# Consensus seed rotation

From Genesis Seed to Seed Rotation: an Overview

Secret Network recently upgraded toversion 1.7.1 , which rotated the network consensus seed during the upgrade.

A consensus seed is a true random 256 bit seed that is used as entropy for generating shareable keypairs between the nodes of the network.

Previously, the consensus seed remained unchanged ever since the network's inception–the network state was encrypted using private keys generated during the network bootstrapping from a consensus seed, similar to a universal trusted zero knowledge setup. However, if anyone were to gain access to this seed they would have the master key to decrypt the state of the entire network. Thus, Secret Network has introduced consensus seed rotation in order to increase network security.

Consensus Seed Rotation

In order to protect against potential future breaches, Secret Network developed a two-part protocol for changing the consensus seed:

1. Rotate the current seed (The "Genesis" seed) and change the encryption scheme
2. Implement contract state migration and allow seed rotation on upgrade (currently in development)
3.

It is important to note that the upgrade to consensus seed rotation does not change the state (and the consensus seed) of the network prior to the upgrade. This means that the new encryption scheme must be able to distinguish between values that were encrypted with the genesis seed and those that will be encrypted with future seeds. To this end, the following features were implemented:

- A way to distinguish between values that were encrypted with the genesis seed and those that will be encrypted with future seeds
- The ability to iterate over all of the keys for a specific contract using CosmWasm iterators (currently in development)
- The ability to decrypt state keys, rotate the seed, and decrypt and re-encrypt all keys & values in the state
-

Creating a New Seed

In order to rotate the Genesis seed, a new seed must be created that will be shared with all current nodes as well as new nodes. To this end, Secret Labs updated all of the existing nodes with the new seed and every new node that joins the network will contain 2 consensus seeds (The genesis and the current) and will use them both based on the encrypted value.

The new seed can be received via three methods:

1. On upgrade: When upgrading to 1.7.1, the node will identify that the current seed is missing and will communicate with Secret Labs' seed service in order to obtain the seed
2. On Registration: On registration, both the current and the genesis seed will be passed to the newly registered node
3. On Bootstrap: Boostrap will access the seed service unless the "use_seed_service_on_bootstrap" feature is off (Which is the default state in localsecret). If so, it will generate one instead.
4.

Previous Encryption Scheme

In the previous encryption scheme, the key and the value were stored as follows:

Previous encryption scheme

New Encryption Scheme

In the new encryption scheme, the plaintext key is no longer necessary in order to decrypt the value. The encrypted value also looks a bit different in the new scheme:

New encryption scheme This new encryption scheme ensures that:

- the encrypted_state_key encrypts differently between different keys
- The salt will verify that on different instances, the same value is encrypted differently for the same key. The salt is the current block's timestamp and the msg id, which is a counter of the messages and allows for different values between different messages in the same block.
-

Node to Service Protocol

In order to authenticate a node, the node first sends an attestation report to the designated /authenticate endpoint as the request body. The server then responds with a challenge, which is a randomly generated 4-byte value that is linked to the public key of the node. The node then creates a new attestation report that incorporates the challenge into its quote, which proves that the node can generate new attestation reports certified by Intel while also rendering old certificates invalid. Subsequently, the node transmits this new attestation report to the /seed/[id] endpoint, where the ID represents the desired seed. Upon receiving the attestation report, the server verifies it and sends the seed to the node.

The port of the service is 4487 and there are 2 DNS names that are used.

On MainNet - sss.scrtlabs.com

On Pulsar - sssd.scrtlabs.com

Conclusion

Secret Network has implemented consensus seed rotation in the upgrade to version 1.7.1. Previously, the consensus seed remained unchanged since the network's inception, but this posed a security risk as anyone with access to the seed would have the master key to decrypt the entire network's state. The new consensus seed rotation includes a two-part protocol to change the genesis seed and encryption scheme, and implement contract state migration to allow seed rotation on upgrade. The upgrade does not change the network state prior to the upgrade, so the new encryption scheme distinguishes between values encrypted with the genesis seed and those encrypted with future seeds. Secret Labs updated all existing nodes with the new seed and new nodes joining the network will have both the genesis and current seeds, using them based on the encrypted value. These updates increase the security of Secret Network by protecting against potential future breaches.

Last updated11 months ago On this page *From Genesis Seed to Seed Rotation: an Overview* Consensus Seed Rotation * Creating a New Seed * Node to Service Protocol * Conclusion

Was this helpful? Edit on GitHub Export as PDF