Open call for contributions by Truffle + PegaSys: EEA private transactions¶

What's missing?¶

In its current state, Truffle doesn't support EEA private transactions. Support for the web3js-eea library is needed for this to work and PegaSys has recently made an implementation of the v4.0 EEA specification.

Why did this happen?¶

Theinterface-adapter in Truffle's codebase doesn't provide an adapter for the EEA 4.0 client spec, therefore private transactions aren't working.

The EEA client specification is a developing specification and Truffle has not yet implemented the EEA client specification around private transactions. PegaSys is providing an implementation of the specification for private transactions. Pegasys and Truffle are teaming up to ensure that Truffle supports the EEA JSON RPC API methods.

What's the solution?¶

PegaSys and Truffle have teamed up to make a single open call for contributions. This guest blog post is made by the PegaSys team and is to be posted on the Truffle blog and promoted by both teams in order to add code to Truffle codebase to add support for them.

The solution involves using a Delegation pattern to help make that work.

The location of the files to be modified can be found heretruffle/packages/interface-adapter/lib

In short, when Truffle receives a transaction, it should determine if its aprivateFor transaction. If so, it would delegate the rest of the business logic to the EEA library. If not, it should just continue processing the transaction with the original web3 methods.

Also, it would be very useful to make sure this plays nicely with the getTransactionReceipt methods later in the chain of events.

What to do Exactly?¶

The initial process involves adding an EEA definition to the shim file. Theweb3-shim.ts file currently contains the overloads for the different interface definitions and their mappings.

import

{

EthereumDefinition

}

from

"./ethereum-overloads" ; import

{

QuorumDefinition

}

from

"./quorum-overloads" ; import

{

FabricEvmDefinition

}

from

"./fabric-evm-overloads" ; const

initInterface

=

async (web3Shim :

Web3Shim)

=>

{

const

networkTypes :

NetworkTypesConfig

=

new

Map (Object . entries ({

"ethereum" :

EthereumDefinition ,

"quorum" :

QuorumDefinition ,

"fabric-evm" :

FabricEvmDefinition

}));

networkTypes . get (web3Shim . networkType). initNetworkType (web3Shim);

} The initial import and mapping has been done in this fork .

import

{

EthereumDefinition

}

from

"./ethereum-overloads" ; import

{

EEADefinition

}

from

"./EEA-overloads" ; import

{

QuorumDefinition

}

from

"./quorum-overloads" ; import

{

FabricEvmDefinition

}

from

"./fabric-evm-overloads" ; const

initInterface

=

async (web3Shim :

Web3Shim)

=>

{

const

networkTypes :

NetworkTypesConfig

=

new

Map (Object . entries ({

"ethereum" :

EthereumDefinition ,

"EEA" :

EEADefinition ,

"quorum" :

QuorumDefinition ,

"fabric-evm" :

FabricEvmDefinition

}));

networkTypes . get (web3Shim . networkType). initNetworkType (web3Shim);

} Therest of the owl is to add the needed logic in theeea-overloads.ts file and theoverrides variable for it to interact with theweb3eeajs library .

Who can do this?¶

Anyone who'd like to try can very much do so we will gladly support any developer who wants to help out. But here are some recommended prerequisites:

The Truffle codebase is mainly written injavascript , and this particular interface-adapter library we'll be looking at is written intypescript . So knowing some javascript is a must.

It will definitely help if you've had some exposure and usage of the Truffle library. Nothing fancy, just using it for compiling and deploying contracts for example. Likewise, having some previous knowledge of the web3js library is going to prove quite helpful.

All of the above somewhat describes a general dapp developer- so in other words; if you've already written a dapp before using Truffle, this contribution could be done by you.

So what's in it for you? Two things: - You'll receive some 😎 PegaSys Swag! - We'll personally thank and credit you on stage at Trufflecon at my Permissioning Talk

For more information or help on this contribution, contact Felipe from PegaSys , or write us on our gitter channel .