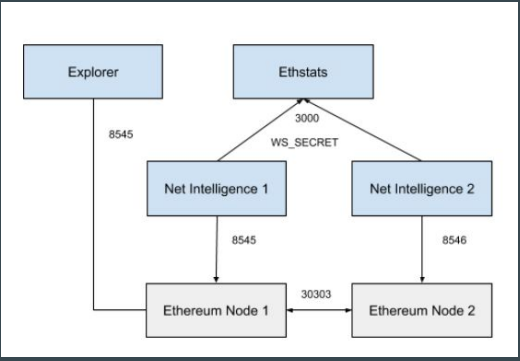
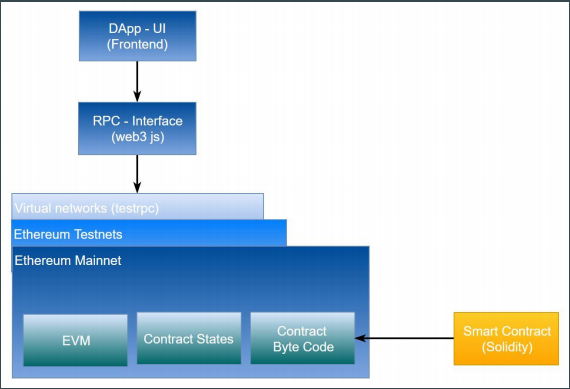
**Hướng Setup Ethereum Private Network And Deploy Token Trên Ubuntu 16.04**

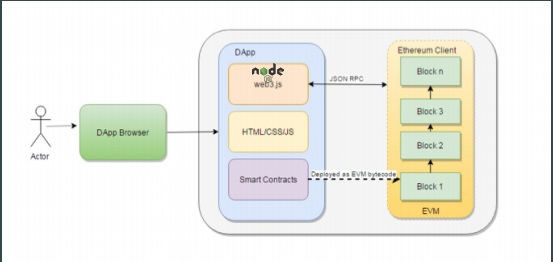
1. **Ethereum Private Network là gì**
2. **Sơ đồ tổng quát:**



1. **S­­­­­ơ đồ dApp:**



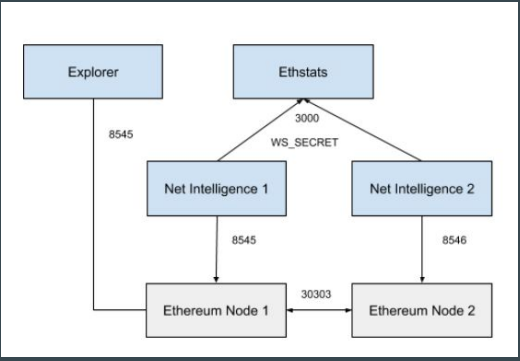
* Build Server API connect to Web3.js
* Reactjs, React Native, Angular, ... use API



1. **Các Thành Phần Chính:**

* Ethereum Node: ethereum private network.
* Net Intelligence: eth-net-intellegence-api.
* Explorer: same ethereum scan(transaction).
* Ethstats: thống kê private network.

1. **Mô Hình Phác Thảo:**



1. **CHUẨN BỊ:**

* 1 Máy tính(ubuntu16.04x64-Ram4G) để xây dựng private network.

1. **TÓM TẮT CÁC BƯỚC THỰC HIỆN:**
2. Account Sử Dụng Để Setup
3. Cài Đặt Công Cụ Cho Ubuntu
4. Setup Ethereum Private Network(Geth)
5. Setup Explorer
6. Eth-netstats
7. Eth-net-intelligence-api
8. Setup Mist-Wallet
9. Demo Create Token
10. Tài liệu tham khảo
11. **CHI TIẾT CÁC BƯỚC THỰC HIỆN:**

|  |  |  |
| --- | --- | --- |
| **INPUT** | **OUTPUT** | **GHI CHÚ** |
| 1. **Account Sử Dụng Để Setup** | | |
| putty ssh login   * Host=103.77.169.248:22 * Username=user * password=\*\*\*\*\*\*\*\*\* |  |  |
|  |  |  |
| 1. **Cài Đặt Công Cụ Cho Ubuntu** | | |
| **$** sudo apt-get update  **$** sudo apt-get install build-essential  **$** sudo apt-get upgrade  **$** ./prereqs-ubuntu.sh |  |  |
|  |  |  |
| 1. **Setup Ethereum Private Network(Geth)** | | |
| sudo add-apt-repository -y ppa:ethereum/ethereum  sudo apt-get update  sudo apt-get install ethereum |  |  |
| Create new account | $geth --datadir=./chaindata/ account new | Chép account address lại |
| Create genesis.json | $vi genesis.json  {  "coinbase" : "0x0000000000000000000000000000000000000001",  "difficulty" : "0x20000",  "extraData" : "",  "gasLimit" : "0x2fefd8",  "nonce" : "0x0000000000000042",  "mixhash" : "0x0000000000000000000000000000000000000000000000000000000000000000",  "parentHash" : "0x0000000000000000000000000000000000000000000000000000000000000000",  "timestamp" : "0x00",  "alloc": {  "account number": { "balance": "0x1337000000000000000000" }  },  "config": {  "chainId": 22,  "homesteadBlock": 0,  "eip155Block": 0,  "eip158Block": 0  }  } | dán account address trong file genesis.json, và balance khởi tạo ban đầu. |
| Init data | $geth --datadir=./chaindata/ init ./genesis.json |  |
| Mining | $geth --identity "process-node" --datadir=./chaindata/ --mine --minerthreads=1 --rpc --networkid 22 --rpcaddr "0.0.0.0" --rpcport 8545 --rpcapi="db,eth,net,web3,personal" --rpccorsdomain "\*" |  |
|  |  |  |
| 1. **Setup Mist-Wallet** | | |
| Tải Mist-linux64-0-11-1.zip | Tải Mist tại <https://github.com/ethereum/mist/releases>  Và giải nén | Nếu dùng Mist-linux64-0-11-1.deb thì tiến hành cài :  $ sudo dpkg -i --force-all Mist-Wallet-linux64-0-10-0.deb |
| Start Mist | ./mist --network 22 --rpc ~/chaindata/geth.ipc |  |
| 1. **Setup Explorer** | | |
| Download source explorer | $git clone <https://github.com/carsenk/explorer> |  |
| Install dependencies | $cd explorer/  $ npm install |
| Install bower | $npm install -g bower  $bower install |  |
| Start explorer | $npm start |  |
| 1. **Eth-netstats** | | |
| Download source eth-getstats | $git clone <https://github.com/cubedro/eth-netstats> |  |
| Install dependencies | $cd eth-netstats  $npm install |  |
| Install grunt | $npm install -g grunt-cli  $ grunt |  |
| Start eth-stats | $WS\_SECRET=my\_secret npm start |  |
| 1. **Eth-net-intelligence-api** | | |
| Download sources eth-net –intelligence-api | $git clone https://github.com/cubedro/eth-net-intelligence-api |  |
| Install dependencies | $cd eth-net-intelligence-api  $npm install |  |
| Install pm2 | $sudo -S npm install -g pm2 |  |
| Config | $Change value app.json with parameter: "name" : "process-node",  "INSTANCE\_NAME" : "name", "WS\_SERVER" : "http://localhost:3000",  "WS\_SECRET" : "my\_secret", |  |
| Start pm2 | $pm2 start app.json |  |
|  |  |  |
| 1. **Demo Create Token** | | |
| Create smartcontract TokenERC20 | $Create TokenERC20.sol  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  pragma solidity ^0.4.16;  interface tokenRecipient { function receiveApproval(address \_from, uint256 \_value, address \_token, bytes \_extraData) external; }  contract TokenERC20 {  // Public variables of the token  string public name;  string public symbol;  uint8 public decimals = 18;  // 18 decimals is the strongly suggested default, avoid changing it  uint256 public totalSupply;  // This creates an array with all balances  mapping (address => uint256) public balanceOf;  mapping (address => mapping (address => uint256)) public allowance;  // This generates a public event on the blockchain that will notify clients  event Transfer(address indexed from, address indexed to, uint256 value);    // This generates a public event on the blockchain that will notify clients  event Approval(address indexed \_owner, address indexed \_spender, uint256 \_value);  // This notifies clients about the amount burnt  event Burn(address indexed from, uint256 value);  /\*\*  \* Constructor function  \*  \* Initializes contract with initial supply tokens to the creator of the contract  \*/  function TokenERC20(  uint256 initialSupply,  string tokenName,  string tokenSymbol  ) public {  totalSupply = initialSupply \* 10 \*\* uint256(decimals); // Update total supply with the decimal amount  balanceOf[msg.sender] = totalSupply; // Give the creator all initial tokens  name = tokenName; // Set the name for display purposes  symbol = tokenSymbol; // Set the symbol for display purposes  }  /\*\*  \* Internal transfer, only can be called by this contract  \*/  function \_transfer(address \_from, address \_to, uint \_value) internal {  // Prevent transfer to 0x0 address. Use burn() instead  require(\_to != 0x0);  // Check if the sender has enough  require(balanceOf[\_from] >= \_value);  // Check for overflows  require(balanceOf[\_to] + \_value >= balanceOf[\_to]);  // Save this for an assertion in the future  uint previousBalances = balanceOf[\_from] + balanceOf[\_to];  // Subtract from the sender  balanceOf[\_from] -= \_value;  // Add the same to the recipient  balanceOf[\_to] += \_value;  emit Transfer(\_from, \_to, \_value);  // Asserts are used to use static analysis to find bugs in your code. They should never fail  assert(balanceOf[\_from] + balanceOf[\_to] == previousBalances);  }  /\*\*  \* Transfer tokens  \*  \* Send `\_value` tokens to `\_to` from your account  \*  \* @param \_to The address of the recipient  \* @param \_value the amount to send  \*/  function transfer(address \_to, uint256 \_value) public returns (bool success) {  \_transfer(msg.sender, \_to, \_value);  return true;  }  /\*\*  \* Transfer tokens from other address  \*  \* Send `\_value` tokens to `\_to` on behalf of `\_from`  \*  \* @param \_from The address of the sender  \* @param \_to The address of the recipient  \* @param \_value the amount to send  \*/  function transferFrom(address \_from, address \_to, uint256 \_value) public returns (bool success) {  require(\_value <= allowance[\_from][msg.sender]); // Check allowance  allowance[\_from][msg.sender] -= \_value;  \_transfer(\_from, \_to, \_value);  return true;  }  /\*\*  \* Set allowance for other address  \*  \* Allows `\_spender` to spend no more than `\_value` tokens on your behalf  \*  \* @param \_spender The address authorized to spend  \* @param \_value the max amount they can spend  \*/  function approve(address \_spender, uint256 \_value) public  returns (bool success) {  allowance[msg.sender][\_spender] = \_value;  emit Approval(msg.sender, \_spender, \_value);  return true;  }  /\*\*  \* Set allowance for other address and notify  \*  \* Allows `\_spender` to spend no more than `\_value` tokens on your behalf, and then ping the contract about it  \*  \* @param \_spender The address authorized to spend  \* @param \_value the max amount they can spend  \* @param \_extraData some extra information to send to the approved contract  \*/  function approveAndCall(address \_spender, uint256 \_value, bytes \_extraData)  public  returns (bool success) {  tokenRecipient spender = tokenRecipient(\_spender);  if (approve(\_spender, \_value)) {  spender.receiveApproval(msg.sender, \_value, this, \_extraData);  return true;  }  }  /\*\*  \* Destroy tokens  \*  \* Remove `\_value` tokens from the system irreversibly  \*  \* @param \_value the amount of money to burn  \*/  function burn(uint256 \_value) public returns (bool success) {  require(balanceOf[msg.sender] >= \_value); // Check if the sender has enough  balanceOf[msg.sender] -= \_value; // Subtract from the sender  totalSupply -= \_value; // Updates totalSupply  emit Burn(msg.sender, \_value);  return true;  }  /\*\*  \* Destroy tokens from other account  \*  \* Remove `\_value` tokens from the system irreversibly on behalf of `\_from`.  \*  \* @param \_from the address of the sender  \* @param \_value the amount of money to burn  \*/  function burnFrom(address \_from, uint256 \_value) public returns (bool success) {  require(balanceOf[\_from] >= \_value); // Check if the targeted balance is enough  require(\_value <= allowance[\_from][msg.sender]); // Check allowance  balanceOf[\_from] -= \_value; // Subtract from the targeted balance  allowance[\_from][msg.sender] -= \_value; // Subtract from the sender's allowance  totalSupply -= \_value; // Update totalSupply  emit Burn(\_from, \_value);  return true;  }  }  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |  |
| Deploy smartcontract |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| 1. **Tài liệu tham khảo** | | |
| Blocklink  <https://ethereum.org/token>  <https://viblo.asia/p/tu-thiet-lap-private-blockchain-dua-tren-nen-tang-ethereum-RQqKLg0457z>  <https://medium.com/@chim/ethereum-how-to-setup-a-local-test-node-with-initial-ether-balance-using-geth-974511ce712> | | |