Consthip - Subnet - Create and Deploy

⚠ Un subnet è gia stato deploiato sul test net di fuji. Questa documentazione serve a creare un nuovo subnet da 0

In questa documentazione, verrà illustrato il processo di creazione di un subnet e il relativo deploy sulla rete test Fuji di Avalanche.

Genesis File

Il genesis file è uno script che contiene tutte le configurazioni necessarie per l'impostazione del subnet. Ora esamineremo il nostro genesis file, che riflette il design del subnet discusso nella documentazione "Contship - Subnet - Proof of Concept and Design".

```
{
    "config": {
      "chainID": 2819,
      "homesteadBlock": 0,
      "eip150Block": 0,
      "eip150Hash": "0x2086799aeebeae135c246c65021c82b4e15a2c451340993aacfd2751886514f0",
      "eip155Block": 0,
      "eip158Block": 0,
      "byzantiumBlock": 0,
      "constantinopleBlock": 0,
      "petersburgBlock": 0,
      "istanbulBlock": 0,
      "muirGlacierBlock": 0,
      "SubnetEVMTimestamp": 0,
      "feeConfig": {
        "gasLimit": 5000000,
        "targetBlockRate": 5,
        "minBaseFee": 60000000000,
        "targetGas": 10000000,
        "baseFeeChangeDenominator": 50,
        "minBlockGasCost": 0,
        "maxBlockGasCost": 5000000,
        "blockGasCostStep": 10000
      },
      "contractDeployerAllowListConfig": {
        "blockTimestamp": 0,
        "adminAddresses": ["0xe267aA4FFcBbeEa372819AA78A7e3d44cc9d9C60"]
      },
      "contractNativeMinterConfig": {
        "blockTimestamp": 0,
        "adminAddresses": ["0x0000000b9af48743ef1188f3F20c9b8B90F52a5b"]
      }
    },
    "alloc": {
      "0xe267aA4FFcBbeEa372819AA78A7e3d44cc9d9C60": {
```

```
        "balance": "0x3635C9ADC5DEA00000"
      },
      "0x628e60b166Fe63013964FEE22A7AB13Ee79E546e": {
        "balance": "0x3635C9ADC5DEA00000"
      }
    },
    "nonce": "0x0",
    "timestamp": "0x0",
    "extraData": "0x00",
    "gasLimit": "5000000",
    "difficulty": "0x0",
    "mixHash": "0x0000000000000000000000000000000000000000000000000000000000000000",
    "coinbase": "0x0000000000000000000000000000000000000000",
    "number": "0x0",
    "gasUsed": "0x0",
    "parentHash": "0x0000000000000000000000000000000000000000000000000000000000000000"
  }
```

Durante questa fase del progetto, ci concentreremo solo su alcune configurazioni pertinenti. In futuro, potremo aggiungere ulteriori configurazioni per dettagli più specifici. Per maggiori informazioni, consultare il seguente link: Avalanche Dev Docs: Create Without Limits

1. Iniziamo impostando la `chainID`, che deve essere unica. Per verificare l'unicità, consultare il seguente link: ChainList
2. Successivamente, configuriamo le `feeConfig` secondo le impostazioni descritte nella documentazione "Contship - Subnet - Proof of Concept and Design".
3. Impostiamo il `contractDeployerAllowListConfig` per configurare la pool di smart contract e assegnare i diritti di deploy. Avremo un'unica admin address che potrà aggiungere o rimuovere i diritti di deploy degli smart contract. Per aggiungere semplici indirizzi senza diritti di amministrazione, sostituire adminAddresses con enabledAddresses.
4. Configuriamo il `contractNativeMinterConfig` per consentire a un wallet di creare token. Questo è utile se gli sviluppatori necessitano di ulteriori test token. I concetti di adminAddress e enabledAddress rimangono invariati.
5. Infine, utilizziamo alloc per distribuire test token ai wallet indicati durante la creazione del subnet.

Creazione e deploy del subnet

Ora possiamo procedere con la creazione del subnet. Per farlo, accediamo alla cartella `subnet` e modifichiamo lo script per assegnare un altro nome al subnet, se lo si desidera. Dopo aver apportato la modifica, possiamo avviare lo script con il seguente comando:

```
./build-genesis.sh
```

Se vedete quest'errore :

```
./build-genesis.sh: line 5: avalanche: command not found
```

Basta avviare lo script e poi fare ctrl + c quando l'installazione e finita:

```
./dependency.sh
```

E dovrete ottenere questo :

```
ava-labs/avalanche-cli info checking GitHub for latest tag
ava-labs/avalanche-cli info found version: 1.3.7 for darwin/arm64
ava-labs/avalanche-cli info installed /Users/patrickdev/bin/avalanche
^CInstallation complete! Please restart your terminal or run 'source /Users/patrickdev/.bashrc' to apply the chan
patrickdev@Patricks-MacBook-Air subnet %
```

Una volta che avete fatto `source /Users/patrickdev/.bashrc` potete riavviare lo script :

```
./build-genesis.sh
```

E scegliamo `Subnet-EVM` e poi `Use latest version` per installare la virtual machine di ethereum.

Adesso possiamo vedere se il subnet e stato creato correttamente usando questo commando :

```
avalanche subnet list
```

E possiamo vedere che ContshipNetV5 e stato creato :

SUBNET	CHAIN	CHAINID	VMID	TYPE	VM VERSION	FROM REPO
ContshipNetV1	ContshipNetV1	2819	WhYY15RH8t2CHtkEa41ZPmML8YQgwUeJhwpaMjtjR6h94k4pf	Subnet-EVM	v0.5.11	false
ContshipNetV3	ContshipNetV3	2819	WhYY15RH8t2CHtkEaEaun1zAV1ok2f8SRNSWxEZbzrmEPan3A	Subnet-EVM	v0.5.11	false
ContshipNetV4	ContshipNetV4	2819	WhYY15RH8t2CHtkEaKsay9JafFWGaFNWH5kVFUu3HjoDAqGFw	Subnet-EVM	v0.6.1	false
ContshipNetV5	ContshipNetV5	2819	WhYY15RH8t2CHtkEaRAGAGczqVCo7qca8o4TYjEUacqD3VCS3	Subnet-EVM	v0.6.1	false

Per vedere se il genesis file a funzionato, possiamo fare :

```
avalanche subnet describe ContshipNetV5
```

E vedremo questo :

```

```

```
 _____       _        _ _
|  __ \     | |      (_) |
| |  | | ___| |_ __ _ _| |___
| |  | |/ _ \ __/ _  | | / __|
| |__| |  __/ || (_| | | \__ \
|_____/ \___|\__\__,_|_|_|___/
+-----------------+------------------------------------------------------+
|    PARAMETER    |                        VALUE                         |
+-----------------+------------------------------------------------------+
| Subnet Name     | ContshipNetV5                                        |
+-----------------+------------------------------------------------------+
| ChainID         | 2819                                                 |
+-----------------+------------------------------------------------------+
| Mainnet ChainID | 0                                                    |
+-----------------+------------------------------------------------------+
| Token Name      | TEST                                                 |
+-----------------+------------------------------------------------------+
| VM Version      | v0.6.1                                               |
+-----------------+------------------------------------------------------+
| VM ID           | WhYY15RH8t2CHtkEaRAGAGczqVCo7qca8o4TYjEUacqD3VCS3 |
+-----------------+------------------------------------------------------+

  _____                 _____              __ _
 / ____|               / ____|            / _(_)
| |  __  __ _ ___  | |     ___  _ __ | |_ _  __ _
| | |_ |/ _  / __| | |    / _ \| '_ \|  _| |/ _  |
| |__| | (_| \__ \ | |___| (_) | | | | | | | (_| |
 \_____|\__,_|___/  \_____\___/|_| |_|_| |_|\__, |
                                              __/ |
                                             |___/
+--------------------------+-------------+
|      GAS PARAMETER       |    VALUE    |
+--------------------------+-------------+
| GasLimit                 |     5000000 |
+--------------------------+-------------+
```

```
| MinBaseFee               | 60000000000 |
+--------------------------+-------------+
| TargetGas (per 10s)      |    10000000 |
+--------------------------+-------------+
| BaseFeeChangeDenominator |          50 |
+--------------------------+-------------+
| MinBlockGasCost          |           0 |
+--------------------------+-------------+
| MaxBlockGasCost          |     5000000 |
+--------------------------+-------------+
| TargetBlockRate          |           5 |
+--------------------------+-------------+
| BlockGasCostStep         |       10000 |
+--------------------------+-------------+


          _         _
    /\   (_)       | |
   /  \   _ _ __ __| |_ __ ___  _ __
  / /\ \ | | '__/ _  | '__/ _ \| '_ \
 / ____ \| | | | (_| | | | (_) | |_) |
/_/    \_\_|_|  \__,_|_|  \___/| .__/
                               | |
                               |_|
+------------------------------------------+------------------------+------------------------+
|                 ADDRESS                  | AIRDROP AMOUNT (10^18) |  AIRDROP AMOUNT (WEI)  |
+------------------------------------------+------------------------+------------------------+
| 0xe267aA4FFcBbeEa372819AA78A7e3d44cc9d9C60 |                   1000 | 10000000000000000000 |
+------------------------------------------+------------------------+------------------------+
| 0x628e60b166Fe63013964FEE22A7AB13Ee79E546e |                   1000 | 10000000000000000000 |
+------------------------------------------+------------------------+------------------------+


  _____                                 _ _
 |  __ \                               (_) |
 | |__) | __ ___  ___ ___  _ __ ___  _ __  _| | ___  ___
 |  ___/ '__/ _ \/ __/ _ \| '_   _ \| '_ \| | |/ _ \/ __|
 | |   | | |  __/ (_| (_) | | | | | | |_) | | |  __/\__ \
 |_|   |_|  \___|\___\___/|_| |_| |_| .__/|_|_|\___||___/
                                    | |
                                    |_|

+---------------------+------------------------------------------+---------+
|     PRECOMPILE      |                  ADMIN                   | ENABLED |
+---------------------+------------------------------------------+---------+
| Native Minter       | 0x0000000b9af48743ef1188f3F20c9b8B90F52a5b |         |
+---------------------+------------------------------------------+---------+
| Contract Allow List | 0xe267aA4FFcBbeEa372819AA78A7e3d44cc9d9C60 |         |
```

Prima di tutto, facciamo il comando :

```
avalanche key list
```

E vedremo questo :

```
+--------+-----------------------+-------------------------------+-------------------------------------------
|  KIND  |          NAME         |             CHAIN             |                   ADDRESS
+--------+-----------------------+-------------------------------+-------------------------------------------
| stored | ContshipDocumentazione | C-Chain (Ethereum hex format) | 0xA0A15211cC5613DD51c8746106a74dCCa088525c
+        +                       +-------------------------------+-------------------------------------------
```

```
|        |                       | P-Chain (Bech32 format)       | P-fuji16e6nad8qm4f62utsje6huk7qx2xl39fnstvc4
+        +-----------------------+-------------------------------+---------------------------------------------
|        | PatrickAdmin          | C-Chain (Ethereum hex format) | 0x7474cE7b605687942543fc9789fbB56BE6849D13
+        +                       +-------------------------------+---------------------------------------------
|        |                       | P-Chain (Bech32 format)       | P-fuji1h6gzhmeefq9yw7nudv5a9msmkgfymwuzm5r5r
+        +-----------------------+-------------------------------+---------------------------------------------
|        | Patricktest           | C-Chain (Ethereum hex format) | 0xb83f5b4A2dCd63eDAF97BF252bAdC122b85Df818
+        +                       +-------------------------------+---------------------------------------------
|        |                       | P-Chain (Bech32 format)       | P-fuji1ymep62cuxhzuey9sw30z852g2myme4jd3lr6t
+--------+-----------------------+-------------------------------+---------------------------------------------
```

Durante questo processo, utilizzero la chiave ContshipDocumentazione

Adesso possiamo deploiare il subnet con questo comando :

```
avalanche subnet deploy ContshipNetV5
```

Scegliamo la rete Fuji, quindi selezioniamo "Use stored key" e scegliamo la chiave creata nella documentazione "Contship - Subnet - Nodi", che nel nostro caso si chiama "ContshipDocumentazione". Ci verrà chiesto di indicare chi sarà il controller key, ovvero chi avrà i diritti amministrativi. Selezioniamo "Custom" e inseriamo l'indirizzo della P-Chain della nostra chiave, che in questo caso è: P-fuji16e6nad8qm4f62utsje6huk7qx2xl39fnstvc4q. Una volta fatto ciò, possiamo avviare il deploy del subnet. Al termine dell'operazione, dovreste vedere un messaggio simile al seguente:

```
Your Subnet's control keys: [P-fuji16e6nad8qm4f62utsje6huk7qx2xl39fnstvc4q]
Your subnet auth keys for chain creation: [P-fuji16e6nad8qm4f62utsje6huk7qx2xl39fnstvc4q]
Subnet has been created with ID: 2m1u8UhSctTVAhxugTdtarZK3ipCscDV5XD1VxCz3yRRVJYiTs
Now creating blockchain...
+--------------------+------------------------------------------------------+
| DEPLOYMENT RESULTS |                                                      |
+--------------------+------------------------------------------------------+
| Chain Name         | ContshipNetV5                                        |
+--------------------+------------------------------------------------------+
| Subnet ID          | 2m1u8UhSctTVAhxugTdtarZK3ipCscDV5XD1VxCz3yRRVJYiTs |
+--------------------+------------------------------------------------------+
| VM ID              | WhYY15RH8t2CHtkEaRAGAGczqVCo7qca8o4TYjEUacqD3VCS3  |
+--------------------+------------------------------------------------------+
| Blockchain ID      | X3mwSCHGqzhPWoKxEBU5QCy2TKZ8kP4SUGC9wjVYbaNuAtTm6  |
+--------------------+                                                      +
| P-Chain TXID       |                                                      |
+--------------------+------------------------------------------------------+
```

Ottimo, il subnet è stato correttamente deployato sulla testnet. Ora possiamo procedere con i seguenti passaggi:

```
avalanche subnet export ContshipNetV5
```

E diamo un nome al file, ad esempio "subnet-details.txt". Questo file conterrà tutti i dettagli del subnet, come il subnet ID, la blockchain ID, ecc. Adesso si puo tornare alla documentazione “ Contship - Subnet - Nodi “ per aggiungere il nodo al subnet.