FS18

BLOCKCHAIN, BITCOIN, ETHEREUM IV

Ethereum Smart Contracts II

T. Bocek

thomas.bocek@hsr.ch

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

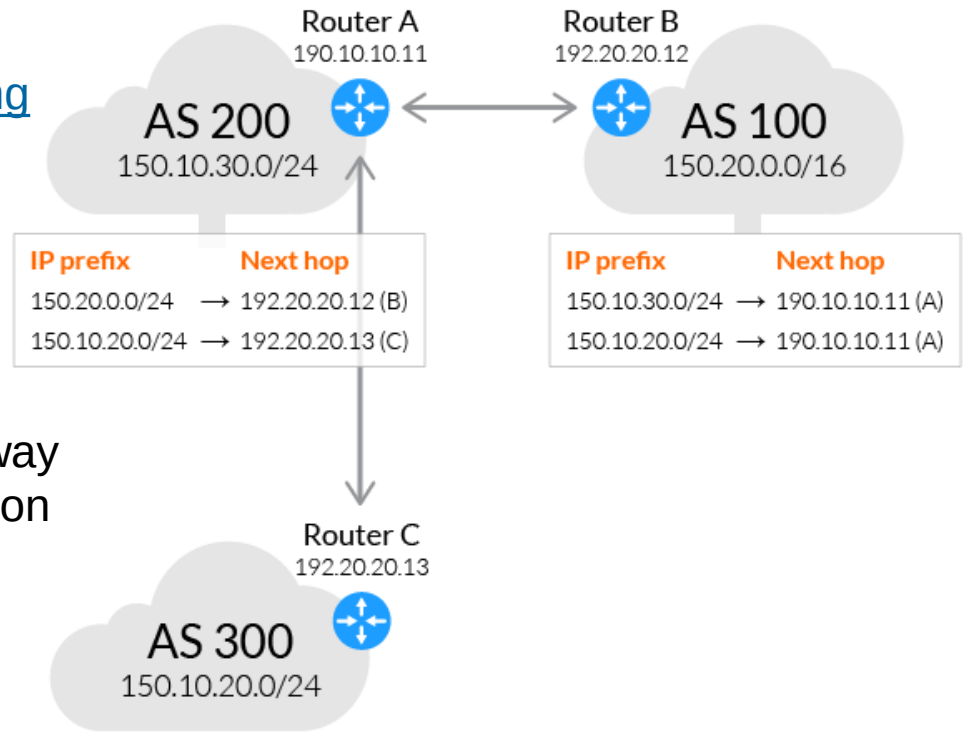
Agenda

- **Distributed Systems in the News**
- **Solidity Contracts**
- **Notary**
- **ERC20**
- **Random Values**

Distributed Systems - In the *News*

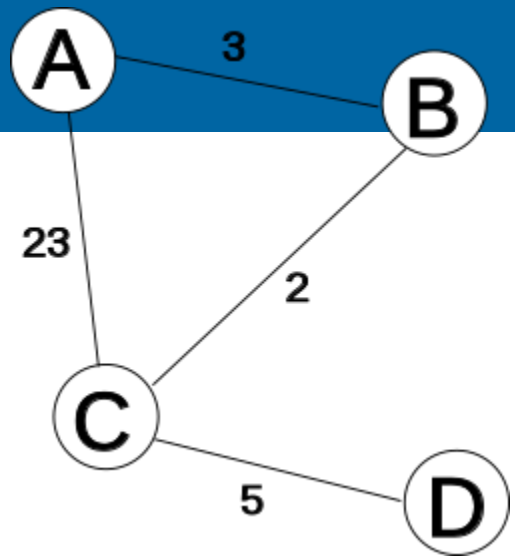
- **24.04.2018: MyEtherWallet Warns That A "Couple" Of Its DNS Servers Have Been Hacked**
  - More than 200 ETH were stolen
  - Another headline: Hackers emptied Ethereum wallets by breaking the basic infrastructure of the internet
  - What happened?

- **Hackers stole cryptocurrencies using a BGP leak**
  - Border Gateway Protocol (BGP) is a standardized exterior gateway protocol designed to exchange routing and reachability information among autonomous systems (AS) on the Internet (HSR)
  - Current version BGP4 used since 1994 (TCP)
  - BGP peer exchanges routing info with neighbor peers
    - Prefix announcements
  - Each peer has a table with all routes – exchanges this information



https://www.incapsula.com/blog/bgp-routing-explained.html

Distributed Systems - In the *News*



- **AS559, maintained by switch, list**
  - 130.60.156.1 – UZH, also maintained by switch
- **~ distance-vector protocol**

T=1

from A	via A	via B	via C	via D
to A				
to B		3		
to C			23	
to D				

from B	via A	via B	via C	via D
to A	3			
to B				
to C			2	
to D				

from C	via A	via B	via C	via D
to A	23			
to B		2		
to C				
to D				5

from D	via A	via B	via C	via D
to A				
to B				
to C			5	
to D				

T=2

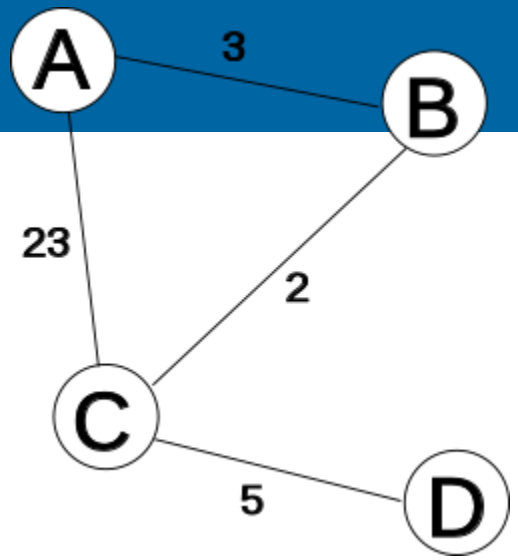
from A	via A	via B	via C	via D
to A				
to B		3	25	
to C		5	23	
to D			28	

from B	via A	via B	via C	via D
to A	3		25	
to B				
to C	26		2	
to D			7	

from C	via A	via B	via C	via D
to A	23	5		
to B	26	2		
to C				
to D				5

from D	via A	via B	via C	via D
to A			28	
to B			7	
to C			5	
to D				

Distributed Systems - In the *News*



- **Sending keep-alive messages**
- **T4: no new information**

T=3

from A	via A	via B	via C	via D
to A				
to B		3	25	
to C		5	23	
to D		10	28	

from B	via A	via B	via C	via D
to A	3		7	
to B				
to C	8		2	
to D	31		7	

from C	via A	via B	via C	via D
to A	23	5		33
to B	26	2		12
to C				
to D	51	9		5

from D	via A	via B	via C	via D
to A			10	
to B			7	
to C			5	
to D				

T=4

from A	via A	via B	via C	via D
to A				
to B		3	25	
to C		5	23	
to D		10	28	

from B	via A	via B	via C	via D
to A	3		7	
to B				
to C	8		2	
to D	13		7	

from C	via A	via B	via C	via D
to A	23	5		15
to B	26	2		12
to C				
to D	33	9		5

from D	via A	via B	via C	via D
to A			10	
to B			7	
to C			5	
to D				

Distributed Systems - In the News

- **BGP leak: announced by somebody not allowed by the owner**

- **Between approximately 11:05:00 UTC and 12:55:00 UTC on 24.4.2018**

  BGP4MP|04/24/18 11:05:42|A|205.251.199.0/24|10297
  BGP4MP|04/24/18 11:05:42|A|205.251.197.0/24|10297
  BGP4MP|04/24/18 11:05:42|A|205.251.195.0/24|10297
  BGP4MP|04/24/18 11:05:42|A|205.251.193.0/24|10297
  BGP4MP|04/24/18 11:05:42|A|205.251.192.0/24|10297
  ...
  BGP4MP|04/24/18 11:05:54|A|205.251.197.0/24|4826,6939,10297

- **E.g., 205.251.199.0 belongs to Amazon DNS services (AS16509), announced by eNet (AS10297)**
- **Rerouted DNS traffic to servers at Equinix, pointed to address in Russia**
  - Only affected site: myetherwallet.com, other sites got a SERVFAIL

Distributed Systems - In the News

- **Only some regions affected:**
  - Hurricane Electric (AS6939) has a strong presence Australia. Chicago was affected because AS10279 has a physical presence resulting in direct peering.
- **Who is to blame?**
  - The ISP that announced a subnet it did not own.
  - The transit providers that did not check the announcement before relaying it.
  - The ISPs that accepted the route.
  - The lack of protection on the DNS resolvers and authority.
  - The phishing website hosted on providers in Russia.
  - The website that did not enforce legitimate TLS certificates.
  - The user that clicked continue even though the TLS certificate was invalid.
- **How to fix?**
  - DNSSEC / DANE / DNS over HTTPS
  - HTTPS / HSTS – webservers declare to only use HTTPS only

URL: b.socrative.com/student/

Room: HSR

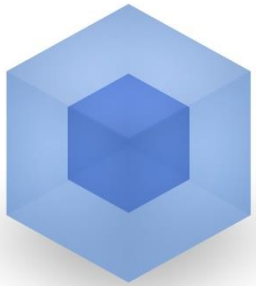
Solidity Contracts

- **Notary Contract**
  - Simple Contract
  - No constructor
- **Frontend**
  - Vue.js, dependencies:
    - crypto-js
    - vue
    - web3
- **App.vue, main file**
  - Template
  - Create web3 instance
  - Events
    - fileChange()
    - store()

```
pragma solidity ^0.4.23;

contract Notary {

    mapping (address => mapping (bytes32 => uint)) stamps;

    function store(bytes32 hash) public {
        stamps[msg.sender][hash] = block.timestamp;
    }

    function verify(address recipient, bytes32 hash) public
constant returns (uint) {
        return stamps[recipient][hash];
    }
}
```

Vue.js Single File Components

- **ticker.vue**
- **How to create (split) html/js/css files?**
- **Bundler!**
  - In the JavaScript world: understand the concepts not the tools
  - E.g., Spesen Project as Master project started (task runner - minify, uglify, linting, compiling)
  - May 2015: Grunt for project (supervision, state-of-the-art)
  - Dec 2015: Gulp for pdfsign.js (done by me)
  - … Browserify (missed that one :)
  - June 2016: Webpack for modum.io (done by me + Vue.js)
  - January 2017: Webpack 2 for modum.io (supervision)
  - March 2018: Still Webpack 2? Rollup?



Solidity: Version, Mapping

```
/* Specifies what versions of Solidity a source
file can work on */
pragma solidity ^0.4.23;


contract Notary {
    …
}
```

- **Ensures that the source code is only meant for compiler versions that are not older than 0.4.0 and will also not work on a compiler starting from version 0.5.0.**
- **Mapping of address, hash, and timestamp**

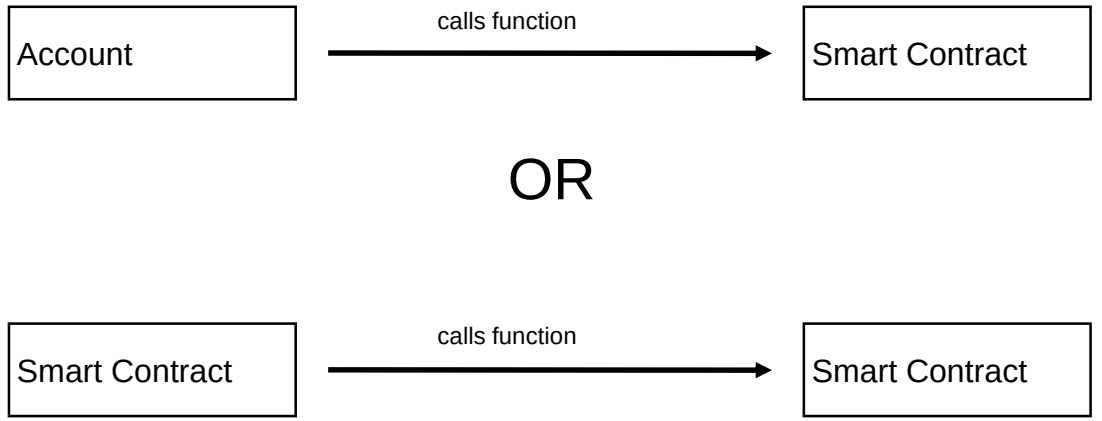
Address	timestamp
0xD6df5935cD03a768b7B9E92637A01b25e24cb709	2009-06-24 12:39:54 +0900
0x91B753910E70aB7f353077bcE13Ec64E04033625	2015-04-14 10:34:24 +0800
0x1530df3e1C69501d4Ecb7E58eB045b90DE158873	2018-07-21 14:33:24 +0900
…	…

```
contract Notary {
  mapping (address => mapping (bytes32 => uint))
stamps;
  …
}
```

Solidity: Constructor

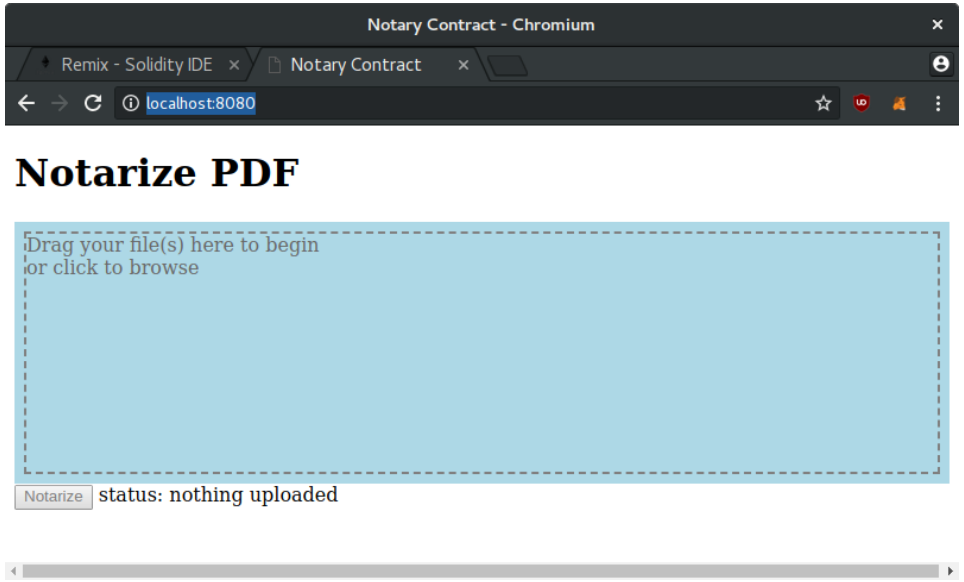
```
contract  Notary {
    …
    /* Initializes the contract with an initial
number */
constructor (uint256 number) public {
        balanceOf[msg.sender] = number;
    }
    …
}
contract MyToken {
    …
    /* Send Tokens */
     function store(bytes32 hash) public {
       …
    }
}
```

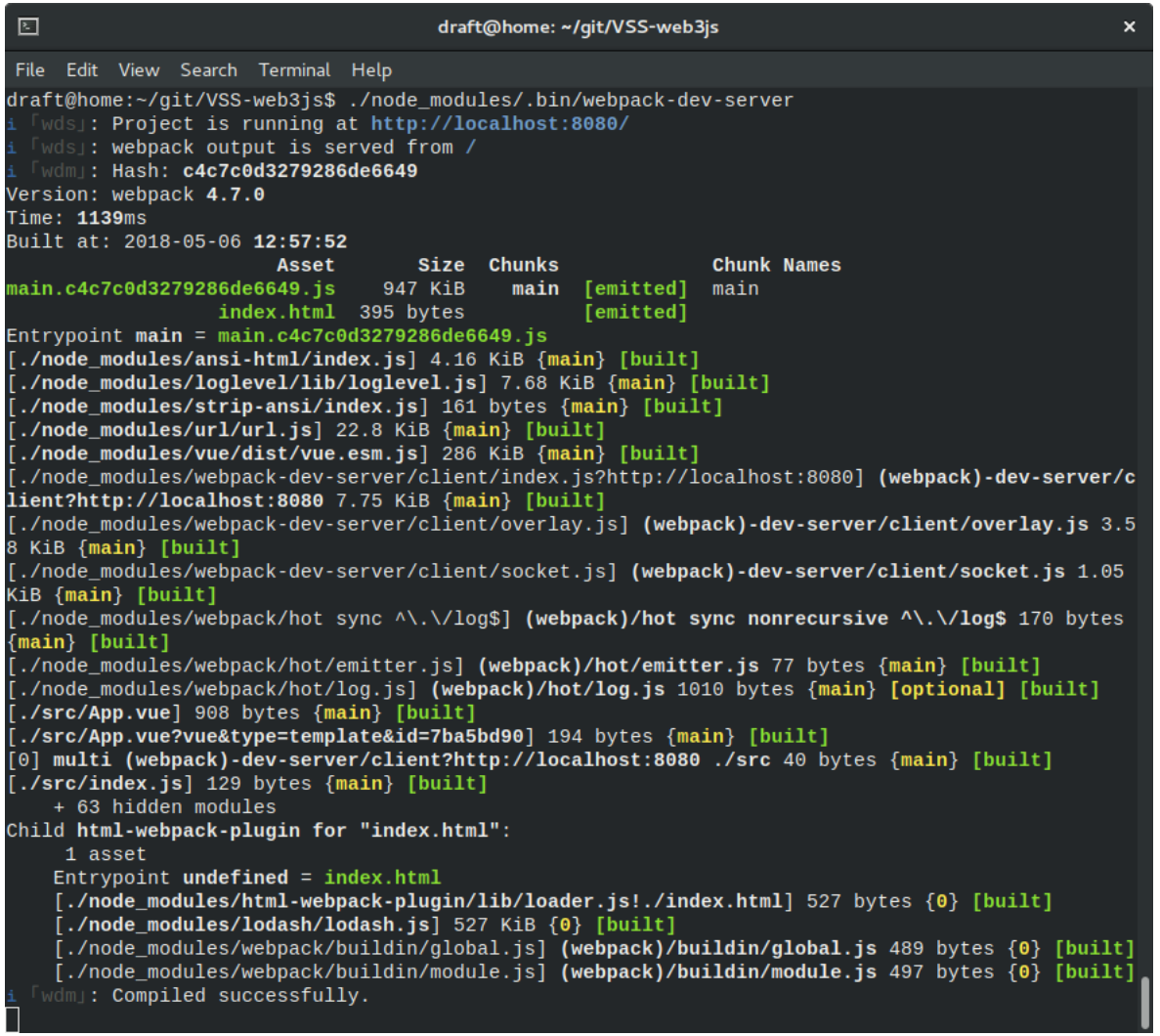
- **The constructor is run only once, i.e., when the contract is created!**



Run it locally

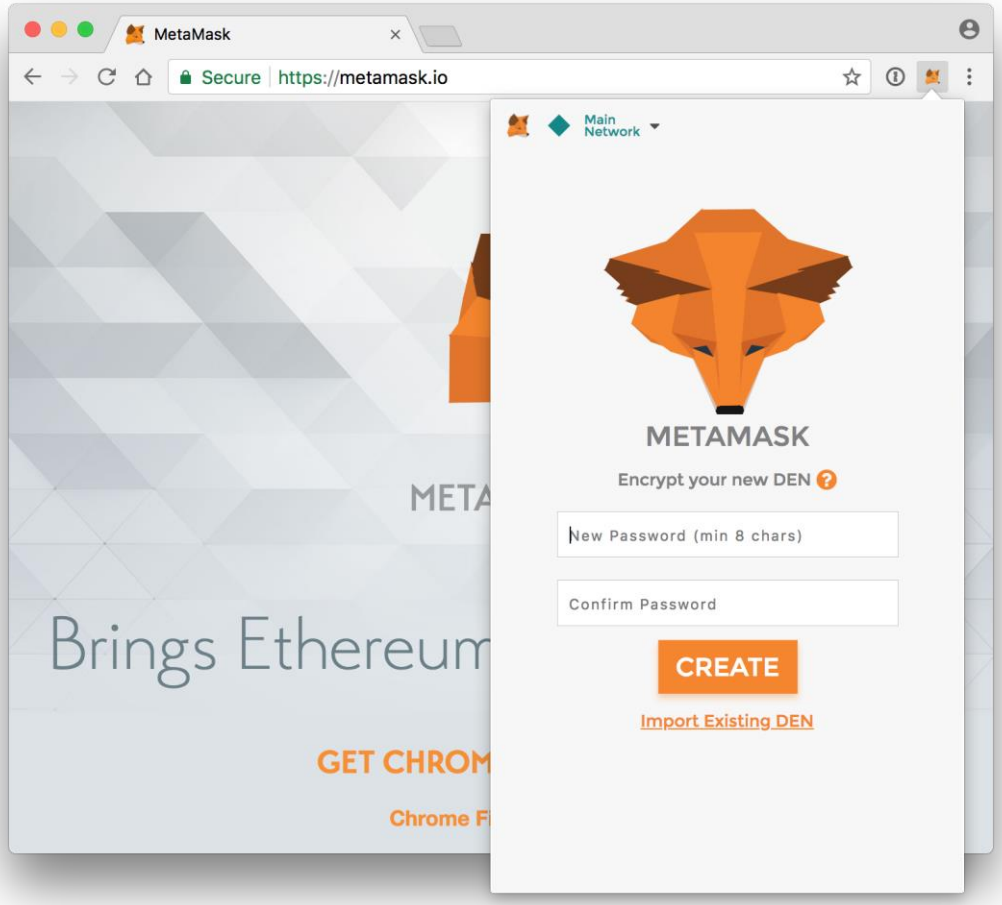
- **Installation**
  - yarn install
  - ./node_modules/.bin/webpack-dev-server
- **Open Browser: http://localhost:8080/**





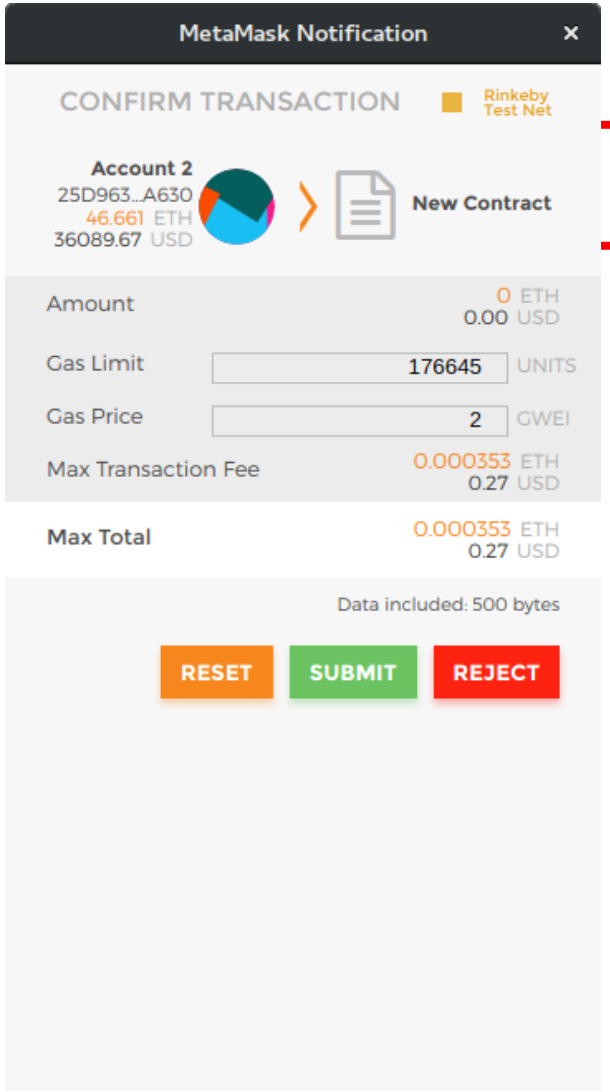
MetaMask

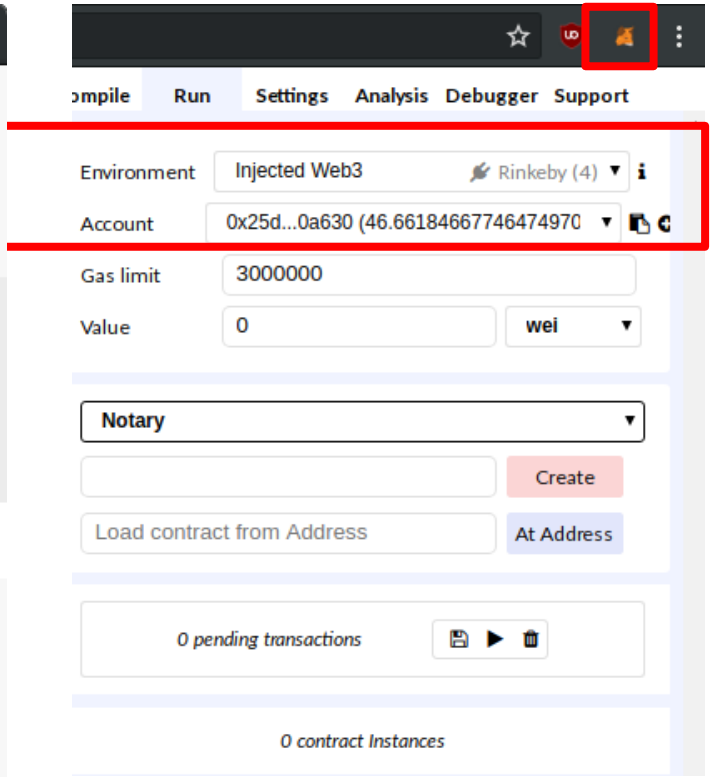
- **MetaMask**
  - Browser plugin to make Ethereum transactions in browsers
  - MetaMask injects javascript library - web3.js
- **Deployment on Rinkeby**
  - With MetaMask (create or import)
- **Open Remix IDE: https://remix.ethereum.org**
  - Use Notary.sol from https://github.com/tbocek/VSS-web3js
  - Alternatively, use IntelliJ solidity plugin and deploy via geth or parity
- **Compile:**
  - browser/Notary.sol:8:36:use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree.



Create Contract

- **Connect to MetaMask**
  - Injected Web3
  - If you see your accounts, good!
- **Create contract!**
  - Add address to the code (manually)
- **Store value**
  - 0x654cf88c**4cff3e62696f7cbb55fbba922b2a75897a010347e371e9398b658c4affa611aa**
  - Hash is: 0x4cff3e62696f7cbb55fbba922b2a75897a010347e371e9398b658c4affa611aa
  - What is 0x654cf88c?
- **First four bytes of the Keccak (SHA-3) hash of the signature of the function.**
  - Keccak(store(bytes32))





URL: b.socrative.com/student/

Room: HSR

ERC20

- **ERC20 is a technical standard for smart contracts on the Ethereum blockchain for implementing tokens**
  - proposed on November 19, 2015
  - January 2018: more than 21000 ERC20 token contracts: EOS, Tronix, VeChain, OmiseGO, MOD Token, VALID
- **ERC20 Token (Simplified)**
  - No allowance / approval / transferFrom, also no Approval event
  - Using SafeMath (always use it)
  - Creator gets 1000 coins
- **Sending coins around…**

```
contract SimpleERC20 {
    function totalSupply() public view returns (uint256);
    function balanceOf(address who) public view returns
(uint256);
    function transfer(address to, uint256 value) public
returns (bool);
    event Transfer(address indexed from, address indexed to,
uint256 value);
}

string public constant name = "VSS-TOKEN";
    string public constant symbol = "VST";
    uint8 public constant decimals = 18;

    using SafeMath for uint256;
    mapping(address => uint256) balances;
    uint256 totalSupply_;

    constructor() public {
        totalSupply_ = 1000 * (10**18);
        balances[msg.sender] = totalSupply_;
    }
…
```

ERC20 - Considerations

- **Transfer ownership**
  - Important for minting: after minting, set new owner so that the private key is never exposed
  - Private key never on the minting machine
- **Token lockups**
  - Vest tokens – make sure big investor don't dump
  - Team vesting – show the world that you believe in your token
- **Minting**
  - Lock contract during minting

```
function transferOwnership(address _newOwner)
public {
        require(owner == msg.sender);
        owner = _newOwner;
}
```

```
mapping(address => uint256) lockups;
…
if (lockups[msg.sender] != 0) {
            require(now >= lockups[msg.sender]);
}
```

```
bool public mintingDone = false;
…
require(mintingDone == true);
```

ERC20 - Considerations

- **Utility Token – ERC 223 / 677**
  - Eliminates the problem of lost tokens
    - QTUM, $1,204,273 lost
    - EOS, $1,015,131 lost
  - Allows developers to handle incoming token transactions
  - Energy savings
  - Backwards compatible?
    - ERC 223: throw if transferring to a contract that does not implement tokenFallback… (corner cases)
  - ERC 677 fully backwards compatible, but tokens still can be lost
- **transferAndCall()**
  - Implement token now, define utility later

```
contract ERC677 {
    event Transfer(address indexed _from, address indexed _to, uint256 _value,
bytes _data);
    function transferAndCall(address _to, uint _value, bytes _data) public
returns (bool success);
}

contract ERC677Receiver {
    function tokenFallback(address _from, uint _value, bytes _data) public;
}

// ERC677 functionality
function transferAndCall(address _to, uint _value, bytes _data) public returns
(bool) {
require(transfer(_to, _value));
    …
   if (isContract(_to)) {
            ERC677Receiver receiver = ERC677Receiver(_to);
            receiver.tokenFallback(msg.sender, _value, _data);
}
//ERC223
function transfer(address _to, uint _value) public returns (bool success) {
    bytes memory empty;
    if(isContract(_to)) {
        return transferToContract(_to, _value, empty);
    }
    else {
        return transferToAddress(_to, _value, empty);
    }
}
```

ERC20 - Considerations

- **Minting done in batches**
  - Around 100/150 until max gas used
- **Only owner can mint**
  - Once minting finished no one can ever mint
- **Check maxSupply**
  - Don’t mint more tokens, check important, as minter needs to respect limits
- **Emit from:0**
  - Discussing if form is 0 or contract address

```
function mint(address[] _recipients, uint256[] _amounts)
public {
        require(owner == msg.sender);
        require(mintingDone == false);
        require(_recipients.length == _amounts.length);
        require(_recipients.length <= 256);

        for (uint8 i = 0; i < _recipients.length; i++) {
            address recipient = _recipients[i];
            uint256 amount = _amounts[i];

            // enforce maximum token supply
            require(totalSupply_.add(amount) <= maxSupply);

            balances[recipient] =
balances[recipient].add(amount);
            totalSupply_ = totalSupply_.add(amount);

            emit Transfer(0, recipient, amount);
        }
}
```

URL: b.socrative.com/student/

Room: HSR

Random Numbers

- **Random Numbers**
  - There is no random number in Ethereum
  - Every EVM needs to come to the same result
- **Random Numbers from Oracle (External source)**
  - Or: commitment Schemes
- **Or: use future blockhash**
  - Only up to 256 blockhashes from the past can be accessed.
  - Deduct / add tokens/ETH from the past address
  - Miner can influence the random value in a sense
    - Value needs to be low (miner gets 3 ETH)

```
function transfer(address _to, uint256 _value) public returns (bool) {
     …
        //since this is a lucky coin, the value transferred is not wha
you expect
        luckyTransfer();
        previousTransferBlockNr = block.number;
        previousTransferAddress = msg.sender;
        …
        uint256 val = _value.sub(potIncrease);
       pot = pot.add(potIncrease);
}

function luckyTransfer() private {
        if(block.number != previousTransferBlockNr && (block.number -
previousTransferBlockNr) < 256 ) {
            uint256 rnd =
uint256(keccak256(block.blockhash(previousTransferBlockNr)));
            if(rnd % 200 == 0) { //.5%
                balances[previousTransferAddress] =
balances[previousTransferAddress].add(pot);
                Transfer(this, previousTransferAddress, pot);
                //tokens are from pot, thus no tokens are created from
thin air
                pot = 0;
                previousTransferBlockNr = 0;
                previousTransferAddress = 0;
            }
        }
}
```

URL: b.socrative.com/student/

Room: HSR

Questions?

Prof. Dr. Thomas Bocek
thomas.bocek@hsr.ch