Aula 18 - Ethereum

Visão Geral

Prof. Rogério Aparecido Gonçalves[1]
rogerioag@utfpr.edu.br

[1]Universidade Tecnológica Federal do Paraná (UTFPR)
Departamento de Computação (DACOM)
Campo Mourão - Paraná - Brasil

Programa de Pós Graduação em Ciência da Computação
Mestrado em Ciência da Computação
PPGCC17 - Tópicos em Redes de Computadores e Cibersegurança

Agenda i

1. Introdução

2. Um pouco mais de Ethereum

3. Próximas Aulas

4. Referências

Introdução

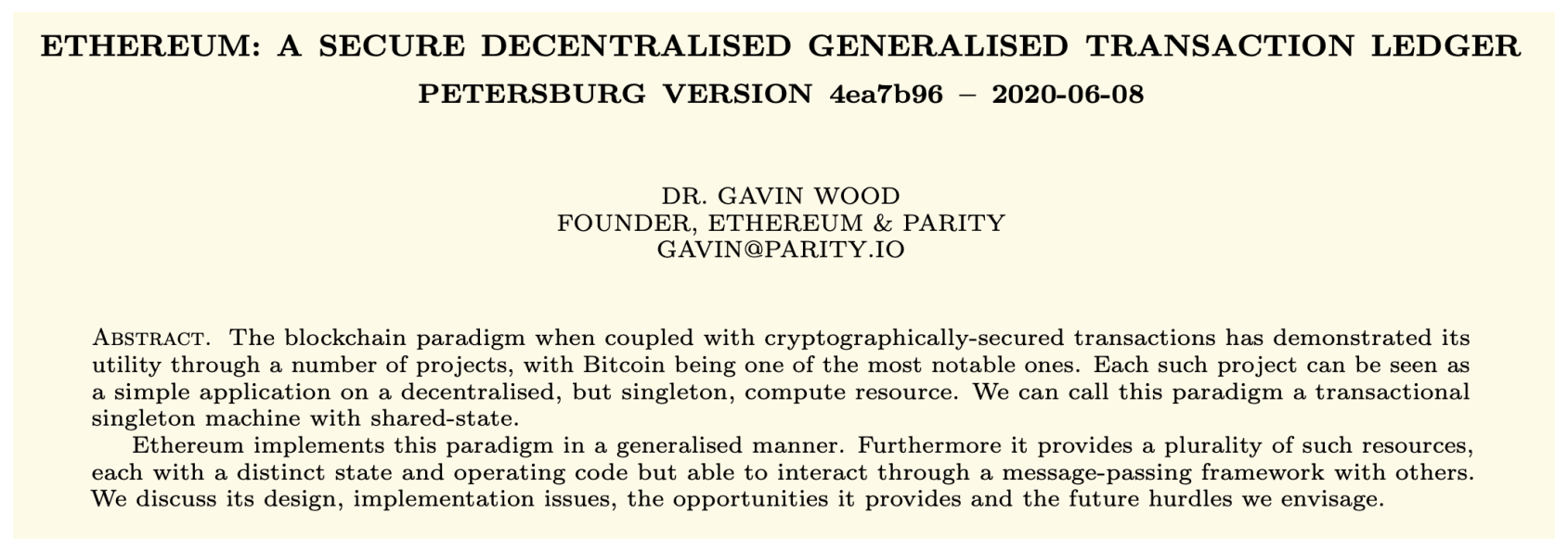
Objetivos

- Apresentação de uma Visão Geral sobre rede **Ethereum**, componentes do Ecossistema *Ethereum*, a __Ethereum Virtual Machine (EVM)_ e Contratos Nativos. Além disso, uma perspectiva do usuário é apresentada, mostrando a estrutura dos blocos do *blockchain* da *Ethereum*, *Wallets* e *softwares* clientes, nós e mineradores, ferramentas e `APIs`, protocolos e Linguagens de Programação Suportados.

Ethereum – Overview i

- Vitalik Buterin (https://vitalik.ca) conceitualizou Ethereum (https://ethereum.org) em Novembro de 2013.
- A ideia central proposta foi o desenvolvimento de uma linguagem **Turing-completa** para permitir o desenvolvimento de programas arbitrários (contratos inteligentes) para *blockchain* e Aplicações Descentralizados (DApps).
- Este conceito difere do *Bitcoin*, onde a linguagem de **script** é limitada e permite apenas as operações necessárias.

Ethereum – Overview ii



ETHEREUM: A SECURE DECENTRALISED GENERALISED TRANSACTION LEDGER

PETERSBURG VERSION 4ea7b96 – 2020-06-08

DR. GAVIN WOOD
FOUNDER, ETHEREUM & PARITY
GAVIN@PARITY.IO

ABSTRACT. The blockchain paradigm when coupled with cryptographically-secured transactions has demonstrated its utility through a number of projects, with Bitcoin being one of the most notable ones. Each such project can be seen as a simple application on a decentralised, but singleton, compute resource. We can call this paradigm a transactional singleton machine with shared-state.

Ethereum implements this paradigm in a generalised manner. Furthermore it provides a plurality of such resources, each with a distinct state and operating code but able to interact through a message-passing framework with others. We discuss its design, implementation issues, the opportunities it provides and the future hurdles we envisage.

Figura 1: O *Ethereum Yellow Paper*[1]

Ethereum – Overview iii

- O *Ethereum Yellow Paper* foi escrito por Dr. Gavin Wood, o fundador do *Ethereum* e da Parity (http://gavwood.com), e serve como uma especificação formal do protocolo da *Ethereum*.
- Qualquer pessoa pode implementar um cliente Ethereum seguindo as especificações de protocolo definidas no artigo.

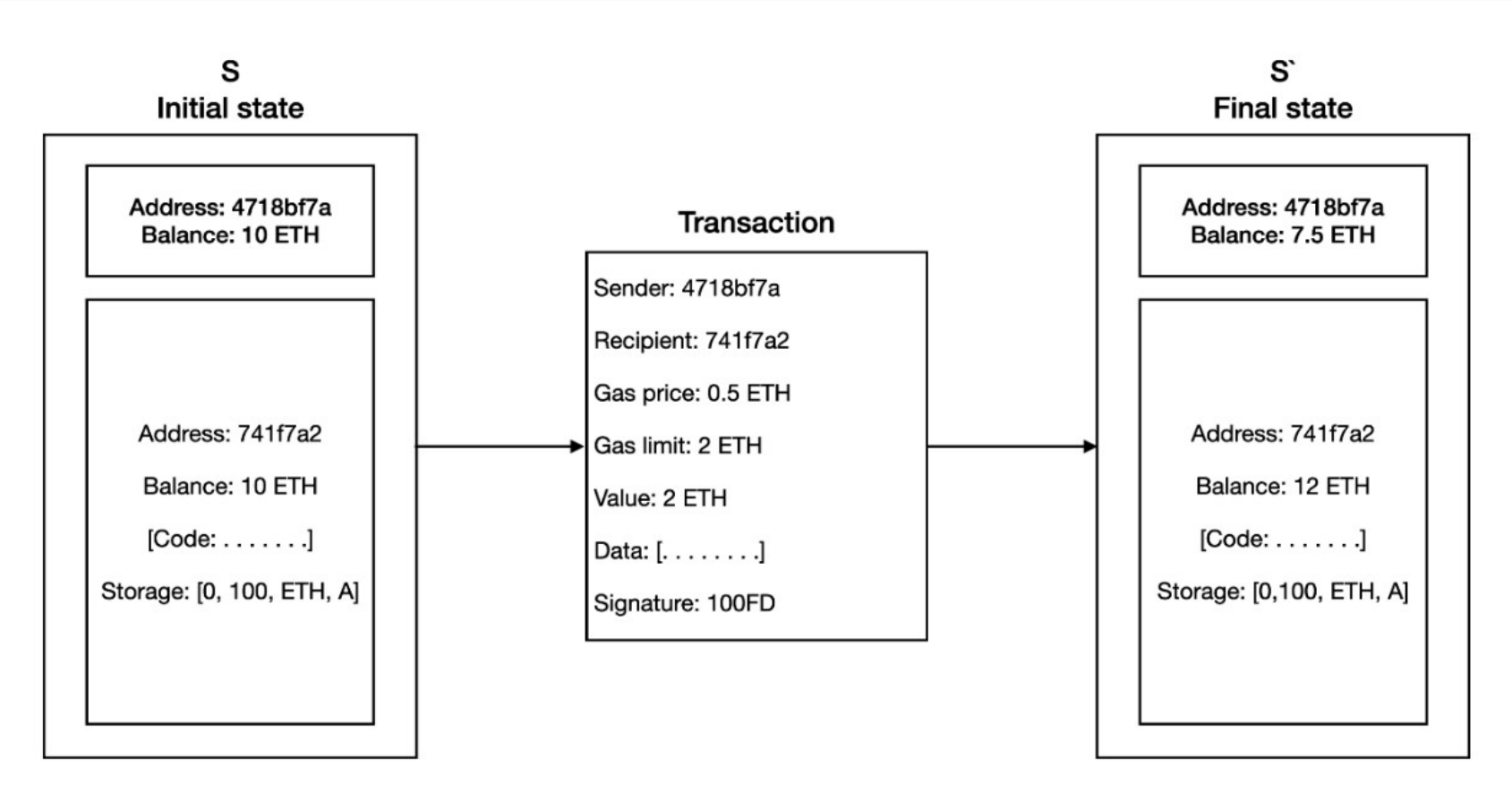
[1] *Ethereum Yellow Paper* - https://ethereum.github.io/yellowpaper/paper.pdf

Ethereum Releases i

- A primeira versão da *Ethereum*, denominada **Olympic**, foi liberada em Maio de 2015. Dois meses mais tarde, a versão chamada de **Frontier** foi liberada em Julho. Outra versão, a **Homestead** com várias melhorias foi liberada em Março de 2016. A release chamada de **Muir Glacier**, que atrasou a **difficulty bomb** (https://eips.ethereum.org/EIPS/eip-2384). Um grande lançamento antes disso foi Istambul, que incluiu mudanças em torno de privacidade e dimensionamento capacidades.
- Uma lista de todas as *releases* anunciadas é mantida em https://github.com/ethereum/go-ethereum/releases.

A Blockchain Ethereum i

- O Ethereum, assim como qualquer outro *blockchain*, pode ser visualizado como uma máquina de estado baseada em transações.



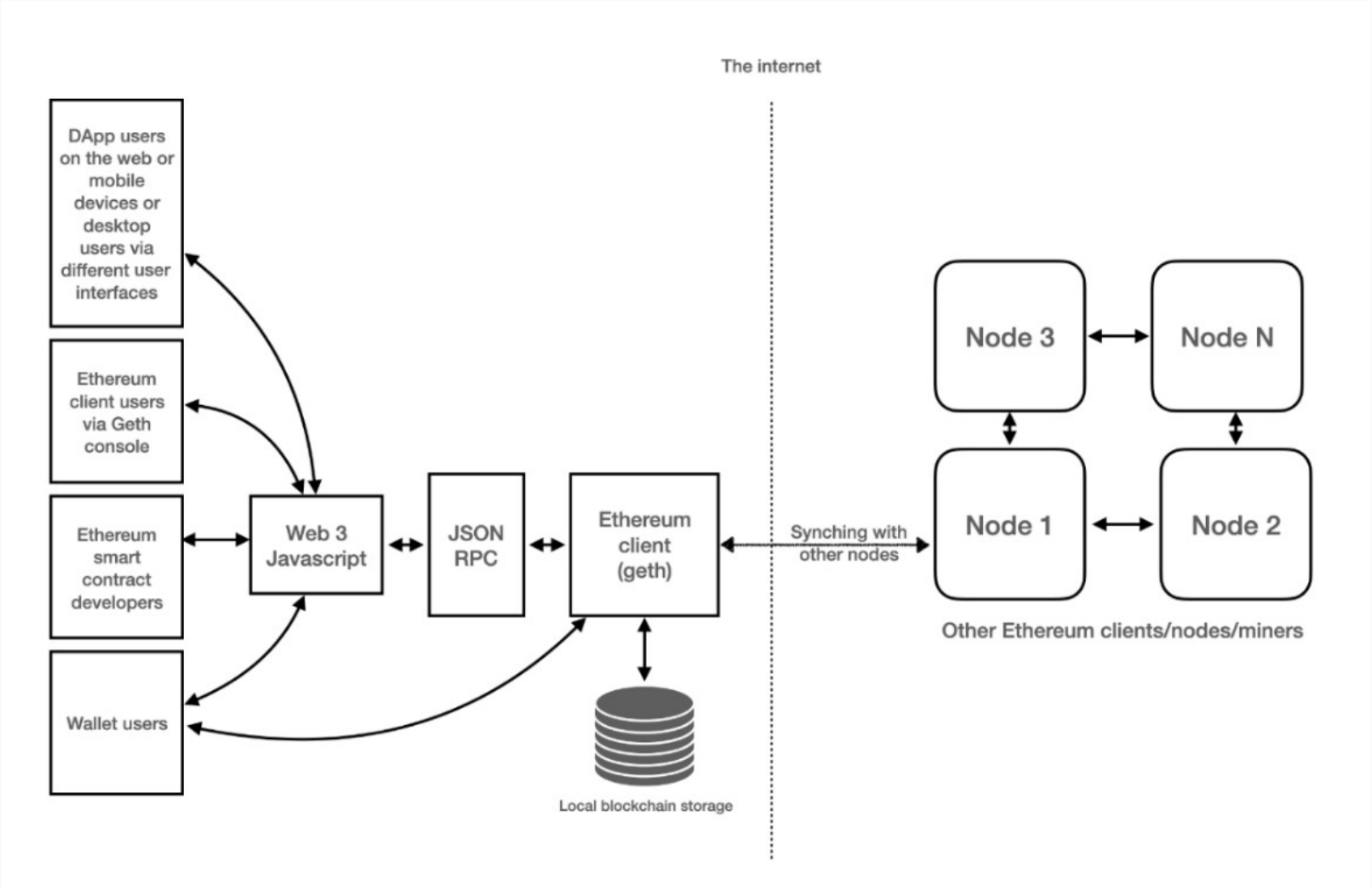
A Blockchain Ethereum ii

- A ideia principal da *blockchain* da *Ethereum*, um estado gênese é transformado em um estado final executando transações de forma incremental. A transformação final é então aceita como a versão absoluta e indiscutível do estado. A função de transição de estado *Ethereum* é mostrada, onde a execução de uma transação resultou em uma transição de estado.

Ethereum – a user's perspective i

- O caso de uso mais comum do rede *Ethereum* é o envio e o recebimento de pagamentos.
- Para isso, o usuário assina a transação e a envia, que se propaga na rede, momento em que os mineradores a pegam, verificam e iniciam a Prova de Trabalho (PoW).
- Se PoW for bem sucedida, o bloco com a transação é finalizado e propagado, e um novo bloco é adicionado à cadeia
- Para enviar e receber transações, um software de carteira é usado: por exemplo, carteiras são usadas em dispositivos móveis.

Arquitetura de Alto Nível da Ethereum i



Rede Ethereum i

A rede *Ethereum* é uma rede *peer-to-peer* onde os nós participantes mantem a *blockchain* e contribuem para o mecanismo de consenso. As redes podem ser divididas em três tipos, com base nos requisitos e uso.

A mainnet

A **mainnet** é a atual rede *Ethereum*. Seu ID de rede é 1 e seu ID de cadeia (*chain*) é também 1. Os IDs de rede e de cadeia são usados para identificar a rede. Um explorador de blocos que mostra informações detalhadas sobre blocos e outras métricas relevantes estão disponíveis em https://etherscan.io, que pode ser usado para explorar a blockchain Ethereum.

Rede Ethereum ii

Testnets

Existem um número de redes de testes (testnets) disponíveis para *Ethereum*. Elas tem como objetivo fornecer um ambiente de testes para contratos inteligentes e DApps antes de serem implantados para produção na rede *blockchain*. Além disso, sendo redes de teste, elas permitem experimentos e pesquisa. A principal testnet é chamada `Ropsten`, que contém todas as características de outras redes de propósito especial menores que foram criados para fins específicos. Por exemplo, outras redes de teste incluem `Kovan` e `Rinkeby`, que foram desenvolvidos para testar as versões do **Byzantium**. As mudanças que foram implementados nessas redes de teste menores também foram implementados em `Ropsten`. Agora a rede de teste `Ropsten` contém todas as propriedades de `Kovan` e `Rinkeby`.

Rede Ethereum iii

Redes Privadas

As *private nets* são redes privadas que podem ser criadas gerando-se um novo *genesis block*. Este é geralmente o caso em redes *blockchain* privadas, onde um grupo privado de entidades iniciam sua rede *blockchain* e a usam como uma blockchain autorizada ou de consórcio.

Elementos do Ecossistema Ethereum i

- Chaves e Endereços
- Contas
- Transações e mensagens
- Criptomoeda/Tokens Ether
- A Ethereum Virtual Machine (EVM)
- Smart contracts e contratos nativos.

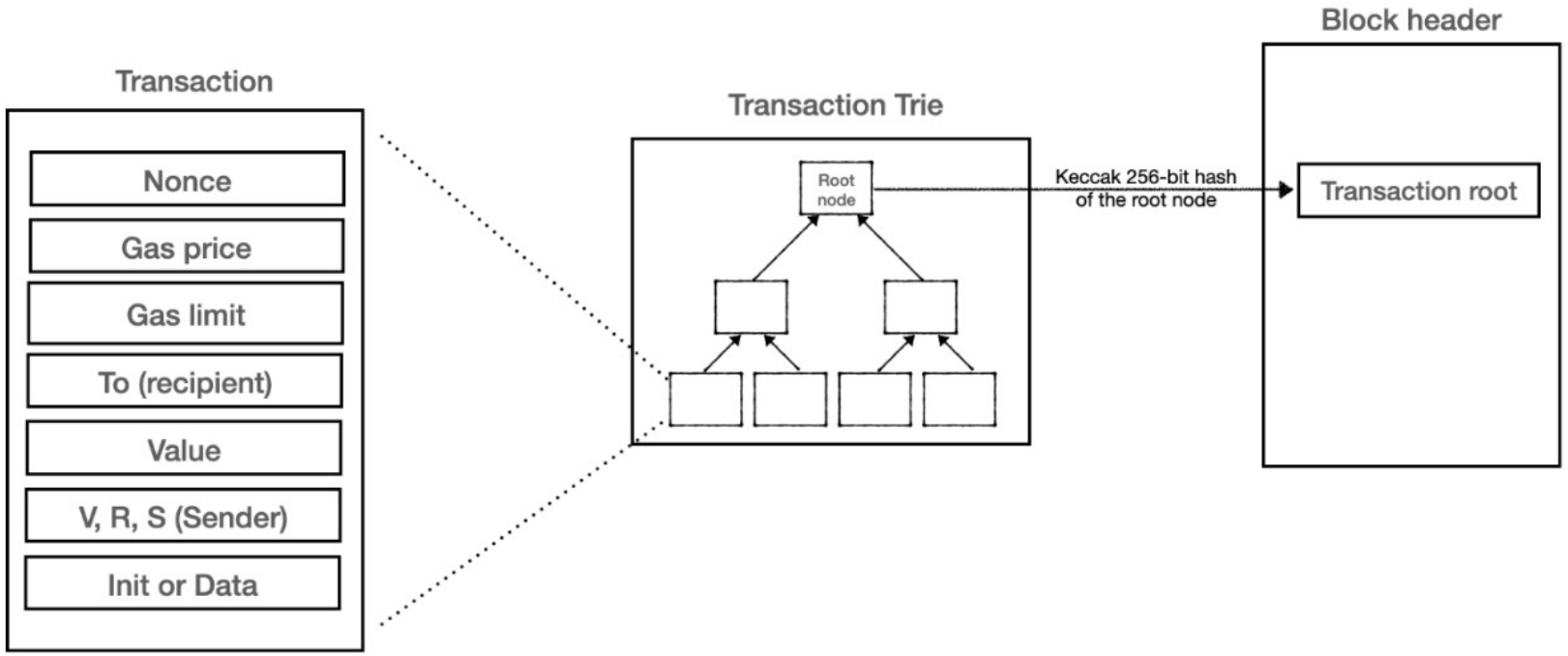
Tipos de contas i

- **EOAs:** *Externally Owned Accounts*. Contas de usuários representadas por um endereço.
- **CAs:** *Contract accounts*. Criadas como resultado do *deployment* de um contrato inteligente, também representado por um endereço.

Transações e Árvore de Transações (trie) i

Transações

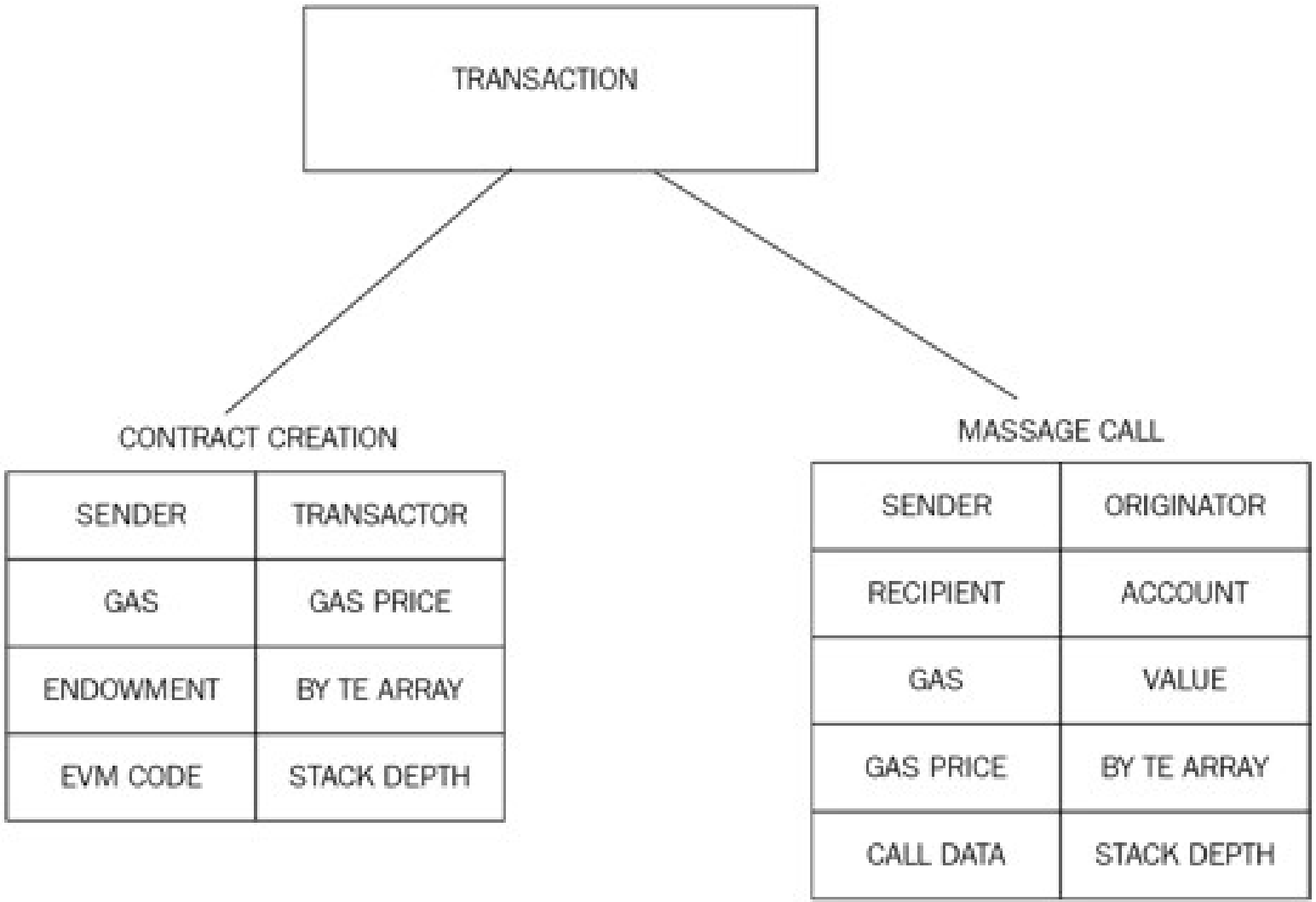
Uma transação no *Ethereum* consiste em vários campos, como mostrado aqui, junto com a *transaction trie*. O diagrama também mostra a relação entre a tentativa de transação e o cabeçalho do bloco.



Tipos de Transações i

- Existem três tipo de transações:
  - Criação de Contrato
  - *Call*
  - Transferência de Valor

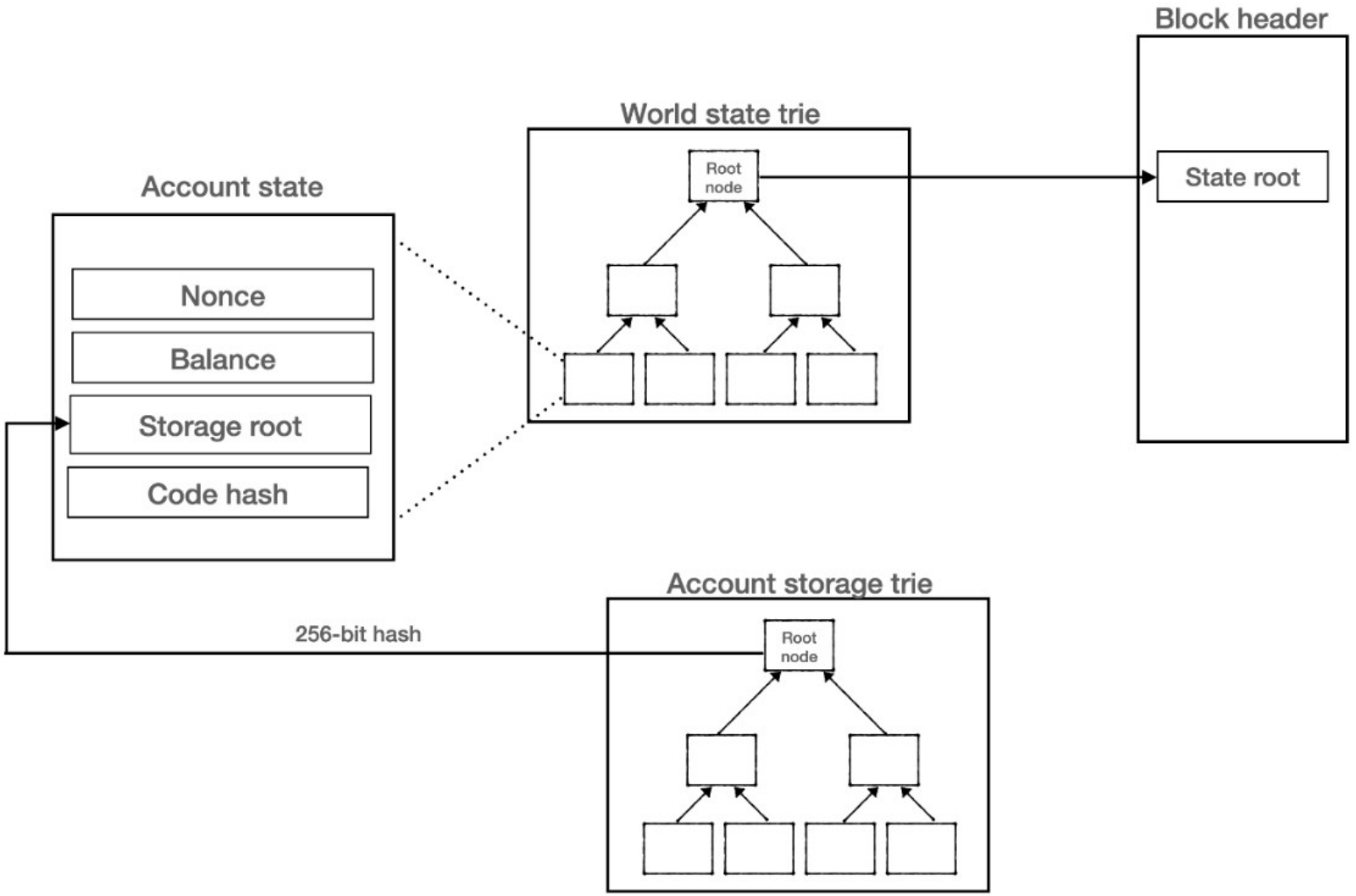
Tipos de Transações ii



Estado da conta e armazenamento na trie i

O diagrama mostra os campos contidos no estado da conta e como os vários elementos estão contidos no *world state* trie: * World state trie * State root * Account state * Account storage trie

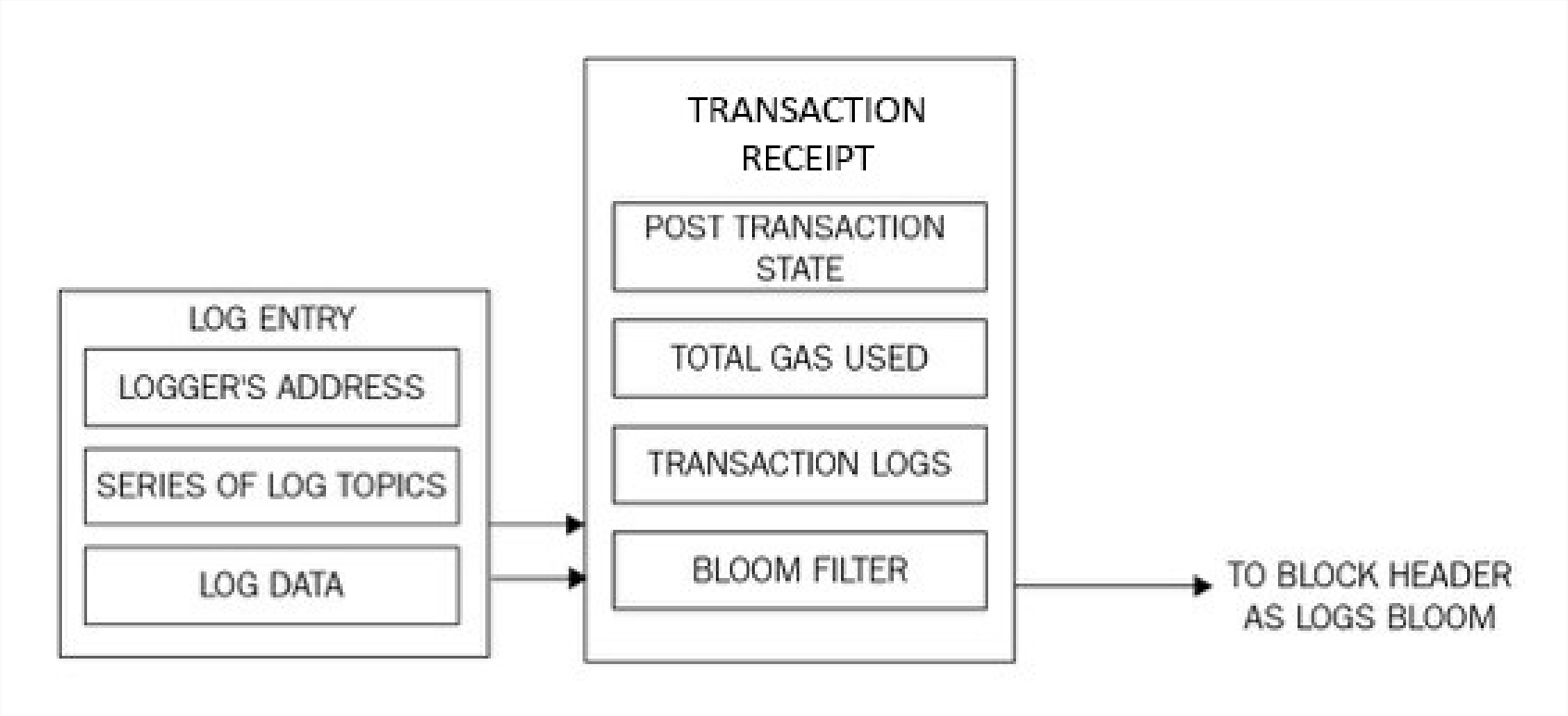
Estado da conta e armazenamento na trie ii



Recibos de Transações i

- Recibos de Transações (transaction receipts) são gerados como resultado da execução de transações.
- Logs também são atualizados em conformidade.
- Ambas as estruturas de dados contêm vários campos, conforme mostrado abaixo:

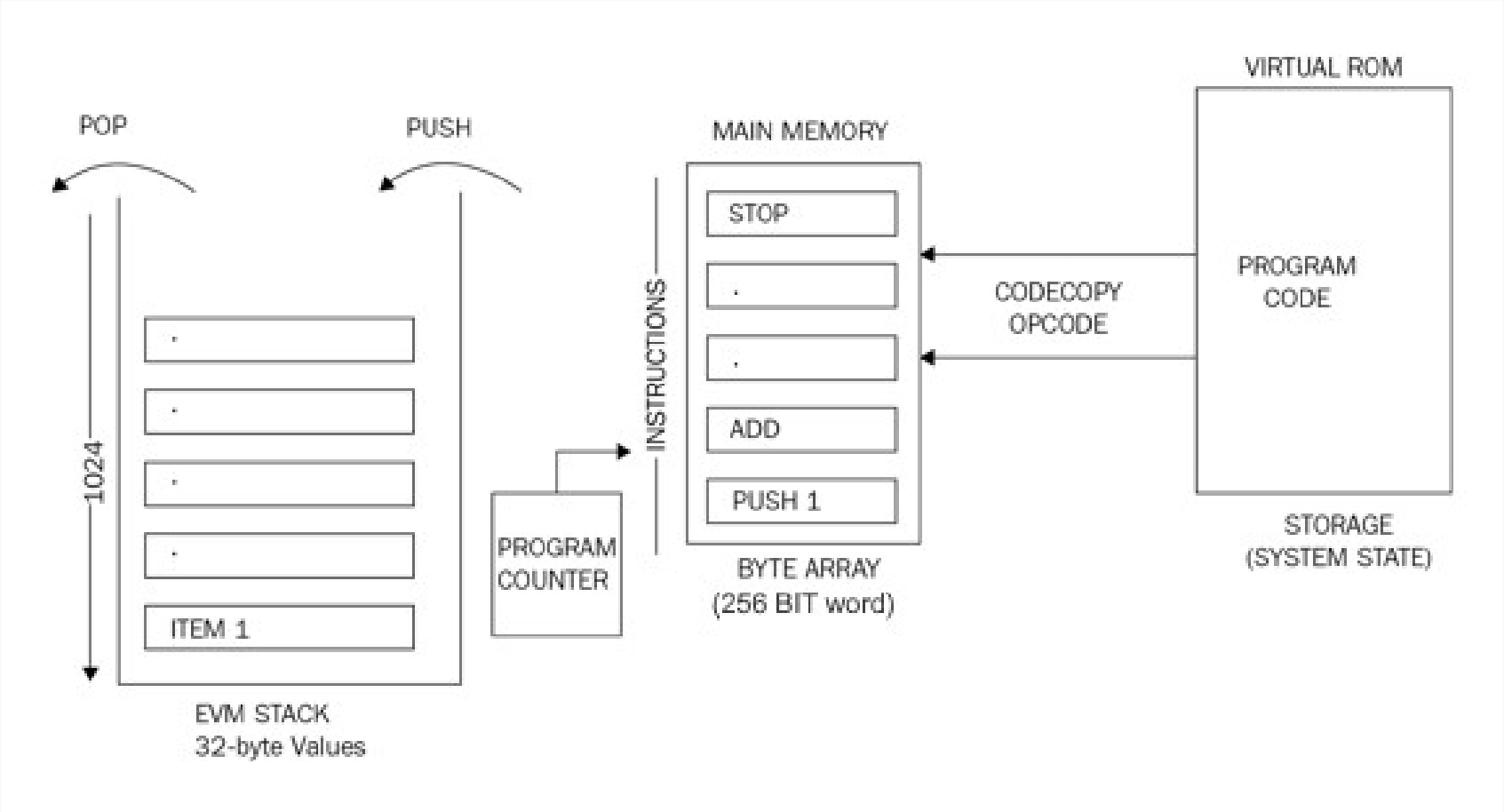
Recibos de Transações ii



The Ethereum Virtual Environment (EVM) i

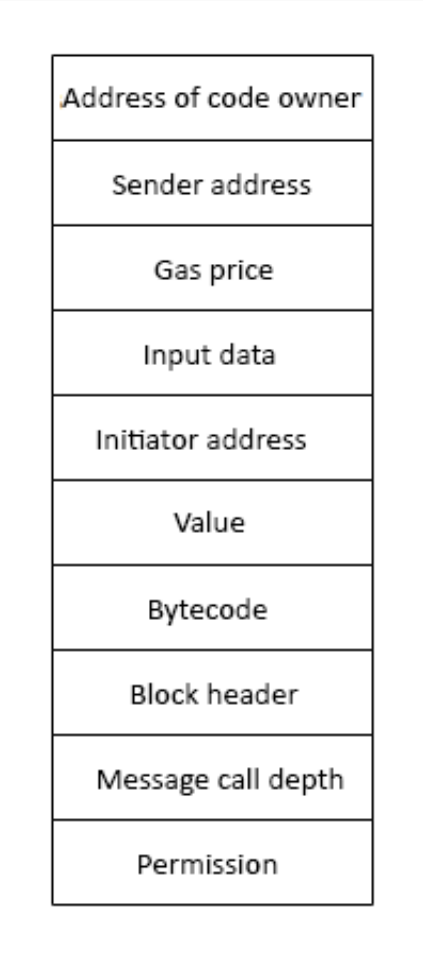
- Stack size based on LIFO queue: Last In, First Out.
- 1024 stack depth limit
- Turing complete but limited by gas, making it quasi-Turing complete
- Big-endian design
- Storage available to EVM
  - Memory
  - Storage
  - Stack

EVM operation design i



Execution environment i

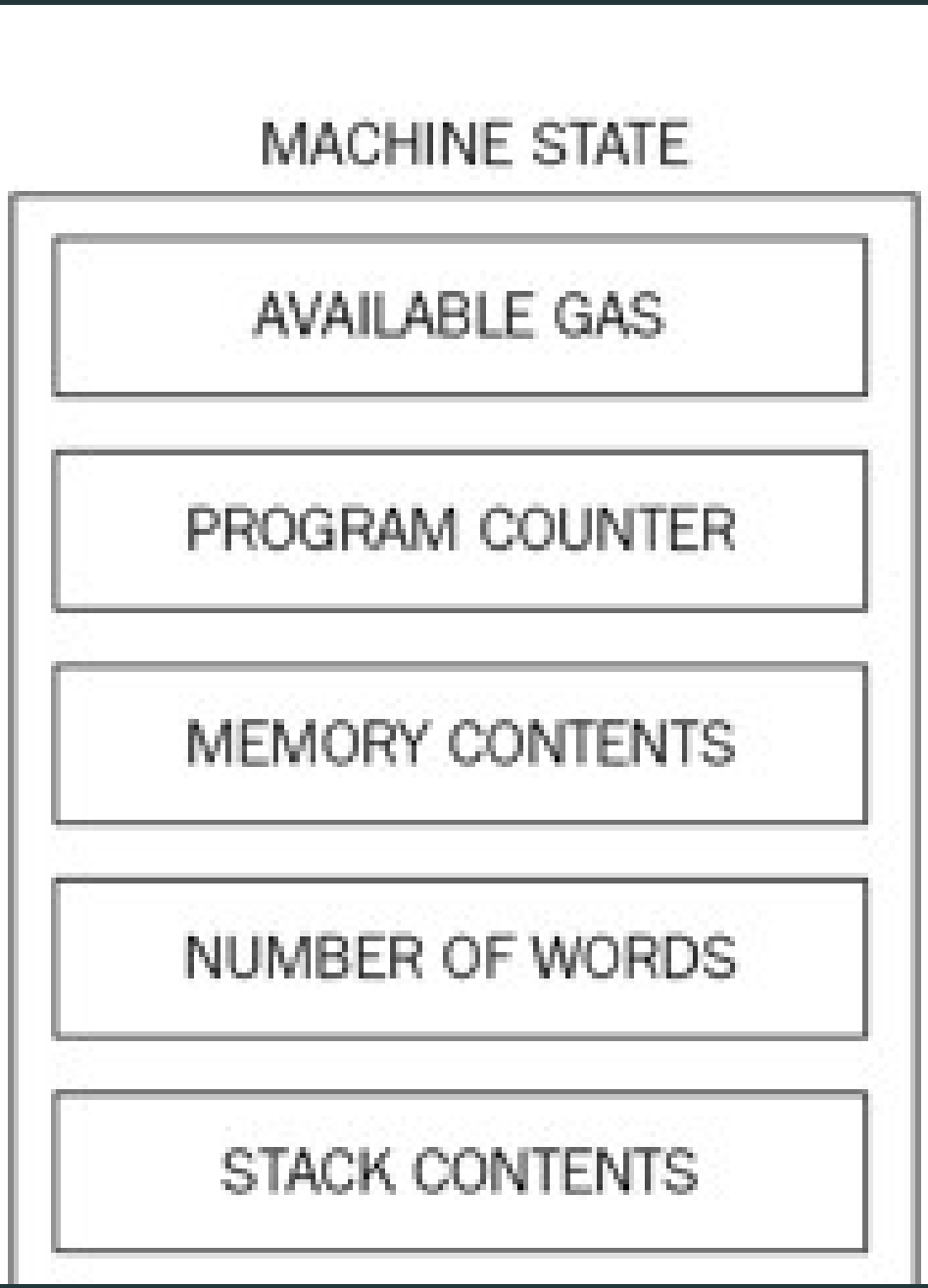
O ambiente de execução do *Ethereum* consiste em vários elementos, como mostrado:



Machine State i

Uma Máquina de Estado ou *Machine state* é uma tupla compreendendo vários campos, como mostrado em:

Machine State ii



Contratos Nativos i

Existem nove contratos pré-compilados ou contratos nativos na versão Ethereum Istanbul: * *The elliptic curve public key recovery function * The SHA-256-bit hash function * The RIPEMD-160-bit hash function * The identity/datacopy function * Big mod exponentiation function * Elliptic curve point addition function * Elliptic curve scalar multiplication * Elliptic curve pairing * Blake2 compression function 'F'*

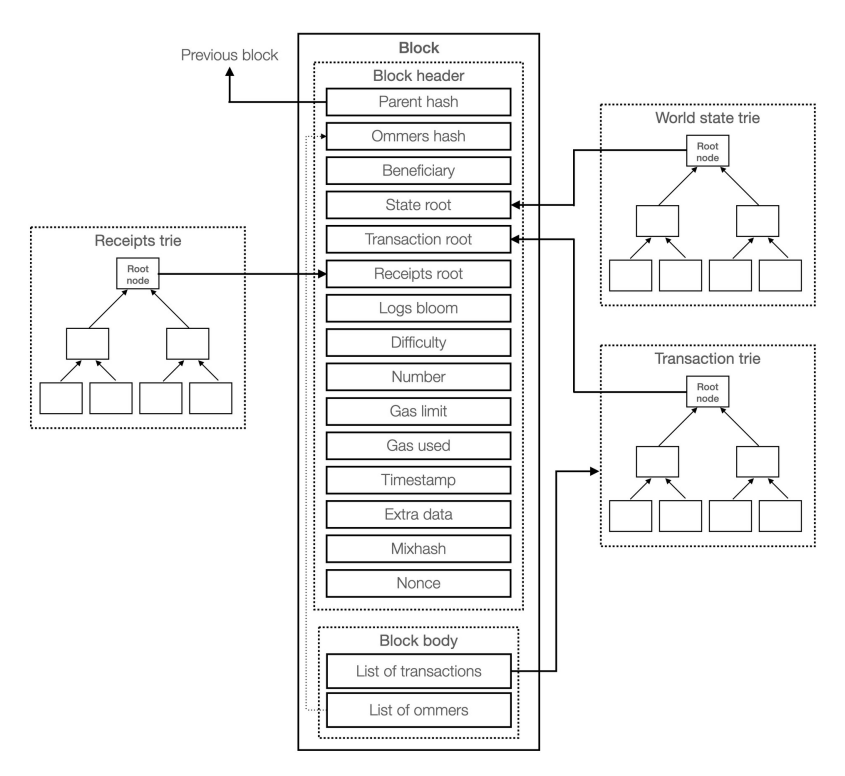
Um pouco mais de Ethereum

Outline i

- Blocos e *Blockchain*
- Wallets e Software Clientes
- Nós e Mineradores
- APIs e ferramentas
- Protocolos suportados
- Linguagens de Programação

Blocks e Blockchain i

Um bloco *Ethereum* consiste em vários campos, conforme diagrama. *State root*, *transaction root* e *receipts root* são *root hashes* de suas respectivas árvores.



Mecanismo de Validação de Blocos i

The Ethereum block validation mechanism checks the following conditions: * If it is consistent with uncles and transactions. This means that all ommers satisfy the property that they are indeed uncles, and also if the Proof of Work (PoW) for uncles is valid. * If the previous block (parent) exists and is valid. * If the timestamp of the block is valid. This means that the current block's timestamp must be higher than the parent block's timestamp. Also, it should be less than 15 minutes into the future. All block times are calculated in epoch time (Unix time).

If any of these checks fails, the block will be rejected. A list of errors for which the block can be rejected is presented here: * The timestamp is older than the parent * There are too many, or duplicate uncles
* The uncle is an ancestor, or the uncle's parent is not an ancestor * There is non-positive difficulty * There is an invalid mix digest or PoW

Finalização de Bloco i

Block finalization is a process that is run by miners to validate the contents of the block and apply rewards. It results in four steps being executed. These steps are described here:

1. **Ommers (uncles) validation.** In the case of mining, determine ommers. The validation process of the headers of stale blocks checks whether the header is valid and whether the relationship between the uncle and the current block satisfies the maximum depth of six blocks. A block can contain a maximum of two uncles.

Finalização de Bloco ii

2. **Transaction validation.** In the case of mining, determine transactions. This process involves checking whether the total gas used in the block is equal to the final gas consumption after the final transaction, in other words, the cumulative gas used by the transactions included in the block.

Finalização de Bloco iii

3. **Reward application.** Apply rewards, which means updating the beneficiary's account with a reward balance. In Ethereum, a reward is also given to miners for stale blocks, which is 1/32 of the block reward. Uncles that are included in the blocks also receive 7/8 of the total block reward. The current block reward is 2 ether. It was reduced first from 5 ether to 3 with the Byzantium release of Ethereum. Later, in the Constantinople release (https://blog.ethereum.org/2019/02/22/ethereum-constantinople-st--petersburg-upgrade-announcement/), it was reduced further to 2 ether. A block can have a maximum of two uncles.

4. **State and nonce validation.** Verify the state and block nonce. In the case of mining, compute a valid state and block nonce.

Mecanismo de Block difficulty i

O mecanismo de dificuldade do bloco é representado pela fórmula abaixo, que garante que os blocos sejam produzidos a uma taxa constante:

$$block\_diff = parent\_diff + parent\_diff//'2048 * \ max(1 - (block\_timestamp - parent\_timestamp)//10, -99) + \ int(2 * *((block.number//100000) - 2))$$

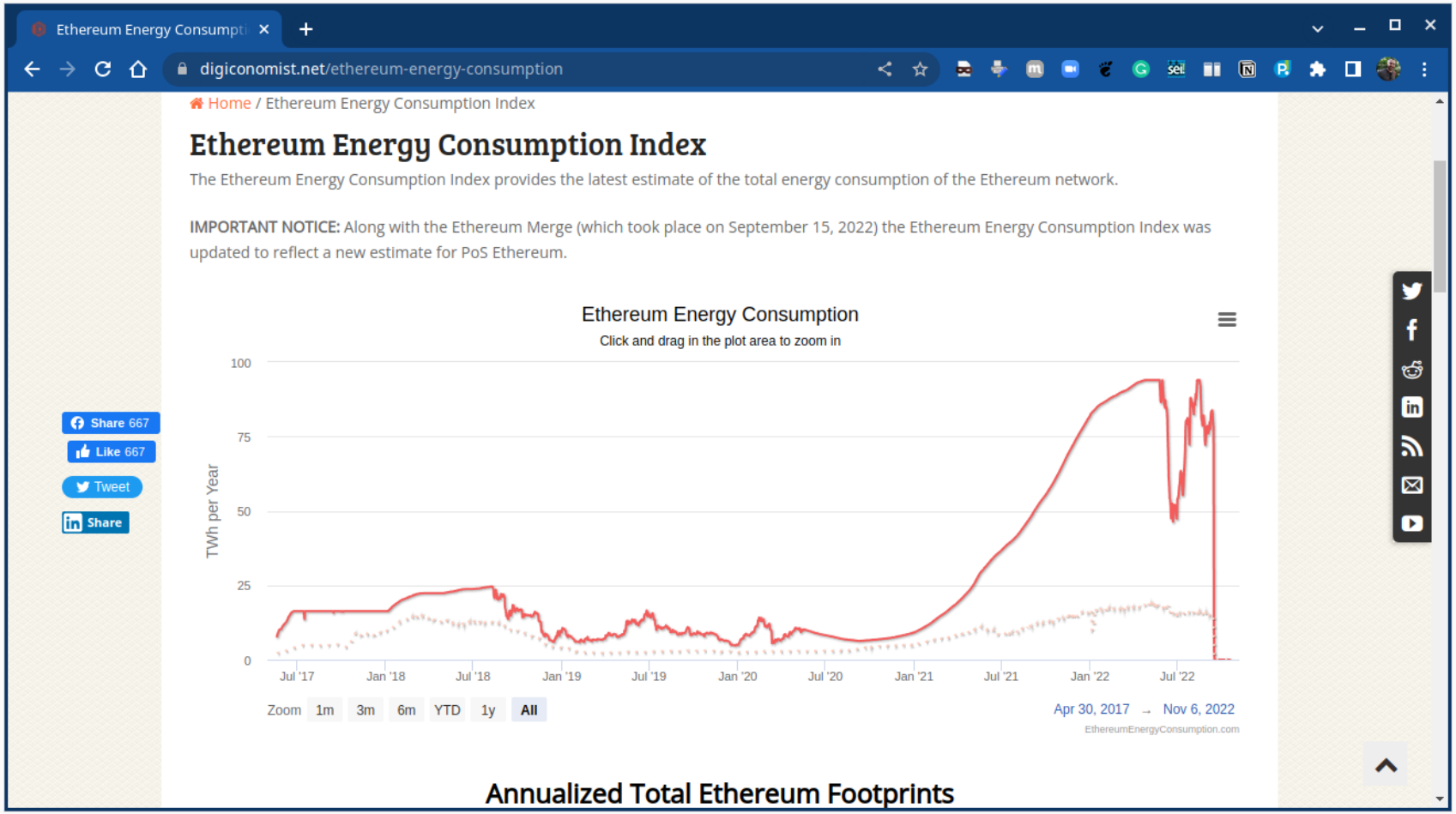
O *gas cost* de uma transação pode ser calculado usando esta fórmula:

$$Totalcost = gasUsed * gasPrice$$

Consumo de Energia i

- Com a última atualização `Merge` que trocaram a *Proof of Work* (PoW) pela *Proof of Stake* (PoS) tendo como uma das motivações a questão ambiental. Houve um grande impacto no consumo de energia.

Consumo de Energia ii



Wallets e Software Clientes i

- Wallets
- Light clients
- Existem três tipos de sincronização de clientes:
  - **Full:** Nesse modo de sincronização, o cliente `Geth` faz um *download* completo da *blockchain* para o nó local. Isso significa que ele obtém todos os cabeçalhos e corpos dos blocos e valida todas as transações e blocos desde o bloco *genesis*. No início de 2020, o tamanho do *blockchain* Ethereum era de aproximadamente $210GB$, e baixar e manter isso pode ser um problema.
  - Hoje, 18 de outubro de 2022 o tamanho chega a $966.06GB$, segundo https://ycharts.com/indicators/ethereum_chain_full_sync_data_size.

Wallets e Software Clientes ii

- **Fast:** Neste modo é feito o *download* completo, mas somente recupera e verifica somente os 64 blocos anteriores ao bloco corrente. Depois disso, ele verifica os novos blocos na íntegra. Não reproduz e verifica todas as transações históricas desde o bloco *genesis*, em vez disso, ele só faz os *downloads* de estado. Isso também reduz significativamente o tamanho do disco do banco de dados *blockchain*. Este é o modo padrão de sincronização do cliente `Geth`.
- **Light:** Este é o modo mais rápido e apenas baixa e armazena o estado atual. Nesse modo, o cliente não baixa nenhum bloco histórico e processa apenas os blocos mais novos.

Ethereum keystore: Key decryption process i

Os arquivos de chave pública e privada são armazenados no diretório keystore. Os arquivos de keystore Ethereum são arquivos JSON, cujo conteúdo se parece com o script abaixo.

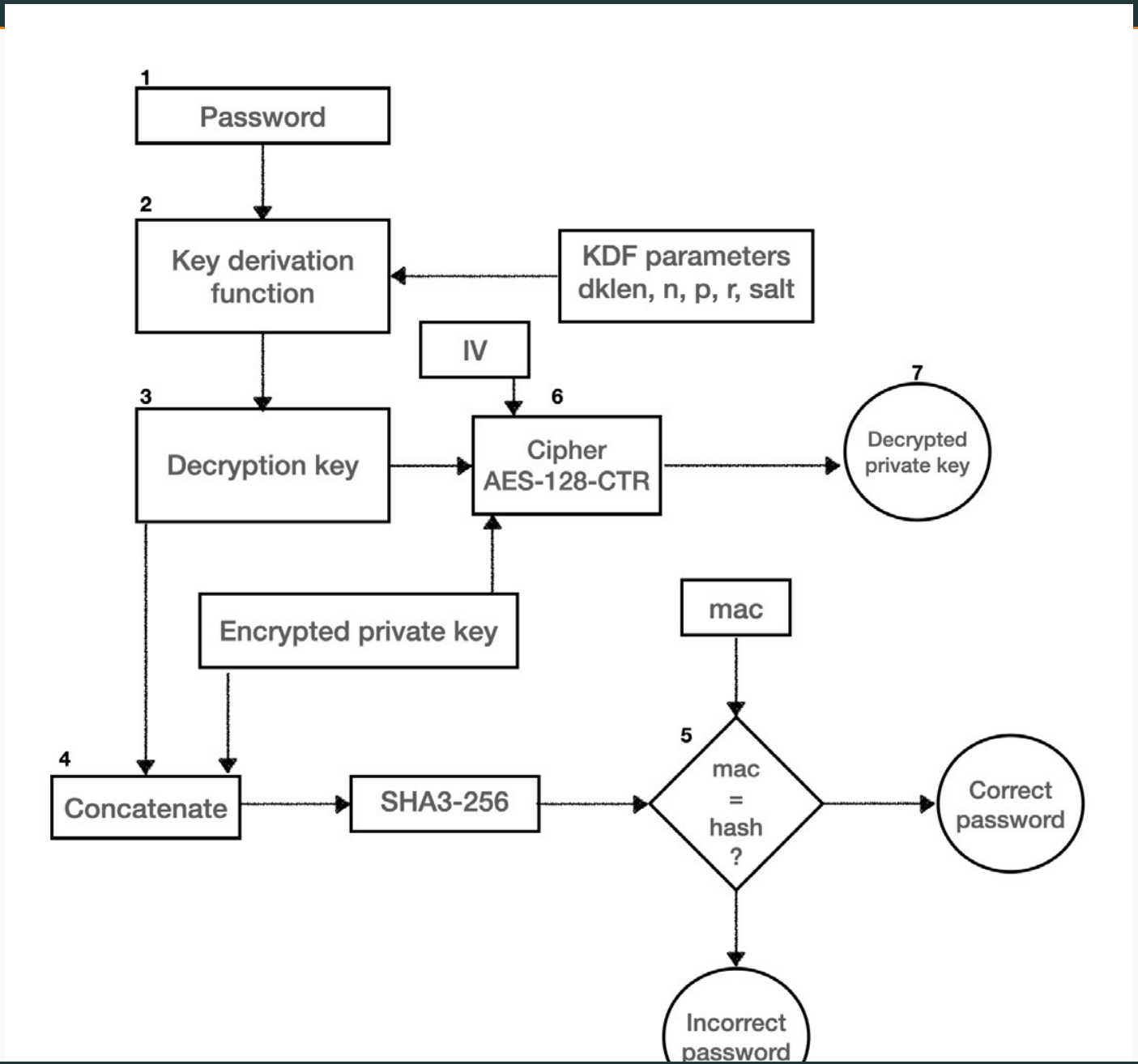
Ethereum keystore: Key decryption process ii


```
{
"address":"ba94fb1f306e4d53587fcdcd7eab8109a2e183c4",
"crypto":
{
    "cipher":"aes-128-ctr",
    "ciphertext":
        "b2ab4f94f5f44ce98e61d99641cd28eb00fd794129be25beb8a5fae89ef93241",
    "cipherparams":
        {"iv":"a0fdfe0a6d314a62ba6a370f438faa57"},
    "kdf":"scrypt",
    "kdfparams":
        {"dklen":32,"n":262144,"p":1,"r":8,
"salt":"be3e99203c24ffcb71a6be2823fc7a211c8cc10d66bc6b448fef420fa0669068"},
"mac":"1b0a42d4bf7a8e96d308179e9714718e902727ead7041b97a646ef1c9d6f9ad7"},
"id":"7e0772e0-965e-4a05-ad93-fc5d11245ba3",
"version":3
}
```

Ethereum keystore: Key decryption process iii

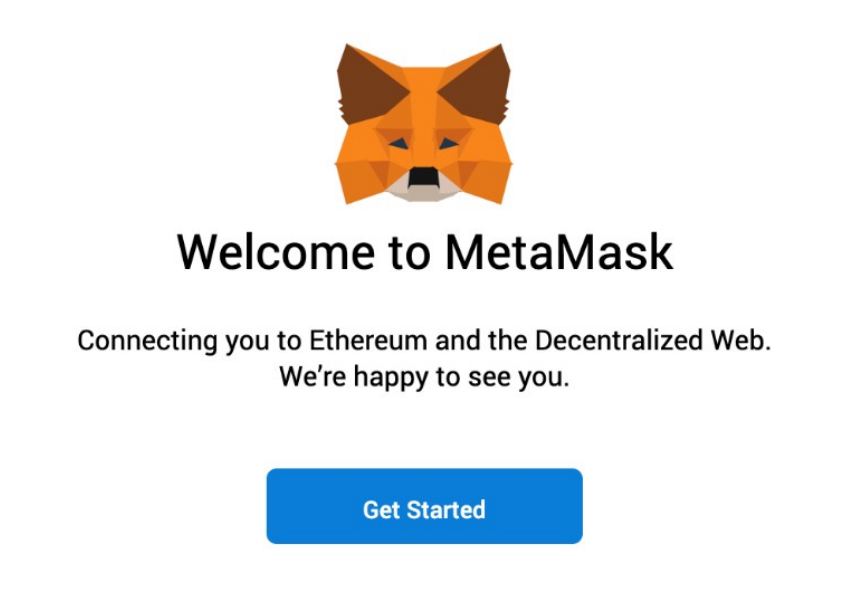
Key decryption process:

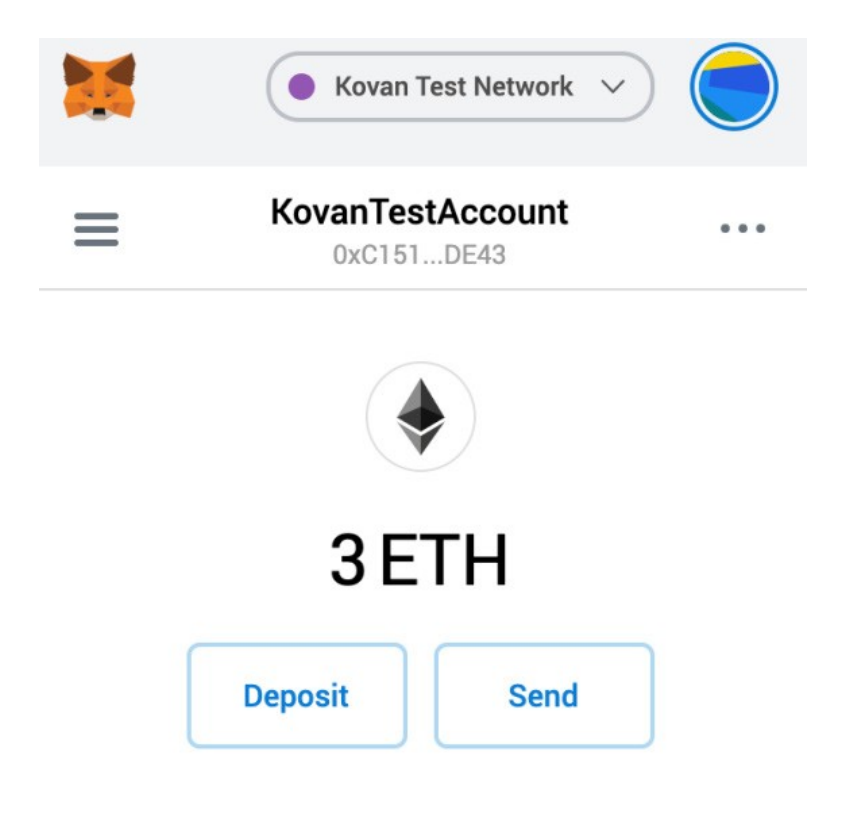
Ethereum keystore: Key decryption process iv



MetaMask i

`MetaMask` é uma carteira de criptomoedas e uma interface para redes *blockchain* sem exigir a instalação de um nó local.





Nós e Mineradores i

A mineração é o processo pelo qual novos blocos são selecionados por meio de um mecanismo de consenso e adicionados ao *blockchain*. O processo segue os seguintes passos: * It listens for the transactions broadcasted on the Ethereum network and determines the transactions to be processed. * It determines stale ommer blocks and includes them in the blockchain. * It updates the account balance with the reward earned from successfully mining the block. * Finally, a valid state is computed and the block is finalized, which defines the result of all state transitions.

Ethash i

`Ethash` é o nome do algoritmo de `Proof of Work` usado no `Ethereum`.

1. First, the header from the previous block and a 32-bit random nonce is combined using Keccak-256. 2. This produces a 128-bit structure called `mix`. 3. `mix` determines which data is to be picked up from the DAG. 4. Once the data is fetched from the DAG, it is "mixed" with the mix to produce the next mix, which is then again used to fetch data from the DAG and subsequently mixed. This process is repeated 64 times. 5. Eventually, the 64th mix is run through a digest function to produce a 32-byte sequence. 6. This sequence is compared with the difficulty target. If it is less than the difficulty target, the nonce is valid, and the PoW is solved. As a result, the block is mined. If not, then the algorithm repeats with a new nonce.

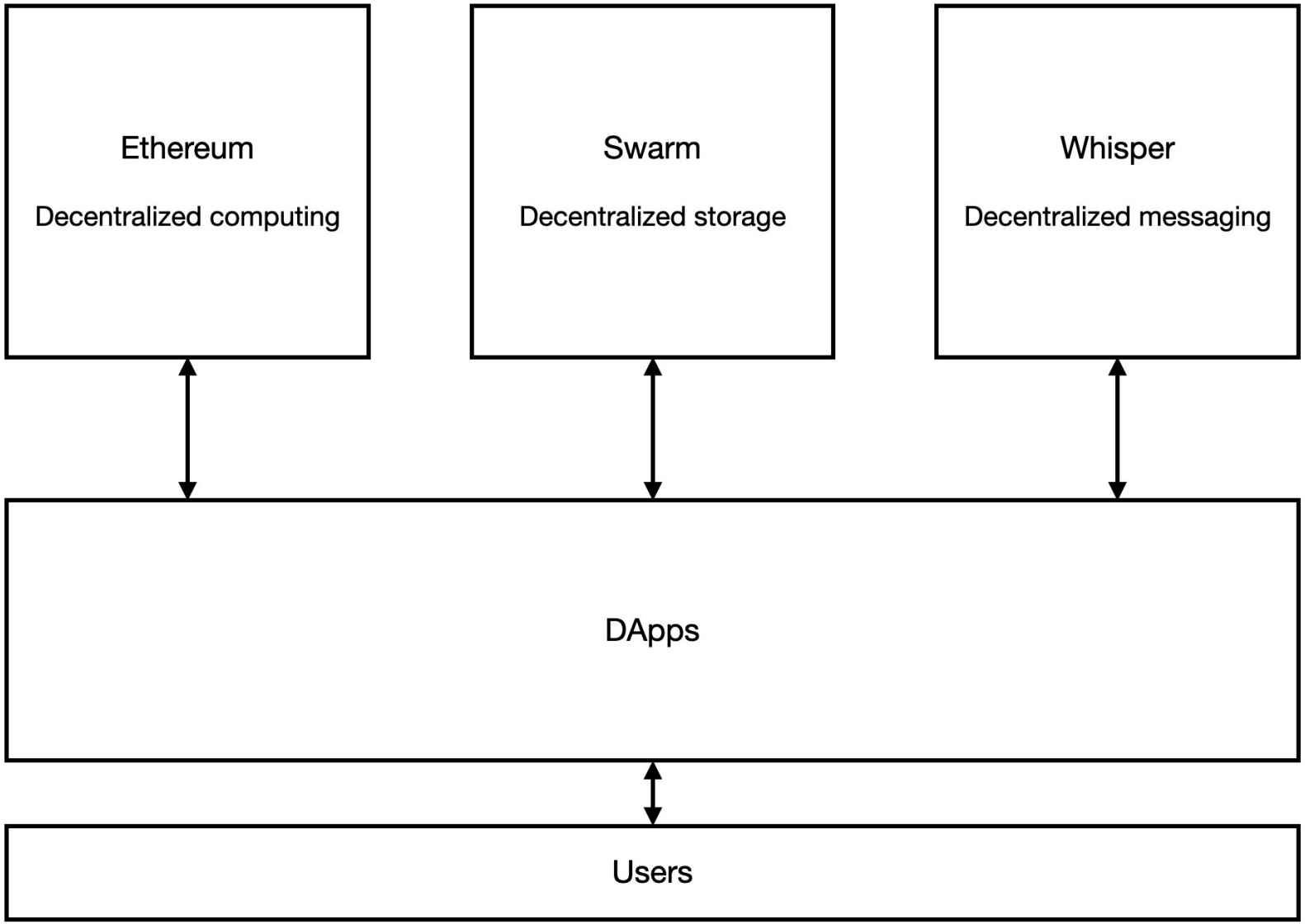
Mineração i

- **CPU:** Mining using a computer's built in CPU.
- **GPU:** Mining using graphical processing units, which provide better performance as compared to CPUs.
- **ASICs:** Specialized hardware—Application-Specific Integrated Circuit designed solely to run mining algorithms. These are currently the most efficient mining tools.
- Mining pools: In mining pools, resources are shared between miners and rewards are split accordingly.

Protocolos Suportados i

Dois principais protocolos de suporte, `Swarm` e `Whisper`, são usados para fornecer armazenamento e mensagens descentralizadas, a fim de criar um ecossistema descentralizado completo.

Protocolos Suportados ii



Linguagens de Programação i

- **Solidity** is one of the high-level languages that has been developed for Ethereum. It uses JavaScript-like syntax to write code for smart contracts. Once the code is written, it is compiled into bytecode that's understandable by the EVM using the Solidity compiler called solc.
- **LLL** is another language that is used to write smart contract code.
- **Serpent** is a Python-like, high-level language that can be used to write smart contracts for Ethereum.
- **Vyper** is a newer language that has been developed from scratch to achieve a secure, simple, and auditable language.

Atividade

- Leitura do `Ethereum yellow paper`:
  https://ethereum.github.io/yellowpaper/paper.pdf
- Instalação do `Geth` e executar ele com a `Kovan testnet`.

Word Cloud

Leitura Recomendada

Leitura Recomendada

Capítulo 11: Ethereum 101

Livro: IMRAN BASHIR. Mastering Blockchain : Distributed Ledger Technology, Decentralization, and Smart Contracts Explained, 2nd Edition.

Capítulo 12: Futher Ethereum

Livro: IMRAN BASHIR. Mastering Blockchain : Distributed Ledger Technology, Decentralization, and Smart Contracts Explained, 2nd Edition.

Próximas Aulas

Próximas Aulas

- Ambientes de Desenvolvimento e Ferramentas.

Referências

Referências i

Imran, Bashir. 2018. *Mastering Blockchain : Distributed Ledger Technology, Decentralization, and Smart Contracts Explained, 2nd Edition.* Packt Publishing. https://search.ebscohost.com/login.aspx?direct=true&db=e000xww&AN=1789486&lang=pt-br&site=eds-live&scope=site.