# Mining a new block
@app.route('/mine_block', methods = ['GET'])
def mine_block():
    previous_block = blockchain.get_previous_block()
    previous_proof = previous_block['proof']
    proof = blockchain.proof_of_work(previous_proof)
    previous_hash = blockchain.hash(previous_block)
    block = blockchain.create_block(proof, previous_hash)
    response = {'message': 'Congratulations, you just mined a
block!',
                'index': block['index'],
                'timestamp': block['timestamp'],
```

```
                'proof': block['proof'],
                'previous_hash': block['previous_hash']}
    return jsonify(response), 200

# Getting the full Blockchain
@app.route('/get_chain', methods = ['GET'])
def get_chain():
    response = {'chain': blockchain.chain,
                'length': len(blockchain.chain)}
    return jsonify(response), 200

# Checking if the Blockchain is valid
@app.route('/is_valid', methods = ['GET'])
def is_valid():
    is_valid = blockchain.is_chain_valid(blockchain.chain)
    if is_valid:
        response = {'message': 'The Blockchain is valid.'}
    else:
        response = {'message': 'Houston, we have a problem. The
Blockchain is not valid.'}
    return jsonify(response), 200

# Running the app
app.run(host = '0.0.0.0', port = 5000)
```

(Figure 1.5 – Blockchain Code ((Ponteves et al., n.d.)))

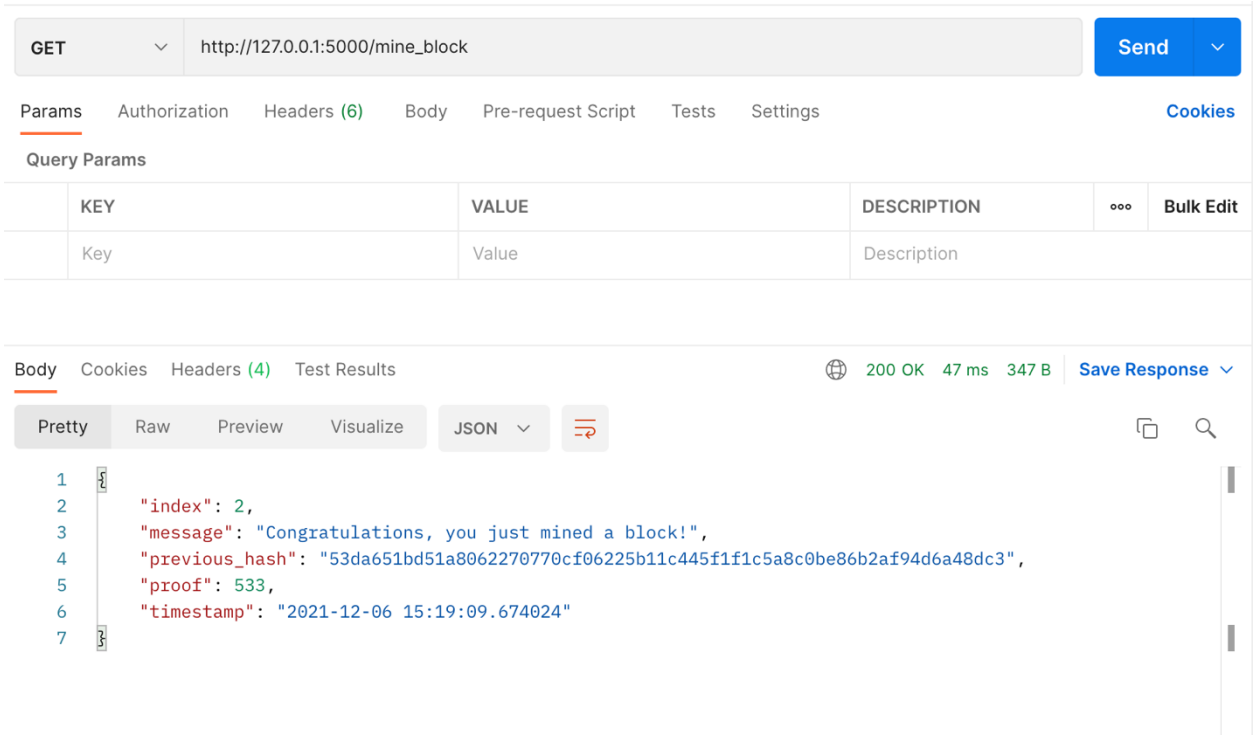
General Summary of how to run the above blockchain

1. Running the blockchain – Begin by copying the above code into a new .py file in the Sypder IDE. Run the code and the console will display "* Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)"
2. Viewing the active blockchain – Open the postman application. In the "send bar" type "http://127.0.0.1:5000/get_chain". This command will show just the genesis block of the blockchain. See Figure A1.



(Figure A1 – Viewing a block on a basic blockchain)

3. Mining a block on the blockchain – To mine the second block on the blockchain, in postman type “http://127.0.0.1:5000/mine_block”.



(Figure A2 – Mining a block on a basic blockchain)

4. Mine several blocks – using the previous step above mine several blocks on the blockchain, following this run the get_chain request in postman again. The results should be similar to figure A3.


```
{"chain":[{"index":1,"previous_hash":"0","proof":1,"timestamp":"2021-12-06
15:07:55.691396"},{"index":2,"previous_hash":"53da651bd51a8062270770cf06225b11c445f1f1c5a8c0be86b2af9
4d6a48dc3","proof":533,"timestamp":"2021-12-06
15:19:09.674024"},{"index":3,"previous_hash":"49b79c2405374be0322f99cd32b33821e38f93ebd0411d9f82b1660
7e956302b","proof":45293,"timestamp":"2021-12-06
15:25:21.181060"},{"index":4,"previous_hash":"2f7a64d7c2fa16ee3dc60bcb4d7b2387cc9c48c68746aee89de312c
fe3f97632","proof":21391,"timestamp":"2021-12-06
15:25:22.387585"},{"index":5,"previous_hash":"a5e7dfd48f56dd6f5806c09762e8340f10975ea4f1c51e172bb4989
d1ee6e2bc","proof":8018,"timestamp":"2021-12-06
15:25:23.220402"},{"index":6,"previous_hash":"d71dba1fdc7ec2de7aa2f64afa0c34ad7324800f1646be8ea652831
2921eb0d8","proof":48191,"timestamp":"2021-12-06
15:25:24.279841"},{"index":7,"previous_hash":"427f1284cac4b205678be6fd43b524f1ae09204b2347f520e6904a7
a593a2aea","proof":19865,"timestamp":"2021-12-06
15:25:24.803341"},{"index":8,"previous_hash":"14fbec5cc01d1daddca258b308ecc0c4a06865ef35976e5fc85c2ca
c354c33bc","proof":95063,"timestamp":"2021-12-06
15:25:25.936087"},{"index":9,"previous_hash":"49a9ceb6727dfd4e8f84552f511f13ae75179803c3bf119d4e58420
8d7855905","proof":15457,"timestamp":"2021-12-06
15:25:26.719538"},{"index":10,"previous_hash":"67ef9702a54d5825fcf34b6784e2262fb600e848dfd16a6c964ac6
5c03b7b23f","proof":15479,"timestamp":"2021-12-06 15:25:27.408784"}],"length":10}
```


(Figure A3 – a basic blockchain with 10 blocks created)

This example displays an oversimplified version of a blockchain with just the core functionality of mining a block and viewing the blockchain. This display is a visual representation of simple blockchain concepts which can be added to for real-world functionality.

References

Blockchain, the next big thing? (2017, July 18). Codecentric AG Blog. https://blog.codecentric.de/en/2017/07/what-is-blockchain/

Kehrli, J. (2016, Nov 22). Blockchain 2.0—From Bitcoin Transactions to Smart Contract applications. 37.

Li, J., Li, N., Peng, J., Cui, H., & Wu, Z. (2019). Energy consumption of cryptocurrency mining: A study of electricity consumption in mining cryptocurrencies. *Energy*, *168*, 160–168. https://doi.org/10.1016/j.energy.2018.11.046

Hyperledger Fabric. (n.d.). *Hyperledger Foundation*. Retrieved November 9, 2021, from https://www.hyperledger.org/use/fabric

Wüst, K., & Gervais, A. (2018). Do you Need a Blockchain? *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, 45–54. https://doi.org/10.1109/CVCBT.2018.00011

Using Blockchain to Drive Supply Chain Transparency and Innovation. (n.d.). Deloitte United States. Retrieved November 9, 2021, from https://www2.deloitte.com/us/en/pages/operations/articles/blockchain-supply-chain-innovation.html

Novo, O. (2019). Scalable Access Management in IoT Using Blockchain: A Performance Evaluation. *IEEE Internet of Things Journal*, *6*(3), 4694–4701. https://doi.org/10.1109/JIOT.2018.2879679

Bach, L. M., Mihaljevic, B., & Zagar, M. (2018). Comparative analysis of blockchain consensus algorithms. *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 1545–1550. https://doi.org/10.23919/MIPRO.2018.8400278

Wang, S., Ouyang, L., Yuan, Y., Ni, X., Han, X., & Wang, F.-Y. (2019). Blockchain-Enabled Smart Contracts: Architecture, Applications, and Future Trends. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *49*(11), 2266–2277. https://doi.org/10.1109/TSMC.2019.2895123

Wang, X., Yu, G., Zha, X., Ni, W., Liu, R. P., Guo, Y. J., Zheng, K., & Niu, X. (2019). Capacity of blockchain based Internet-of-Things: Testbed and analysis. *Internet of Things*, *8*, 100109. https://doi.org/10.1016/j.iot.2019.100109

Pedersen, A., Risius, M., & Beck, R. (2019). A Ten-Step Decision Path to Determine When to Use Blockchain Technologies. *MIS Quarterly Executive*, *18*, 99–115. https://doi.org/10.17705/2msqe.00010

Ponteves, H., Kirill, E., & Ligency Team. (n.d.). *Blockchain A-Z™: Learn How To Build Your First Blockchain*. Udemy. Retrieved November 30, 2021, from https://www.udemy.com/course/build-your-blockchain-az/