F

Ethereum Consortium

Simulated Multi Member

Contents

[Overview 3](#_Toc496833437)

[About blockchain 3](#_Toc496833438)

[Mining Node Details 3](#_Toc496833439)

[Transaction Node Details 3](#_Toc496833440)

[Ethereum configuration 4](#_Toc496833441)

[Deployment Architecture 4](#_Toc496833442)

[Getting Started 4](#_Toc496833443)

[Post Deployment Sanity Checks 11](#_Toc496833444)

[Administrator page 11](#_Toc496833445)

[Accessing Nodes 13](#_Toc496833446)

[Create Ethereum Account 14](#_Toc496833447)

[Initiate Initial Ether Allocation 16](#_Toc496833448)

[Transfer of Ether between Accounts 18](#_Toc496833449)

[Next Steps 19](#_Toc496833450)

# Overview

This solution template is designed to make it sufficiently easier and quicker to deploy and configure a multi-member consortium Ethereum network with minimal Azure and Ethereum knowledge.

With a handful of user inputs and a single-click deployment through the Azure portal, each member can provision their network footprint, using Microsoft Azure compute, networking, and storage services across the globe. Each member's network footprint consists of a set of load-balanced transaction nodes with which an application or user can interact to submit transactions, a set of mining nodes to record transactions, and a VPN gateway. A subsequent connection step connects the gateways to create a fully configured multi-member blockchain network.

# About blockchain

For those of you who are new to the blockchain community, the release of this solution is a great opportunity to learn about the technology in an easy and configurable manner on Azure. However, to get started, we recommend deploying the simpler Ethereum consortium network topology with this guided walkthrough, before building out multi-member networks.

## Mining Node Details

Each consortium member is given a separate, identical subnet containing one or more mining nodes, backed by a managed disk. The first default VM in the subnet is configured as a boot node to support dynamic discoverability of the nodes in the network. Mining nodes communicate with other mining nodes to come to consensus on the state of the underlying distributed ledger. There is no need for your application to be aware of or communicate with these nodes. Since we are focused on private networks, these nodes are isolated from inbound public internet traffic adding a secondary layer of protection. Outbound traffic is allowed, but not to the Ethereum discovery port. While each member’s VMs are in a separate subnet, the individual nodes are still connected and communicating with one another via Ethereum’s discovery protocol.

All nodes have the latest stable Go Ethereum (Geth) client software and are configured to be mining nodes. All nodes use the same Ethereum address and key pair that is protected by the Ethereum account password. The Ethereum passphrase you provided is used to generate the default account (coinbase) for each mining node. As mining nodes mine, they collect fees that are added to this account.

The number of mining nodes per consortium member depends on the overall size of the network desired and amount of hashing power dedicated to each member. The larger the network the more nodes that need to be compromised to gain an unfair advantage. The template supports up to 15 mining nodes per consortium member.

## Transaction Node Details

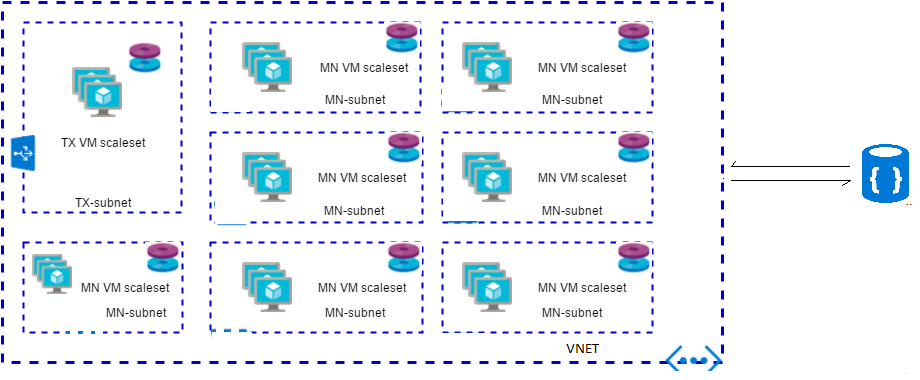
All consortium members share a set of load-balanced transaction nodes. These nodes are reachable from outside the virtual network so that applications can use these nodes to submit transactions or execute smart contracts within the blockchain networks. All nodes have the latest stable Go Ethereum (Geth) client software and are configured to maintain a complete copy of the distributed ledger. These nodes use the same Ethereum account, protected by the Ethereum account password provided.

We have explicitly separated the nodes that accept transactions from the nodes that mine transactions to ensure that the two actions are not competing for the same resources. We have also load-balanced the transaction nodes using VM scale sets. The template supports up to 5 transaction nodes in the VM scale sets.

## Ethereum configuration

Besides the infrastructural footprint and configuration of nodes, the blockchain network itself is created. The genesis block is configured with the desired Ethereum network id, an appropriate mining difficulty, and a preconfigured account. The mining difficulty varies depending on the number of mining nodes deployed to ensure mining time remains short even in the beginning. The pre-configured account contains 1 trillion Ether to seed the consortium network with enough gas (Ethereum’s fuel) to handle millions of transactions. Since the mining nodes use this account, their collected fees feed back into the account to ensure continual funds

# Deployment Architecture



This solution template can deploy single region simulated members Ethereum consortium network.

# Getting Started

This process requires an Azure subscription that can support deploying several virtual machines scale sets and managed disks. If necessary, [create a free Azure account](https://azure.microsoft.com/en-us/free/) to begin.

Once a subscription is secured, go to Azure portal. Select ‘+’, Marketplace (‘See all’), and search for ‘Ethereum Consortium Single region simulated’.

The Template Deployment will walk you through configuring the first member’s footprint in the network. The deployment flow is divided into three steps: Basics, Network Configurations and Ethereum configuration.

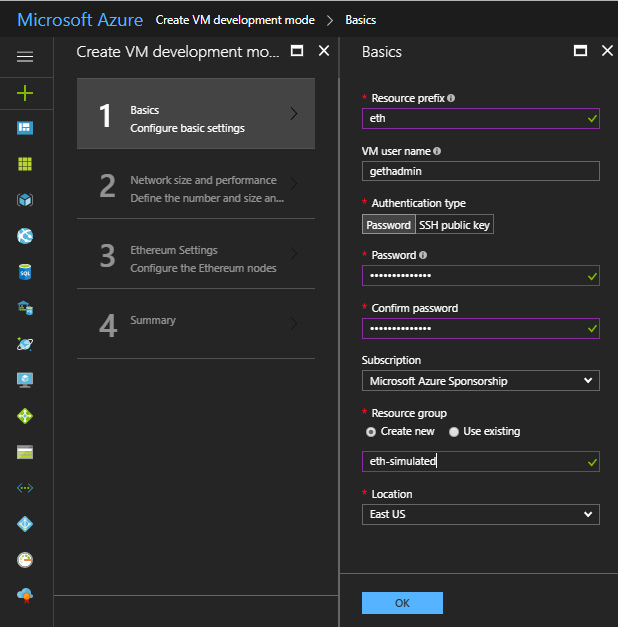
**Basics**

Under the ‘Basics’ blade, specify values for standard parameters for any deployment, such as subscription, resource group and basic virtual machine properties.

A detailed description of each parameter follows:

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter Name** | **Description** | **Allowed Values** | **Default Values** |
| Resource Prefix | String used as a base for naming resources (2 to 4 alphanumeric characters). A unique hash is prepended to the string for some resources, while resource-specific information is appended. | Alphanumeric characters with length 2 to 4 | NA |
| VM user name | Administrator username of each deployed VM (alphanumeric characters only) | 1-64 characters | gethadmin |
| Authentication type | The method to authenticate to the virtual machine. | Password or SSH public key | Password |
| Password (Authentication type = Password) | The password for the administrator account for each of the virtual machines deployed. The password must contain 3 of the following: 1 upper case character, 1 lower case character, 1 number, and 1 special character.  While all VMs initially have the same password, you can change the password after provisioning. | 12 -72 characters | NA |
| SSH Key (Authentication type = Public Key) | The secure shell key used for remote login. |  | NA |
| Subscription | The subscription to which to deploy the consortium network |  | NA |
| Resource Group | The resource group to which to deploy the consortium network. |  | NA |
| Location | The Azure region for resource group. |  | NA |

**A sample deployment is shown below**



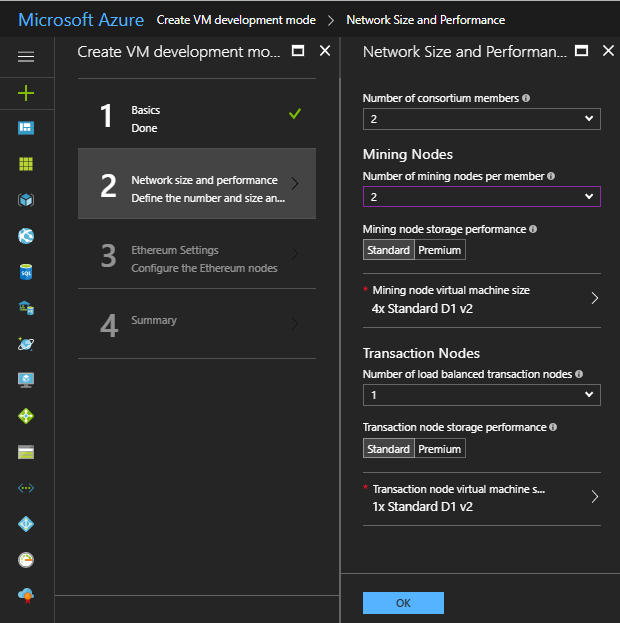
**Network size and performance**

Next, under ‘Network size and performance’ specify inputs for the size of the consortium network, such as number and size of mining nodes and transaction nodes.

A detailed description of each parameter follows:

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter Name** | **Description** | **Allowed Values** | **Default Values** |
| Number of Consortium Members | The number of members to simulate within the network. Each consortium member receives a subnet containing one mining node (virtual machine) backed by one storage account as the initial footprint  Member ID should be unique across different organizations in the same network. A unique member ID is needed even when the same organization deploys to multiple regions.    Make note of the value of this parameter, since you will need to share it with other joining members. | 2-12 | 2 |
| Number of mining nodes per member | The number of mining nodes deployed per subnet.    The total number of mining nodes equals NumConsortiumMembers x NumMiningNodesPerMember. | 2-15 | 2 |
| Mining node storage performance | The type of managed disk backing each of the deployed mining nodes. | Standard or Premium | Standard |
| Mining node virtual machine size | The virtual machine size used for mining nodes. | Standard A, Standard D, Standard D-v2, Standard F series, Standard DS, and Standard FS | Standard\_A1 |
| Number of load balanced transaction nodes | The number of transaction nodes to provision as part of the network. | 1-5 | 2 |
| Transaction node storage performance | The type of managed disk backing each of the deployed transaction nodes. | Standard or Premium | Standard |
| Transaction node virtual machine size | The virtual machine size used for transaction nodes. | Standard A, Standard D, Standard D-v2, Standard F series, Standard DS, and Standard FS | Standard\_A1 |

**A sample deployment is shown below:**

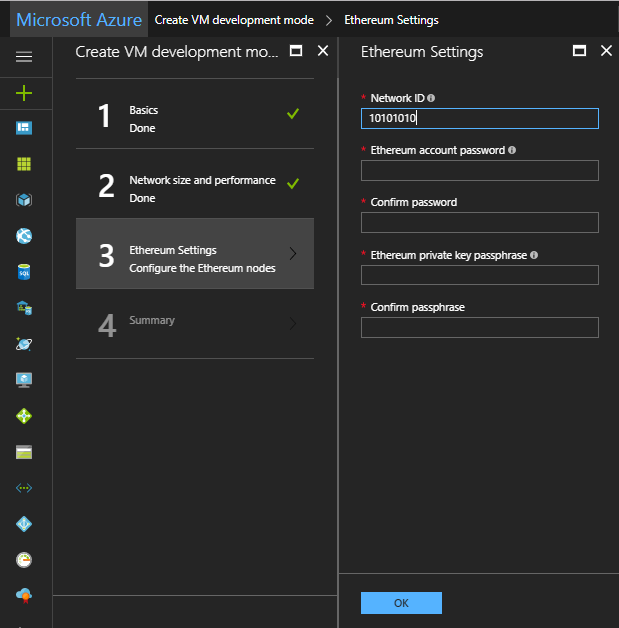


**Ethereum Settings**

Next, under Ethereum settings, specify Ethereum-related configuration settings, like the network ID and Ehterum account password or genesis block.

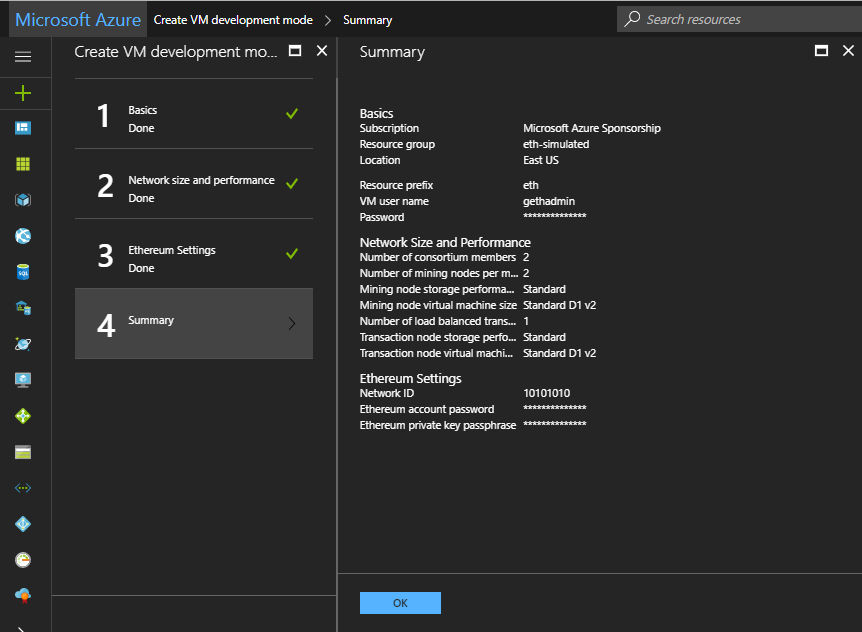
A detailed description of each parameter follows:

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter Name** | **Description** | **Allowed Values** | **Default Values** |
| Ethereum Network ID | The network ID for the consortium Ethereum network being deployed. Each Ethereum network has its own Network ID, with 1 being the ID for the public network. While we have restricted network access for mining nodes, we still recommend using a large number to prevent collisions. | 5 - 999,999,999 | 10101010 |
| Ethereum Account Password | The administrator password used to secure the Ethereum account imported into each node.  The password must contain the following: 1 upper case character, 1 lower case character, and 1 number. | 12 or more characters | NA |
| Ethereum private key passphrase | The passphrase used to generate the ECC private key associated with the default Ethereum account that is generated. A pre-generated private key does not need to be explicitly passed in.    Consider a passphrase with sufficient randomness to ensure a strong private key and no overlap with other consortium members. The passphrase must contain the following at a minimum: 1 upper case character, 1 lower case character, and 1 number. | 12 or more characters | NA |



**Summary**

Click through the summary blade to review the inputs specified and to run basic pre-deployment validation.



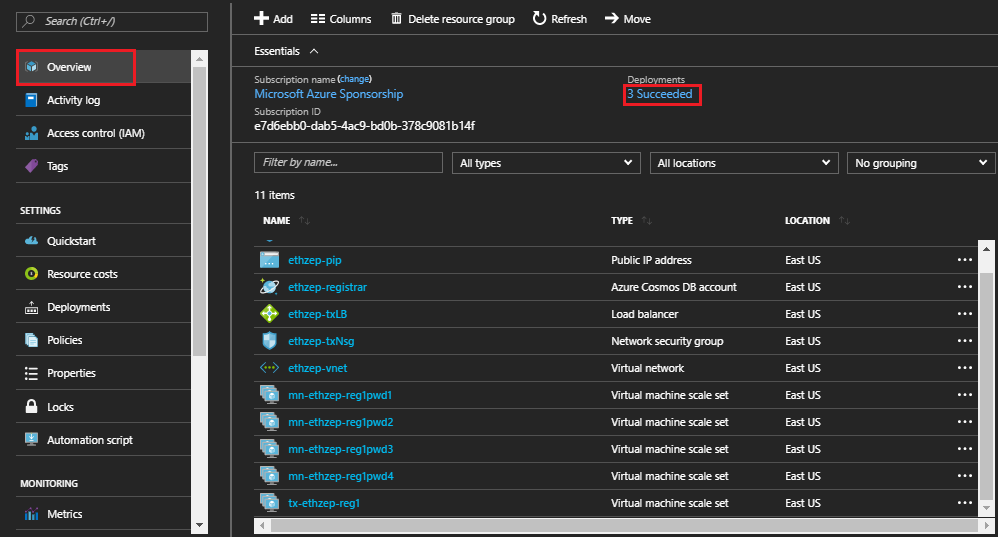
Finally, review legal and privacy terms and click ‘Purchase’ to deploy. Depending on the number of VMs being provisioned, deployment time can vary from a few minutes to tens of minutes.

# Post Deployment Sanity Checks

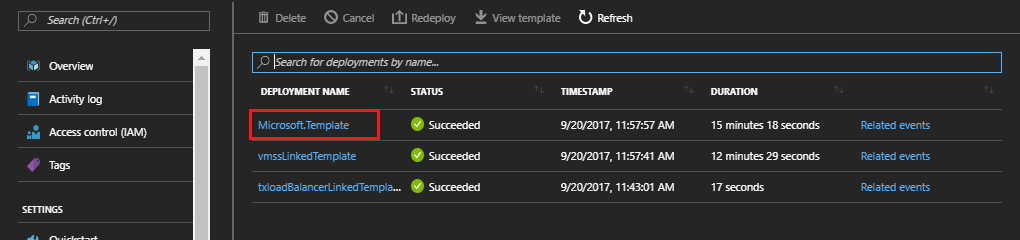
## Administrator page

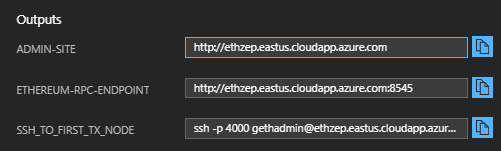
Once the deployment has completed successfully and all resources have been provisioned, you can go to the administrator page to get a simple view of your blockchain network.

The admin site URL is the DNS name of the load balancer; it is also the output of the template deployment. To find the template output, select the resource group just deployed. Select the Overview tab, then Last Deployment.

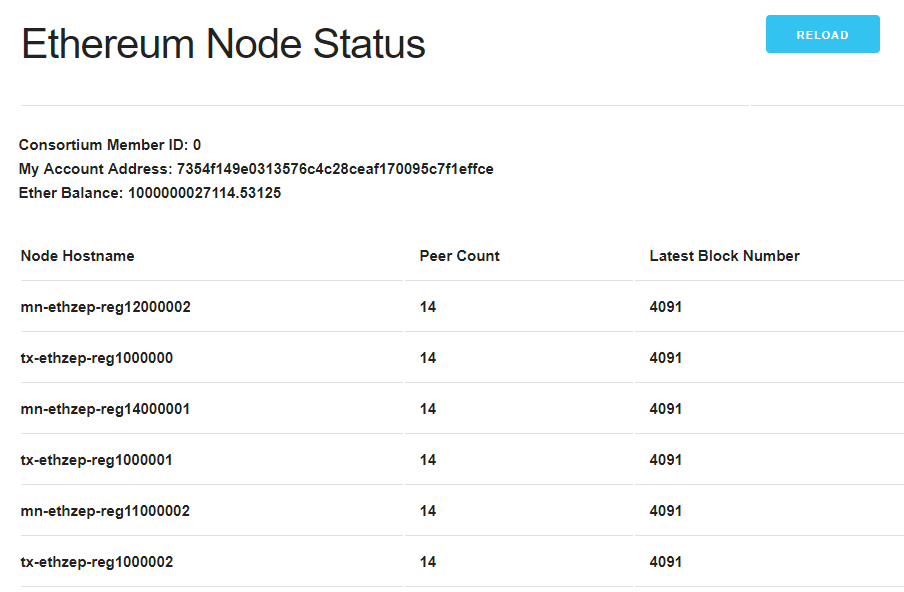


The new screen shows deployment history. Finally, select ‘microsoft-azure-blockchain.azure-blockchain.azure-blockchain-servi…’ and look for the outputs section. You’ll see the URL for the admin page listed in the output parameter named as ADMIN-SITE.





You can get a high-level overview of the topology you just deployed by reviewing the Ethereum Node Status section. This section includes all node hostnames and the participant to which the node belongs. It also displays node connectivity with the peer count. Peer count is the minimum of the number of mining nodes in the network and twenty-five where twenty-five is the configured maximum peer count, as in the public Ethereum network. Note, that peer count does not restrict the number of nodes that can be deployed within the network. Occasionally, you will see peer count fluctuate and be less for certain nodes. This is not always a sign that the nodes are unhealthy, since forks in the ledger can cause minor changes in peer count. Finally, you can inspect the latest block seen by each node in the network to determine forks or lags in the system.



The node status is refreshed every 10 seconds. Reload the page via the browser or "Reload" button to update the view.

## Accessing Nodes

You can remotely connect to one of the TX nodes from the VM scale sets on which the transaction nodes run via SSH with your provided admin username and password. Since the transaction node VMs do not have their own public IP addresses, you will need to go through the load balancer and specify the port number. The SSH command to run to access the first transaction node is in the template deployment output named as SSH\_TO\_FIRST\_TX\_NODE (for the sample deployment: ssh -p 4000 gethadmin@ethnet7tl.southeastasia.cloudapp.azure.com). To get to additional transaction nodes, increment the port number by one (e.g. the first transaction node is on port 4000, second, is 4001, third is 4002, etc.).

Since the VM scale sets on which the mining nodes run are not externally accessible, you will need to go through one of the transaction nodes. Once you have SSH’ed into a transaction node, install your private key on the transaction node and then SSH into any of the mining nodes.

**Note**

The hostnames can be obtained from Admin Site or from the Azure portal. In Azure portal, the hostnames of nodes present in the virtual machine scale set (VMSS) resource is listed under **Instances** which differs from the actual hostnames. For example, the hostname in Azure Portal may look like **mn-asdfmv-reg1ssh1\_0** but the actual hostname would be like **mn-asdfmv-reg11000000**

**Few examples**

**Azure Portal hostname Actual hostname**

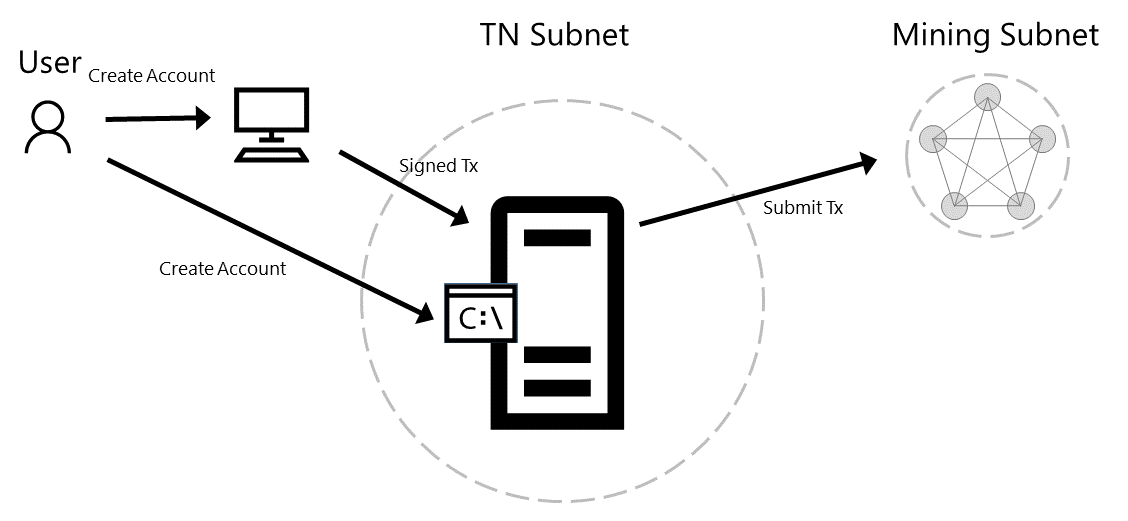
mn-ethwvu-reg1pwd2\_0 mn-ethwvu-reg12000000

mn-ethwvu-reg1ssh2\_1 mn-ethwvu-reg12000001

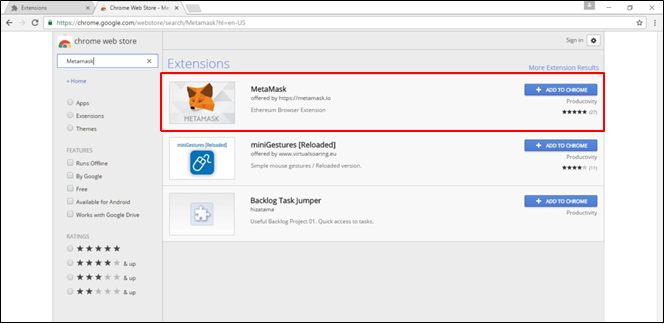
mn-ethwvu-reg1pwd1\_0 mn-ethwvu-reg11000001

# Create Ethereum Account

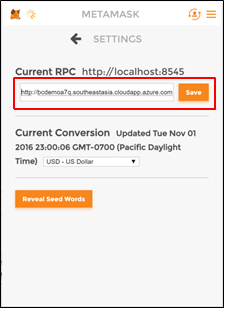
To create an additional account, you can use a variety of solutions. One such solution is MetaMask, a Chrome extension that provides an “identity vault” and connection to an Ethereum network, public, test or custom. MetaMask formulates a transaction to register the account in the network. This transaction, like any other transaction, will go to one of the transaction nodes, and eventually be mined into a block as illustrated below.



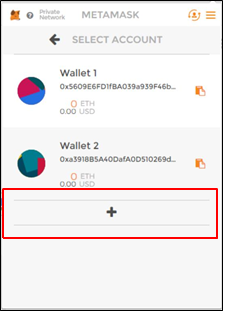
To install the extension in Chrome, go to Customize and control Google Chrome (Overflow button), More Tools, Extensions, Get More Extensions, and search for MetaMask.



Once installed, open MetaMask and create a new vault. By default, the vault will be connected to the Morden Test Network. You will need to change this to connect to the deployed private consortium network, specifically to the load balancer in front of the transaction nodes. From the template output, retrieve the exposed Ethereum RPC endpoint at port 8545 named as ETHEREUM-RPC-ENDPOINT, and enter it in custom RPC as shown below.

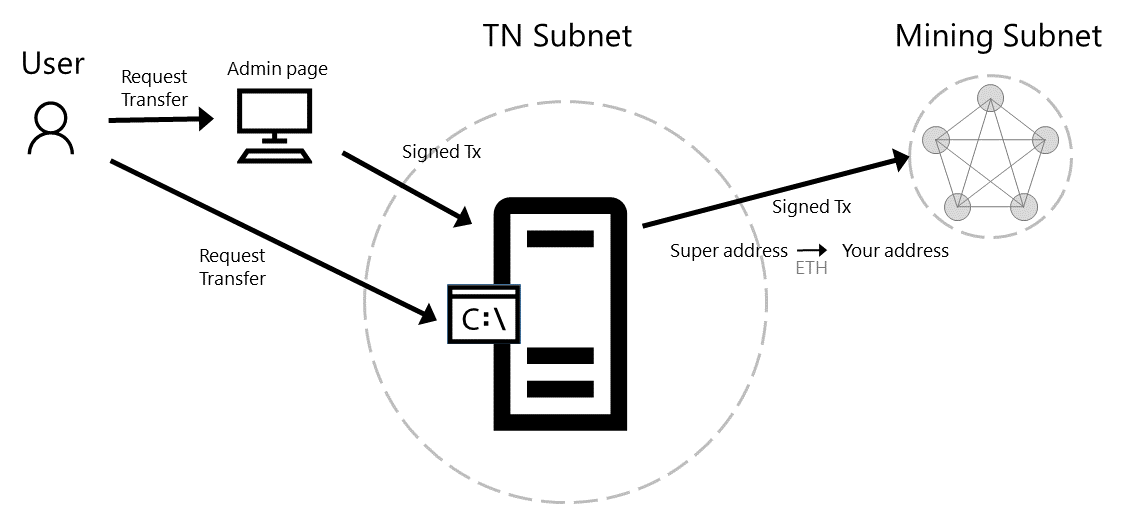


By creating the vault, you create a wallet containing an account. To create additional accounts, select Switch Accounts and then the ‘+’ button as shown below.

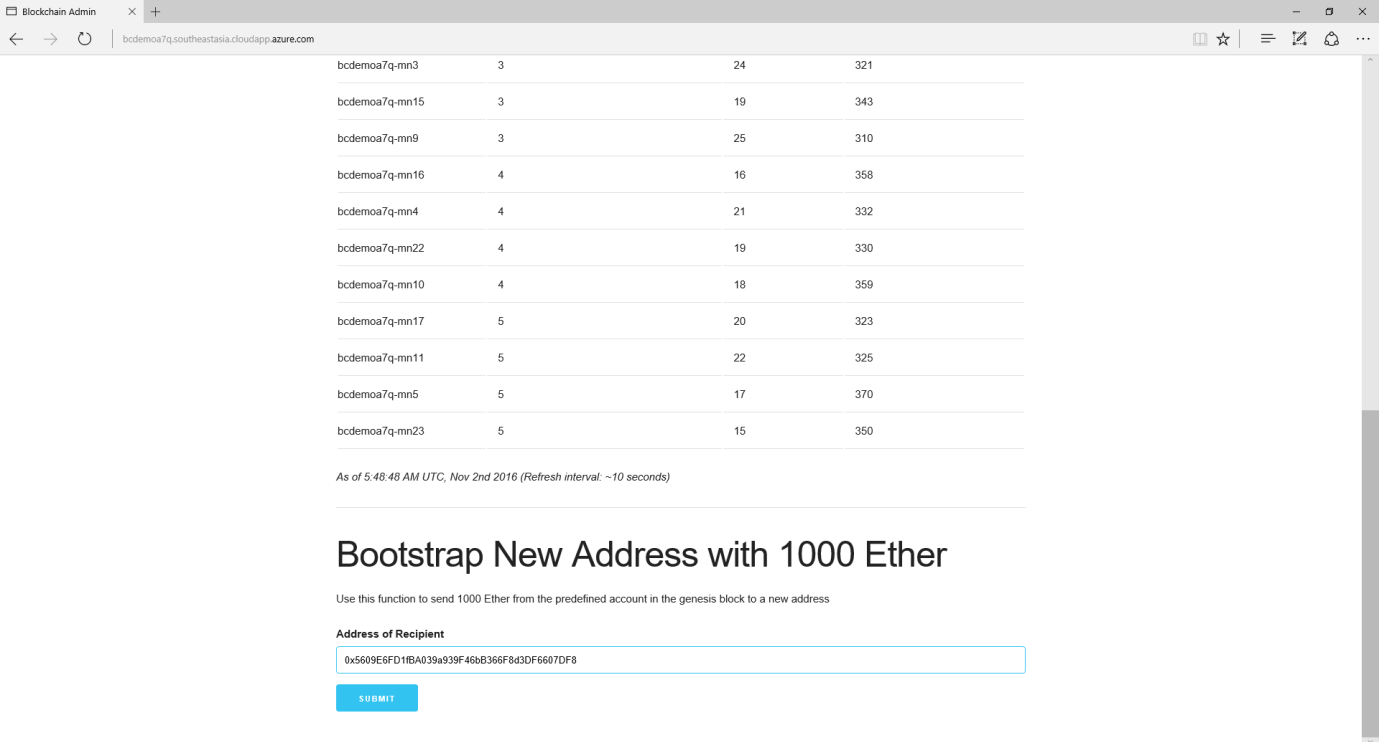


## Initiate Initial Ether Allocation

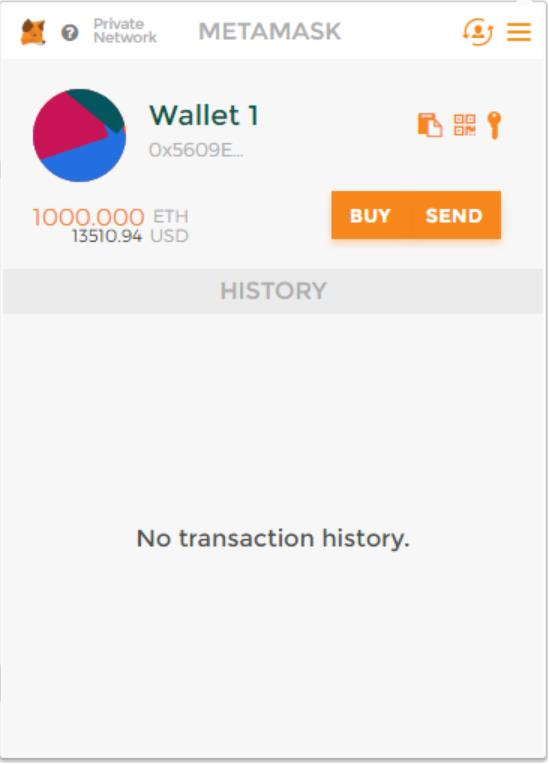
Through the administrator page, you can formulate a transaction to transfer Ether from the pre-allocated account to another Ethereum account. This Ether transfer is a transaction that is sent to the transaction node and mined into a block as illustrated below.



Via the clipboard icon in the MetaMask wallet, copy the address of the Ethereum account to which you want to transfer ether and go back to the administrator page. Paste the copied account into the input field to transfer 1000 ether from the pre-allocated Ethereum account to your newly created account. Click submit and wait for the transaction to be mined into a block.



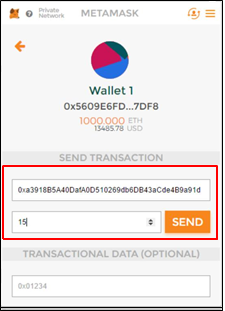
Once the transaction is committed into a mined block, the account balance in MetaMask for your account will reflect the transfer of 1000 Ether.



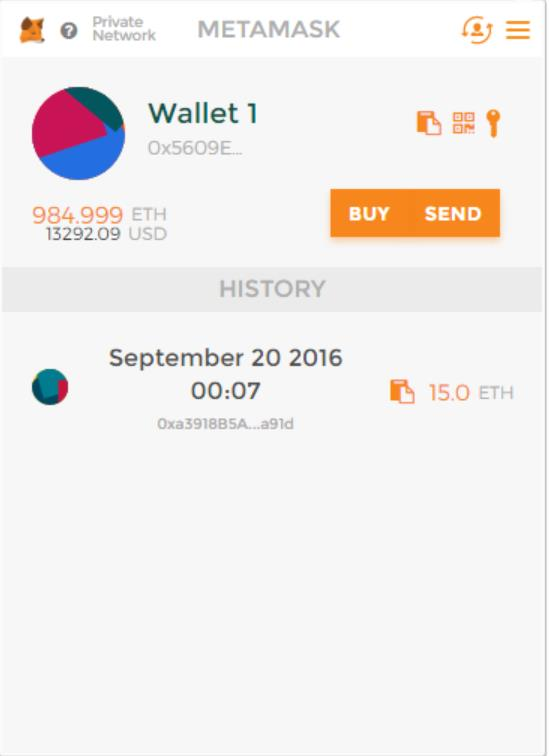
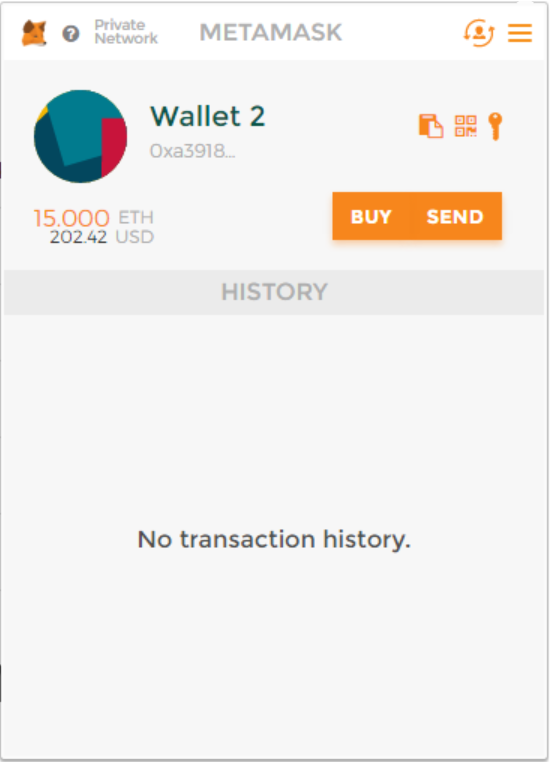
## Transfer of Ether between Accounts

At this point, you are ready to execute transactions within your private consortium network. The simplest transaction is to transfer Ether from one account to another. To formulate such a transaction, you can use MetaMask once again, transferring money from the first account used above to a second account.

From Wallet 1 in MetaMask, click on send. Copy the address of the second wallet created into Recipient Address input field and amount of Ether to transfer in the Amount input field. Click send and accept the transaction.



Once again, when the transaction is mined and committed into a block, the account balances will be reflected accordingly. Note, wallet 1’s balance is deducted a bit more than 15 Ether, since you had to pay a mining fee to process the transaction.

# Next Steps

You are now ready to focus on application and smart contract development against your private consortium blockchain network. Happy coding!