

Consensus in Bitcoin

Gallersdörfer, U., Holl, P., & Matthes, F. (2020). "Blockchain-based Systems Engineering". Lecture Slides. TU Munich.

Chair of Software Engineering for Business Information Systems (sebis)
Faculty of Informatics
Technische Universität München
www.matthes.in.tum.de

1. Consensus

- Cryptocurrency with a central authority
- Byzantine generals problem
- Block propagation
- Double spending problem

2. Proof-of-Work (Mining)

- Search puzzle
- Difficulty determination
- Incentives
- Amount of Bitcoin
- Mining hardware
- Mining pools

Bitcoin with a central authority (CA)

- In the last two lectures, we talked about the **Blockchain** of Bitcoin, of the **data** contained in it, about **transaction** vs **account-based ledgers**, about the **content of transactions** and **Bitcoin script** and many more.
 - If we wanted to design a digital currency with a central authority, we would be done:
 - Central authority **signs** every new block, publishes it to the network.
 - Other nodes **validate** the content and append the new block to their own chain.
- ➔ What are the **disadvantages** with this approach? What could the CA do?
- The CA has to be **nominated** or somehow defined.
 - The CA could **unilaterally ignore** or delay **transactions** of certain parties or with certain properties.
 - The CA could render the network **unavailable** by being overloaded or intentionally shut down.

- Because of these disadvantages for the network, we do **not** want have a central authority.

Definition *Distributed Consensus*

A network consists out of N nodes. All of these nodes have an input value and propose it to all other nodes. Some of the nodes are faulty (not responding) or malicious, trying to propose a wrong input.

Two properties must hold:

- The process has to terminate with all **honest nodes** in agreement on **one input value**.
 - The value must have been **generated** by an **honest node**.
-
- We want the network to agree on the following input:
 - Which of the proposed **transactions** are **valid**?
 - In which **order** do the **transactions** appear in the ledger?

Difficulties with distributed consensus

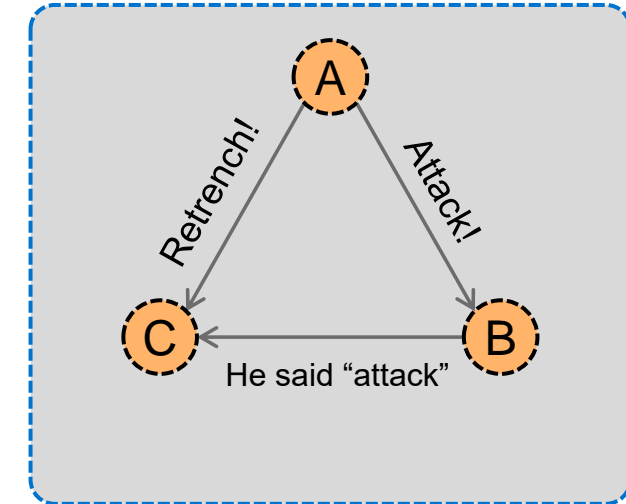
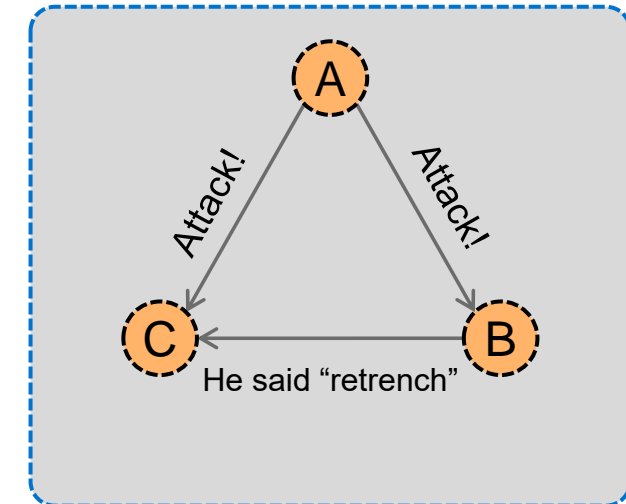
The Byzantine generals problem

The Byzantine army wants to invade an enemy city, however, it is separated into multiple divisions. They want to attack at the same time, therefore they have to communicate in between the divisions to find a common time to attack.

A general is responsible for one division. These generals communicate by messenger. Some of the generals may be traitors, sending wrong messages to other generals. The **goal** is for all loyal generals to **derive** the **same plan** without the traitors being able to convince other generals of the wrong plan.

It can be shown that if more or equal to one third of the generals are malicious, it is impossible for the honest nodes to derive a common plan.

Three generals



C does not know to what agree on.

[LAM*1982] Lamport, L., Shostak, R., & Pease, M. (1982). [The Byzantine generals problem](#). *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 4(3), 382-401.

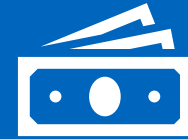
- Problem with the number of nodes:
 - Many nodes join the network, also many leave after a short time. How do we know how many there are?
 - Approaches like “more than 50% positive votes on a block” would not work, as we do not know how many nodes are in the network.
 - How do we prevent an attacker to create an arbitrary number of nodes to increase her chance to be selected? This is called a sybil-attack.
- Problem with time:
 - In decentralized networks, there is **no general notion of time**, as a time server (e.g. NTP) would constitute a central node in the network.
 - This makes the development of distributed consensus algorithms difficult.



Ongoing consensus



Notion of randomness



Incentives

Probabilistic consensus

The consensus mechanism is an ongoing process in Bitcoin, therefore the order of blocks or transactions is never 100% safe.

Randomness in consensus

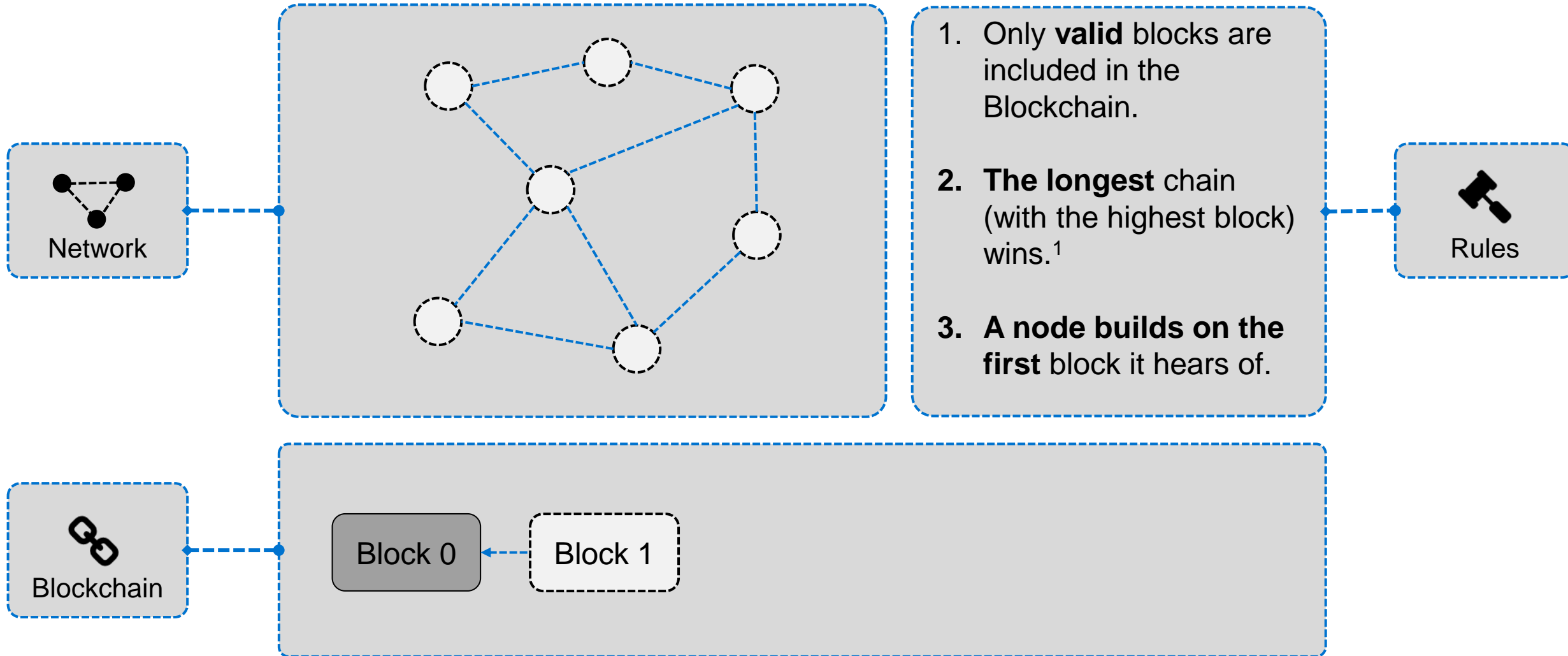
The network selects a random node to propose a new block. As we will see later, this ensures that if over 50% are honest, a probabilistic consensus can be reached.

Incentivized nodes

The network incentivizes nodes to participate in the consensus algorithm. They receive bitcoins for created blocks which are included in the longest chain.

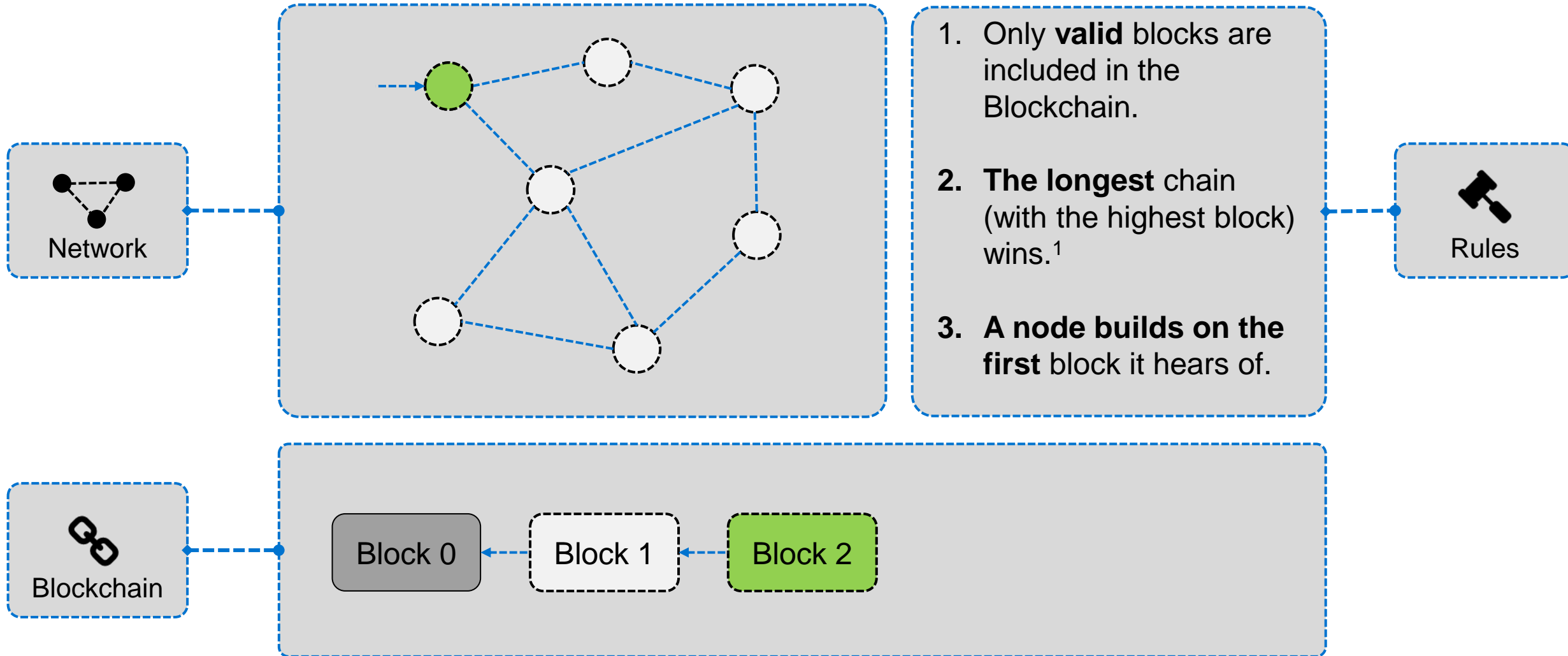
- 1 *Transaction Broadcast:* Every node who receives transactions or creates them, broadcasts them to the network, making everyone aware of new transactions.
- 2 *Block Building:* Every node collects the valid transactions, orders them and creates a new block containing the transactions.
- 3 *Random Node Selection:* A node is randomly chosen out of the network. It is able to propose its block to the network.
- 4 *Block Validation:* Other nodes receive the block from the randomly chosen node and validate whether it is correct. A correct block only contains valid transactions.
- 5 *Block Acceptance:* Other nodes show their acceptance for this block if the nodes build new blocks on top of the recently proposed block.

How do blocks propagate through the network?



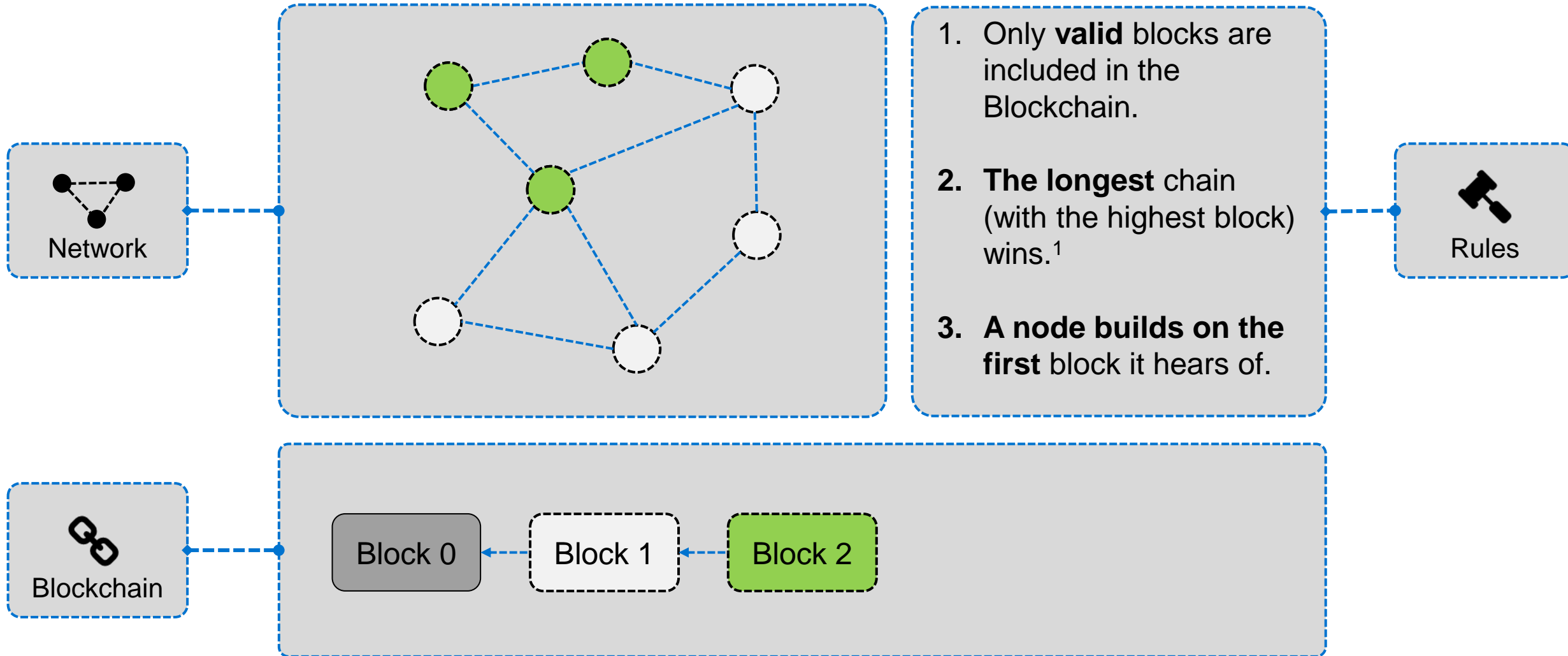
¹ The term "longest chain" is actually the chain with the highest weight. We will explain this term later on.

How do blocks propagate through the network?



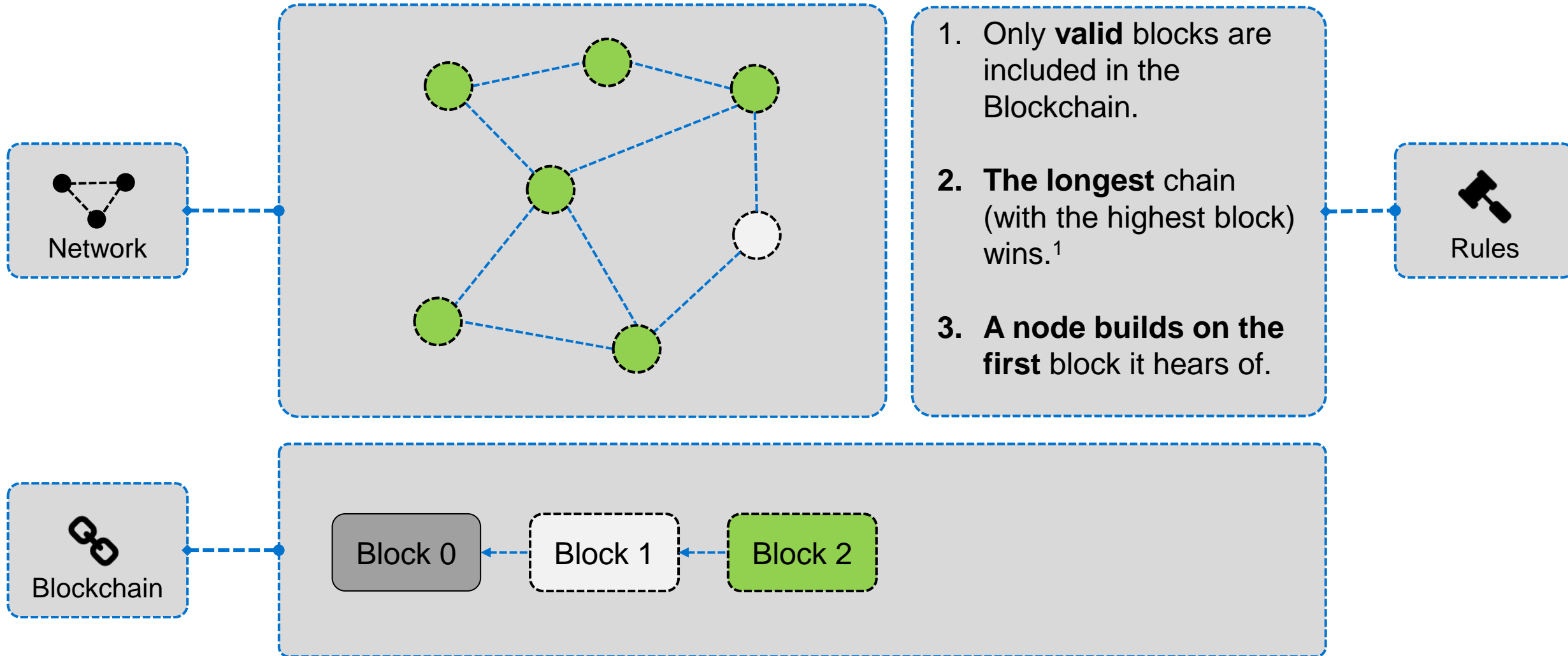
¹ The term “longest chain” is actually the chain with the highest weight. We will explain this term later on.

How do blocks propagate through the network?



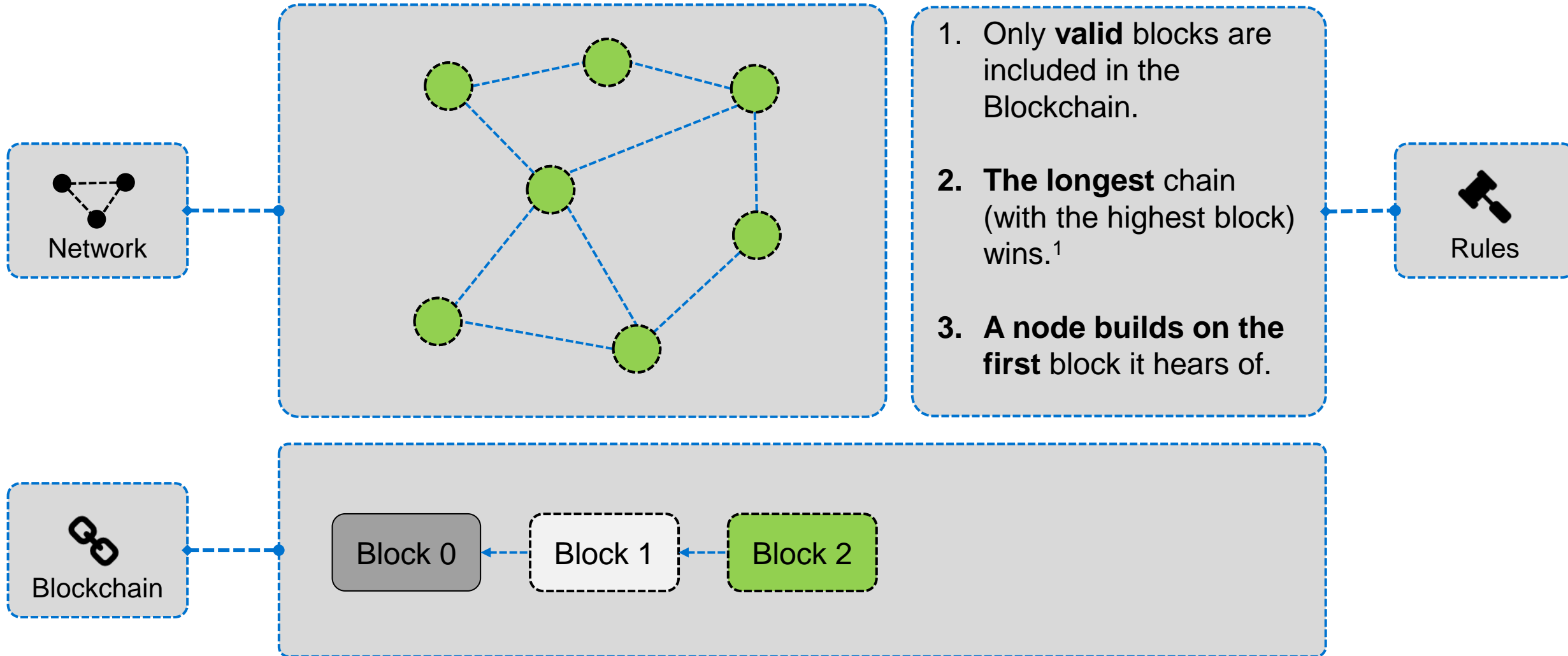
¹ The term “longest chain” is actually the chain with the highest weight. We will explain this term later on.

How do blocks propagate through the network?



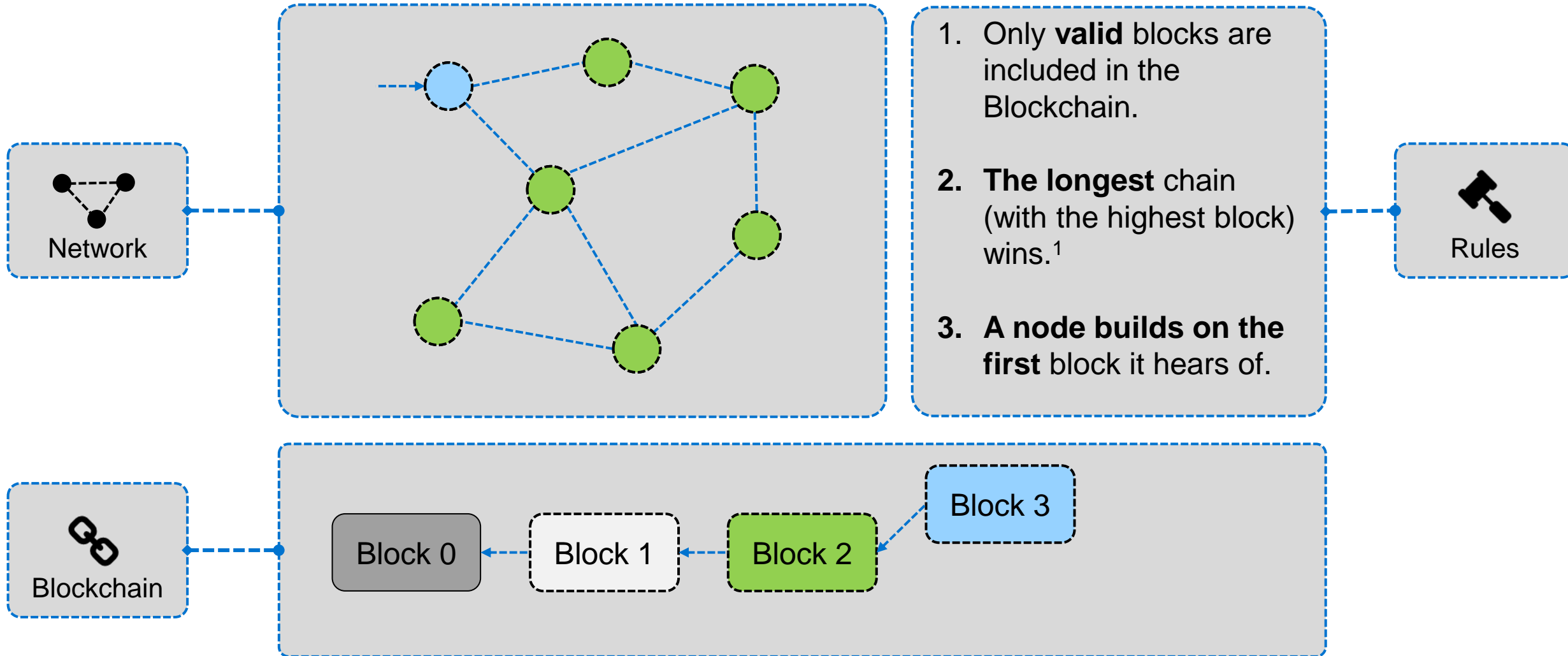
¹ The term “longest chain” is actually the chain with the highest weight. We will explain this term later on.

How do blocks propagate through the network?



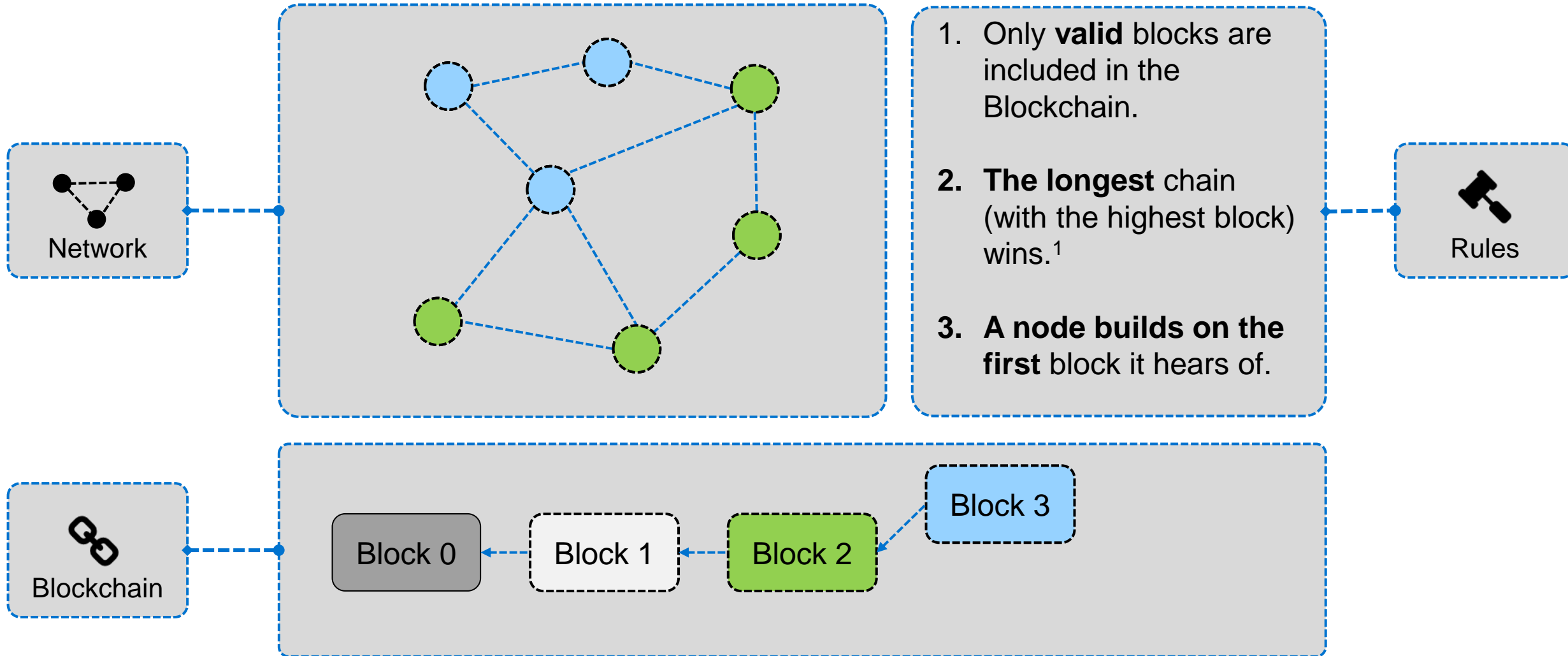
¹ The term "longest chain" is actually the chain with the highest weight. We will explain this term later on.

How do blocks propagate through the network?



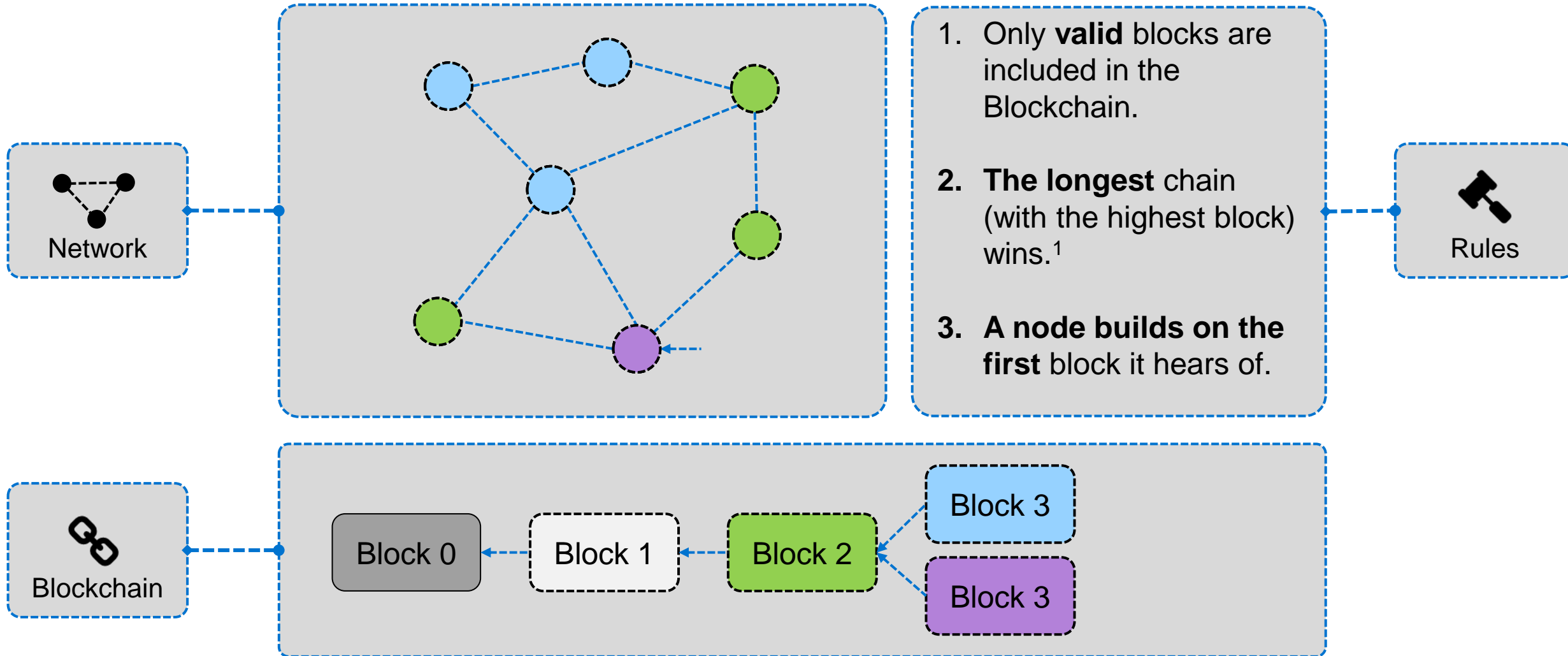
¹ The term “longest chain” is actually the chain with the highest weight. We will explain this term later on.

How do blocks propagate through the network?



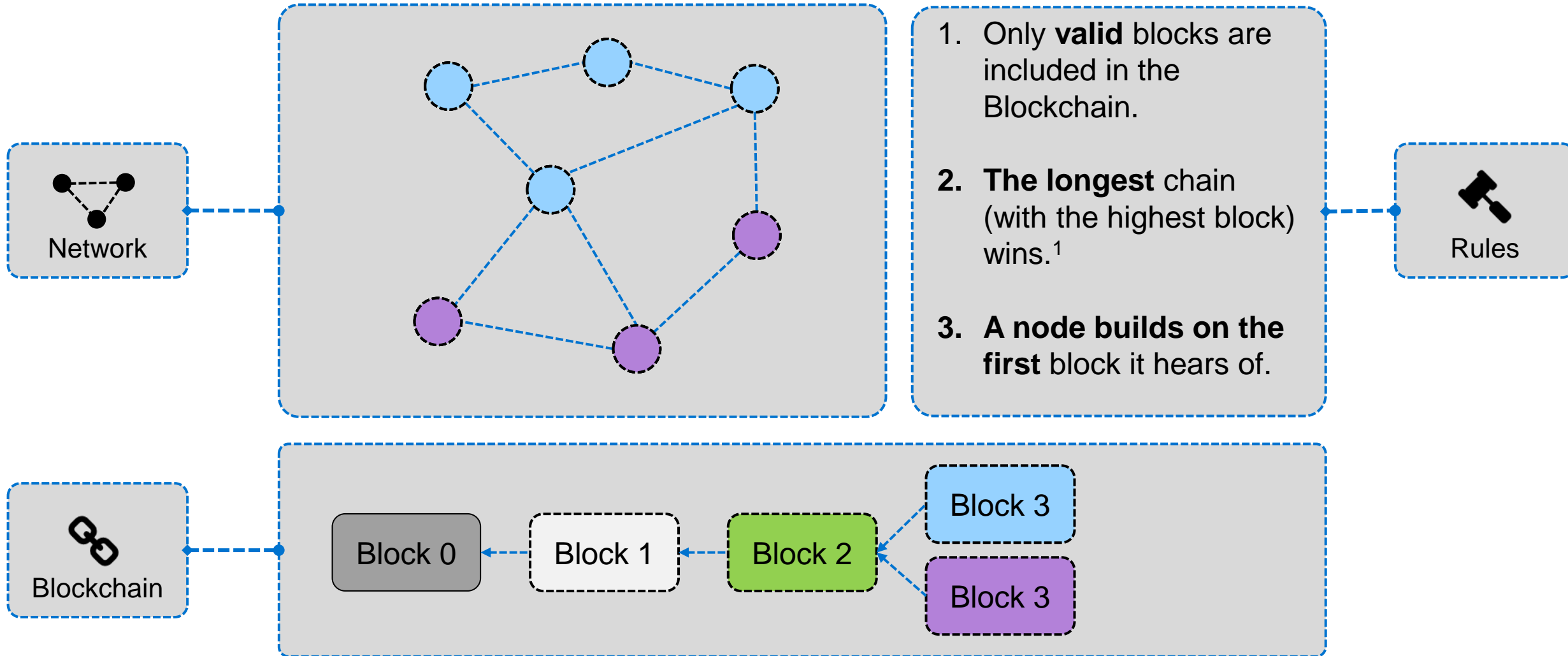
¹ The term “longest chain” is actually the chain with the highest weight. We will explain this term later on.

How do blocks propagate through the network?



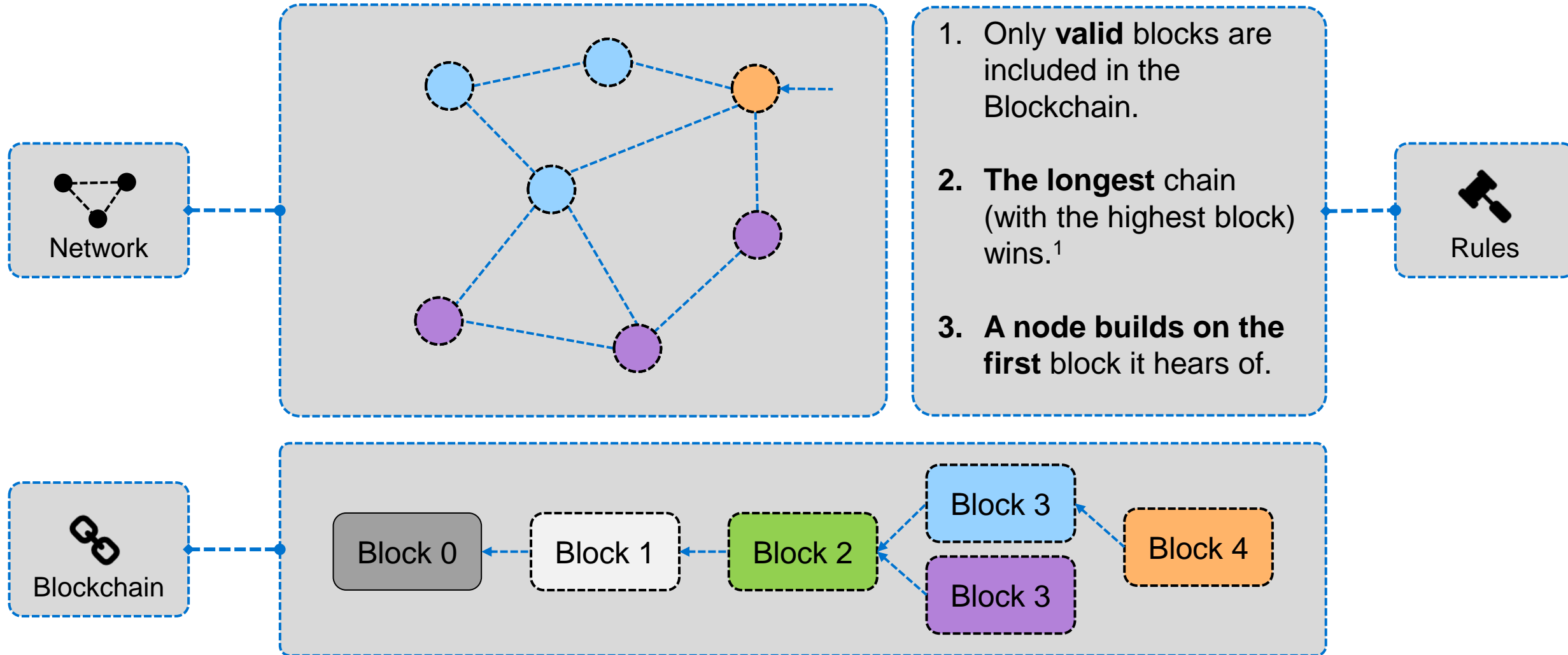
¹ The term “longest chain” is actually the chain with the highest weight. We will explain this term later on.

How do blocks propagate through the network?



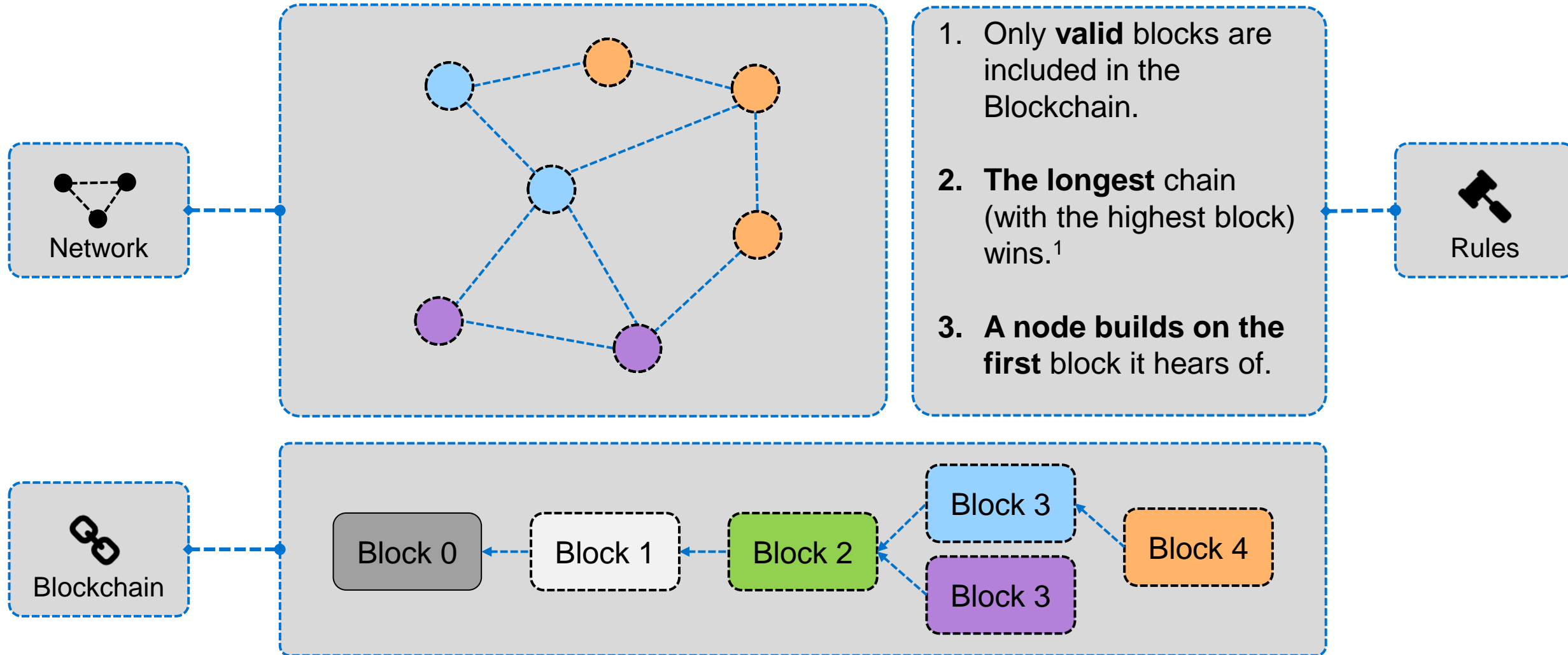
¹ The term "longest chain" is actually the chain with the highest weight. We will explain this term later on.

How do blocks propagate through the network?



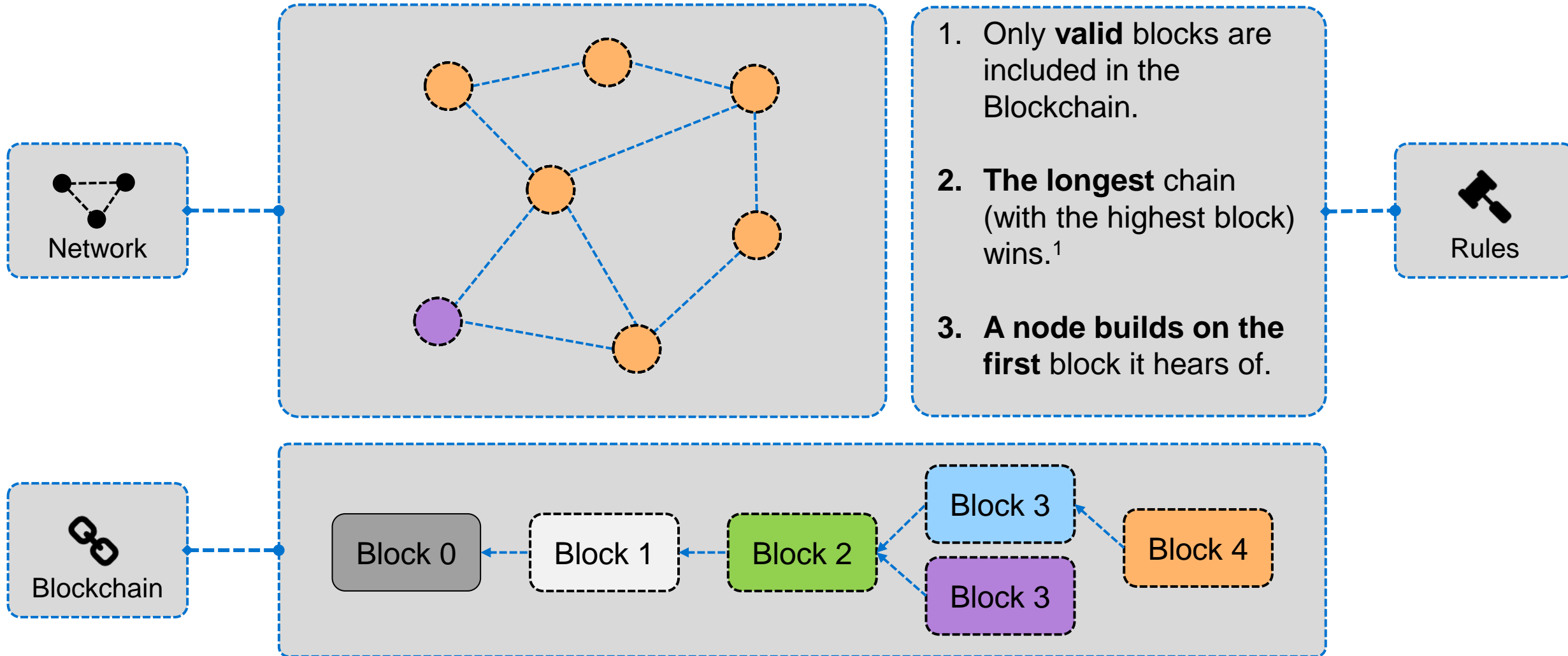
¹ The term “longest chain” is actually the chain with the highest weight. We will explain this term later on.

How do blocks propagate through the network?



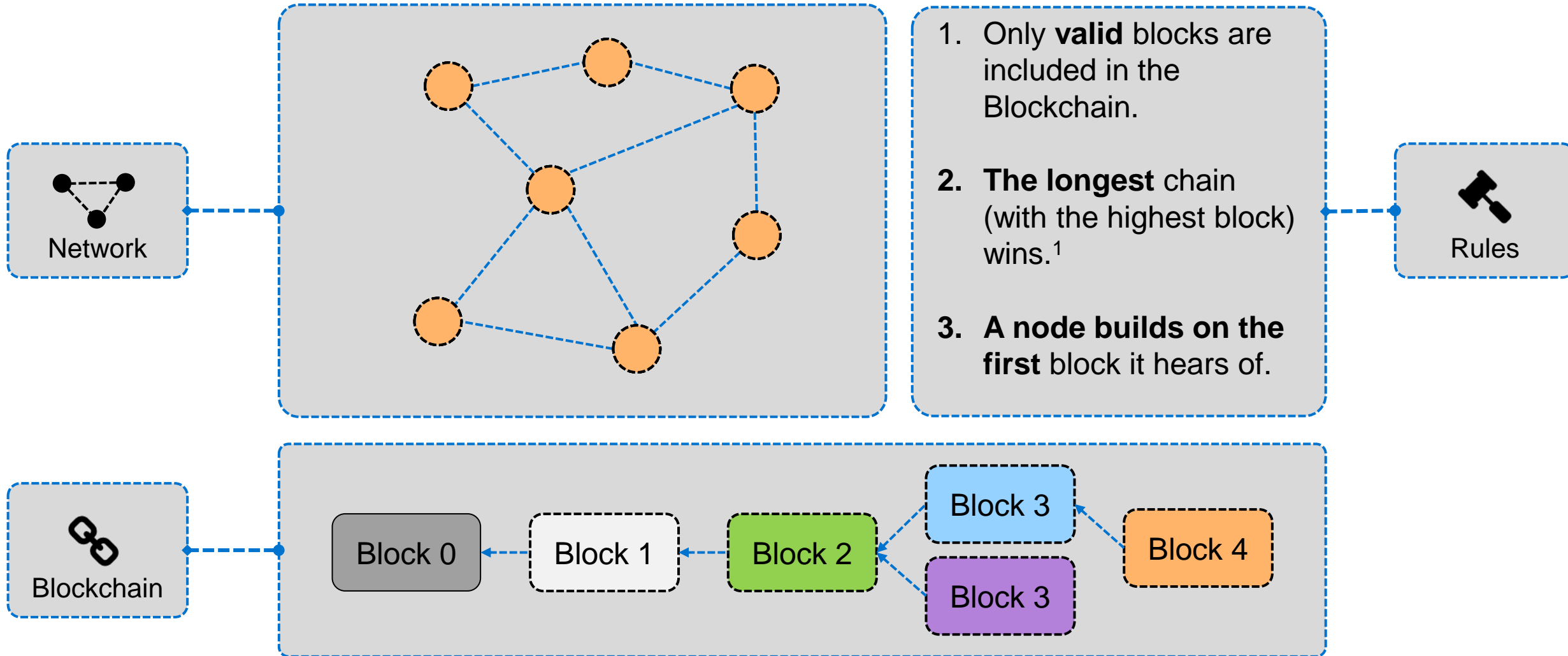
¹ The term “longest chain” is actually the chain with the highest weight. We will explain this term later on.

How do blocks propagate through the network?



¹ The term "longest chain" is actually the chain with the highest weight. We will explain this term later on.

How do blocks propagate through the network?

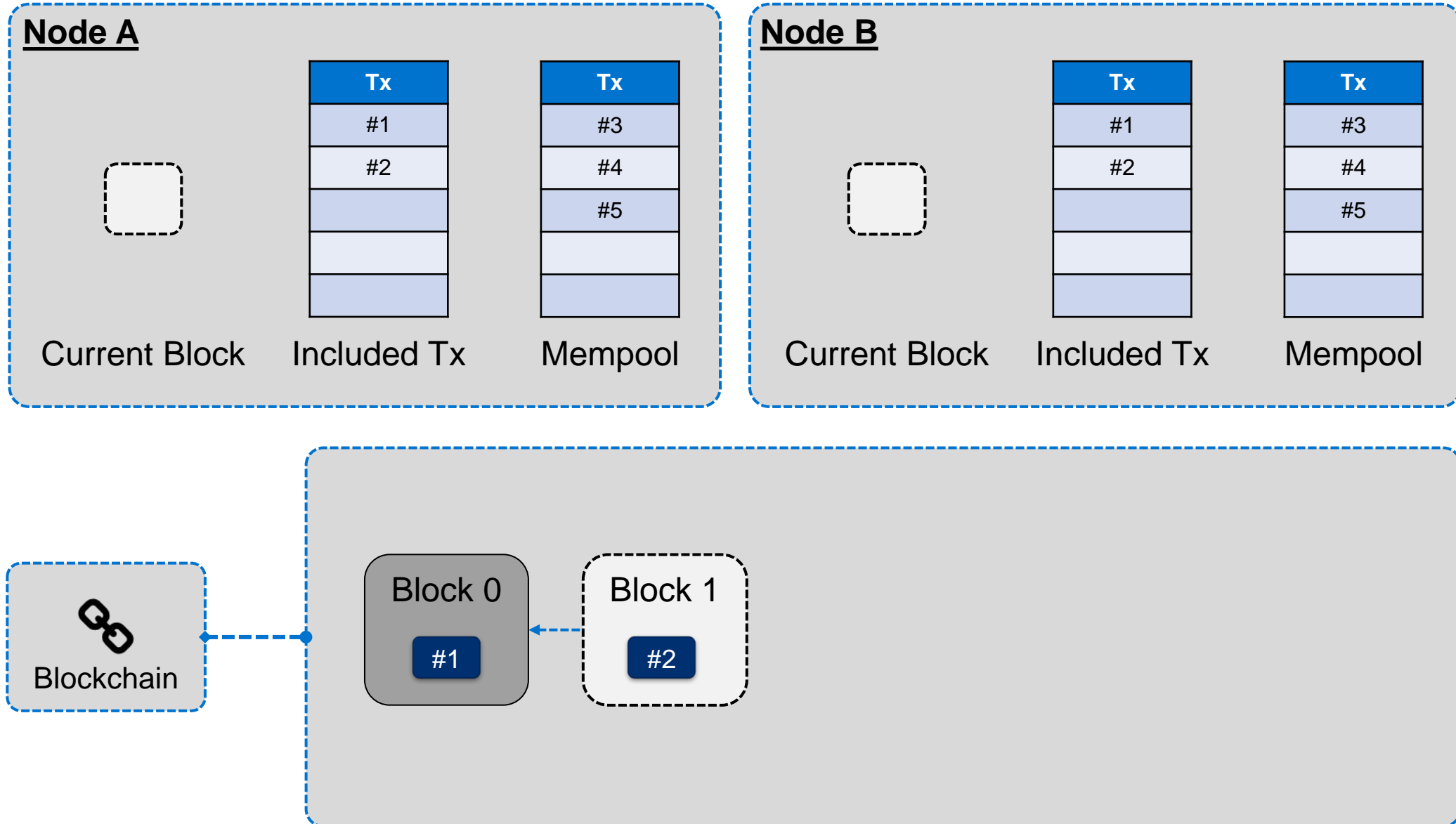


¹ The term “longest chain” is actually the chain with the highest weight. We will explain this term later on.

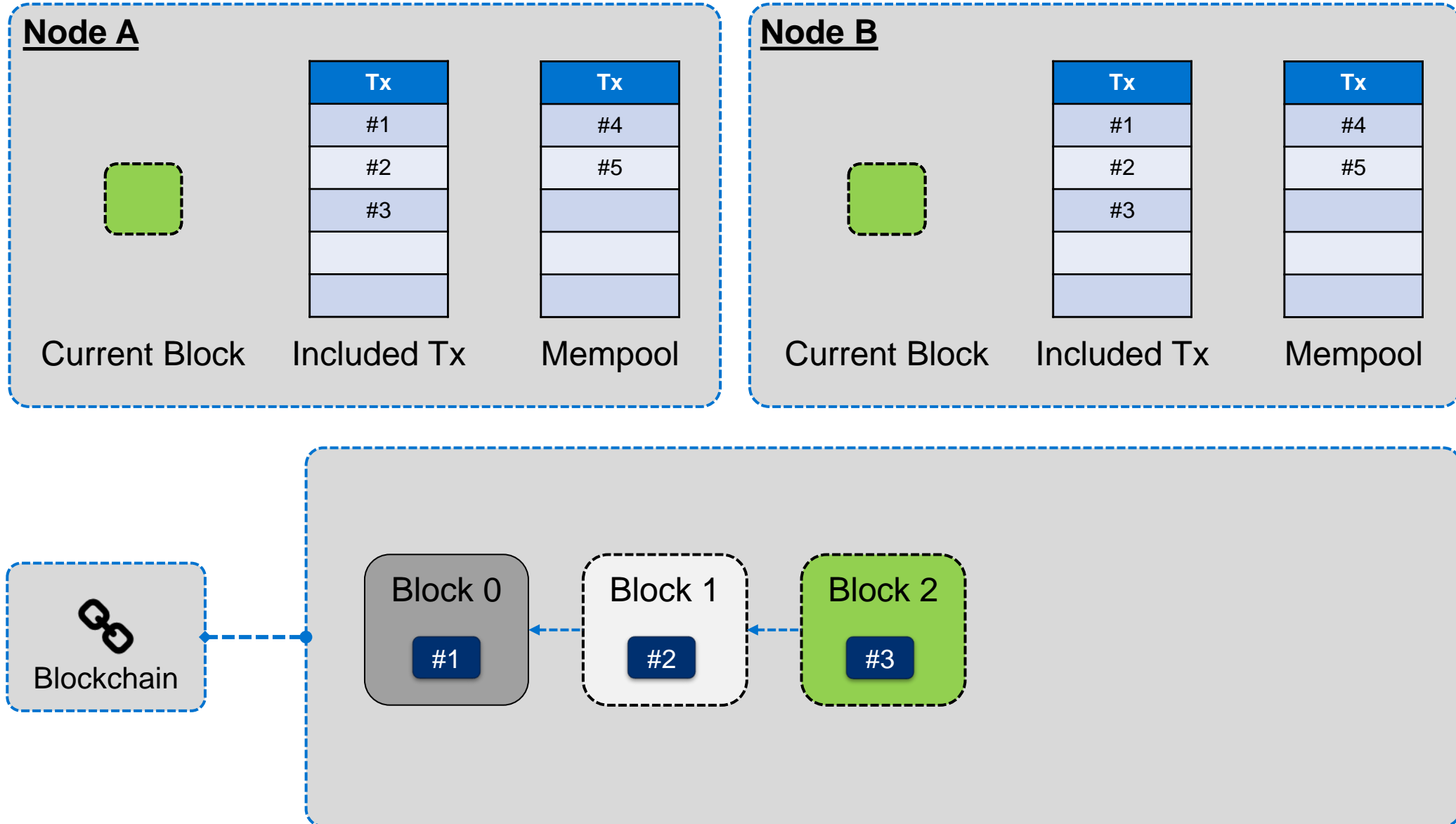
What happens to transactions which are included in the orphan block(s)?

- Unconfirmed transactions are stored in the mempool before they get added to a block.
- As unconfirmed transactions get “gossiped” in the network, every node will know of all transactions.
- As a new block is proposed, all nodes update their mempool and remove the transactions which were included.
- As a consequence, the transactions in an orphan block are simply considered as unconfirmed, waiting to be included in a later block.

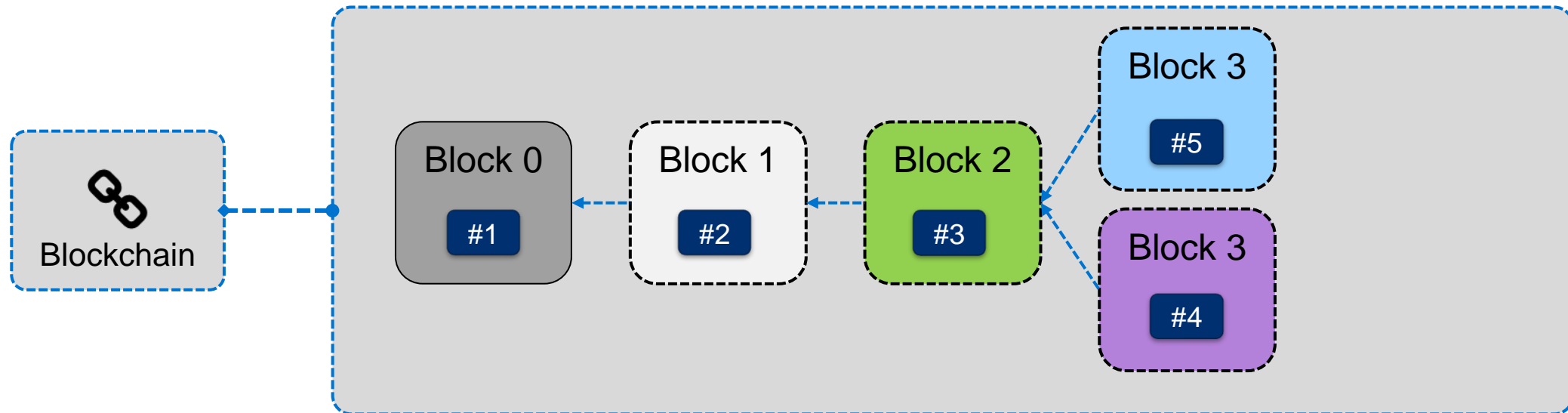
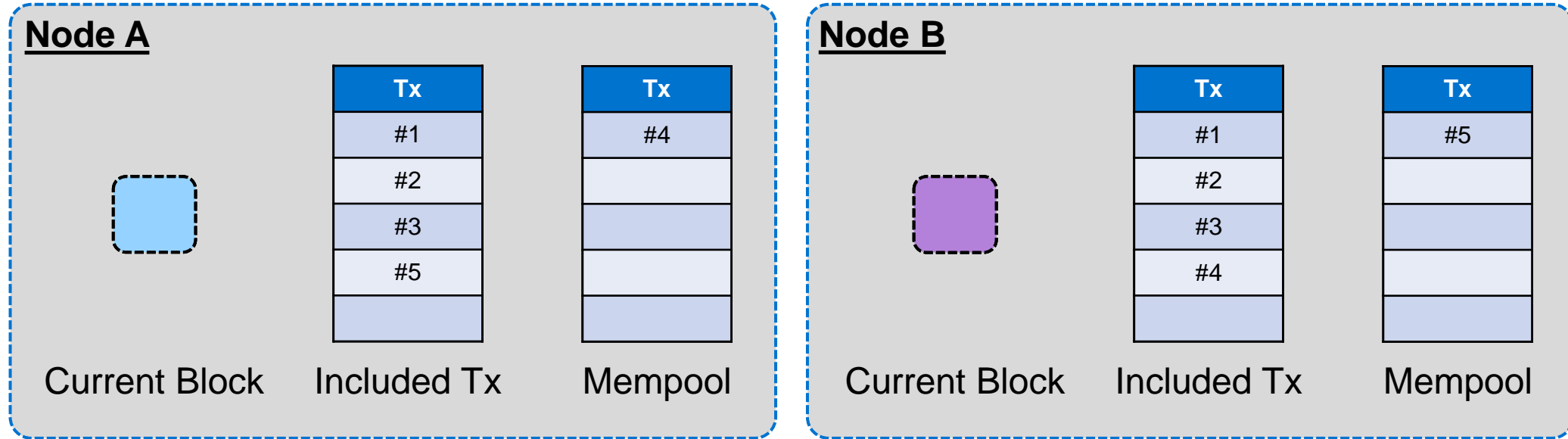
What happens to transactions which are included in the orphan block(s)?



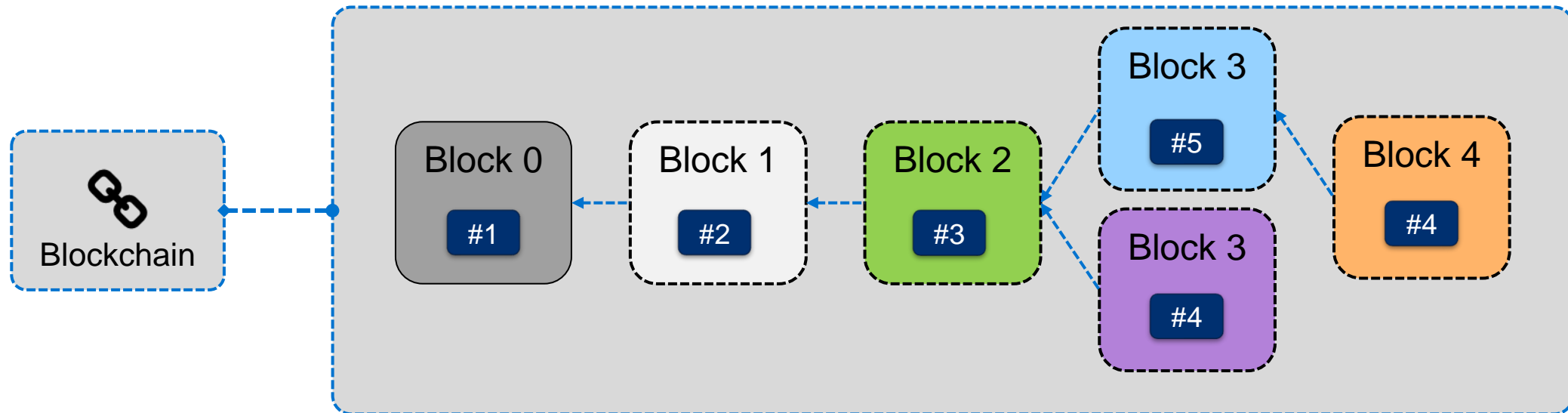
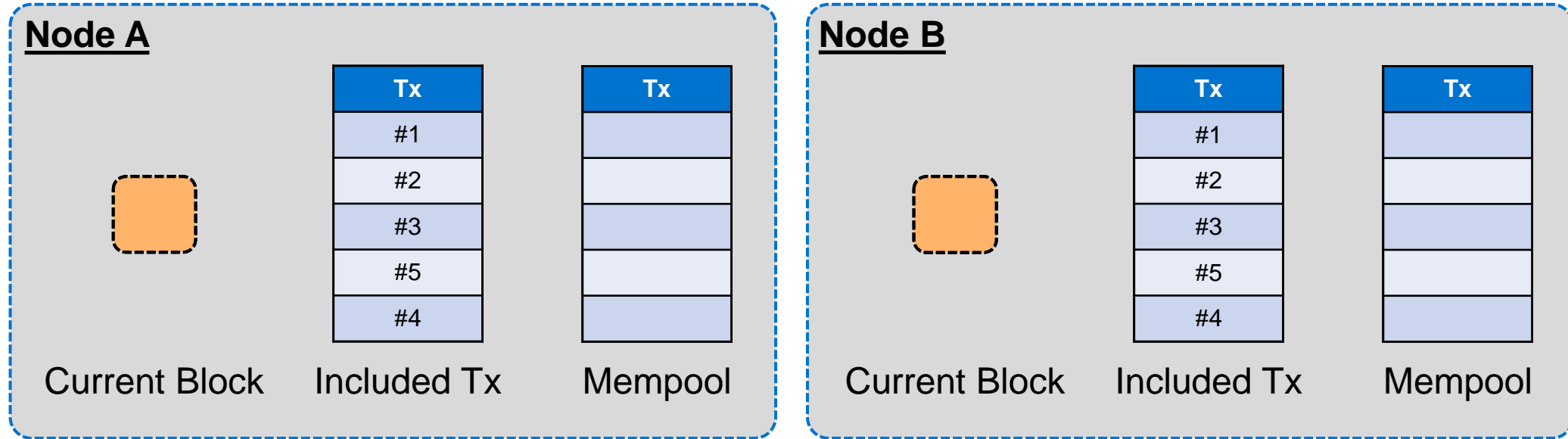
What happens to transactions which are included in the orphan block(s)?



What happens to transactions which are included in the orphan block(s)?



What happens to transactions which are included in the orphan block(s)?



Could the consensus mechanism be attacked?

It is possible to steal bitcoins?

No: Since UTXOs are secured with the hash of the public key of a user¹, the attacker cannot generate a valid transaction spending these UTXOs.

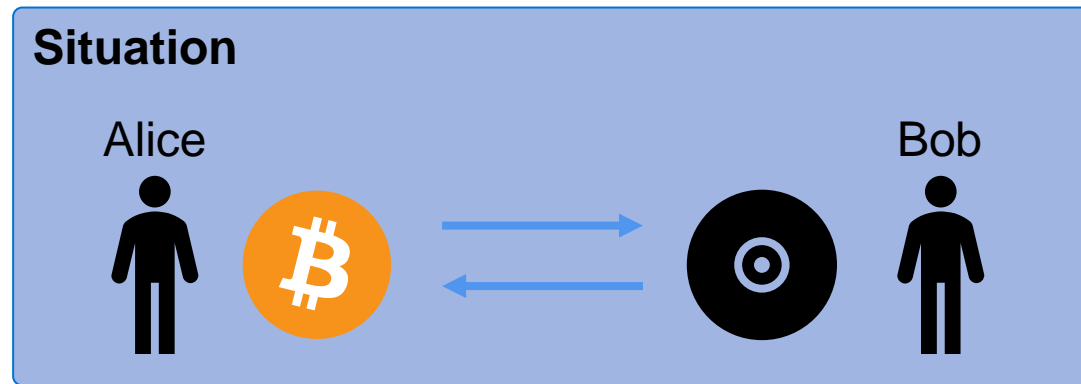
It is possible to block a participant (Wallet Owner) in the Blockchain network?

Assume that a malicious node wants to block all transactions by Bob. This malicious node was selected randomly by the network to propose the new block. It does not include the transaction from Bob, blocking his transaction from getting into its block. However, the next random node (if it is honest) will include Bobs transaction.

¹ The script of the UTXO can also define other requirements for spending.

Attacking the consensus – Double spending

As we saw, the idea of double spending of coins is prevalent. The idea of digital cash did evolve around the idea that we need to prevent a double spend. Two transactions spending the same Txout is somehow hard, but not impossible. We will go into the details of this attack.



Alice wants to buy a music file from Bob's online shop with Bitcoin. She creates a transaction which sends the bitcoins to Bob. A honest node sees Alice's transaction and includes it in its block. Bob sees the new block and sees his transaction included in it. He sends the file to Alice, in good faith to have his money.

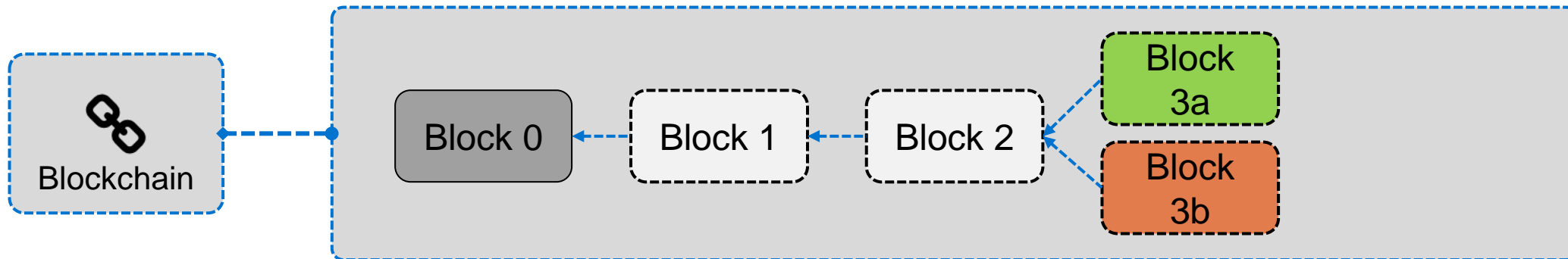
So far so good: What are the options for Alice to "double spend" the bitcoins she sent to Bob?

Attacking the consensus – Double spending (cont.)

Take a look at the underlying Blockchain:

- The green block (block 3a) contains Alice valid transaction to Bob.
- After an honest node proposed this block, Alice was selected to propose the new block.
- What can she do?
 - Option 1: Build on top of block 3a, she accepts the fact that the transaction has happened. This is not what she wants, she wants to double spend!
 - Option 2: Build on top of block 2 a new block 3b (orange), not containing the transaction she sent to Bob, but a transaction spending the same coins (she would have sent to Bob) to herself.¹ This is described as forking.

➔ Classical double-spend pattern



¹ She has to create a transaction spending the UTXO, as if not, the transaction would simply be included in a later block.

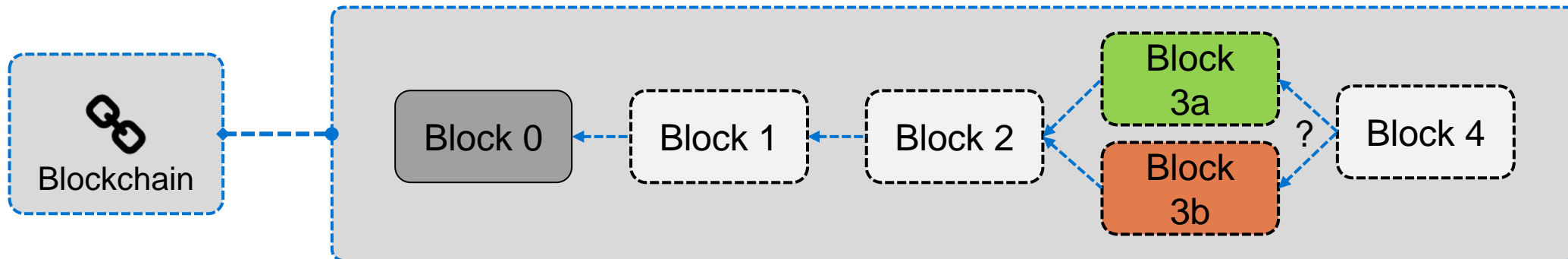
Attacking the consensus – Double spending (cont.)

Does this mean double spending is possible?

No. It is impossible to create a valid block or Blockchain with two transactions consuming the same UTXO. What happens is that two “realities” are created. Block 3a (green) declares a reality where Bob is paid and block 3b (red) declares a reality in which Alice sends the money to herself.

How is this conflict resolved?

The next node that get selected proposing a new block resolves the issue. It has to select the block on which it wants to create its new block. As all nodes adopt the longest chain, one reality (one of the blocks 3) is “orphaned”, meaning this block does not have any relevance to the network anymore.



Attacking the consensus – Double spending (cont.)

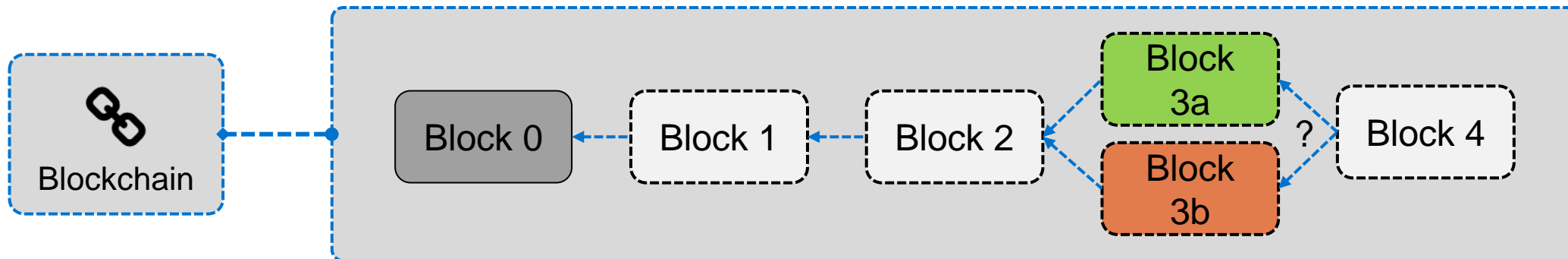
When is the attack successful?

The attack is successful, if Alice convinces the network that her block (block orange) is the valid block that should be included in the longest Blockchain.

From our story, we know that the green block is the “valid” block. From the perspective of an individual node, both blocks are equally valid.

What should Bob do to prevent such attack?

Bob should wait until it is clear that the payment to him is actually included in the longest Blockchain, ideally with several confirmations (blocks on top of the block containing his transaction) before sending the file.



Who is allowed to select the next block?

As already stated before, we want to select a random user to propose our new block.

Problems

- Decentralized network, the network has to agree on who to select
- Unknown user base, users joining and leaving
- Should be resistant to Sybil attacks

How do we do that?

→ **Select a scarce resource!**

→ **Do you know one?**

1. Consensus

- Cryptocurrency with a central authority
- Byzantine generals problem
- Block propagation
- Double spending problem

2. Proof-of-Work (Mining)

- Search puzzle
- Difficulty determination
- Incentives
- Amount of Bitcoin
- Mining hardware
- Mining pools



- Facilitates search puzzle
- Requires large amount of tries
- High investment costs
- High energy costs
- Leads to arms race
- High attack costs
- Fully anonymous mining

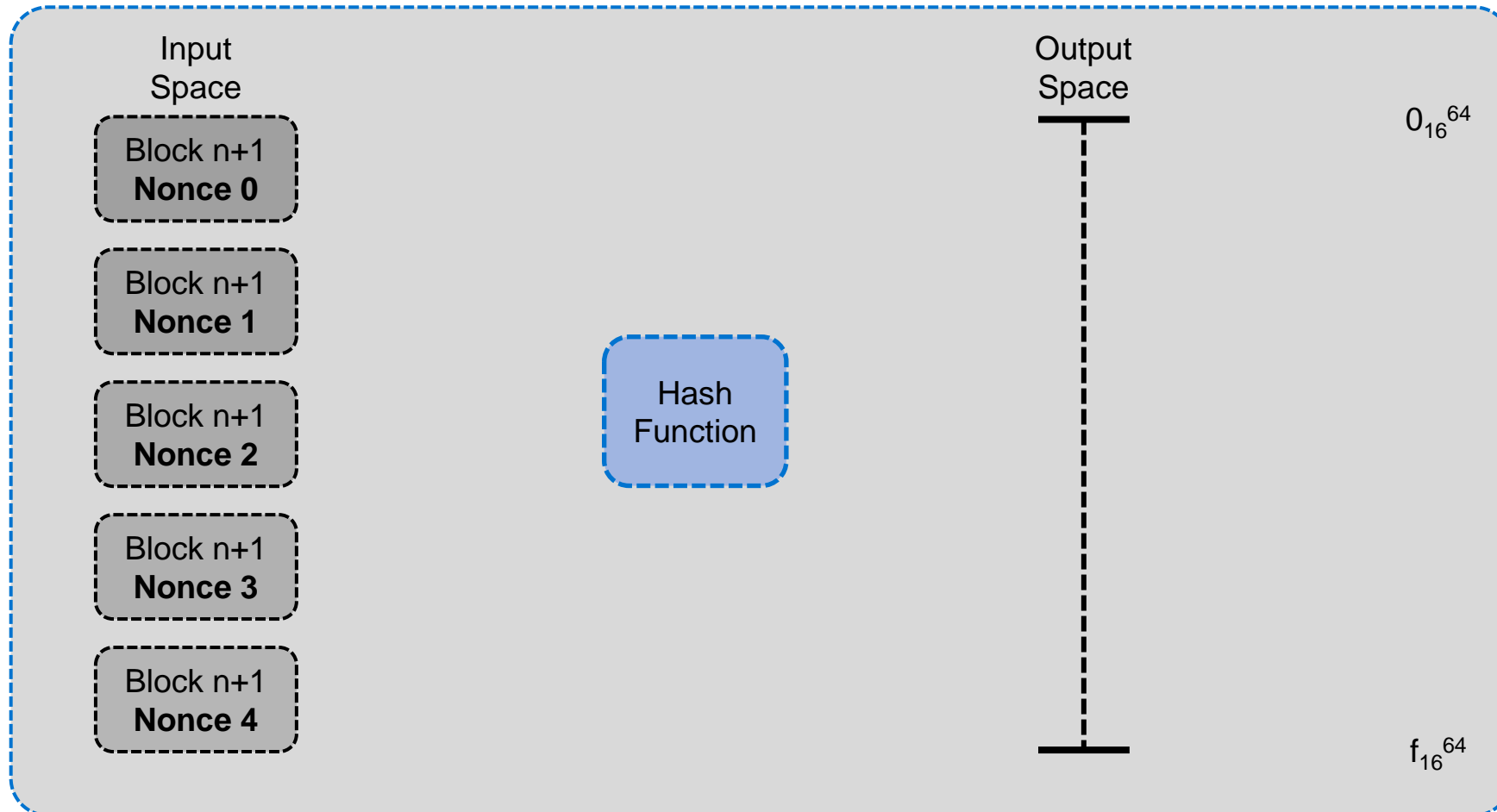
- Coins are deposited to propose new block
- Requires large amount of stake¹
- Low energy costs
- “Rich people getting richer”
- Low attack costs (discouraged by penalties)

Bitcoin uses PoW!

¹ Note that PoS is not suited to bootstrap a Blockchain, as the value of the currency is not bound to another currency.

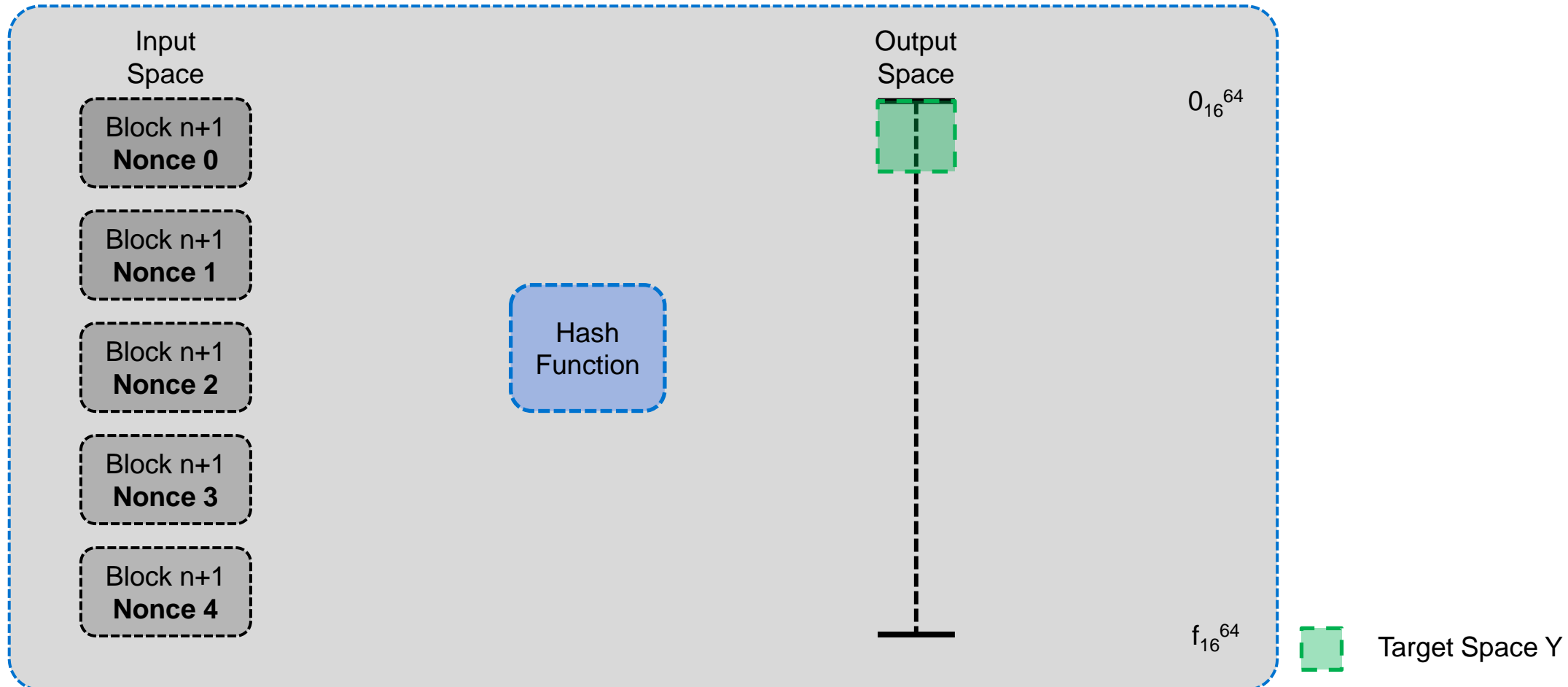
The mining puzzle – Proof of Work (PoW)

Idea: We use the search puzzle introduced in the chapter about cryptographic foundations. The header of the hash has to be included in Y. Bitcoin uses double SHA-256. ($\text{sha256}(\text{sha256}(\text{block}))$)



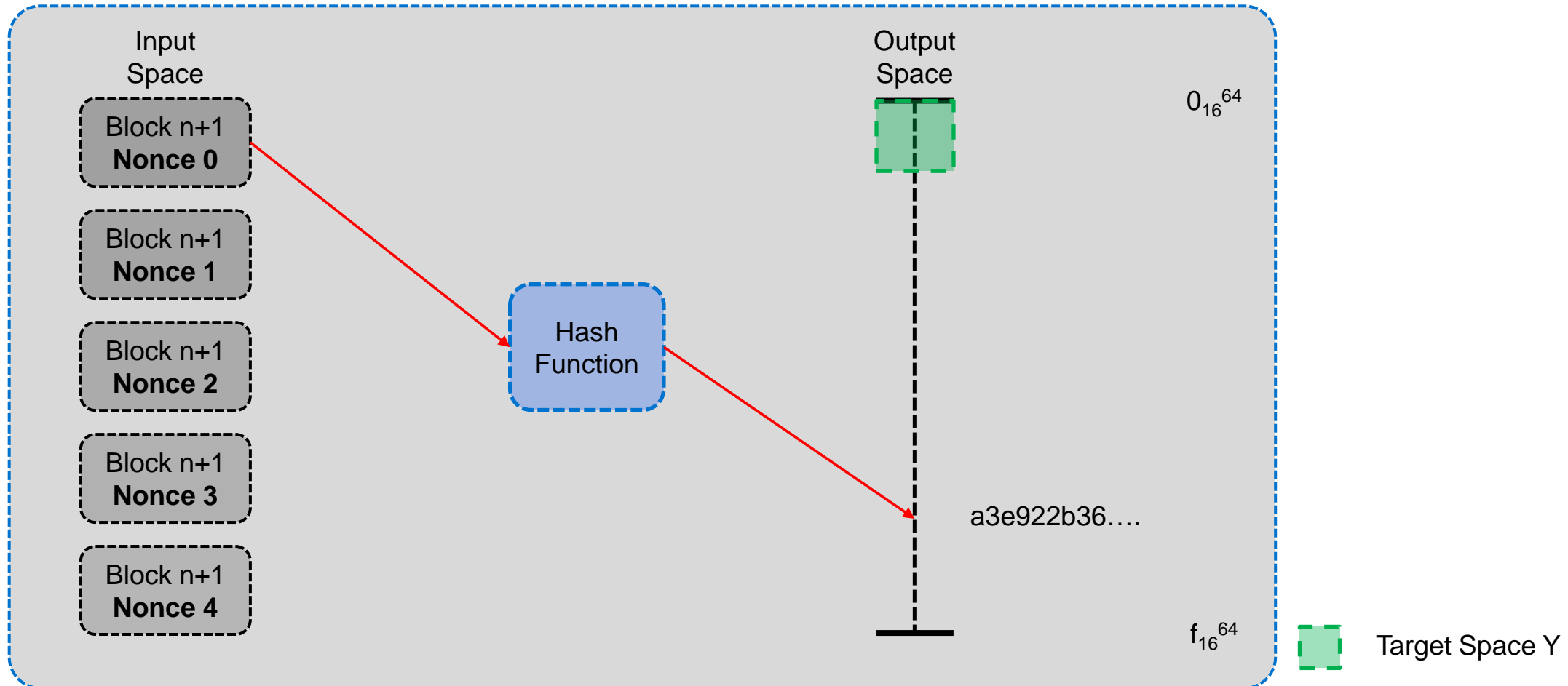
The mining puzzle – Proof of Work (PoW)

Idea: We use the search puzzle introduced in the chapter about cryptographic foundations. The header of the hash has to be included in Y. Bitcoin uses double SHA-256. ($\text{sha256}(\text{sha256}(\text{block}))$)



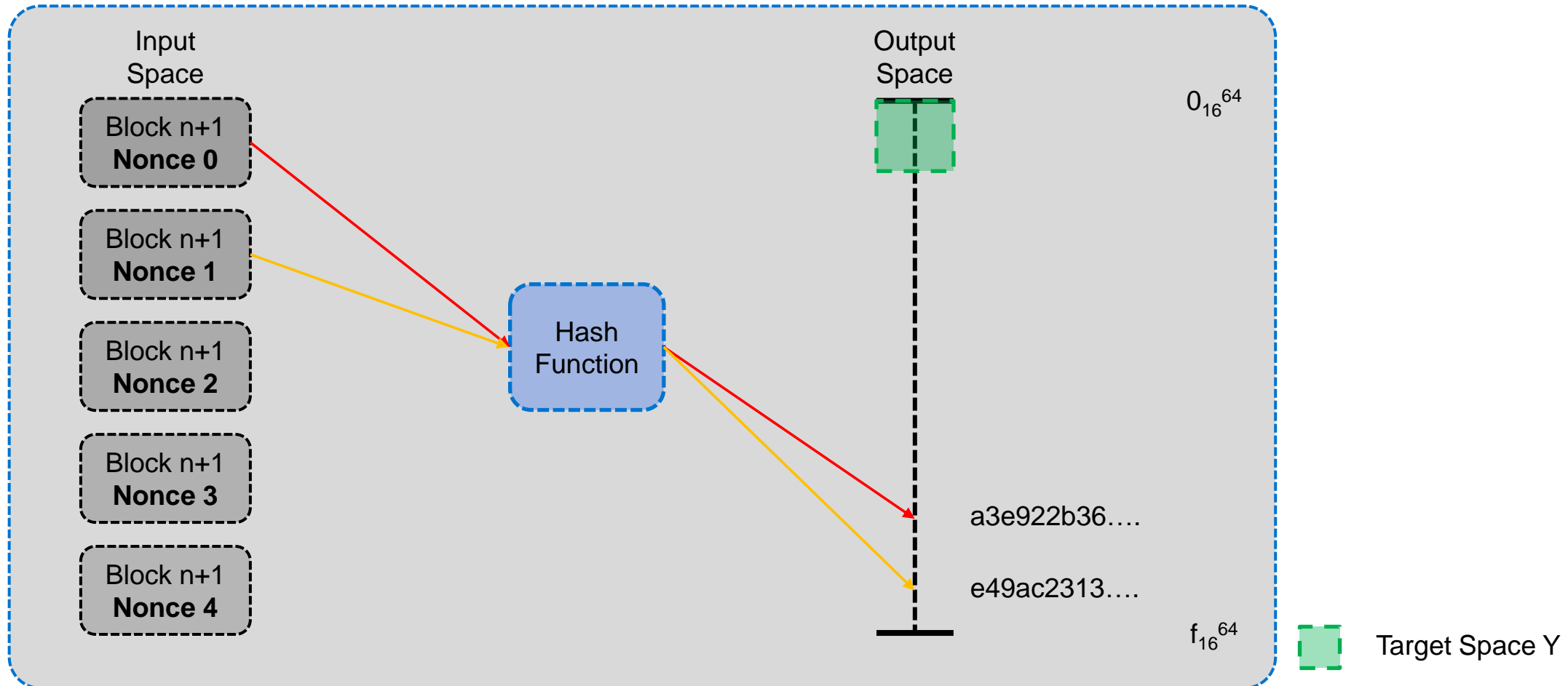
The mining puzzle – Proof of Work (PoW)

Idea: We use the search puzzle introduced in the chapter about cryptographic foundations. The header of the hash has to be included in Y. Bitcoin uses double SHA-256. ($\text{sha256}(\text{sha256}(\text{block}))$)



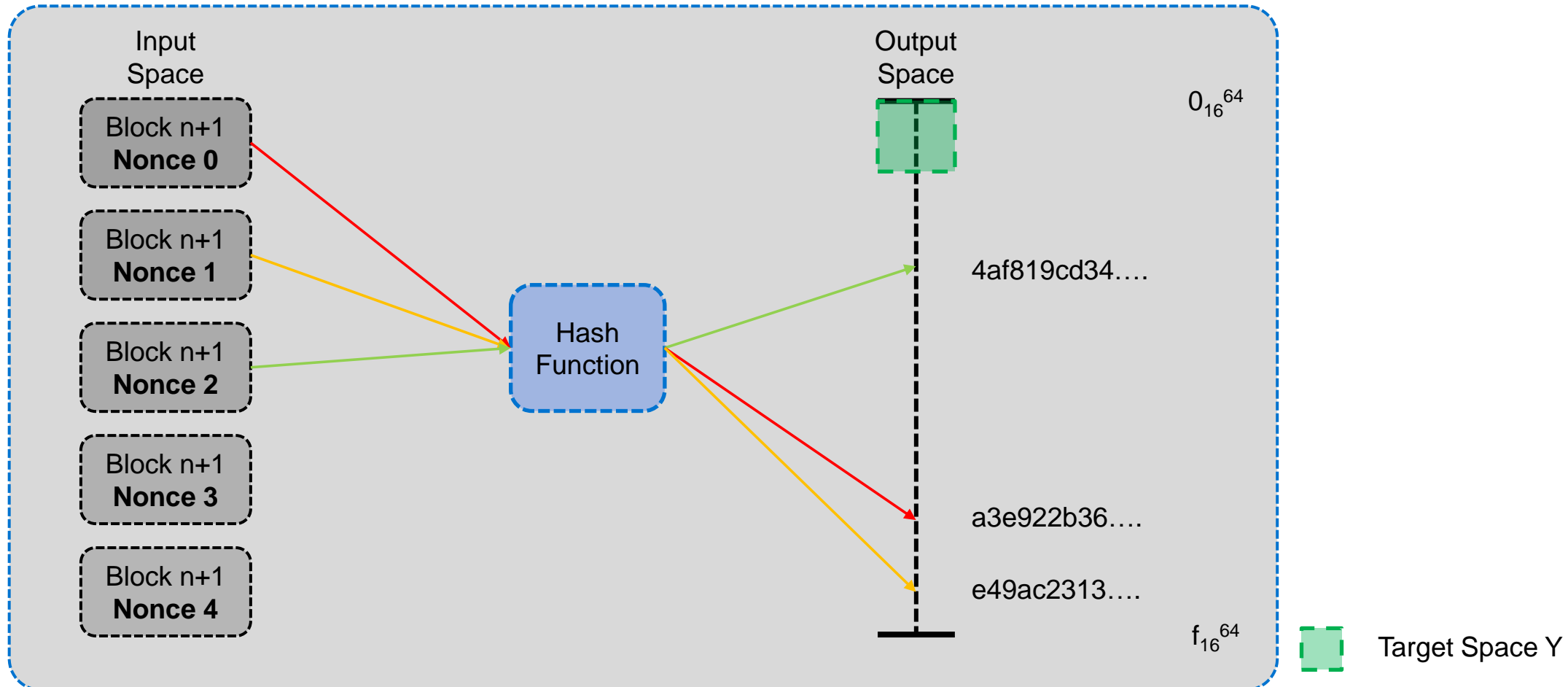
The mining puzzle – Proof of Work (PoW)

Idea: We use the search puzzle introduced in the chapter about cryptographic foundations. The header of the hash has to be included in Y. Bitcoin uses double SHA-256. ($\text{sha256}(\text{sha256}(\text{block}))$)



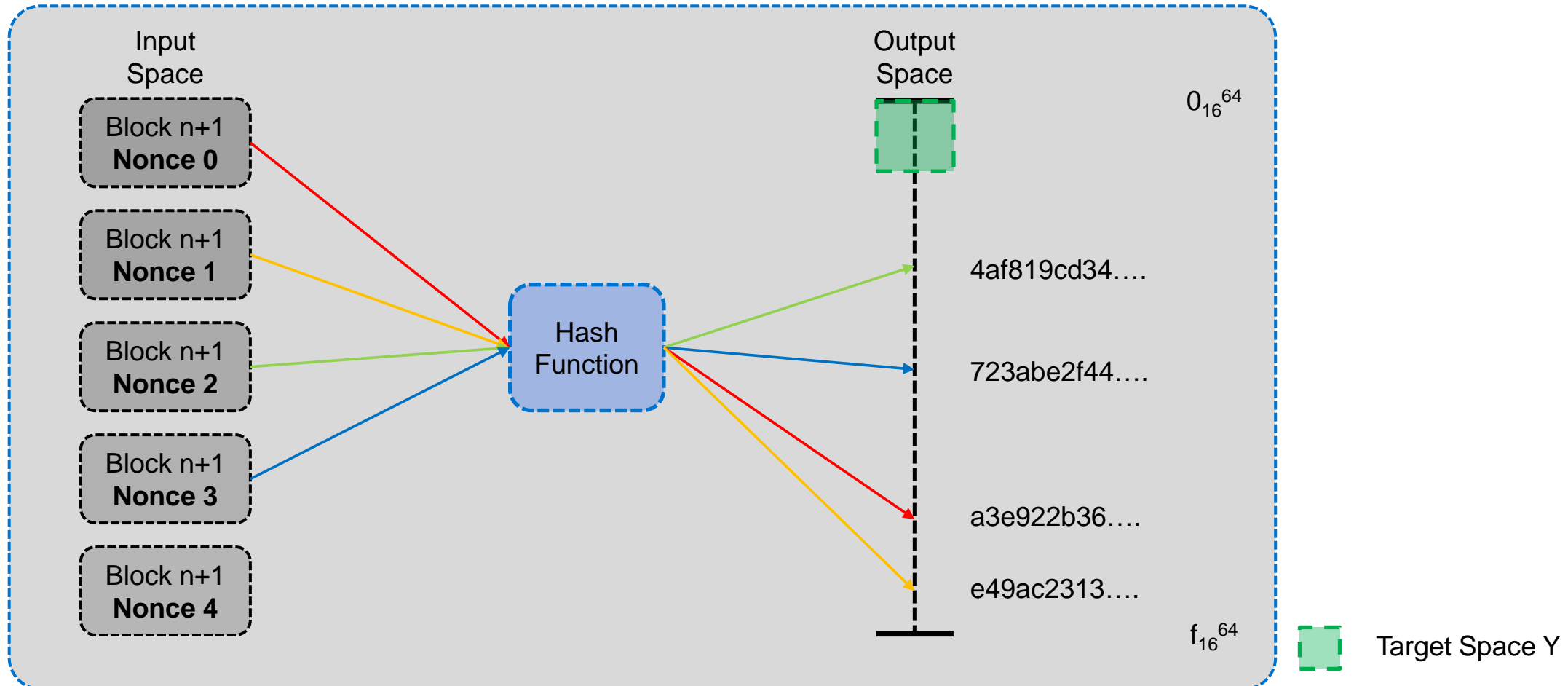
The mining puzzle – Proof of Work (PoW)

Idea: We use the search puzzle introduced in the chapter about cryptographic foundations. The header of the hash has to be included in Y. Bitcoin uses double SHA-256. ($\text{sha256}(\text{sha256}(\text{block}))$)



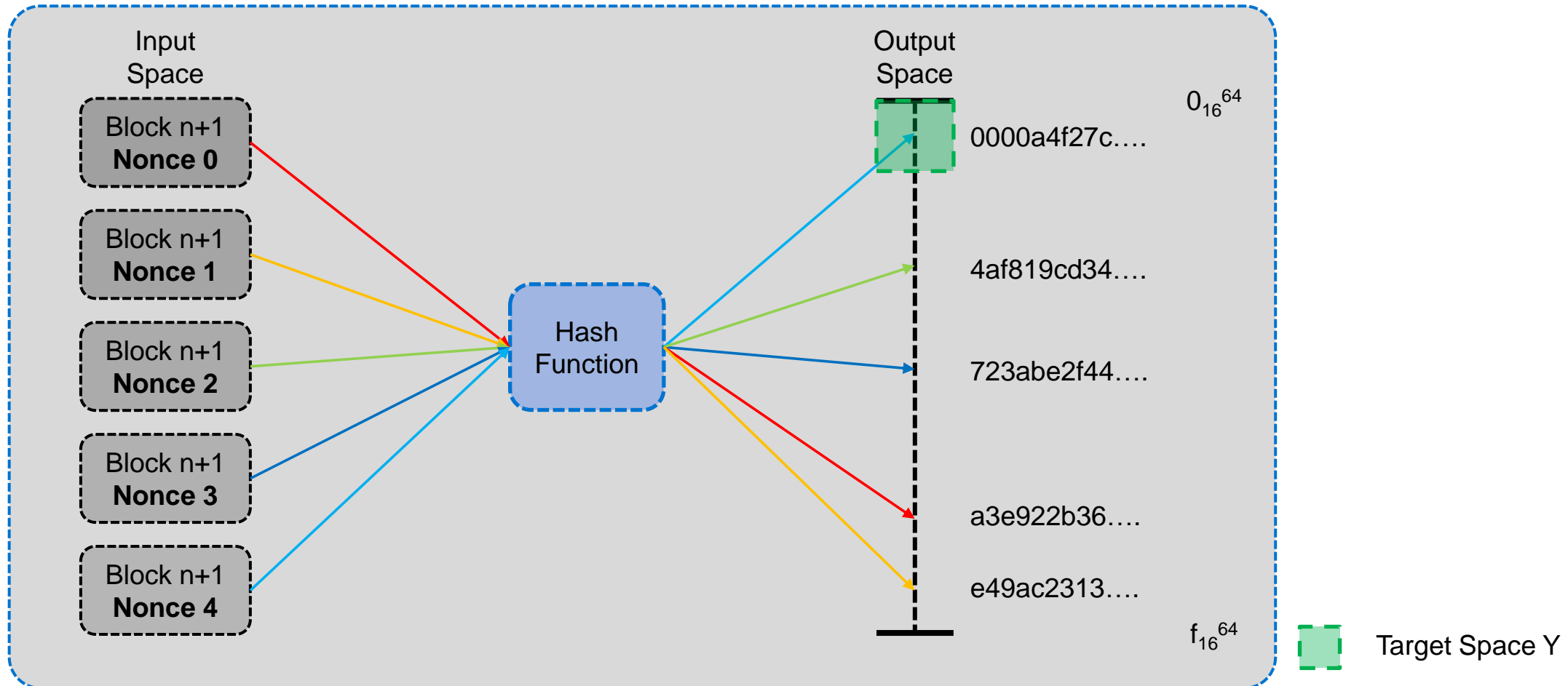
The mining puzzle – Proof of Work (PoW)

Idea: We use the search puzzle introduced in the chapter about cryptographic foundations. The header of the hash has to be included in Y. Bitcoin uses double SHA-256. ($\text{sha256}(\text{sha256}(\text{block}))$)



The mining puzzle – Proof of Work (PoW)

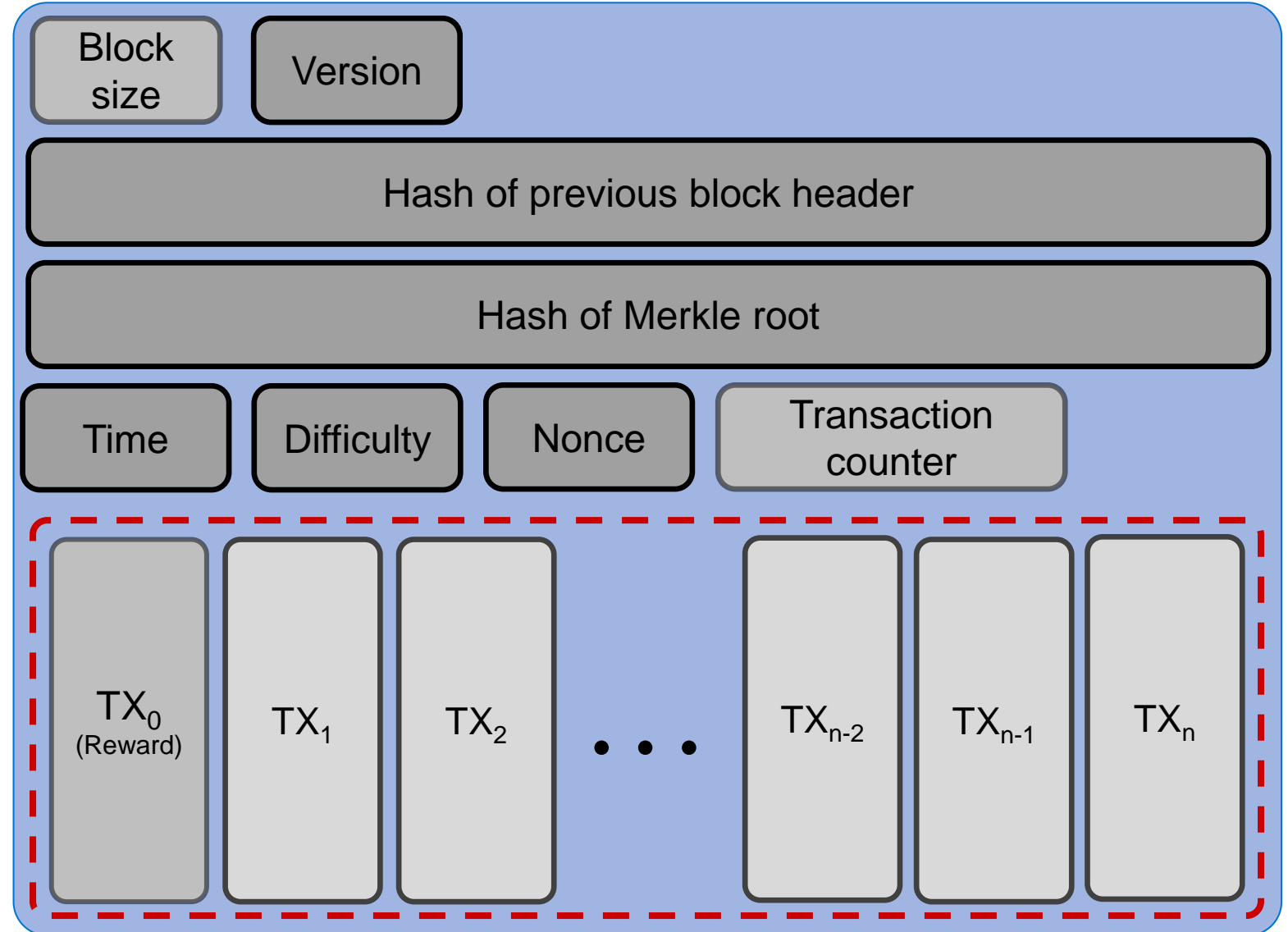
Idea: We use the search puzzle introduced in the chapter about cryptographic foundations. The header of the hash has to be included in Y. Bitcoin uses double SHA-256. ($\text{sha256}(\text{sha256}(\text{block}))$)



Has everyone the same search puzzle?

Recap:

- The block's hash used for chaining is calculated from the *version* until the *nonce* field.
- Assume:
 - There are only 10 Tx in the memory pool. Every node includes all of them in the new block.
 - Every node uses the same time and version.
- Has everyone the same search puzzle?
If not, why not?
- No, every node has a different puzzle, as the TX_0 (the reward-address) is different from node to node.




Difficulty calculation & block time

- The block time defines the average time between the creation of two blocks
- In Bitcoin, block time = 10 minutes
- Why has the block time to be constant?
 - Too slow:
 - Transactions take longer to be included
 - Network capacity decreases
 - To fast:
 - Higher possibility of chain forking, leading to multiple “realities”.
 - Network has to keep track of these forks even if many will be orphaned.
 - Empty blocks
- How do we design the search puzzle in such way that it keeps a constant block time?
- Every 2016 blocks, the difficulty of the puzzle is adapted to the current network speed.
- The longest chain is considered as the chain with the accumulated highest difficulty.

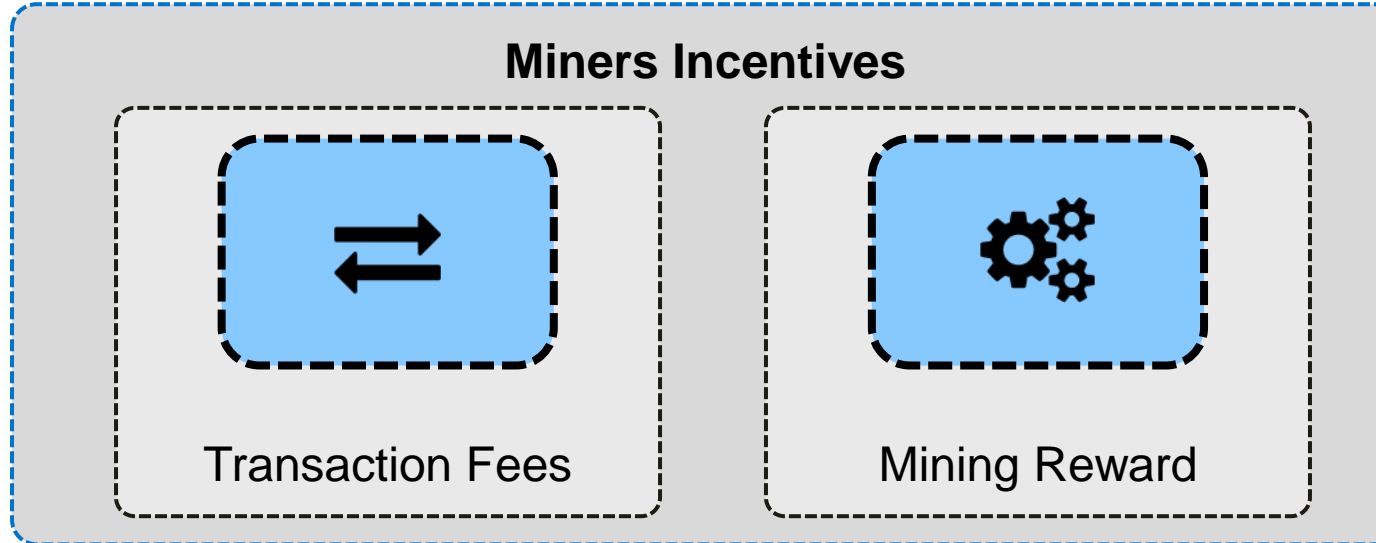
- 1 Measure, how long the last 2016 blocks took to get mined. ($=T$)
- 2 Calculate the factor of speed (two Weeks / T) ($=F$)
- 3 The difficulty gets increased ($F > 1$) or decreased ($F < 1$).
- 3a Maximum increase: 4. Maximum decrease: 0,25.
- 4 The process is done every 2016¹ blocks.

¹14 Days x 24 Hours x 6 (every 10 mins) = 2016

Description	Bitcoin 
Size of Data Field	8 Byte
Representation	Unsigned integer
Smallest Unit	1 Satoshi
Base Unit	1 BTC = 100.000.000 Satoshi
Maximum Amount of BTC	20.999.999,9769 BTC

Why would anyone waste energy on solving a stupid puzzle?

Because of incentives!



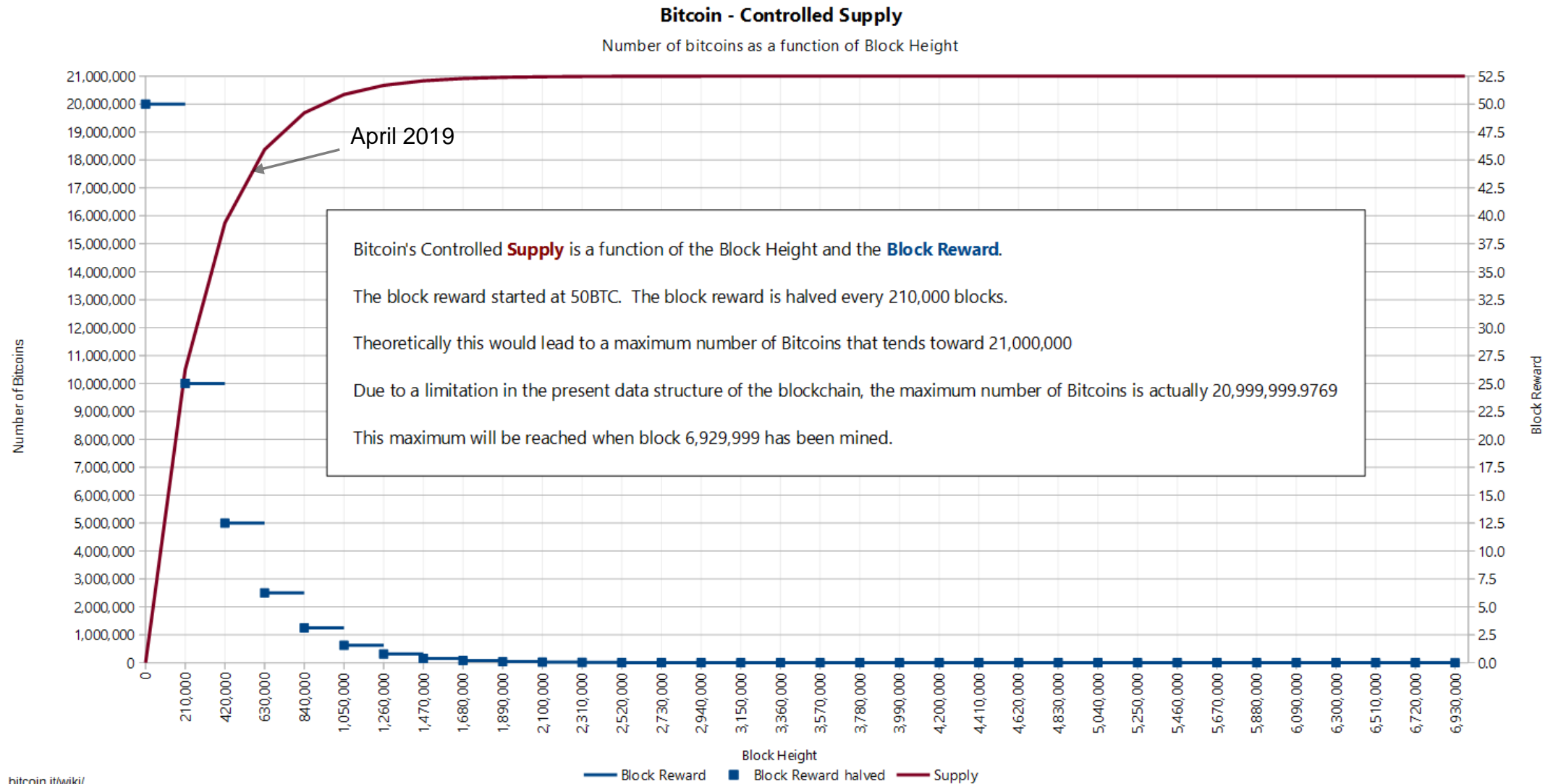
Transaction Fee

Every transaction includes a transaction fee. It is the difference between all inputs and outputs.

Mining Reward

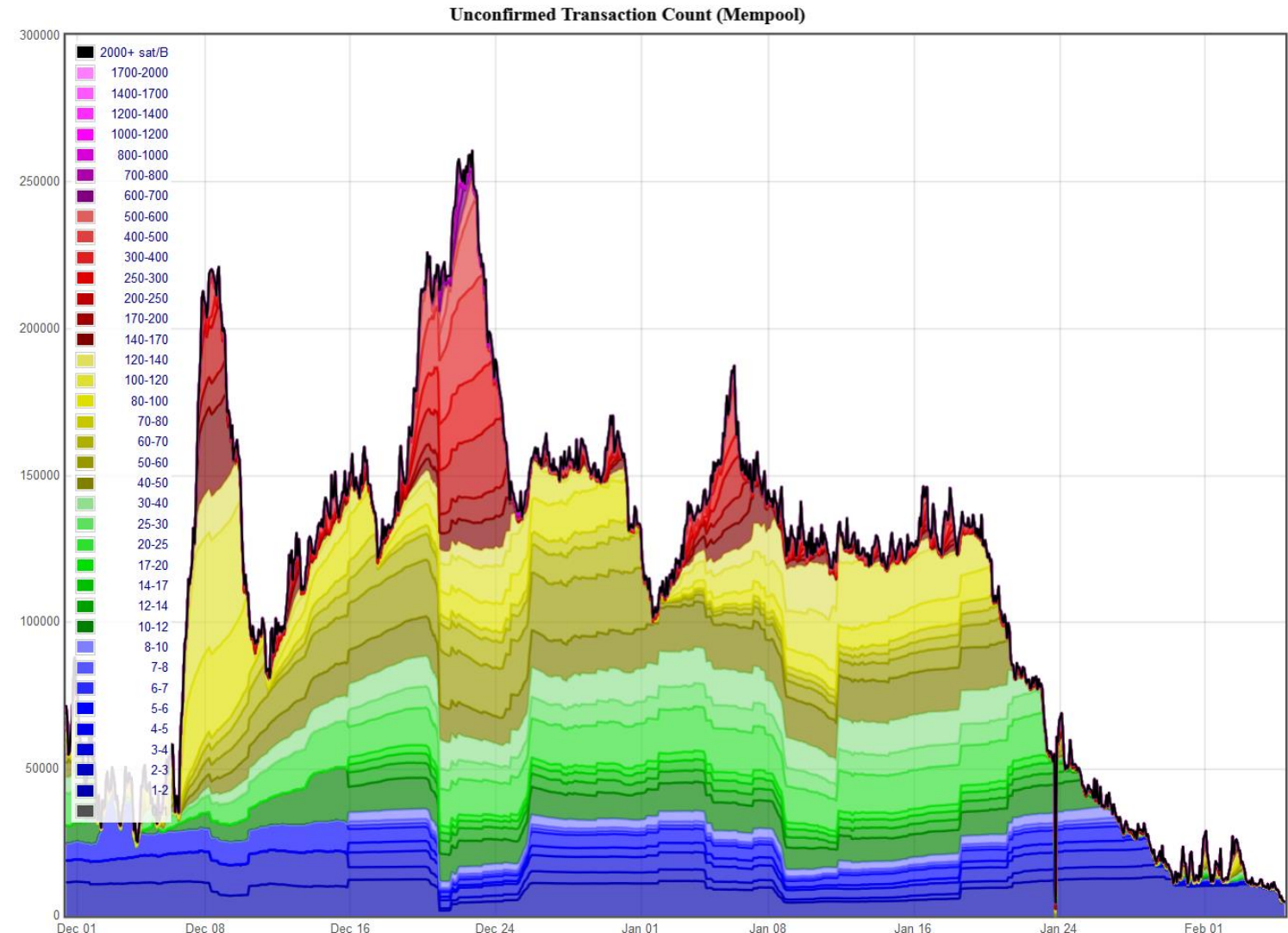
For a newly created block, the miner is allowed to issue new bitcoins to her own wallet.

There is an upper bound of 21 million Bitcoins issued over time.



Transaction fee

- Every transaction includes a transaction fee. It is the difference between all inputs and outputs.
- The miner of the block obtains the transaction fees in addition to the block reward.
- The network can only process between 3 and 6 transactions per second, therefore some transactions have to wait longer.
- The higher the transaction fee, the faster the transaction gets included in the Blockchain.
- The miners are incentivized to mine the high-fee transactions first.
- The fee is calculated in Satoshi¹/byte.



The website representing this graph can be found here: <https://jochen-hoenicke.de/queue/>. This image shows the mempool from 01.12.2017 to 15.02.2018.

¹ Satoshi equals 10^{-8} bitcoin. It is the smallest value in the Bitcoin network.

Coinbase transaction

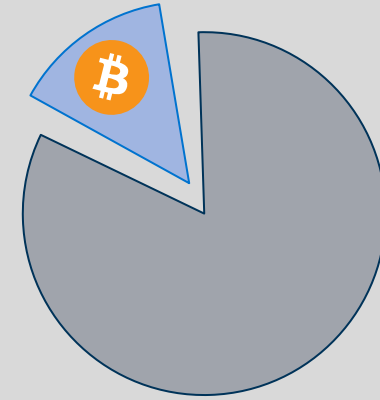
We already mentioned the special type of transaction, the so called **Coinbase transaction**

- The coinbase transaction is the first transaction in a block
- The coinbase transaction has a Txin that references no Txout (called **Coinbase**)
- The miner who finds the block is entitled to the coinbase transaction and therefore the block reward consisting out of
 - Block reward (newly available Bitcoins which are introduced in the system)
 - Transaction fees
- The contents of the coinbase transaction are
 - (In Bitcoin) the block height
 - Up to 100 arbitrary bytes can be put into the transaction input (scriptSig)

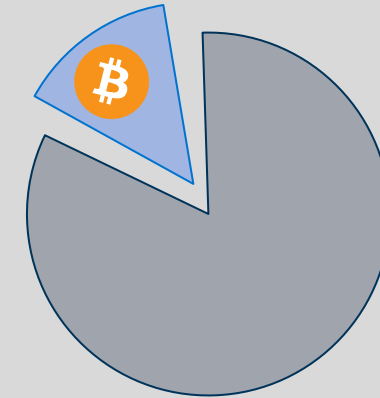
Arms race in mining

- The process of mining can be a profitable business. However, there are some remarks:
- 1.800 Bitcoins are generated on a daily basis. If computing power is increased, the difficulty of the search puzzle increases, too.
→ The overall output of mining reward stays the same.
- Example
10 entities own hardware with 10.000 Th/s. Each receives roughly 180 Bitcoins per day. Each of them decides to double their hash-rate to gain more Bitcoins. Now everyone has 20.000 Th/s, but still earns 180 Bitcoins per day as his share stays at 10% of the network hash rate.
- → If a miner wants to increase its revenue, it has to invest more than the others. As everyone thinks this way (otherwise his revenue will decrease), this leads to an **arms race**.

Situation 1: 10.000 Th/s



Situation 2: 20.000 Th/s



Mining hardware



2009
CPU

CPUs were the first hardware to mine Bitcoins.



2010
GPU

GPUs are faster than CPUs. First mining software was introduced in 2010.



2011
FPGA

FPGA (field programmable gate array) are much more energy effective than GPUs.



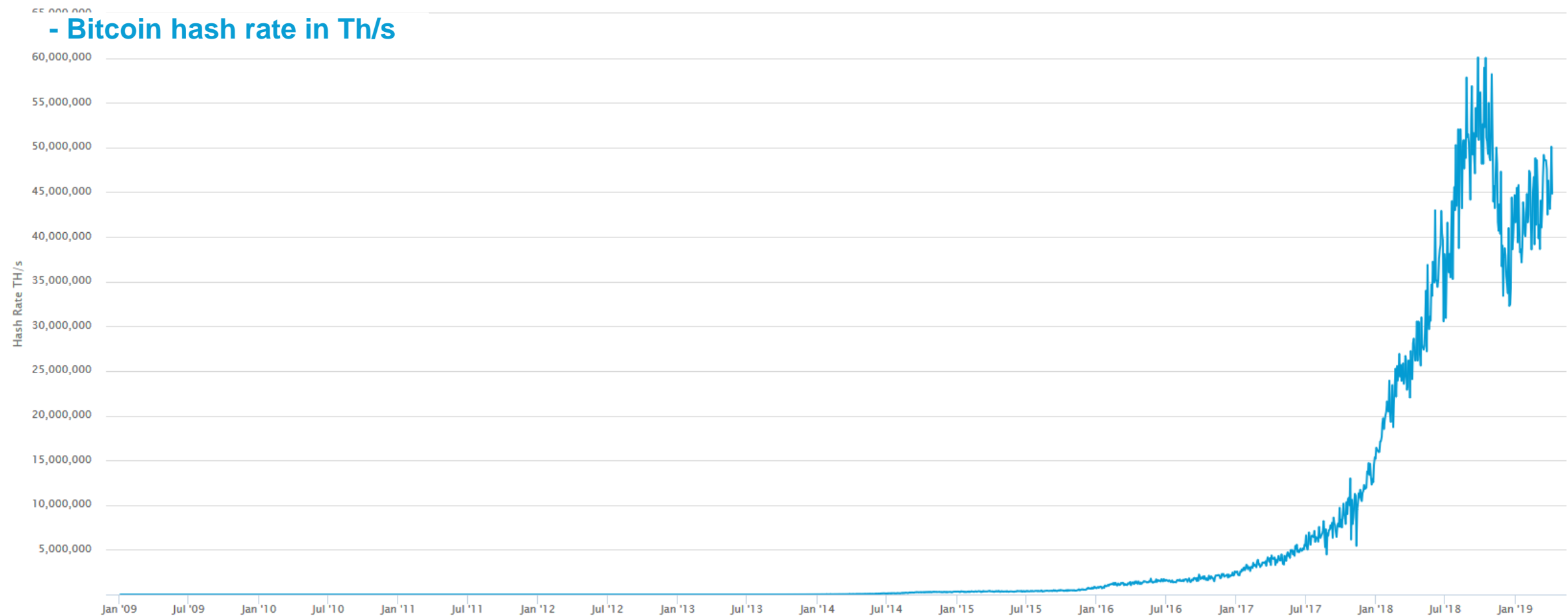
2013
ASIC

ASIC (application-specific integrated circuit) are chips specially designed for mining. Fastest mining.

FPGA image taken from https://en.wikipedia.org/wiki/File:Icarus_Bitcoin_Mining_rig.jpg by Xiangfu, cropped. CC BY-SA 4.0

ASIC image taken from https://commons.wikimedia.org/wiki/File:Cryptocurrency_Mining_Farm.jpg by Marco Krohn, not modified. (CC BY-SA 4.0)

Mining Hardware and difficulty (cont.)



With increasing difficulty, miners face problems:

- Hardware costs are high (high fixed costs)
- Electricity and cooling costs are high (high variable costs)
- Decreasing market share (own hash rate vs. overall hash rate)
- A block is either found or not → no condolence reward

Solution:

- Miners work together in mining pools to stabilize their monthly income
- A pool is organized by the pool manager

Assume percentage of overall hash rate:
 $1 / 2.000.000^1 = 0,0000003$ (0,00003%)

Blocks proposed per year:
 $6 * 24 * 365 = 52.560$ blocks

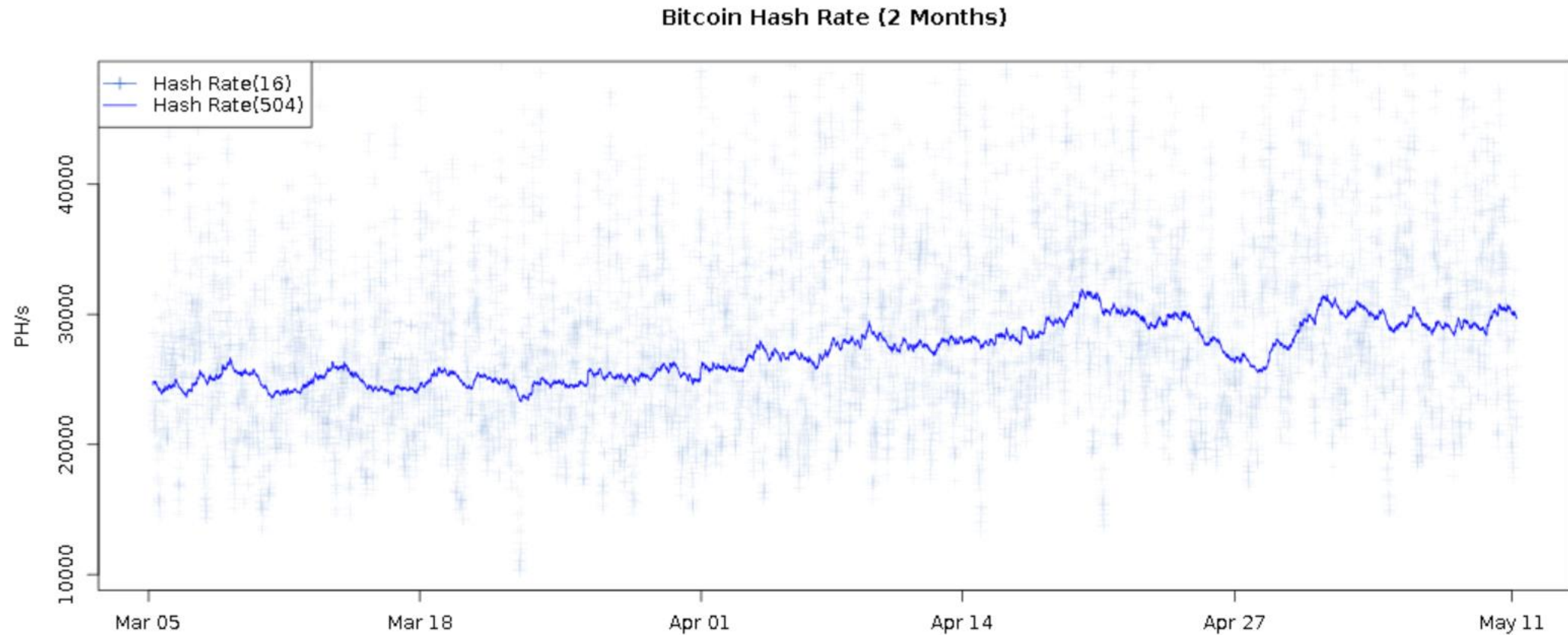
Expected number of blocks per year:
 $0,0000003 * 52.560 = 0,0158$ Blocks / year

How does a mining pool work?

- The pool manager proposes for each new block height a “block prototype” to his pool, requiring the PoW done. ($TX_0 \rightarrow$ pool manager)
- Near-solutions are sent to the pool manager to prove some work. Each proof is called a share.
- Solutions are also sent to the pool manager and pushed into the network.
- Pool manager receives mining reward and distributes it among the shares, keeps a fee.

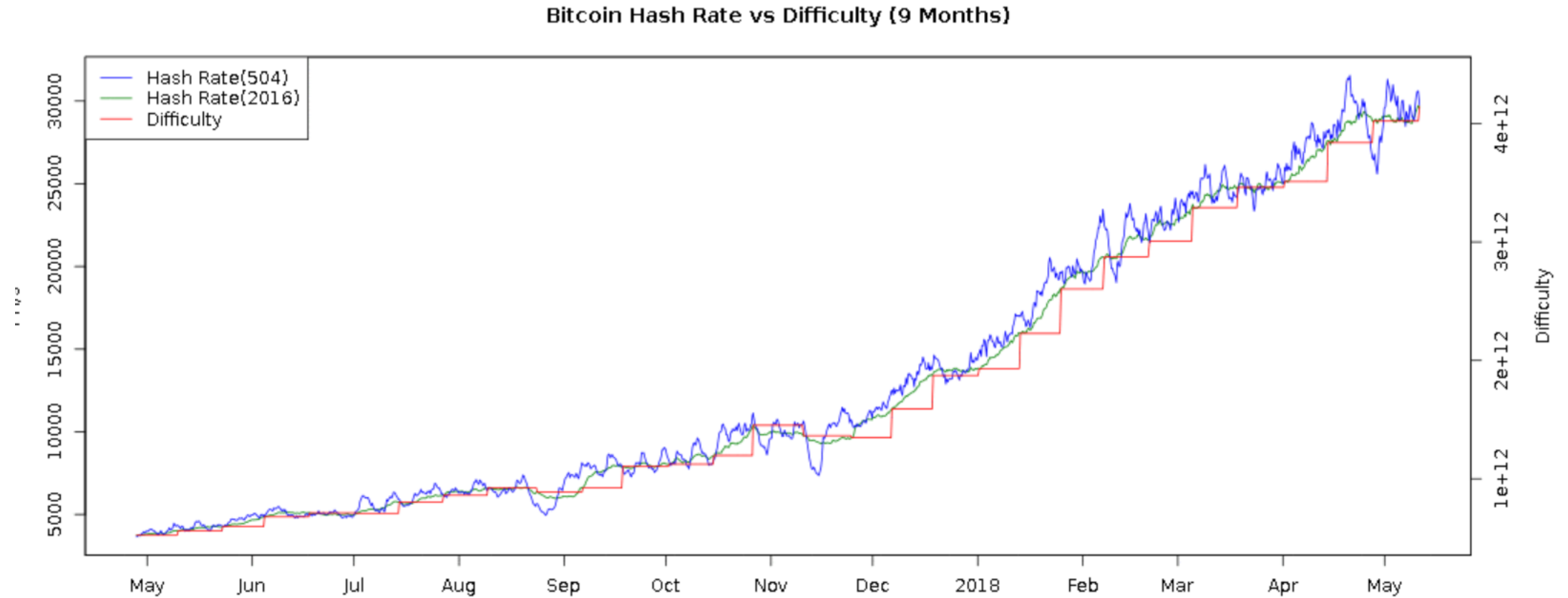
¹We will see later where this number comes from.

The hash rate is a stochastic variable



Graph at <https://bitcoinwisdom.com/bitcoin/difficulty>

The difficulty adapts to the hash rate



Graph at <https://bitcoinwisdom.com/bitcoin/difficulty>