Deploying a dapp on Bubs testnet

First, review theBubs testnet page and theDeploy a smart contract to Bubs testnettutorial.

You will need a funded account to deploy your smart contract.

Next, clone thegm-portal from Github and start the frontend:

bash cd HOME git

clone

https://github.com/jcstein/gm-portal.git cd

gm-portal/frontend yarn && yarn

dev cd HOME git

clone

https://github.com/jcstein/gm-portal.git cd

gm-portal/frontend yarn && yarn

dev In a new terminal instance, set your private key for the faucet as a variable and the RPC URL you're using:

bash export PRIVATE_KEY = ac0974bec39a17e36ba4a6b4d238ff944bacb478cbed5efcae784d7bf4f2ff80 export BUBS_RPC_URL = https://bubs.calderachain.xyz/http export PRIVATE_KEY = ac0974bec39a17e36ba4a6b4d238ff944bacb478cbed5efcae784d7bf4f2ff80 export BUBS_RPC_URL = https://bubs.calderachain.xyz/http Now, change into thegm-portal/contracts directory in the same terminal and deploy the contract using Foundry:

bash cd HOME /gm-portal/contracts forge

script

script/GmPortal.s.sol:GmPortalScript

--rpc-url BUBS_RPC_URL --private-key PRIVATE_KEY --broadcast cd HOME /gm-portal/contracts forge

script

script/GmPortal.s.sol:GmPortalScript

--rpc-url BUBS_RPC_URL --private-key PRIVATE_KEY --broadcast

In the output of the deployment, find the contract address and set it as a variable:

bash export CONTRACT_ADDRESS =< your-contract-address-from-the-output-abov e

export CONTRACT_ADDRESS =< your-contract-address-from-the-output-abov e

Next, you're ready to interact with the contract from your terminal!

First, send a "gm" to the contract:

bash cast

send CONTRACT_ADDRESS \ "gm(string)"

"gm"

\ --private-key PRIVATE_KEY \ --rpc-url BUBS_RPC_URL cast

send CONTRACT_ADDRESS \ "gm(string)"

"gm"

\ --private-key PRIVATE_KEY \ --rpc-url BUBS_RPC_URL Now that you've posted to the contract, you can read all "gms" (GMs) from the contract with this command:

bash cast

call CONTRACT_ADDRESS "getAllGms()"

--rpc-url BUBS_RPC_URL cast

call CONTRACT_ADDRESS "getAllGms()"

--rpc-url BUBS_RPC_URL Next, query the total number of gms, which will be returned as a hex value:

bash cast

call CONTRACT_ADDRESS "getTotalGms()"

--rpc-url BUBS_RPC_URL cast

call CONTRACT_ADDRESS "getTotalGms()"

--rpc-url BUBS_RPC_URL In order to interact with the contract on the frontend, you'll need to fund an account that you have in your Ethereum wallet. Transfer to an external account with this command:

bash export RECEIVER =< receiver

ETH

addres s

cast

send

--private-key PRIVATE_KEY RECEIVER --value

1 ether

--rpc-url BUBS_RPC_URL export RECEIVER =< receiver

ETH

addres s

cast

send

--private-key PRIVATE_KEY RECEIVER --value

1 ether

--rpc-url BUBS_RPC_URL If you are in a different terminal than the one you set the private key in, you may need to set it again.

Update the frontend

Next, you will need to update a few things before you can interact with the contract on the frontend:

1. Change the contract address ongm-portal/frontend/src/App.tsx
2. to your contract address
3. Match the chain info ongm-portal/frontend/src/main.tsx
4. with the chain config of your L2
5. If you changed the contract, update the ABI ingm-portal/frontend/GmPortal.json
6. fromgm-portal/contracts/out/GmPortal.sol/GmPortal.json
7. . This can be done with:

bash cd HOME cp

dev/gm-portal/contracts/out/GmPortal.sol/GmPortal.json

dev/gm-portal/frontend cd HOME cp

dev/gm-portal/contracts/out/GmPortal.sol/GmPortal.json

dev/gm-portal/frontend

Interact with the frontend

Now, login with your wallet that you funded, and post a GM on your GM portal!

Next steps

There are many possibilities of what could be built with this stack. These projects would be good to build on this stack:

- onchain gaming
- decentralized social media
- an NFT ticketing rollup
- Optimism on CelOPstia
- OP Craft on Celestia [][Edit this page on GitHub] Last updated: Previous page Deploy a smart contract on Bubs testnet Next page Deploy an OP Stack devnet []