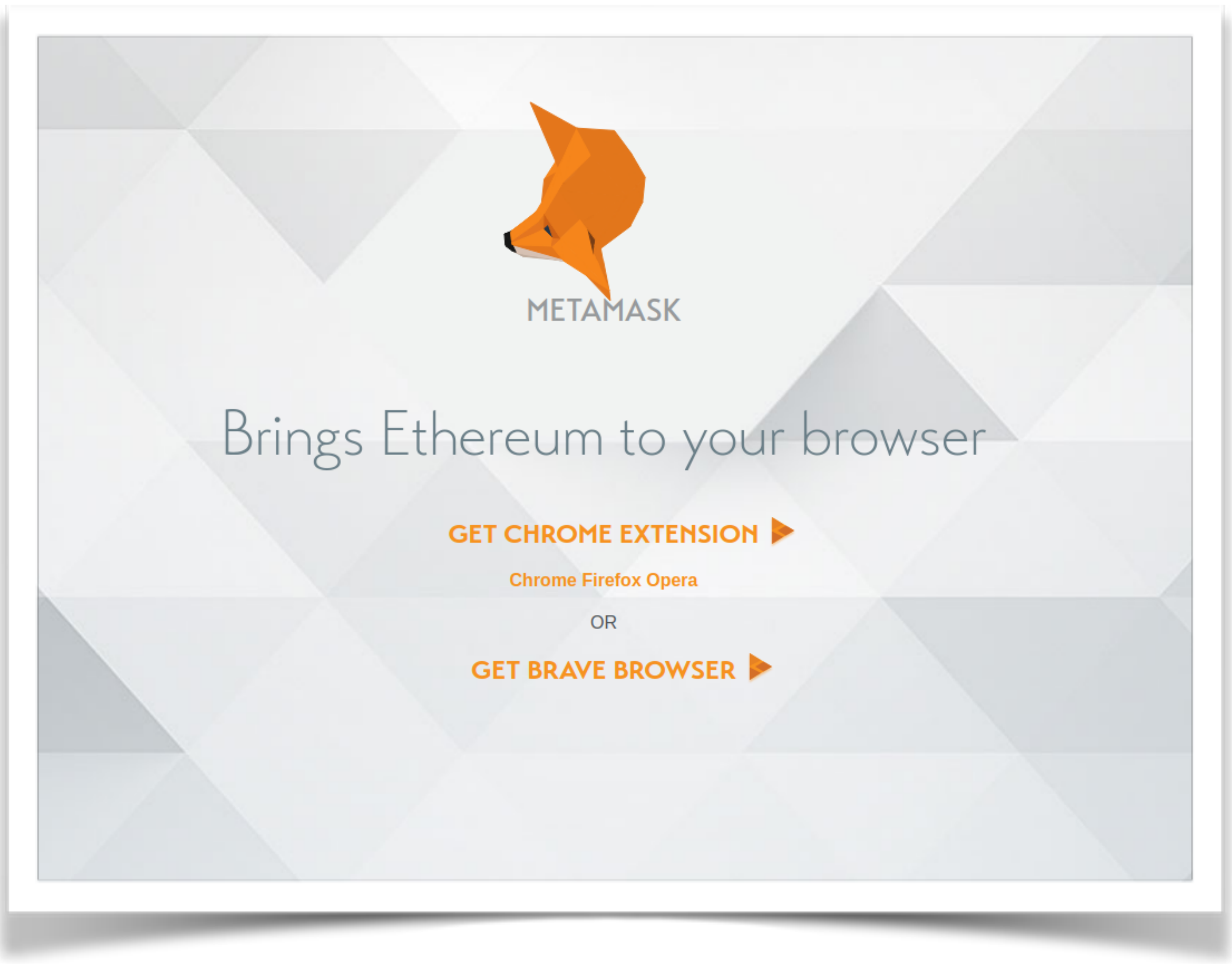
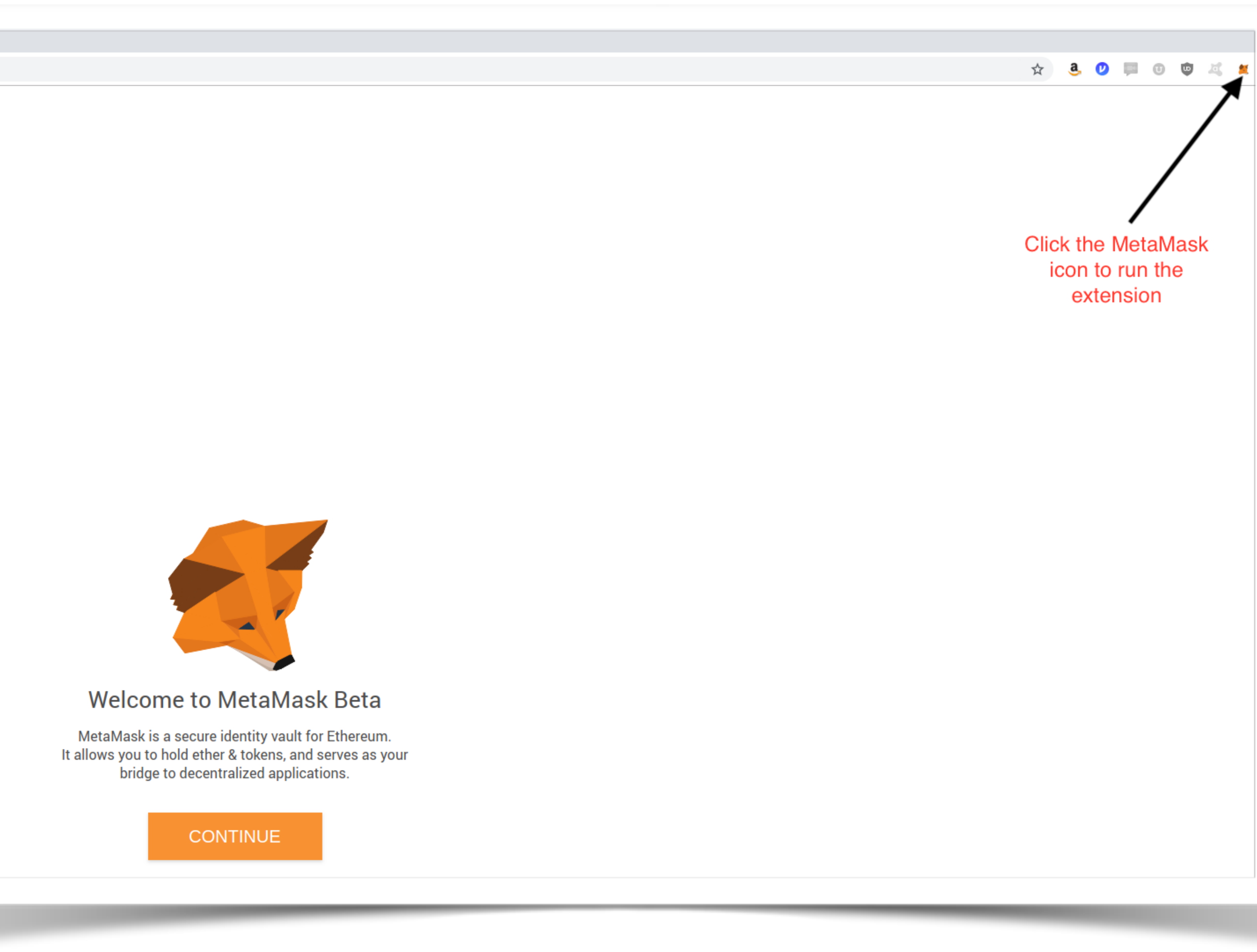
How to Configure Metamask and Remix to Transact with the NSA Codebreaker Challenge Blockchain

Step 1: Go to http://metamask.io and either install one of the browser extensions or the Brave browser

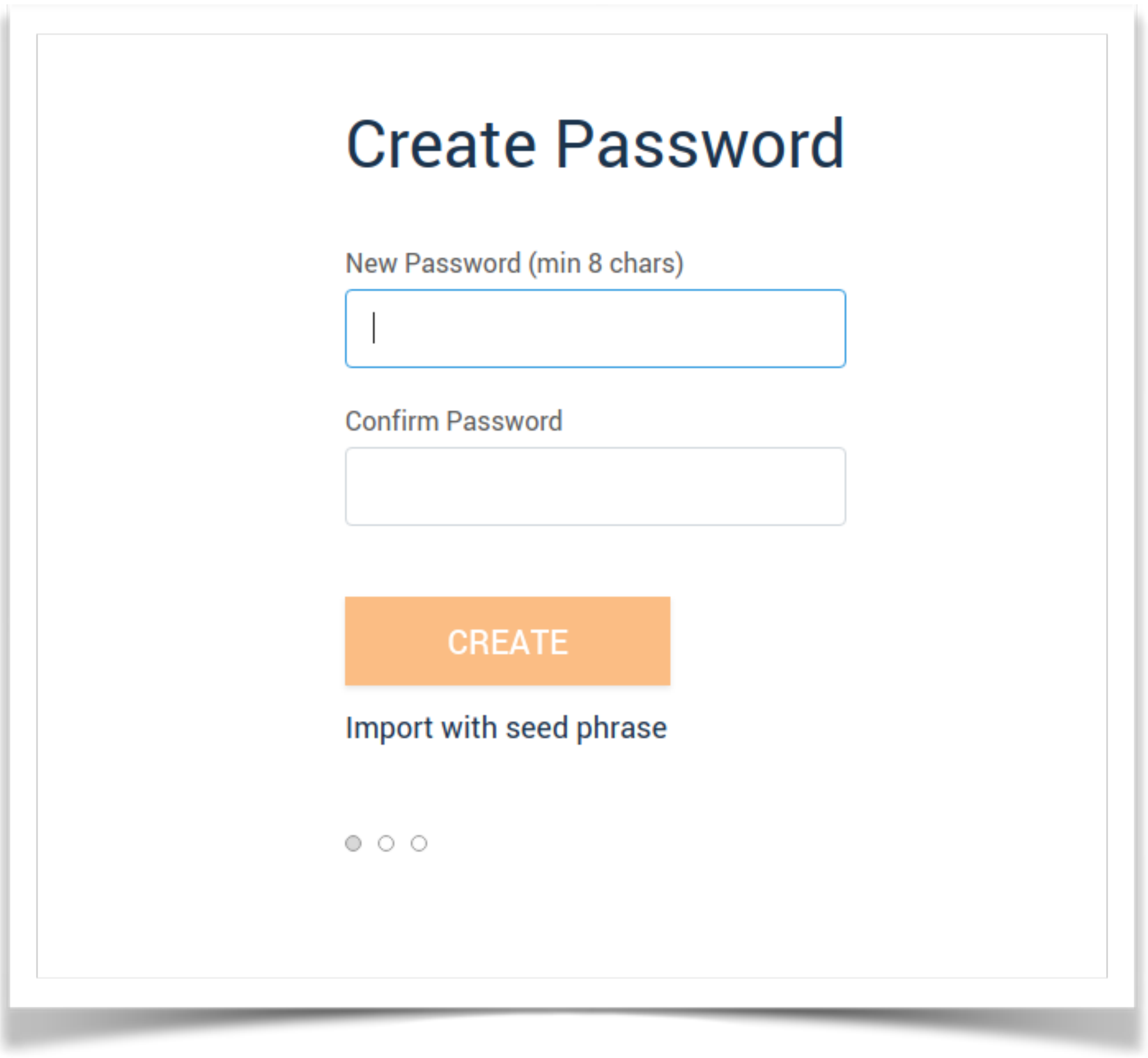


Step 2: Click the Metamask icon in your browser. The latest version will ask if you'd like to try the new interface. These instructions assume you choose yes.

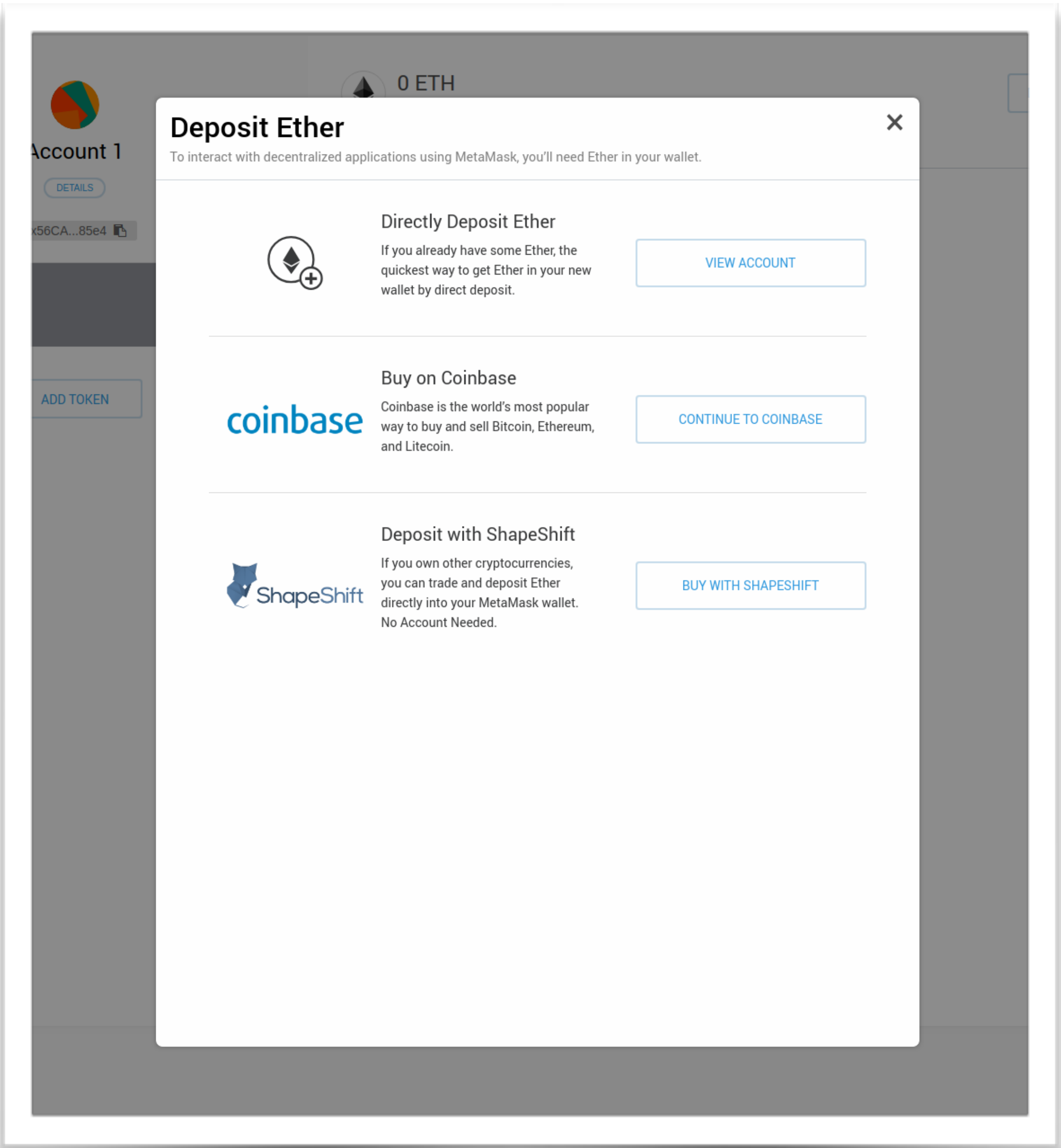


MetaMask is a secure identity vault for Ethereum. It allows you to hold ether & tokens, and serves as your bridge to decentralized applications.

Step 3: Choose a password to protect your keys. After you click create, you will need to copy a secret backup phrase and re-enter it in Metamask for verification.



Step 4: Close this window and ignore it.



Account 1

DETAILS

x56CA...85e4

ADD TOKEN

0 ETH

Deposit Ether

To interact with decentralized applications using MetaMask, you'll need Ether in your wallet.

Directly Deposit Ether

If you already have some Ether, the quickest way to get Ether in your new wallet by direct deposit.

VIEW ACCOUNT

Buy on Coinbase

Coinbase is the world's most popular way to buy and sell Bitcoin, Ethereum, and Litecoin.

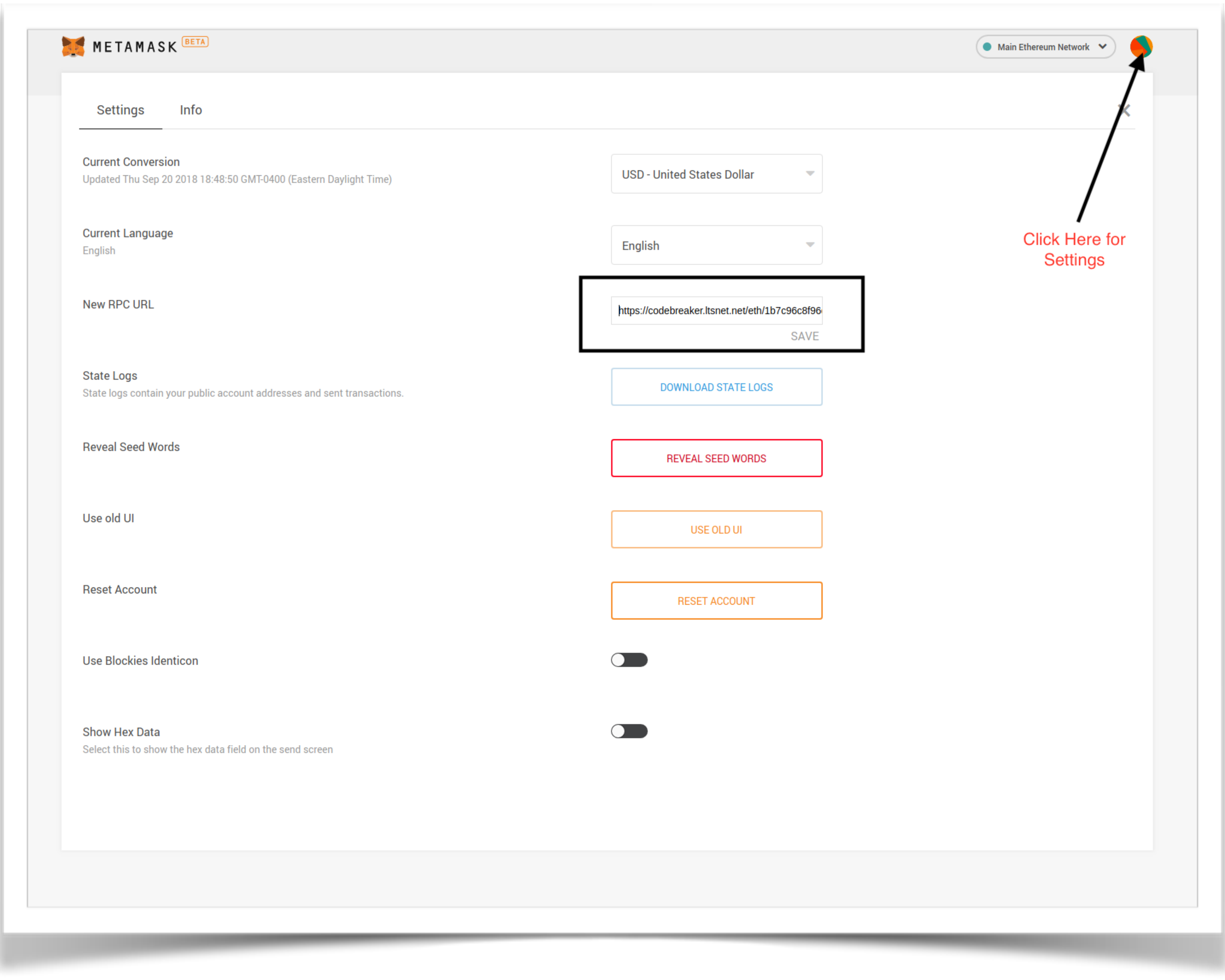
CONTINUE TO COINBASE

Deposit with ShapeShift

If you own other cryptocurrencies, you can trade and deposit Ether directly into your MetaMask wallet. No Account Needed.

BUY WITH SHAPESHIFT

Step 5: Click the icon shown in the picture and select ‘Settings’. You will enter the URL provided in blockchain_information.txt where it says ‘New RPC URL’.



Current Conversion
Updated Thu Sep 20 2018 18:48:50 GMT-0400 (Eastern Daylight Time)

Current Language
English

New RPC URL
https://codebreaker.ltsnet.net/eth/1b7c96c8f96

State Logs
State logs contain your public account addresses and sent transactions.

Reveal Seed Words

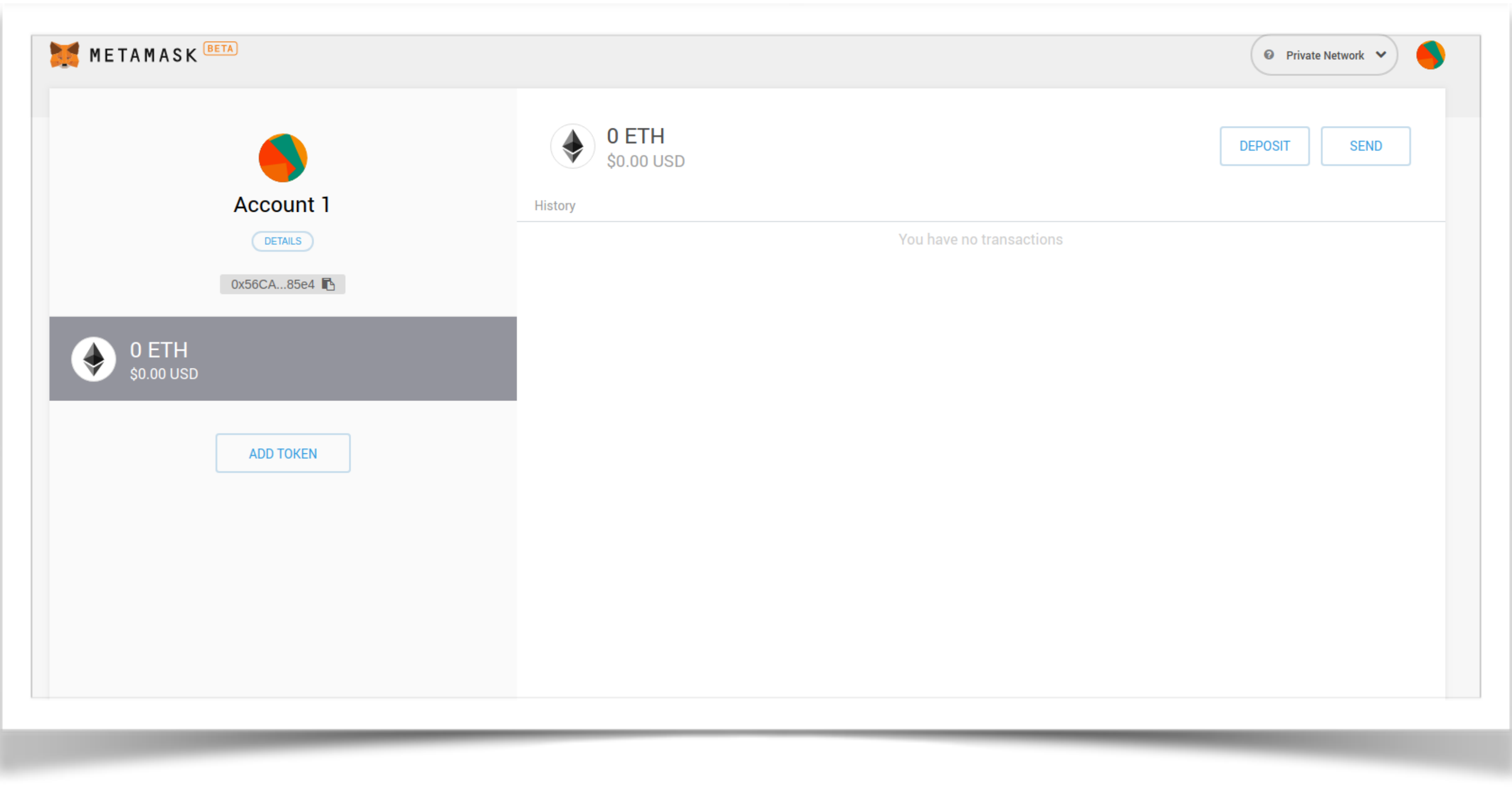
Use old UI

Reset Account

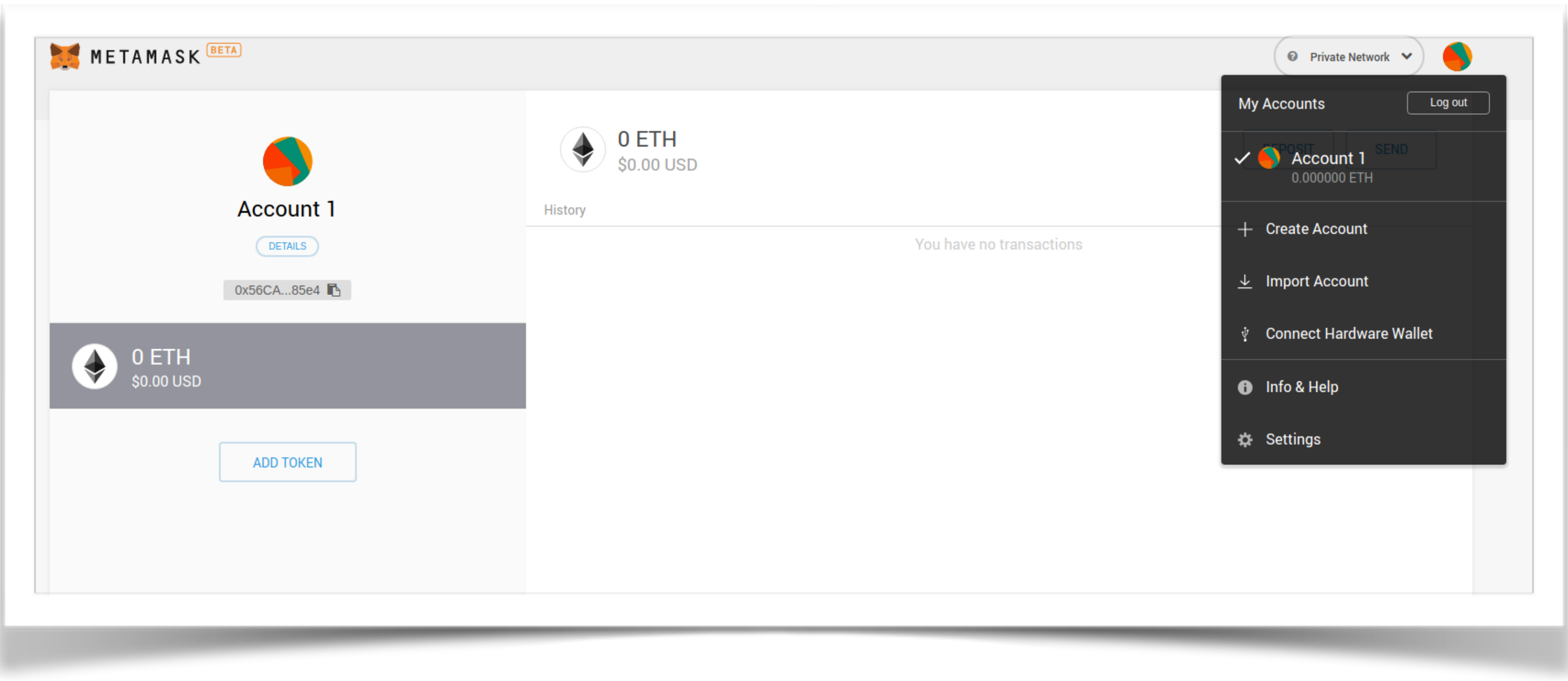
Use Blockies Identicon

Show Hex Data
Select this to show the hex data field on the send screen

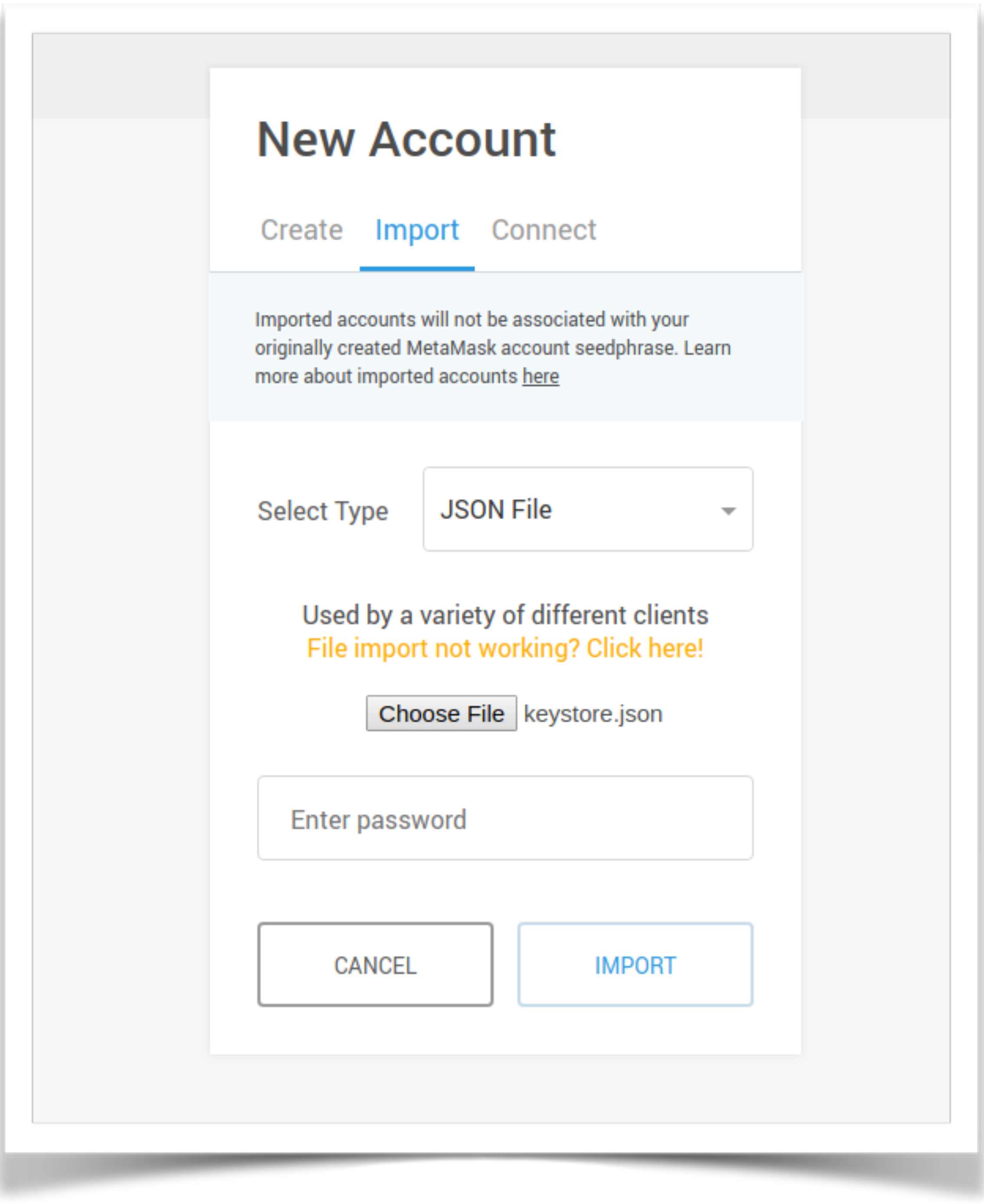
Step 6: Verify that Metamask shows you are connected to a “Private Network” as shown below.



Step 7: Click the same icon as you did to access settings, but now select ‘ Import Account’.



Step 8: Choose ‘JSON File’ in the ‘Select Type’ drop-down box, choose the keystore.json file that you downloaded from the Codebreaker Challenge website, and enter the password in the text box.



New Account

Imported accounts will not be associated with your originally created MetaMask account seedphrase. Learn more about imported accounts here

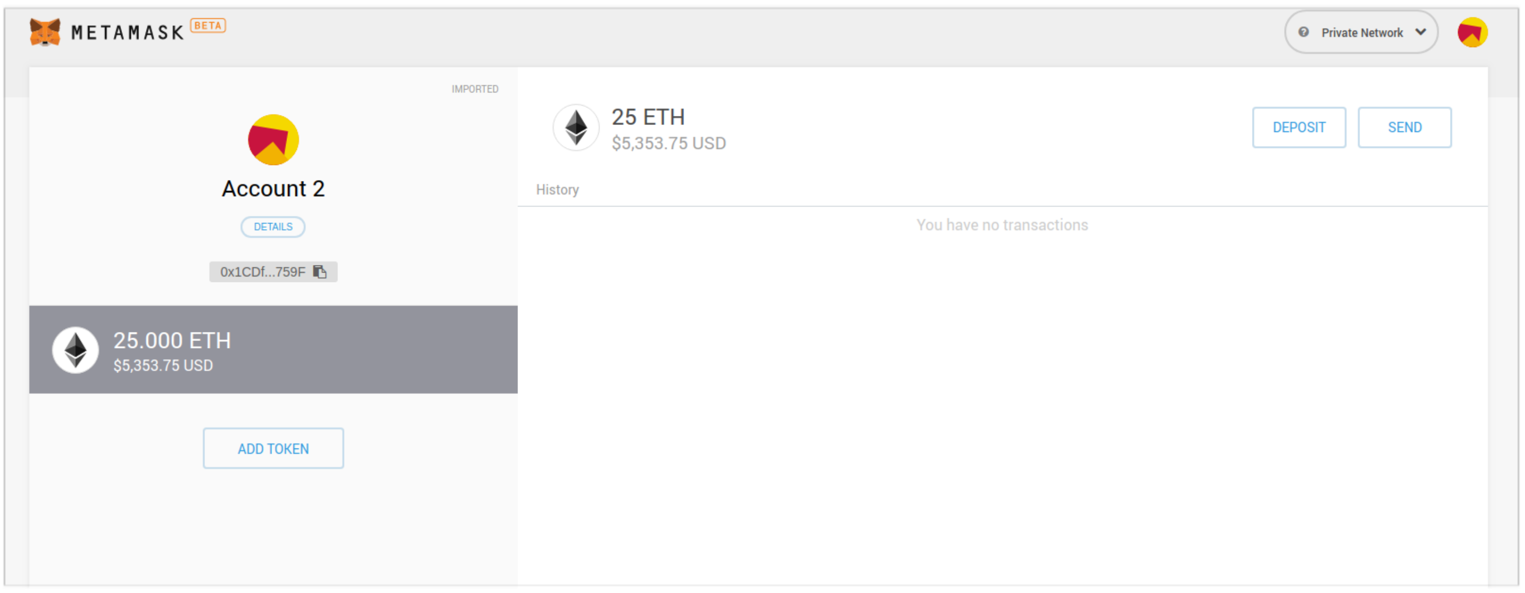
Select Type: JSON File

Used by a variety of different clients
File import not working? Click here!

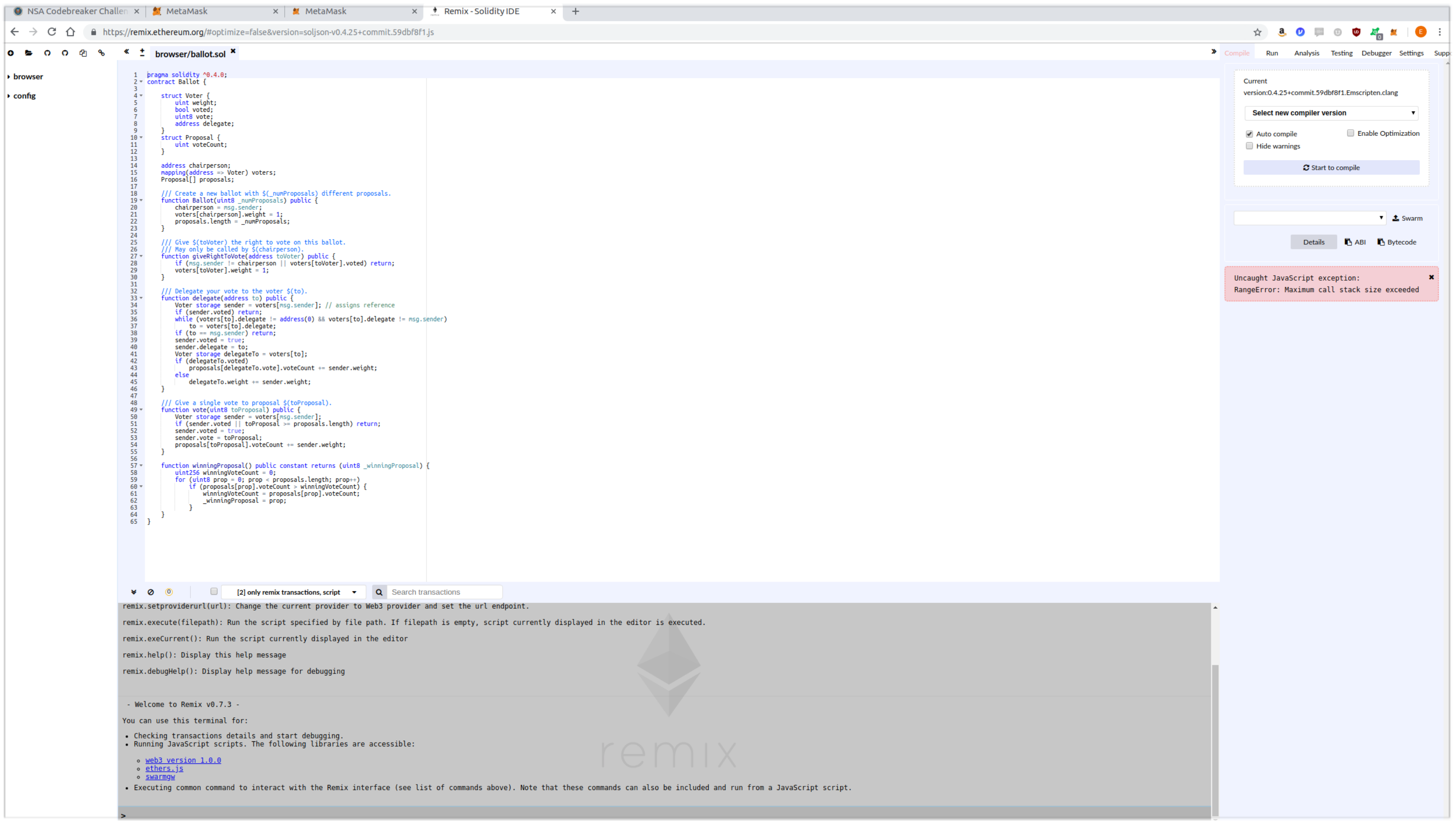
keystore.json

Step 9: You should now see your imported account in Metamask with a 25 ETH balance. Congratulations, your account is now ready to use!

Next we will configure Remix to use your account through Metamask.



Step 10: Go to http://remix.ethereum.org. When it first loads, you'll see a screen like the below with a sample 'ballot' Solidity file.



```
pragma solidity ^0.4.0;
contract Ballot {

    struct Voter {
        uint weight;
        bool voted;
        uint8 vote;
        address delegate;
    }
    struct Proposal {
        uint voteCount;
    }

    address chairperson;
    mapping(address => Voter) voters;
    Proposal[] proposals;

    /// Create a new ballot with $(_numProposals) different proposals.
    function Ballot(uint8 _numProposals) public {
        chairperson = msg.sender;
        voters[chairperson].weight = 1;
        proposals.length = _numProposals;
    }

    /// Give $(toVoter) the right to vote on this ballot.
    /// May only be called by $(chairperson).
    function giveRightToVote(address toVoter) public {
        if (msg.sender != chairperson || voters[toVoter].voted) return;
        voters[toVoter].weight = 1;
    }

    /// Delegate your vote to the voter $(to).
    function delegate(address to) public {
        Voter storage sender = voters[msg.sender]; // assigns reference
        if (sender.voted) return;
        while (voters[to].delegate != address(0) && voters[to].delegate != msg.sender)
            to = voters[to].delegate;
        if (to == msg.sender) return;
        sender.voted = true;
        sender.delegate = to;
        Voter storage delegateTo = voters[to];
        if (delegateTo.voted)
            proposals[delegateTo.vote].voteCount += sender.weight;
        else
            delegateTo.weight += sender.weight;
    }

    /// Give a single vote to proposal $(toProposal).
    function vote(uint8 toProposal) public {
        Voter storage sender = voters[msg.sender];
        if (sender.voted || toProposal >= proposals.length) return;
        sender.voted = true;
        sender.vote = toProposal;
        proposals[toProposal].voteCount += sender.weight;
    }

    function winningProposal() public constant returns (uint8 _winningProposal) {
        uint256 winningVoteCount = 0;
        for (uint8 prop = 0; prop < proposals.length; prop++)
            if (proposals[prop].voteCount > winningVoteCount) {
                winningVoteCount = proposals[prop].voteCount;
                _winningProposal = prop;
            }
    }
}
```

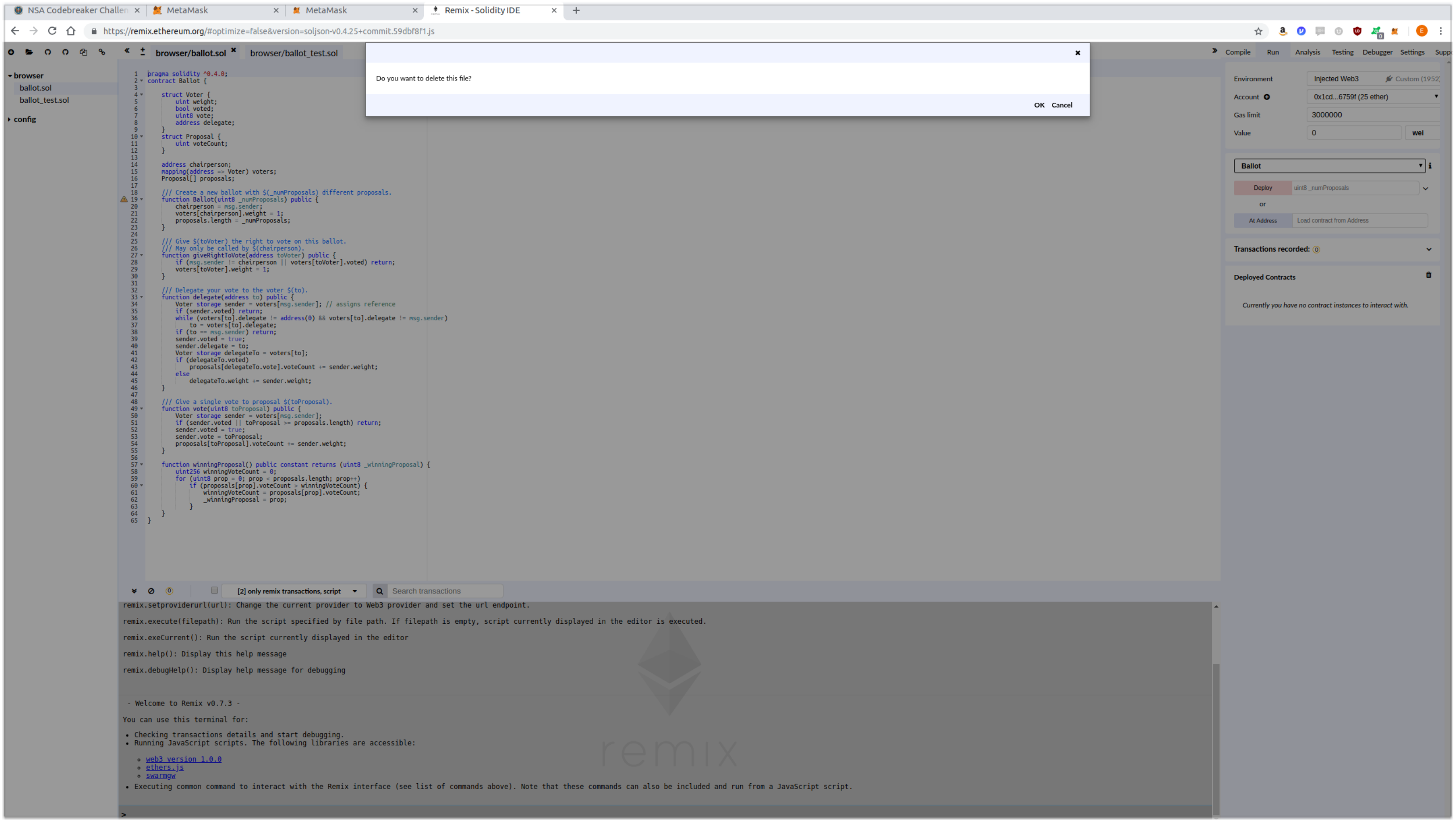
Current
version:0.4.25+commit.59dbf8f1.Emscripten.clang

[✓] Auto compile
[ ] Enable Optimization
[ ] Hide warnings

```
Uncaught JavaScript exception:
RangeError: Maximum call stack size exceeded
```

```
remix.setproviderurl(url): Change the current provider to Web3 provider and set the url endpoint.

remix.execute(filepath): Run the script specified by file path. If filepath is empty, script currently displayed in the editor is executed.

remix.exeCurrent(): Run the script currently displayed in the editor

remix.help(): Display this help message

remix.debugHelp(): Display help message for debugging


 - Welcome to Remix v0.7.3 -

You can use this terminal for:

• Checking transactions details and start debugging.
• Running JavaScript scripts. The following libraries are accessible:

  ○ web3 version 1.0.0
  ○ ethers.js
  ○ swarmgw
• Executing common command to interact with the Remix interface (see list of commands above). Note that these commands can also be included and run from a JavaScript script.

>
```

Step 11: Expand the ‘browser’ explorer in the left column and delete both *.sol files.



https://remix.ethereum.org/#optimize=false&version=soljson-v0.4.25+commit.59dbf8f1.js

browser/ballot.sol
browser/ballot_test.sol

Do you want to delete this file?

OK Cancel

```
pragma solidity ^0.4.0;
contract Ballot {

    struct Voter {
        uint weight;
        bool voted;
        uint8 vote;
        address delegate;
    }
    struct Proposal {
        uint voteCount;
    }

    address chairperson;
    mapping(address => Voter) voters;
    Proposal[] proposals;

    /// Create a new ballot with $(_numProposals) different proposals.
    function Ballot(uint8 _numProposals) public {
        chairperson = msg.sender;
        voters[chairperson].weight = 1;
        proposals.length = _numProposals;
    }

    /// Give $(toVoter) the right to vote on this ballot.
    /// May only be called by $(chairperson).
    function giveRightToVote(address toVoter) public {
        if (msg.sender != chairperson || voters[toVoter].voted) return;
        voters[toVoter].weight = 1;
    }

    /// Delegate your vote to the voter $(to).
    function delegate(address to) public {
        Voter storage sender = voters[msg.sender]; // assigns reference
        if (sender.voted) return;
        while (voters[to].delegate != address(0) && voters[to].delegate != msg.sender)
            to = voters[to].delegate;
        if (to == msg.sender) return;
        sender.voted = true;
        sender.delegate = to;
        Voter storage delegateTo = voters[to];
        if (delegateTo.voted)
            proposals[delegateTo.vote].voteCount += sender.weight;
        else
            delegateTo.weight += sender.weight;
    }

    /// Give a single vote to proposal $(toProposal).
    function vote(uint8 toProposal) public {
        Voter storage sender = voters[msg.sender];
        if (sender.voted || toProposal >= proposals.length) return;
        sender.voted = true;
        sender.vote = toProposal;
        proposals[toProposal].voteCount += sender.weight;
    }

    function winningProposal() public constant returns (uint8 _winningProposal) {
        uint256 winningVoteCount = 0;
        for (uint8 prop = 0; prop < proposals.length; prop++)
            if (proposals[prop].voteCount > winningVoteCount) {
                winningVoteCount = proposals[prop].voteCount;
                _winningProposal = prop;
            }
    }
}
```

```
remix.setproviderurl(url): Change the current provider to Web3 provider and set the url endpoint.

remix.execute(filepath): Run the script specified by file path. If filepath is empty, script currently displayed in the editor is executed.

remix.exeCurrent(): Run the script currently displayed in the editor

remix.help(): Display this help message

remix.debugHelp(): Display help message for debugging


 - Welcome to Remix v0.7.3 -

You can use this terminal for:

• Checking transactions details and start debugging.
• Running JavaScript scripts. The following libraries are accessible:

    ◦ web3 version 1.0.0
    ◦ ethers.js
    ◦ swarmgw
• Executing common command to interact with the Remix interface (see list of commands above). Note that these commands can also be included and run from a JavaScript script.

>
```

Environment: Injected Web3 Custom (1952)
Account: 0x1cd...6759f (25 ether)
Gas limit: 3000000
Value: 0 wei

Ballot

Deploy uint8 _numProposals

or

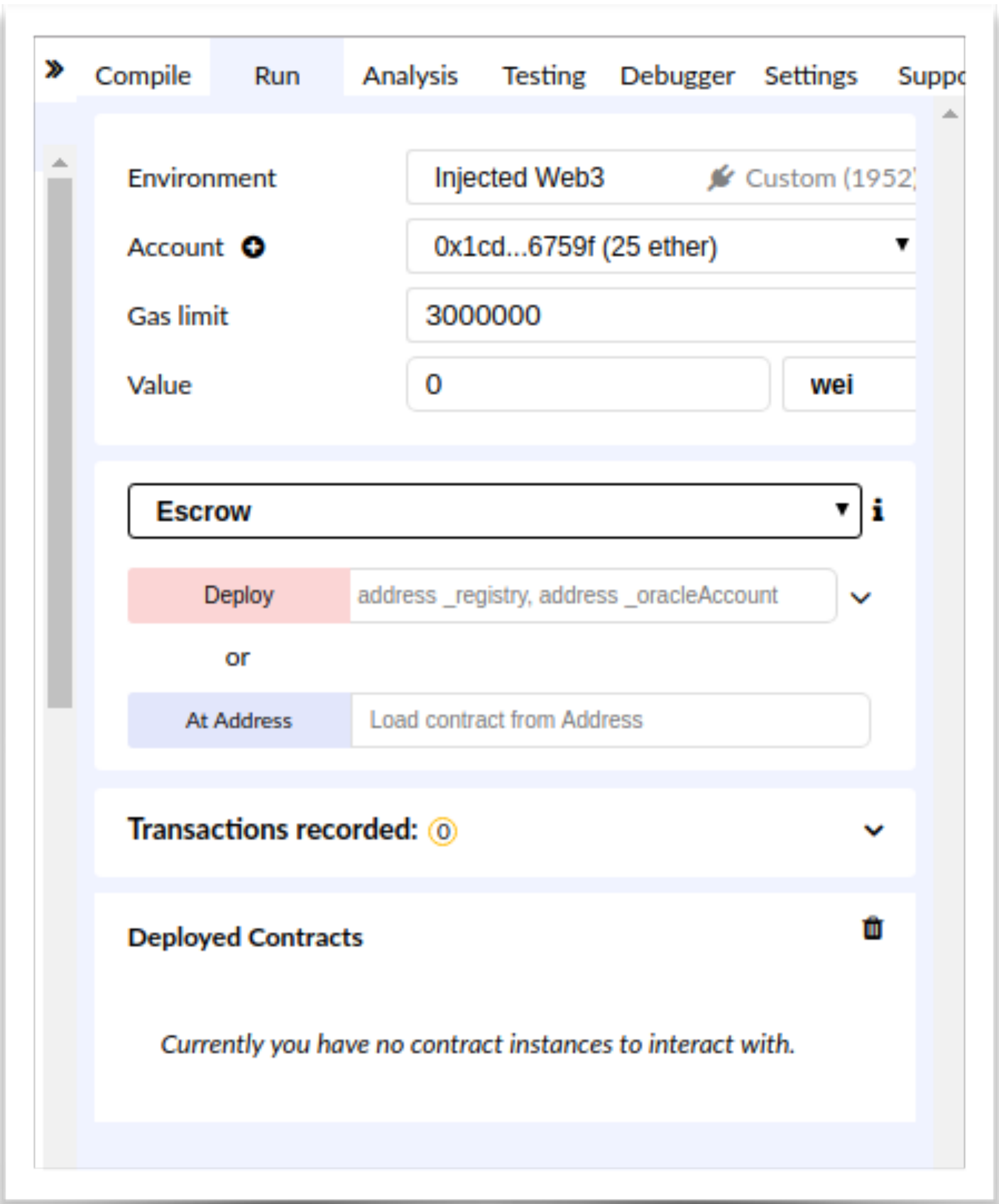
At Address Load contract from Address

Transactions recorded: 0

Deployed Contracts

Currently you have no contract instances to interact with.

Step 12: On the right-hand side of your browser window, select the ‘Run’ tab and make sure ‘Environment’ is set to ‘Injected Web3’. This is how Remix will interact with Metamask. You should now see your account listed.


Compile
Run
Analysis
Testing
Debugger
Settings
Suppo

Environment	Injected Web3 Custom (1952
Account	0x1cd...6759f (25 ether)
Gas limit	3000000
Value	0 wei

Escrow

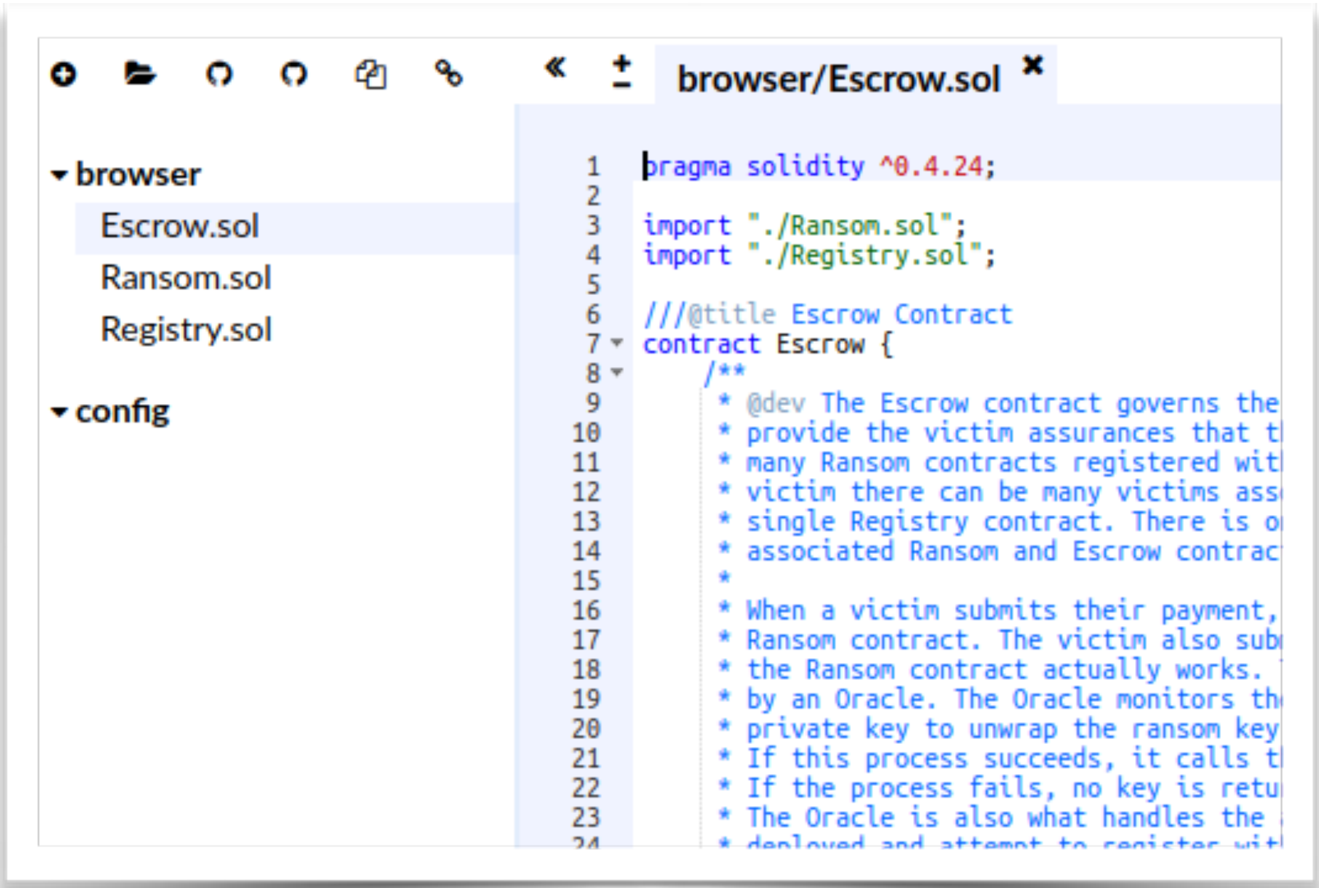
Deploy
address _registry, address _oracleAccount
or
At Address
Load contract from Address

Transactions recorded: 0

Deployed Contracts

Currently you have no contract instances to interact with.

Step 13: Back in the browser explorer, click the icon that looks like an open file and add the Escrow, Ransom, and Registry smart contracts.



```
pragma solidity ^0.4.24;

import "./Ransom.sol";
import "./Registry.sol";

///@title Escrow Contract
contract Escrow {
    /**
     * @dev The Escrow contract governs the
     * provide the victim assurances that t
     * many Ransom contracts registered wit
     * victim there can be many victims ass
     * single Registry contract. There is o
     * associated Ransom and Escrow contrac
     *
     * When a victim submits their payment,
     * Ransom contract. The victim also sub
     * the Ransom contract actually works.
     * by an Oracle. The Oracle monitors th
     * private key to unwrap the ransom key
     * If this process succeeds, it calls t
     * If the process fails, no key is retu
     * The Oracle is also what handles the
     * deployed and attempt to register wit
```

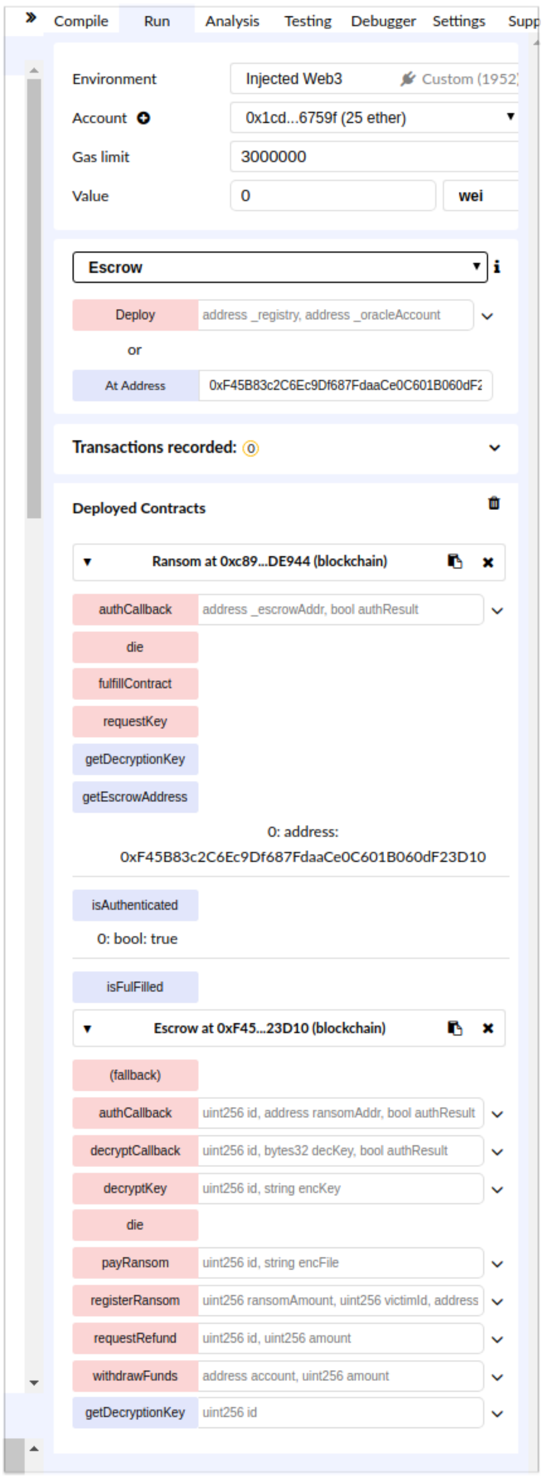
Step 14: Once you've added the smart contracts to your browser, click the Compile tab and make sure the contracts compiled (if auto-compile is turned on). Otherwise click 'start to compile'.

Next click the Run tab and select Escrow. In the 'At Address' text box, paste the Escrow contract address you were given and click the 'At Address' button. You should see all the Escrow functions become visible under Deployed Contracts.

Now do the same for the Ransom contract. Your screen should look similar to the image on the next page.

The blue boxed functions can be executed for 'free' and do not require a transaction. The red boxed functions change the contract state and require a transaction to be sent.

As a test, click 'isAuthenticated' and 'getEscrowAddress' on the Ransom contract. It should report 'true' and the Escrow address should match the value you already entered.



Environment Injected Web3 Custom (1952

Account 0x1cd...6759f (25 ether)

Gas limit 3000000

Value 0 wei

Escrow

Deploy address _registry, address _oracleAccount

or

At Address 0xF45B83c2C6Ec9Df687FdaaCe0C601B060dF2

Transactions recorded: 0

Deployed Contracts

Ransom at 0xc89...DE944 (blockchain)

authCallback address _escrowAddr, bool authResult

0: address: 0xF45B83c2C6Ec9Df687FdaaCe0C601B060dF23D10

isAuthenticated

0: bool: true

isFulFilled

Escrow at 0xF45...23D10 (blockchain)

authCallback uint256 id, address ransomAddr, bool authResult

decryptCallback uint256 id, bytes32 decKey, bool authResult

decryptKey uint256 id, string encKey

payRansom uint256 id, string encFile

registerRansom uint256 ransomAmount, uint256 victimId, address

requestRefund uint256 id, uint256 amount

withdrawFunds address account, uint256 amount

getDecryptionKey uint256 id

You are now ready to use Remix + Metamask to transact with the Codebreaker Challenge blockchain. There are many other options besides these two tools, but this is an easy way to get started.

Note that Remix can also be used to write and deploy your own smart contracts. There is a javascript console at the bottom of the screen that can be used to run web3js commands too. For more help, check out the Resources page on the codebreaker.ltsnet.net website.

Good luck and have fun with the challenge!

- NSA Codebreaker Team