

Criando uma rede privada Ethereum passo a passo com Geth

Lançada em 2015, Ethereum é a principal blockchain programável do mundo. Como outras blockchains, Ethereum tem uma criptomoeda nativa chamada Ether (ETH). ETH é dinheiro digital. Se você ouviu falar de Bitcoin, o ETH tem muitas das mesmas características. É puramente digital e pode ser enviado instantaneamente a qualquer pessoa do mundo. O fornecimento de ETH não é controlado por nenhum governo ou empresa - é descentralizado, e é escasso. Pessoas ao redor de todo o mundo usam ETH para fazer pagamentos, como um meio de valor ou como garantia.

A Ethereum é programável, o que significa que os desenvolvedores podem usá-la para construir novos tipos de aplicativos.

Estes aplicativos descentralizados (ou "dapps") ganham os benefícios da tecnologia de criptomoeda e blockchain. Eles podem ser confiáveis, o que significa que, uma vez que são "enviados" para a Ethereum, eles serão sempre executados como programado. Podem controlar os ativos digitais para criar novos tipos de aplicações financeiras. Podem ser descentralizados, o que significa que nenhuma entidade ou pessoa os controla.

1. Rede privada e teste

Vamos começar a estudar um pouco dos conceitos por trás do ethereum usando geth juntos.

Se desejar acesse: ithub : <https://github.com/fcd007/private-ethereum-network>

Iremos construir nossa rede privada e testar se está funcionando bem. No entanto, usaremos apenas 1 nó de exemplo. Vamos tentar vários nós mais tarde em projetos futuros.

Tutorial Criando uma rede privada Ethereum passo a passo com Geth v.01

Índice

- Ambiente
- Construir rede privada
- Testar rede privada
- Transação para testar a rede privada

Ambiente

- Linux Ubuntu 20.04 LTS
- Go (version 1.10 or later)
- geth (go-ethereum)

2. Download e instalação geth

Meu caminho para clonar o repositório é `/home/dantas007/Documents/blockchain/` assim eu clonei Geth para dentro da pasta de destino abaixo:

```
git clone https://github.com/ethereum/go-ethereum.git
```

```
dantas007 in ~  
> ls  
Android  compose-demo  Desktop  Documents  Downloads  eclipse-workspace  go  Music  node_modules  
Pictures  Public  snap  Templates  Videos  
dantas007 in ~  
> cd /home/dantas007/Documents/blockchain/ethereum_private01/
```

Em seguida, o geth será baixado. após o download, vá para o go-ethereum diretório e abra o terminal.

```
dantas007 in Documents/blockchain/ethereum_private01  
> git clone https://github.com/ethereum/go-ethereum.git  
Cloning into 'go-ethereum'...  
remote: Enumerating objects: 89780, done.  
remote: Total 89780 (delta 0), reused 0 (delta 0), pack-reused 89780  
Receiving objects: 100% (89780/89780), 138.82 MiB | 1.55 MiB/s, done.  
Resolving deltas: 100% (59358/59358), done.  
dantas007 in Documents/blockchain/ethereum_private01 took 1m 37s  
>
```

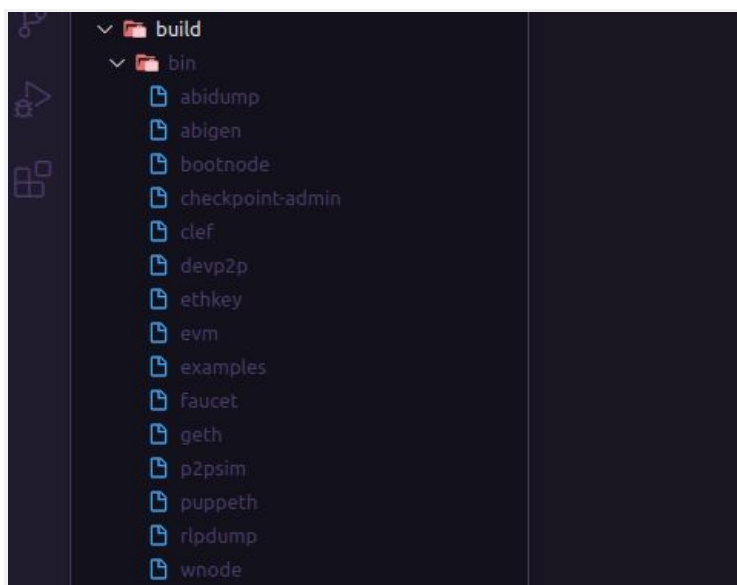
Tutorial Criando uma rede privada Ethereum passo a passo com Geth v.01

Em seguida, o geth será baixado. após o download, vá para o diretório go-ethereum e abra o terminal, execute o comando abaixo: `make all` ou pelo `geth console` com o `make geth`

```
> ls
go-ethereum
dantas007 in Documents/blockchain/ethereum_private01
> cd go-ethereum
dantas007 in go-ethereum on master
> make all
env GO111MODULE=on go run build/ci.go install
>>> /usr/local/go/bin/go install -ldflags -X main.gitCommit=56a319b9daa5228a6b22ecb1d07f8183ebd98
```

No entanto, se você quiser testar com vários nós algum dia, precisará fazer o comando novamente com o `make all`

Se a compilação for bem-sucedida, você poderá ver `geth` em `/go-ethereum/build/bin`



Você pode ver o arquivo `geth`

3. Inicializar a rede privada

Primeiro de tudo você tem que fazer um `genesis.json`. Você pode fazê-lo em qualquer pasta, em qualquer diretório, exceto em um diretório que precise de permissão, mas por conveniência, recomendo fazê-lo na mesma pasta com `geth` (`/go-ethereum/build/bin`), podemos criar de duas maneiras o arquivo `genesis.json`,

Tutorial Criando uma rede privada Ethereum passo a passo com Geth v.01

com o comando `nano genesis.json` e será aberto o editor para salvar a configuração ou com o comando `touch genesis.json` e após isso abrir com arquivo com o seu editor de preferência, no nosso caso será usando comando `nano genesis.json`:



```
1 2 {
3  "config": {
4      "chainId": 15,
5      "homesteadBlock": 0,
6      "eip150Block": 0,
7      "eip155Block": 0,
8      "eip158Block": 0
9  },
10 "alloc" : {},
11 "difficulty" : "0x400",
12 "extraData" : "",
13 "gasLimit" : "0x7A1200",
14 "parentHash" : "0x0000000000000000000000000000000000000000000000000000000000000000",
15 "timestamp" : "0x00"
16 }
```

Após as alterações no arquivo salve-o, no nano `Ctrl+X`, e depois um `enter` para confirmar o nome do arquivo.

Agora temos que criar um diretório para armazenar informações.

Eu fiz uma pasta chamada `geth-test`, você pode fazer isso em qualquer lugar.

O meu diretório de exemplo é:

```
/home/dantas007/Documents/blockchain/ethereum_private01/geth-test/
```

Finalmente, estamos prontos para começar, vá para `/go-ethereum/build/bin`, abra o terminal e siga os passos abaixo:

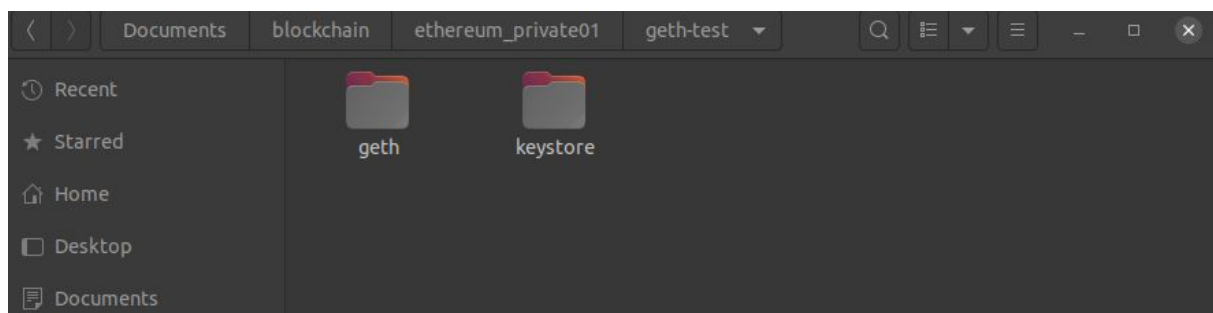
```
./geth --datadir
/home/dantas007/Documents/blockchain/ethereum_private01/geth-t
est/ init genesis.json
```

Tutorial Criando uma rede privada Ethereum passo a passo com Geth v.01

```
dantas007@dantas007: ~/Documents/blockchain/ethereum_private01/go-ethereum/...
dantas007 in Documents/blockchain/ethereum_private01
> ls
go-ethereum
dantas007 in Documents/blockchain/ethereum_private01
> mkdir geth-test
dantas007 in Documents/blockchain/ethereum_private01
> cd geth-test
dantas007 in blockchain/ethereum_private01/geth-test
> cd ../go-ethereum/build/bin
dantas007 in go-ethereum/build/bin on master [?]
>

dantas007@dantas007: ~/Documents/blockchain/ethereum_private01/go-ethereum/...
dantas007 in go-ethereum/build/bin on master
> ./geth --datadir /home/dantas007/Documents/blockchain/ethereum_private01/geth-test init genesis
.json
INFO [06-17|19:54:00.417] Maximum peer count                ETH=50 LES=0 total=50
INFO [06-17|19:54:00.417] Smartcard socket not found, disabling err="stat /run/pcscd/pcscd.com
m: no such file or directory"
INFO [06-17|19:54:00.421] Allocated cache and file handles  database=/home/dantas007/Docum
ents/blockchain/ethereum_private01/geth-test/geth/chaindata cache=16.00MiB handles=16
INFO [06-17|19:54:00.548] Writing custom genesis block
INFO [06-17|19:54:00.549] Persisted trie from memory database nodes=0 size=0.00B time="7.787
µs" gcnodes=0 gcsizes=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [06-17|19:54:00.550] Successfully wrote genesis state database=chaindata hash="fcf2d
3...874b60"
INFO [06-17|19:54:00.550] Allocated cache and file handles  database=/home/dantas007/Docum
ents/blockchain/ethereum_private01/geth-test/geth/lightchaindata cache=16.00MiB handles=16
INFO [06-17|19:54:00.661] Writing custom genesis block
INFO [06-17|19:54:00.661] Persisted trie from memory database nodes=0 size=0.00B time="5.84µ
s" gcnodes=0 gcsizes=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [06-17|19:54:00.662] Successfully wrote genesis state database=lightchaindata hash="
fcf2d3...874b60"
dantas007 in go-ethereum/build/bin on master
>
```

Se a iniciação for bem-sucedida, você poderá ver `geth` e `keystore` em sua pasta de armazenamento:



4. Executar rede privada!

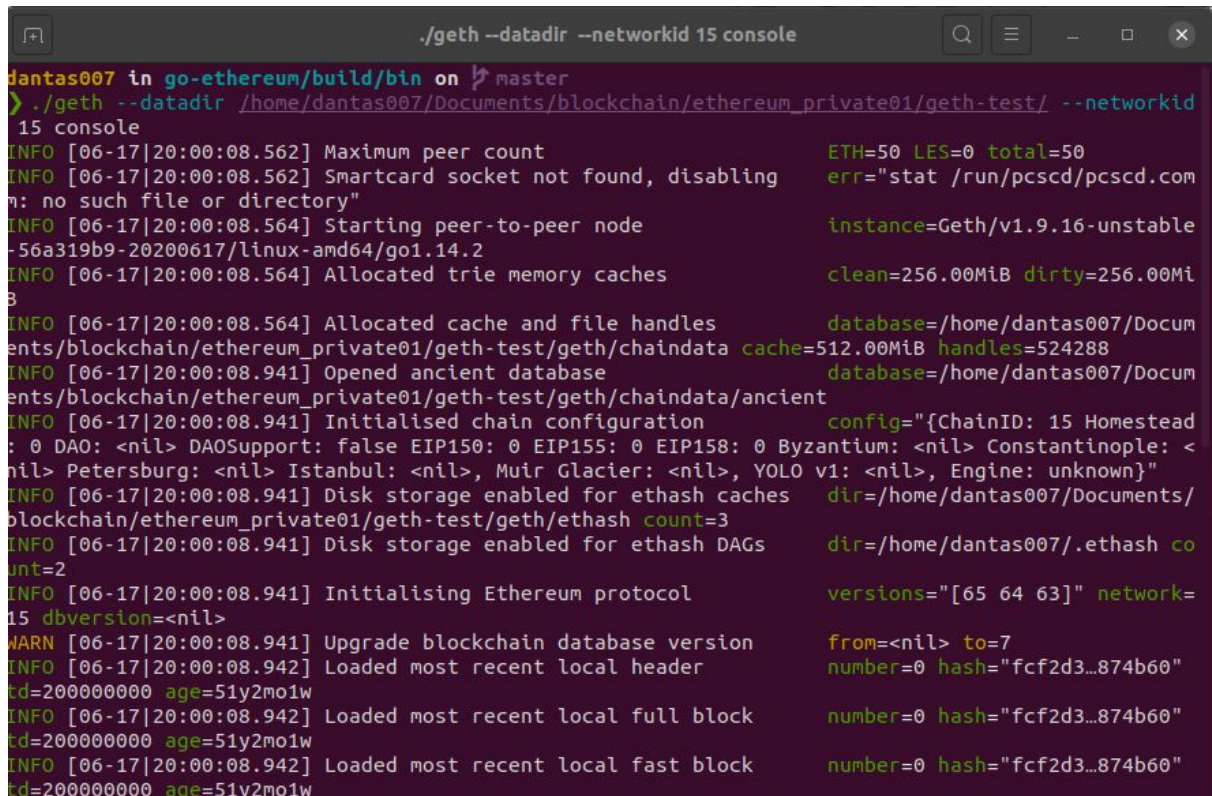
abra o terminal `/go-ethereum/build/bin` e agora pode executar o comando abaixo:

Tutorial Criando uma rede privada Ethereum passo a passo com Geth v.01

```
./geth --datadir  
/home/dantas007/Documents/blockchain/ethereum_private01/geth-test/  
--networkid 15 console
```

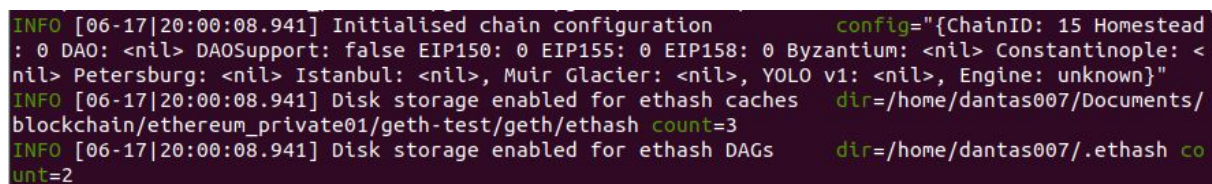
Definimos chainId como 15 no arquivo genesis.json. Portanto, nossos networkId 15 também.

Então você pode ver o comando executado abaixo:



```
./geth --datadir --networkid 15 console  
dantas007 in go-ethereum/build/bin on master  
> ./geth --datadir /home/dantas007/Documents/blockchain/ethereum_private01/geth-test/ --networkid  
15 console  
INFO [06-17|20:00:08.562] Maximum peer count                ETH=50 LES=0 total=50  
INFO [06-17|20:00:08.562] Smartcard socket not found, disabling err="stat /run/pcscd/pcscd.com  
m: no such file or directory"  
INFO [06-17|20:00:08.564] Starting peer-to-peer node        instance=Geth/v1.9.16-unstable  
-56a319b9-20200617/linux-amd64/go1.14.2  
INFO [06-17|20:00:08.564] Allocated trie memory caches      clean=256.00MiB dirty=256.00MiB  
INFO [06-17|20:00:08.564] Allocated cache and file handles  database=/home/dantas007/Docum  
ents/blockchain/ethereum_private01/geth-test/geth/chaindata cache=512.00MiB handles=524288  
INFO [06-17|20:00:08.941] Opened ancient database           database=/home/dantas007/Docum  
ents/blockchain/ethereum_private01/geth-test/geth/chaindata/ancient  
INFO [06-17|20:00:08.941] Initialised chain configuration    config="{ChainID: 15 Homestead  
: 0 DAO: <nil> DAOSupport: false EIP150: 0 EIP155: 0 EIP158: 0 Byzantium: <nil> Constantinople: <  
nil> Petersburg: <nil> Istanbul: <nil>, Muir Glacier: <nil>, YOLO v1: <nil>, Engine: unknown}"  
INFO [06-17|20:00:08.941] Disk storage enabled for ethash caches dir=/home/dantas007/Documents/  
blockchain/ethereum_private01/geth-test/geth/ethash count=3  
INFO [06-17|20:00:08.941] Disk storage enabled for ethash DAGs dir=/home/dantas007/.ethash co  
unt=2  
INFO [06-17|20:00:08.941] Initialising Ethereum protocol    versions="[65 64 63]" network=  
15 dbversion=<nil>  
WARN [06-17|20:00:08.941] Upgrade blockchain database version from=<nil> to=7  
INFO [06-17|20:00:08.942] Loaded most recent local header    number=0 hash="fcf2d3...874b60"  
td=2000000000 age=51y2mo1w  
INFO [06-17|20:00:08.942] Loaded most recent local full block number=0 hash="fcf2d3...874b60"  
td=2000000000 age=51y2mo1w  
INFO [06-17|20:00:08.942] Loaded most recent local fast block number=0 hash="fcf2d3...874b60"  
td=2000000000 age=51y2mo1w
```

Na última linha, você pode ver o ponteiro do console (>). existem algumas linhas que precisamos verificar



```
INFO [06-17|20:00:08.941] Initialised chain configuration    config="{ChainID: 15 Homestead  
: 0 DAO: <nil> DAOSupport: false EIP150: 0 EIP155: 0 EIP158: 0 Byzantium: <nil> Constantinople: <  
nil> Petersburg: <nil> Istanbul: <nil>, Muir Glacier: <nil>, YOLO v1: <nil>, Engine: unknown}"  
INFO [06-17|20:00:08.941] Disk storage enabled for ethash caches dir=/home/dantas007/Documents/  
blockchain/ethereum_private01/geth-test/geth/ethash count=3  
INFO [06-17|20:00:08.941] Disk storage enabled for ethash DAGs dir=/home/dantas007/.ethash co  
unt=2
```

Nesta linha, podemos verificar que o ChainID é 15 como esperávamos. Agora nossa rede privada está funcionando!

5. Testar rede privada

Agora vamos testar nossa rede privada, vamos executar o comando `eth.blockNumber` para verificar o número de blocos. Nós apenas geramos essa rede. Portanto, há 0 bloco.

Execute o comando outro comando o `eth.accounts` agora vamos verificar as contas da rede. Não há contas.

```
> eth.blockNumber
0
> eth.accounts
[]
> INFO [06-17|20:08:46.451] Looking for peers      peercount=0 tried=59 static
```

Vamos gerar conta: `personal.newAccount("Alice")`

```
> personal.newAccount("Alice")
INFO [06-17|20:36:33.917] Your new key was generated      address=0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d
d2E66fc6123EF949C5E7d
WARN [06-17|20:36:33.917] Please backup your key file!    path=/home/dantas007/Documents/blockchain/ethereum_private01/geth-test/keystore/UTC--2020-06-17T23-36-31.211966084Z--4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d
WARN [06-17|20:36:33.917] Please remember your password!  "0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d"
> INFO [06-17|20:36:35.997] Looking for peers      peercount=0 tried=80 static
```

É gerado o endereço de Alice: "0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d",

Podemos vê-lo usando `geth`:

```
INFO [06-17|20:37:39.806] Looking for peers      peercount=0 tried=90 static=0
> eth.accounts
["0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d"]
> INFO [06-17|20:38:10.043] Looking for peers      peercount=1 tried=70 static=0
=0
```

Nós o usaremos como endereço do mineiro, para que a recompensa da geração de blocos seja enviada para o endereço de Alice.

Tutorial Criando uma rede privada Ethereum passo a passo com Geth v.01

Antes da mineração, vamos verificar o equilíbrio de Alice, copiamos o endereço de Alice e aplicamos o comando:

```
eth.getBalance("0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d")  
eth.getBalance(eth.accounts[0])
```

```
0  
> eth.getBalance(INFO [06-17|20:38:33.337] Looking for peers peercount=0  
tried=62 static=0  
> eth.getBalance("0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d")  
0  
> INFO [06-17|20:38:44.176] Looking for peers peercount=0 tried=121 stati  
c=0
```

Como esperamos, não há ether. agora vamos minerar.

Primeiro temos que definir o endereço do mineiro

Vamos usar 3 comandos

Define o endereço do mineiro. Recompensa de mineração será enviada para esta conta: `miner.setEtherbase(address)`

```
c=0 tried=60 static=0  
> miner.setEtherbase("0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d")  
true  
> INFO [06-17|20:39:50.344] Looking for peers peercount=0 tried=56 stati  
c=0  
INFO [06-17|20:40:00.392] Looking for peers peercount=1 tried=51 static=
```

Comece a mineração. Você pode definir quantos threads você usará. Vou usar 1 thread: `miner.start(number of threads)`

```
> miner.start(1)  
INFO [06-17|20:20:29.697] Updated mining threads threads=1  
INFO [06-17|20:20:29.697] Transaction pool price threshold updated price=1000000000  
null
```

Parando de minerar: `miner.stop()`

```
> miner.stop()  
null
```


Tutorial Criando uma rede privada Ethereum passo a passo com Geth v.01

Terminamos a mineração. Agora vamos verificar se funcionou bem.

```
> eth.blockNumber
322
> INFO [06-17|21:02:05.889] Looking for peers          peercount=0 tried=93 stati
```

No meu caso, mineramos 322 blocos até pará-lo.

```
> ether.stop()
null
> eth.getBalance("0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d")
1.61e+21
```

Nós recebemos uma recompensa. Exatamente wei não ether. wei é uma pequena unidade de ether. Podemos convertê-lo em éter por comando abaixo:

```
web3.fromWei(eth.getBalance(address), "ether")
```

```
rs          peercount=0 tried=64 static=0
> web3.fromWei(eth.getBalance(eth.accounts[0]), "ether")
1610
> INFO [06-17|21:06:11.903] Looking for peers          peercount=0 tried=56 stati
```

6. Transação para testar a rede privada

Gerar nova conta para poder realizar uma transação.

```
unt=1 tried=113 static=0
> personal.newAccount("Bob")
INFO [06-17|21:08:23.028] Your new key was generated          address=0xe8f88660fcBeA9530Aa
6A5f2775077247Bb4635D
WARN [06-17|21:08:23.028] Please backup your key file!          path=/home/dantas007/Document
s/blockchain/ethereum_private01/geth-test/keystore/UTC--2020-06-18T00-08-20.122547785Z--e8f88660
fcbea9530aa6a5f2775077247bb4635d
WARN [06-17|21:08:23.028] Please remember your password!
"0xe8f88660fcbea9530aa6a5f2775077247bb4635d"
```

Vamos verificar o balanço da conta de Bob:

```
> eth.getBalance("0xe8f88660fcbea9530aa6a5f2775077247bb4635d")
> eth.getBalance("0xe8f88660fcbea9530aa6a5f2775077247bb4635d")
INFO [06-17|21:10:02.480] Looking for peers          peercount=1 tried=70 static=0
0
```

Tutorial Criando uma rede privada Ethereum passo a passo com Geth v.01

Eu recebi a conta de Bob: "0xe8f88660fcbea9530aa6a5f2775077247bb4635d" Alice enviará éter para a conta de Bob.

```
eth.sendTransaction({from: "0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d", to: "0xe8f88660fcbea9530aa6a5f2775077247bb4635d", value: web3.toWei(22, "ether")})
```

Vamos enviar 22 ether para a conta de Bob

- Conta de Alice: "0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d"
- A conta de Bob: "0xe8f88660fcbea9530aa6a5f2775077247bb4635d"

Ou podemos inicializá-los usando a variável

```
> from = "0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d"
> to = "0xe8f88660fcbea9530aa6a5f2775077247bb4635d"
> eth.sendTransaction({from: from, to: to, value: web3.toWei(5, "ether")})
```

Nós enviamos ether no entanto, ***não esqueça que nunca usamos a chave privada de Alice***. Portanto, nós iremos ver o erro abaixo:

```
> eth.sendTransaction({from: "0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d", to: "0xe8f88660fcbea9530aa6a5f2775077247bb4635d", value: web3.toWei(22, "ether")})
WARN [06-17|21:16:17.498] Served eth_sendTransaction reqId=27 t=2.951948ms err="authentication needed: password or unlock"
Error: authentication needed: password or unlock
    at web3.js:6347:37(47)
    at web3.js:5081:62(37)
    at <eval>:1:20(15)
```

Temos que desbloquear a conta de Alice. Vamos ver o status da conta de Alice.

```
> personal.listWalletsINFO [06-17|21:17:22.873] Looking for peers peercount=0 tried=78 static=0
> personal.listWallets[0].status
"Locked"
> INFO [06-17|21:17:32.897] Looking for peers peercount=0 tried=47 static=0
```

Sim ... está bloqueada ... Então temos que desbloqueá-la para enviar ether de Alice para Bob.

Tutorial Criando uma rede privada Ethereum passo a passo com Geth v.01

```
peercount=0 tried=80 static=0
> web3.personal.unlockAccount("0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d") INFO [06-17|21:19:35.7
55] Looking for peers peercount=0 tried=86 static=0
> web3.personal.unlockAccount("0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d")
Unlock account 0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d
Passphrase:
INFO [06-17|21:19:45.761] Looking for peers peercount=0 tried=80 static=
0
true
```

Verificando as transações pendentes:

```
0
> eth.pendingTransactions
[]
> INFO [06-17|21:20:27.208] Looking for peers peercount=0 tried=61 stati
```

Realizando novamente a transação de Alice para Bob:

```
0
> eth.sendTransaction({from: "0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d", to: "0xe8f88660fcbea9
530aa6a5f2775077247bb4635d", value: web3.toWei(22, "ether")})
INFO [06-17|21:21:14.159] Setting new local account address=0x4b89D0DE7C870E7e4B2
d2E66fc6123EF949C5E7d
INFO [06-17|21:21:14.159] Submitted transaction fullhash=0x2707c0b127193b8c8b
6df4d047e01d807baca72f181ec4b9d3c850bdb2f3d85a recipient=0xe8f88660fCBeA9530Aa6A5f2775077247Bb46
35D
"0x2707c0b127193b8c8b6df4d047e01d807baca72f181ec4b9d3c850bdb2f3d85a"
> INFO [06-17|21:21:18.621] Looking for peers peercount=0 tried=123 stati
```

Agora a conta de Alice está desbloqueada. Vamos voltar para a transação. Podemos ver transações pendentes

```
0
> eth.pendingTransactions
[
  {
    blockHash: null,
    blockNumber: null,
    from: "0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d",
    gas: 21000,
    gasPrice: 1000000000,
    hash: "0x2707c0b127193b8c8b6df4d047e01d807baca72f181ec4b9d3c850bdb2f3d85a",
    input: "0x",
    nonce: 0,
    r: "0x6c74dfd43762c27aee7d8afe1e060cef8a10424f0fc75ec3eb20fecbabde153c",
    s: "0x4aa1d7dcc3586ca0710aaa8735ed7a4d3a51755445c7b5589e6719788a861693",
    to: "0xe8f88660fcbea9530aa6a5f2775077247bb4635d",
    transactionIndex: null,
    v: "0x41",
    value: 22000000000000000000
  }
]
```

Como ainda não mineramos nenhum bloco. então Ainda não há mudança de equilíbrio. Voltamos minerar!

Tutorial Criando uma rede privada Ethereum passo a passo com Geth v.01

```
INFO [06-17|21:24:02.317] Looking for peers      peercount=0 tried=61 static=0
> eth.getBalance("0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d")
1.61e+21
> eth.getBalance("0xe8f88660fcbea9530aa6a5f2775077247bb4635d")
INFO [06-17|21:24:22.368] Looking for peers      peercount=1 tried=63 static=0
> eth.getBalance("0xe8f88660fcbea9530aa6a5f2775077247bb4635d")
0
```

```
0
> eth.blockNumber
358
> INFO [06-17|21:27:46.436] Looking for peers      peercount=1 tried=57 static=0
```

Não há transação pendente. A transação de Alice e Bob está concluída!

```
INFO [06-17|21:28:27.459] Looking for peers      peercount=0 tried=65 static=0
> eth.getBalance("0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d")
INFO [06-17|21:28:37.526] Looking for peers      peercount=0 tried=68 static=0
> eth.pendingTransactions
[]
```

Vamos verificar os balanços:

```
> eth.getBalance("0x4b89d0de7c870e7e4b2d2e66fc6123ef949c5e7d")
1.768e+21
> eth.blockNumber
INFO [06-17|21:29:48.217] Looking for peers      peercount=1
tried=67 static=0
> eth.getBalance("0xe8f88660fcbea9530aa6a5f2775077247bb4635d")
22000000000000000000
> INFO [06-17|21:29:58.246] Looking for peers      peercount=2 tried=84 static=0
```

Como esperávamos, Alice tem 1768 ether, Bob tem 22 ether. Conseguimos!

```
> web3.fromWei(eth.getBalance(eth.accounts[0]), "ether")
1768
> web3.fromWei(eth.getBalance(eth.accounts[1]), "ether")
22
> INFO [06-17|21:31:44.271] Looking for peers      peercount=0 tried=58 static=0
```

Isso é tudo que foi testado neste passo a passo. Meu objetivo é estudar o algoritmo de consenso de geth para estudar. Portanto, se meu processo for bem-sucedido, se houver erros ou você quiser adicionar mais detalhes, contribua.

7. Referências bibliográficas

<https://ethereum.org/pt-br/what-is-ethereum/>

<https://github.com/ethereum/go-ethereum>

<https://github.com/ethereum/go-ethereum/wiki/Private-network#creating-the-genesis-block>

<https://github.com/HyungsungKim/Studying-ethereum>