

Introduction à la Blockchain Ethereum



M2SRIV / Jean-Patrick GELAS
Université Claude Bernard – Lyon 1
2018-2019

Agenda

Nous ne parlerons pas ...

- Algorithmes de consensus (PoW, PoS, PoA, DPoS,...)
- Plateformes d'échanges (Binance, Kraken,...)
- Comment miner de la crypto monnaie
- Comment devenir **crypto millionaire ! :-)**

2

Comparatif énergétique

	Yearly Cost	Energy Used (GJ)
Gold Mining	\$105B	475M
Gold Recycling	\$40B	25M
Paper Currency and Minting	\$28B	39M
Banking System	\$1,870B	2,340M
Governments	\$27,600B	5,861M
Bitcoin Mining	\$4.5B	183M

<https://blog.picks.co/pow-is-efficient-aa3d442754d3>

3

Blockchain : Rappel

- Structure de données simple
- La technologie Blockchain : « Base de données » sécurisées et décentralisées.



Démo : <https://anders.com/blockchain/>

4

Les mineurs

- Héberge une copie de la blockchain
- Ajoutent de nouvelles liste de transactions (i.e. des blocs) à la chaîne.
- Vérifient l'intégrité de la blockchain
- Génèrent de nouveaux *coins*
- (Exécutent les Smart Contracts)



5

Ethereum : Définition

« Protocole d'échanges décentralisés permettant la création par les utilisateurs de contrats intelligents grâce à un langage Turing-complet. »

- Wikipedia

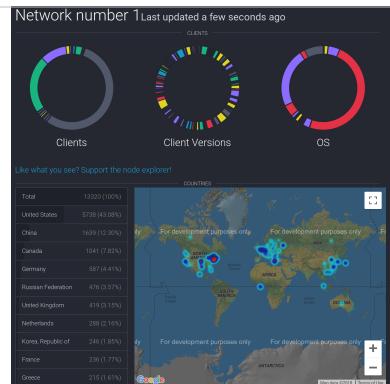
6

Ethereum

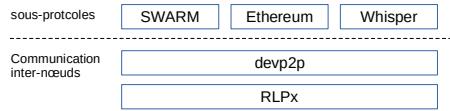
- Blockchain de seconde génération.
- Développée par *Vitalik Buterin*, lancée en juillet 2015.
- Fréquence moyenne des blocs : 14-15 secondes
- Taille des blocs : dynamique
- Symbole boursier : ETH
- Quantité maximale : non limitée

7

<https://www.ethernodes.org>



Architecture Ethereum



- **devp2p**: Abstraction de la couche matériel pour communiquer entre les nœuds.
- **Ethereum** : La blockchain que nous connaissons permettant de stocker la logique des *Dapps* (smart-contract).
- **Swarm** : Un système de fichiers décentralisé proche de IPFS permettant de servir les *Dapps*.
- **Whisper** : Un protocole de communication décentralisé pour communiquer entre les *Dapps*.

8

Smart Contract

- Programme autonome
- Déployé et répliqué
- Non modifiable
- Adapté pour gérer des transactions



11

Infrastructure Ethereum

9000 nœuds contre 7000 pour Bitcoin (mars 2019).



Source: https://twitter.com/peter_szilagyi/status/887272506914213888 - 18/07/2017

9

Smart Contract

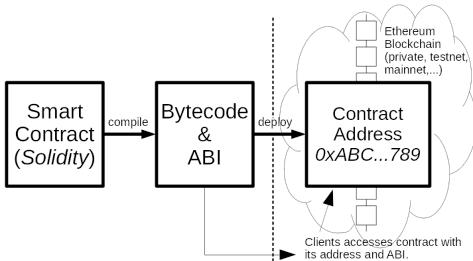
- Du bytecode stocké dans la blockchain
- Rédigé dans un langage de haut niveau : *Solidity*
- Compilé (solc)
- Accessible via une adresse codée sur 160 bits
- Exécuté dans l'Ethereum Virtual Machine (EVM)

0x71c7656ec7ab88b098defb751b7401b5f6d8976f



12

Cycle de vie du Smart Contract



13

Clients Ethereum

- Coté client on a un *provider* qui se connecte à un nœud en particulier et qui communique avec l'EVM avec des requêtes au format JSON-RPC.
- Les clients Ethereum proposent une API avec un ensemble de méthodes JSON-RPC.

• <https://github.com/ethereum/wiki/wiki/JSON-RPC#json-rpc-methods>
curl -X POST -d '{"jsonrpc":"2.0","method":"web3_clientVersion","params":[],"id":67}' <http://127.0.0.1:8545>
{"id":67,"jsonrpc":"2.0","result":"EthereumJS TestRPC/v2.3.3|ethereumjs"}

16

Code machine

```
"opcodes": "PUSH1 0x80 PUSH1 0x40 MSTORE  
CALLVALUE DUP1 ISZERO PUSH2 0x10 JUMP1 PUSH1 0x0  
DUP1 REVERT JUMPDEST POP PUSH1 0x40 MLOAD PUSH1  
0x20 DUP1 PUSH2 0x487 DUP4 CODECOPY DUP2 ADD  
PUSH1 0x40 SWAP1 DUP2 MSTORE SWAP1 MLOAD PUSH1..."  
  
"object":  
"608060405234801561001057600080fd5b5060405160  
208061048783398101604090815290516000805460016  
0a060020a0319163317808255600160a060020a031681  
52600160208190529290209190915560ff81166100606..."
```

14

Communiquer avec Ethereum

- Envoyez des requêtes JSON-RPC avec son propre provider n'est pas forcément simple à mettre en place et à manipuler.
- Un des projets les plus utilisés sur Ethereum propose d'englober ce provider et ces requêtes avec une API JavaScript (existe aussi dans d'autres langages). Il s'agit de *Web3*.
- Une alternative populaire à Web3 est *ethers.js*

17

Smart Contract

- Le *bytecode* est déployé sur Ethereum dans une transaction (dans le champ *data* de la transaction).
- Une fois sur la blockchain le contrat ne peut être ni modifié ni supprimé.
- Il est accessible sur tous les noeuds à jour.
- Pour interagir avec le contrat il faut une machine virtuelle qui interprète le *bytecode*.
- L'EVM (Ethereum Virtual Machine) mise à disposition par les clients Ethereum qui gère un noeuds (ex : Geth, Parity, Cpp-Ethereum,...)

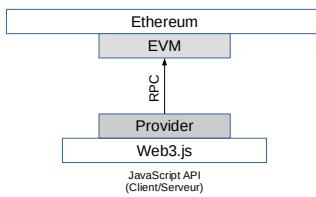
15

Web3.js / Ethers.js

- Ces librairies proposent une librairie JavaScript simple d'utilisation permettant d'interagir avec un noeud.
- La première chose à faire d'ailleurs lorsque l'on utilise Web3 est de lui donner un *provider* avec l'adresse du noeud à contacter.
- Idéalement le noeud auquel se connecte Web3 tourne sur la machine de l'utilisateur.
- Mais de plus en plus les utilisateurs se connectent à des noeuds distants grâce à des services comme Infura (<https://infura.io/>)

18

Web3 pour interagir avec la blockchain



19

Le langage Solidity



- Langage de haut niveau
- Influencé par C++, Python et Javascript.
- Typé statiquement.
- Supporte l'héritage,
- l'appel à des bibliothèques,
- la définition de type complexe par les utilisateurs.

22

Dapps

- Application accessible de façon décentralisée (en principe).
 - Pas de serveur centralisé
 - ni pour gérer une base de données,
 - ni pour servir l'application,
 - ni pour communiquer entre plusieurs applications.
- Le projet Ethereum propose une solution pour chacun de ces points
- *Swarm, Whisper* ne sont pas encore suffisamment matures...
- Une Dapp se "limite" donc à la connexion entre une application communiquant avec des smart-contracts sans s'appuyer sur un serveur pour gérer une base de données centralisée.

20

Hello World

```

pragma solidity ^0.4.18;

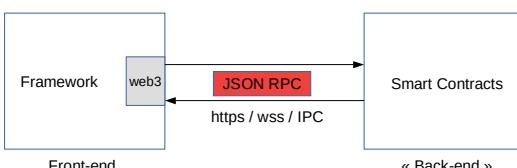
contract Hello {
    string message = "Default message";

    function getMessage () public view returns (string) {
        return message;
    }
    function setMessage (string _message) public payable {
        message = _message;
    }
}

```

23

Dapp



21

Simple Token

```

pragma solidity ^0.4.20;

contract MyToken {
    /* This creates an array with all balances */
    mapping (address => uint256) public balanceOf;

    /* Initializes contract with initial supply tokens to the creator of the contract */
    function MyToken(
        uint256 initialSupply
    ) public {
        balanceOf[msg.sender] = initialSupply;                         // Give the creator all initial tokens
    }

    /* Send coins */
    function transfer(address _to, uint256 _value) public returns (bool success) {
        require(balanceOf[msg.sender] >= _value);                      // Check if the sender has enough
        require(balanceOf[_to] + _value >= balanceOf[_to]);           // Check for overflows
        balanceOf[msg.sender] -= _value;                                // Subtract from the sender
        balanceOf[_to] += _value;                                         // Add the same to the recipient
        return true;
    }
}

```

24

Nexium (NXC) – Token ERC-20

```
contract Nexium {  
    ...  
    function Nexium() {  
        initialSupply = 100000000000;  
        balanceOf[msg.sender] = initialSupply;  
        name = "Nexium";  
        symbol = "NxC";  
        decimals = 3;  
        burnAddress = 0x1b32000000000000000000000000000000000000000000000000000000000000  
    }  
    function totalSupply() returns(uint){  
        return initialSupply - balanceOf[burnAddress];  
    }  
    function transfer(address _to, uint256 _value) ...  
    function transferFrom(address _from, ...  
}
```

25

Ethereum Virtual Machine

- Machine (*quasi-*) Turing complète
- Environnement d'exécution des Smart Contracts
- Émule une machine 256 bits avec des pseudo-registres
- Registres émulés par une *stack* virtuelle.

28

0x (ZRX) – Token ERC-20

```
contract ZRXTOKEN is UnlimitedAllowanceToken {  
  
    uint8 constant public decimals = 18;  
    uint public totalSupply = 10**27; // 1 billion tokens  
    string constant public name = "0x Protocol Token";  
    string constant public symbol = "ZRX";  
  
    function ZRXTOKEN() {  
        balances[msg.sender] = totalSupply;  
        ...  
    }  
}
```

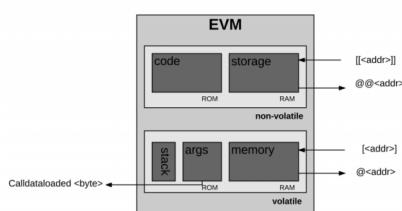
26

Ethereum et unités de mesure

- **Ether (ETH)** : le nom de la crypto monnaie
- **Wei** : une fraction d'Ether (1 ETH = 10^{18} Wei)
- **GAS** : unité de mesure en terme de quantité de calcul
- **GAS price** : défini le prix (en GWei) que vous êtes prêt à payer au mineur.
- **GAS limit** : Quantité maximum de GAS que vous êtes prêt à payer pour une transaction.

29

Ethereum Virtual Machine



27

Analogie

- GAS limit : capacité du réservoir d'une voiture en litre.
- GAS price le prix du litre de carburant.
 - Voiture : 1,50 EUR (prix) par litre (unité)
 - Ethereum : 20 GWei (prix) par GAS (unité)
- Pour remplir le réservoir il faut :
 - 50 litres à 1,50 EUR = 75 EUR
 - 21000 unités de GAS à 20 GWei = 0.00042 ETH

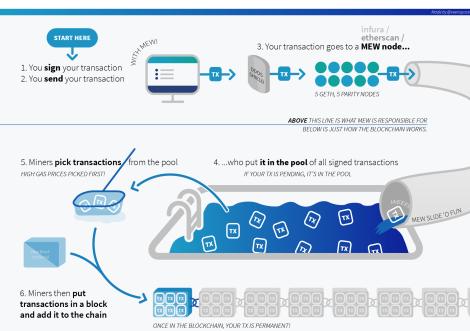
30

Remarques

- Fixer un *GAS limit* évite de dépenser une fortune en cas de problème.
- La quantité de *GAS* requise est définie par le coût et la quantité d'instructions exécutées.
- Fixer un *GAS limit* trop petit a peu d'intérêt.

31

MyEtherWallet Behind-The-Scenes

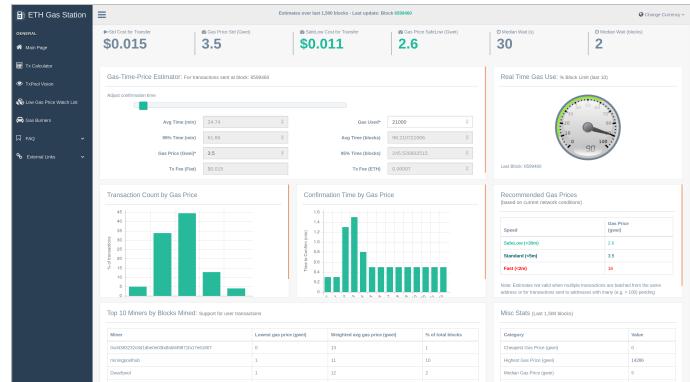


34

Coût des instructions

Value	Mnemonic	Gas Used	Subset	Removed from stack	Added to stack	Notes
0x00	STOP	0	zero	0	0	Halts execution.
0x01	ADD	3	verrylow	2	1	Addition operation
0x02	MUL	5	low	2	1	Multiplication operation.
0x03	SUB	3	verrylow	2	1	Subtraction operation.
0x04	DIV	5	low	2	1	Integer division operation.
0x51	MILOAD	3	verrylow	1	1	Load word from memory.
0x52	MISTORE	3	verrylow	2	0	Save word to memory
0x53	MSTORE8	3	verrylow	2	0	Save byte to memory.
0x54	SLOAD	200		1	1	Load word from storage
0x55	SSTORE	((value != 0) && (storage_location == 0)) ? 20000 : 5000		1	1	Save word to storage.

32

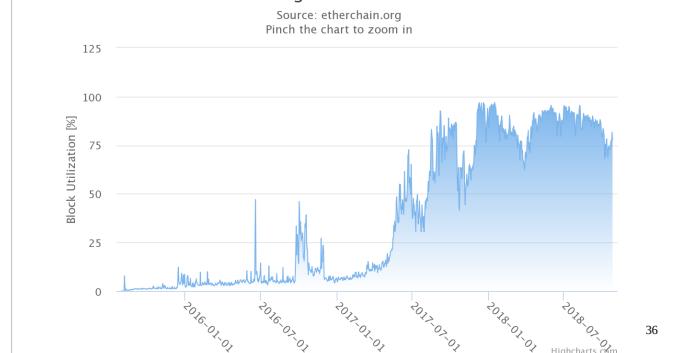


Coût d'une transaction

- Coût total d'une transaction = *GAS_price* x *GAS_used*
- Priorité aux transactions avec un *GAS_price* élevé.
- Plus l'utilisateur est prêt à payer, plus vite la transaction sera traitée.

33

Evolution of the average utilization of Ethereum blocks



36

Performances actuelles

- **block time 15 sec, 4 blocks/min**
- 5959 block/day, 2 174 897 block/year
- **Block Gas limit 8 000 000**
- Daily Gas cap 47 668 965 517
- 76 364 avg gas/tx
- 624 236 tx/day cap
- 433 tx/min
- **7 tx/sec**

37

448 000 USD / GBYTES
Temps de transfert : ~45 Heures
(novembre 2018 - ~200 USD/ETH)



40

En résumé : Pour le calcul...

- L'infrastructure Ethereum est un énorme ordinateur Turing complet distribué
- Ultra tolérant aux pannes
- Très mal exploité car tous les nœuds exécutent les mêmes instructions avec les mêmes données :-(

38

Outils de développement

- Ganache
- Remix
- Truffle

41

En résumé : Pour le stockage...

On est limité par conception à :

- la capacité de stockage des nœuds
- le débit (90 kB / 15 sec => 50 kbytes/s)
- le prix du GAS qui fluctue en fonction de l'Ether
- SSTORE : 20.000 GAS/Word -> 640.000 GAS/KB
- 3,5 GWei / GAS -> 0,00224 ETH/KB
- 200 \$/ETH -> 0,448 \$/KB

39

Ganache

CURRENT BLOCK				SEARCH FOR BLOCK NUMBERS OR TX HASHES			
ACCOUNTS	BLOCKS	TRANSACTIONS	LOGS				
MNEMONIC split worry machine adult rock glow learn common size sting turtle neither	CURRENT BLOCK 20000000000	LAST INDEX 6731975	LAST TX ID 5775	HTTP://127.0.0.1:545	VIRTUAL STATE AUTONOMOUS	HD PATH	R/44'/60'/0/0/account_index
ADDRESS 0x52Bf313DcDf10d477D0e3C336A941B16894d938	BALANCE 96.86 ETH	TX COUNT 1	INDEX 0				
ADDRESS 0xd99A5Ab0014a5a5674729c1E1eE1491E1b4bAD991	BALANCE 103.14 ETH	TX COUNT 0	INDEX 1				
ADDRESS 0x6F2dA64681f2b8820886276c3Aa8f012F57CcEb	BALANCE 100.00 ETH	TX COUNT 0	INDEX 2				
ADDRESS 0xC16bC2D11b1bB7B65E6b6EEA5c3d950900eca84D	BALANCE 100.00 ETH	TX COUNT 0	INDEX 3				

42

Metamask

Visualisation d'une transaction

44

Conclusion

- Miner de l'ether = Sécuriser le réseau = Vérifier les traitements
- Par conception la blockchain Ethereum garantie :
 - l'immutabilité des données,
 - l'exécution et l'accès sans censure possible.
- Analyser un Smart Contract en terme de consommation de GAS pour maîtriser le coût opérationnel
- Maximiser les calculs et le stockage offchain.

47

45

Liens utiles

- <https://hackernoon.com/ether-purchase-power-df40a38c5a2f>
- OPCODE list + GAS : <https://docs.google.com/spreadsheets/d/1m89CvujrQe5LAFjB-YAUConK950uZuMOPMJBXRGCs/edit#gid=0>
- Ethernodes (28/10/2018 -> 13320 nodes) : <https://www.ethernodes.org/network/1>
- <https://www.etherchain.org/charts/averageBlockUtilization>
- <https://etherscan.io/>
- <https://medium.com/coinmonks/storing-on-ethereum-analyzing-the-costs-922d41d6b316>

48