A Sweet Upgradeable Contract Experience with OpenZeppelin and Truffle

Immutability is a cornerstone of blockchain technology. This is great for many reasons, but presents a problem when it's time to upgrade our smart contracts. There are a few different upgrade patterns, but one that's proven to be robust is the proxy pattern. This pattern has, until now, involved some additional overhead for development teams. Thanks to OpenZeppelin though, you can now deploy upgradeable contract systems with ease using the familiar Truffle tool suite!

The Proxy Pattern¶

At a high level, the proxy upgrade pattern involves deploying a proxy contract that delegates function calls to your logic and storage contracts. The proxy is storing addresses of the logic contracts and these addresses can be changed. This allows you to deploy a new version of the logic contract and point the proxy to that new version.

The proxy contract can point to any number of different logic contracts. There are many concerns around this pattern you'll need to keep in mind, such as making sure your old contracts cannot be used maliciously. For more information on proxy patterns,check out OpenZeppelin's proxy pattern guide .

Using the Plugin¶

First, install the package:

npm install --save-dev @openzeppelin/truffle-upgrades Note : This package requires Truffle version 5.1.35 or greater.

Then, in your migration script, use the new deployProxy and upgradeProxy functions:

```
const

{

deployProxy ,

upgradeProxy

}

=

require ( '@openzeppelin/truffle-upgrades' ); const

Box

=

artifacts . require ( 'Box' ); const

BoxV2

=

artifacts . require ( 'BoxV2' ); module . exports

=

async

function

( deployer )

{

const

instance

=

await
deployProxy ( Box ,
[ 42 ],
{
deployer
});
const
upgraded
=
await
upgradeProxy ( instance . address ,
BoxV2 ,
{
deployer
}); }
```

Now your contract systems can be much more flexible while still leveraging all of the benefits of a blockchain network. For more information on the truffle-upgrades plugin, check outOpenZeppelin's Truffle documentation .

We're very excited about this plugin and would love to hear more about how you're using it and how we can make the experience even better! Please let us know how you feel inthe Truffler Slack community or onOpenZeppelin's forum !