Experiment No.7
To create a private ethereum blockchain using Geth
Date of Performance:
Date of Submission:

AIM: To create a private ethereum blockchain using Geth

Objective: To understand the concept of private ethereum blockchain

Theory:

Ethereum is a decentralized blockchain platform that establishes a peer-to-peer network that securely executes and verifies application code, called smart contracts. Smart contracts allow participants to transact with each other without a trusted central authority. Transaction records are immutable, verifiable, and securely distributed across the network, giving participants full ownership and visibility into transaction data. Transactions are sent from and received by user-created Ethereum accounts. A sender must sign transactions and spend Ether, Ethereum's native cryptocurrency, as a cost of processing transactions on the network.

An Ethereum Private Network is a completely private Blockchain which is isolated from the Main Ethereum network. Ethereum Private Network is mainly created by organizations to restrict the read permissions of the Blockchain. Only the nodes with the right permissions will be able to access this Blockchain. The nodes in this network are not connected to the main network nodes and their reach is restricted only to this private Blockchain.

Ethereum Private Network is used by organizations to store private data which should not be visible to people outside their organization. Ethereum Private Network is also used for testing and experimenting the Blockchain if someone doesn't want to use the public test networks.

Ethereum Private Network has its own set of features as listed below:

- It acts as a Distributed Database
- Blockchain in the Ethereum Private Network can contain private data (because the network is not public)
- Access can be permission-based
- Making transactions can be free

- Accounts can be allocated with ethers by ourselves which does not even require buying virtual ethers

Geth

Geth is an Ethereum client written in Go. This means running Geth turns a computer into an Ethereum node. Ethereum is a peer-to-peer network where information is shared directly between nodes rather than being managed by a central server. Nodes compete to generate new blocks of transactions to send to its peers because they are rewarded for doing so in Ethereum's native token, ether (ETH). On receiving a new block, each node checks that it is valid and adds it to their database. The sequence of discrete blocks is called a "blockchain". The information provided in each block is used by Geth to update its "state" - the ether balance of each account on Ethereum. There are two types of account: externally-owned accounts (EOAs) and contract accounts. Contract accounts execute contract code when they receive transactions. EOAs are accounts that users manage locally in order to sign and submit transactions. Each EOA is a public-private key pair, where the public key is used to derive a unique address for the user and the private key is used to protect the account and securely sign messages. Therefore, in order to use Ethereum, it is first necessary to generate an EOA.

Process:

Step 1. Install NodeJs for Windows 10 from URL 'https://nodejs.org/en/download/'.

Step 2. Install Ethereum Mist Wallet for Windows 10 from https://github.com/ethereum/mist/releases.

Step 3. Download and install '**geth**' from URL https://geth.ethereum.org/downloads/

Step 4. Create Genesis block [A file genesis.json is required to be created and store in c:/Users/Admin]

Step 5. Initialize the genesis block using command

Open the command prompt and navigate to the folder wherein "genesis.json" file is located. Run following command:

"geth init customGenesis.json"

Step 6. Once the genesis block is created, run the following command to start the private network:

"geth — networkid=5 console"

Step 7. Launch Ethereum Wallet

The Ethereum wallet can be seen as part of the private blockchain

Step 8. Create Address

Create an address in the Ethereum Wallet application. The address can be from the Ethereum Wallet. In '**Wallets'** section in the Ethereum Mist Wallet application, click on **'Add Account'** to create a new account address.

Step 9. Start mining

After creating the address, go back to the command prompt where the network is running and run the following command:

"miner.start(1)"

Step 10. In case you need to stop mining, run the following command;

"miner.stop()"

Output: Screenshorts

```
C:\Users\admin>geth init --datadir node genesis.json
INFO [10-05|07:42:41.273] Maximum peer count                       ETH=50 LES=0 total=50
INFO [10-05|07:42:41.281] Defaulting to pebble as the backing database
INFO [10-05|07:42:41.281] Allocated cache and file handles         database=C:\Users\admin\node\geth\chaindata cache=512.00MiB handles=8192
INFO [10-05|07:42:41.356] Opened ancient database                  database=C:\Users\admin\node\geth\chaindata\ancient/chain readonly=false
INFO [10-05|07:42:41.358] State schema set to default              scheme=hash
INFO [10-05|07:42:41.392] Set global gas cap                       cap=50,000,000
INFO [10-05|07:42:41.393] Initializing the KZG library             backend=gokzg
INFO [10-05|07:42:41.414] Using pebble as the backing database
INFO [10-05|07:42:41.414] Allocated cache and file handles         database=C:\Users\admin\node\geth\chaindata cache=16.00MiB  handles=16
INFO [10-05|07:42:41.460] Opened ancient database                  database=C:\Users\admin\node\geth\chaindata\ancient/chain readonly=false
INFO [10-05|07:42:41.462] State schema set to default              scheme=hash
INFO [10-05|07:42:41.462] Writing custom genesis block
INFO [10-05|07:42:41.494] Successfully wrote genesis state         database=chaindata                          hash=8da729..3e3e9f
INFO [10-05|07:42:41.496] Defaulting to pebble as the backing database
INFO [10-05|07:42:41.498] Allocated cache and file handles         database=C:\Users\admin\node\geth\lightchaindata cache=16.00MiB  handles=16
INFO [10-05|07:42:41.579] Opened ancient database                  database=C:\Users\admin\node\geth\lightchaindata\ancient/chain readonly=false
INFO [10-05|07:42:41.580] State schema set to default              scheme=hash
INFO [10-05|07:42:41.580] Writing custom genesis block
INFO [10-05|07:42:41.615] Successfully wrote genesis state         database=lightchaindata                          hash=8da729..3e3e9f

C:\Users\admin>
```

```
Microsoft Windows [Version 10.0.22000.2416]
(c) Microsoft Corporation. All rights reserved.

C:\Users\admin>geth init customGenesis.json
INFO [10-05|06:34:03.570] Maximum peer count                       ETH=50 LES=0 total=50
INFO [10-05|06:34:03.576] Using pebble as the backing database
INFO [10-05|06:34:03.576] Allocated cache and file handles         database=C:\Users\student\AppData\Local\Ethereum\geth
\chaindata cache=512.00MiB handles=8192
INFO [10-05|06:34:03.587] Opened ancient database                  database=C:\Users\student\AppData\Local\Ethereum\geth
\chaindata\ancient/chain readonly=true
INFO [10-05|06:34:03.587] State scheme set to already existing     scheme=hash
INFO [10-05|06:34:03.588] Set global gas cap                       cap=50,000,000
INFO [10-05|06:34:03.589] Initializing the KZG library             backend=gokzg
INFO [10-05|06:34:03.608] Using pebble as the backing database
INFO [10-05|06:34:03.609] Allocated cache and file handles         database=C:\Users\student\AppData\Local\Ethereum\geth
\chaindata cache=16.00MiB  handles=16
INFO [10-05|06:34:03.699] Opened ancient database                  database=C:\Users\student\AppData\Local\Ethereum\geth
\chaindata\ancient/chain readonly=false
INFO [10-05|06:34:03.700] State scheme set to already existing     scheme=hash
Fatal: Failed to write genesis block: database contains incompatible genesis (have d4e56740f876aef8c010b86a40d5f56745a11
8d0906a34e69aec8c0db1cb8fa3, new d51512d0d6c2d58ccca7dfd8ced6ae36c505deeb01cd322d491babda51e4b107)

C:\Users\admin>
```

Conclusion:

Geth is a well-justified choice for creating a private Ethereum blockchain for several reasons. It offers compatibility with the Ethereum ecosystem, facilitating the seamless deployment of DApps and smart contracts on the public network. Geth is highly customizable, allowing you to tailor your blockchain to specific needs with features like permissioning, consensus algorithms, and privacy controls. Its robust developer support, active open-source community, and access to Ethereum standards make it easier to integrate and interoperate with other systems. Furthermore, Geth's isolation from the public Ethereum network provides enhanced security and performance reliability, making it a dependable tool for building consortium or enterprise blockchain solutions.