Section 1

Vault Replication

Overview

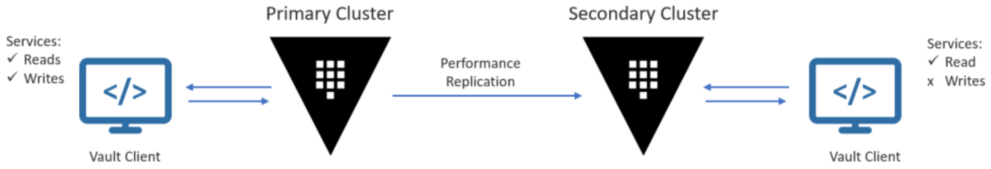
Vault Replication

- Setting the Stage
- Replication Architecture
- Replication Requirements
- Secondary Token
- How Communication Works
- How to Configure Replication
- Monitoring Replication
- Vault Lab Environment

Setting the Stage

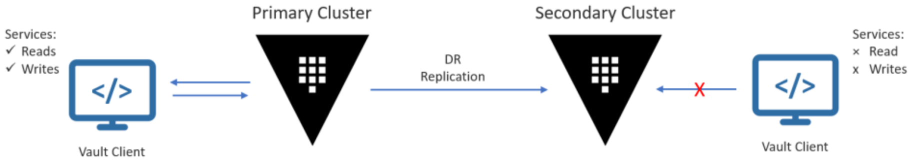
Performance Replication

- Replicates the underlying configuration, policies, and other data
- Ability to service reads from client requests
- Clients will authenticate to the performance replicated cluster separately
- Does not replicate tokens or leases

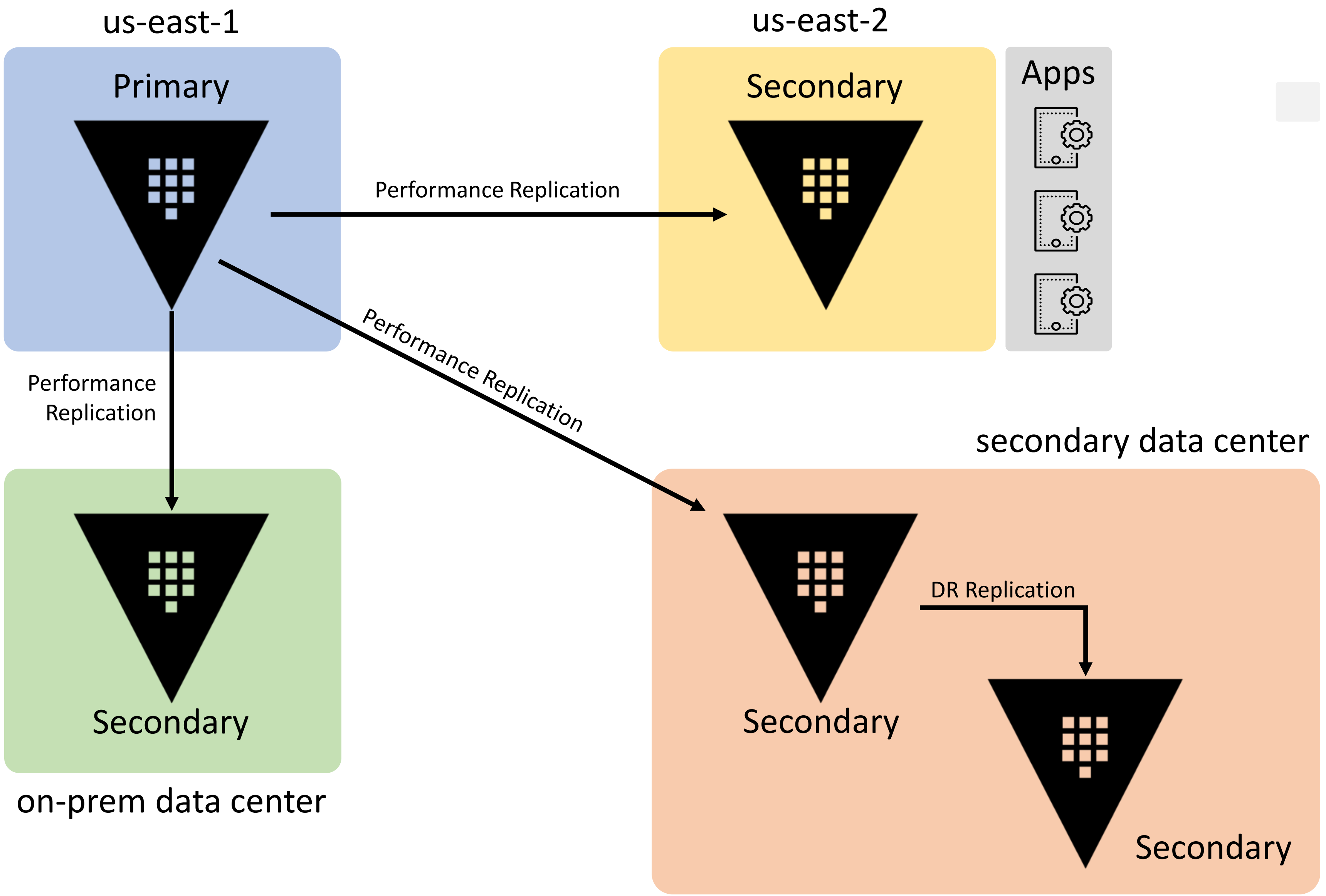


Disaster Recovery Replication

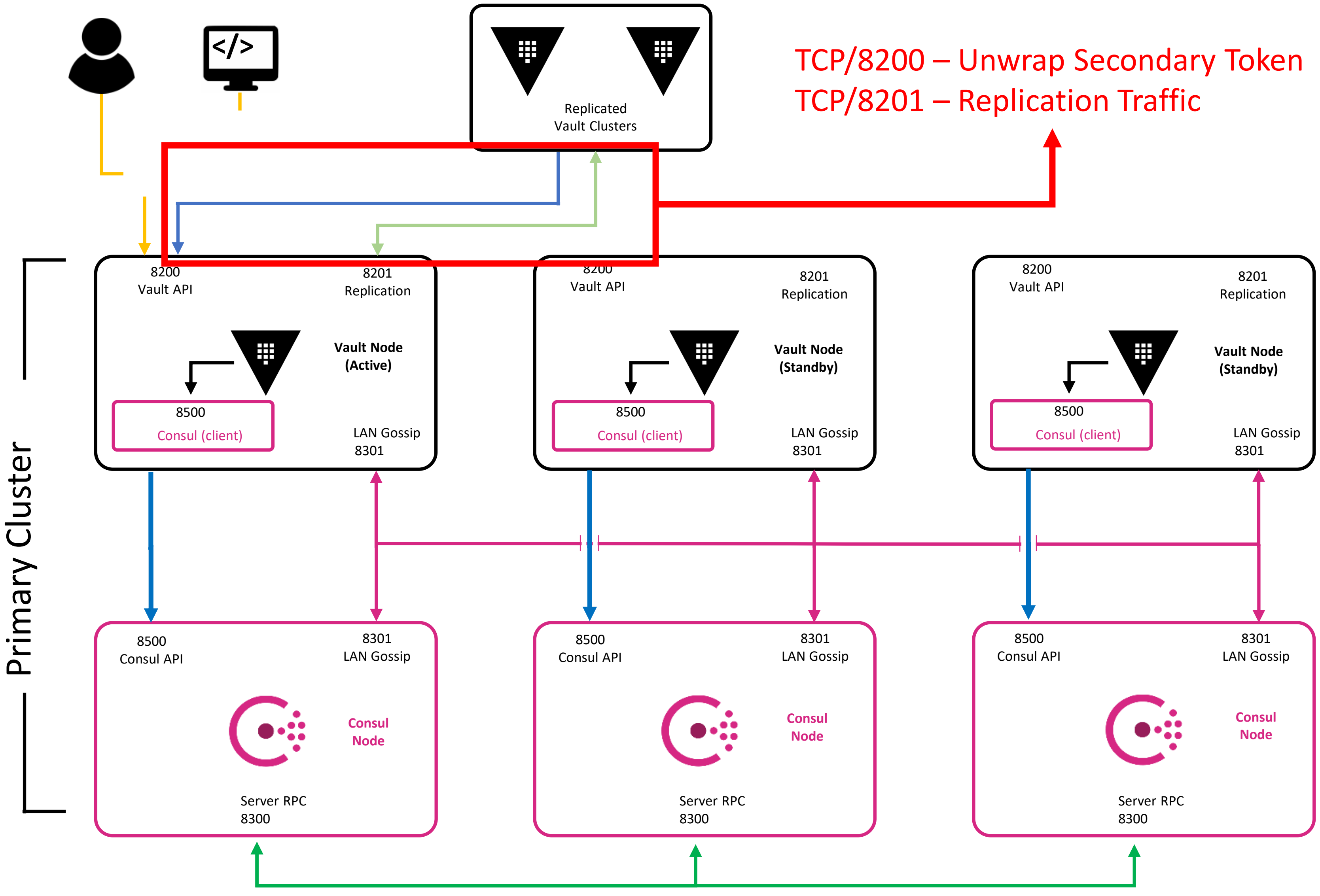
- Replicates the underlying configuration, policies, and all other data
- Cannot service reads from client requests
- Clients should authenticate with the primary cluster only (or a perf cluster)
- Will replicate tokens and leases created on the primary cluster



Architecture



Ports Required



TCP/8200 – Unwrap Secondary Token
TCP/8201 – Replication Traffic

Secondary Token

Secondary token is required to permit a secondary cluster to replicate from the primary cluster

Due to its sensitivity, the secondary token is protected with response wrapping.

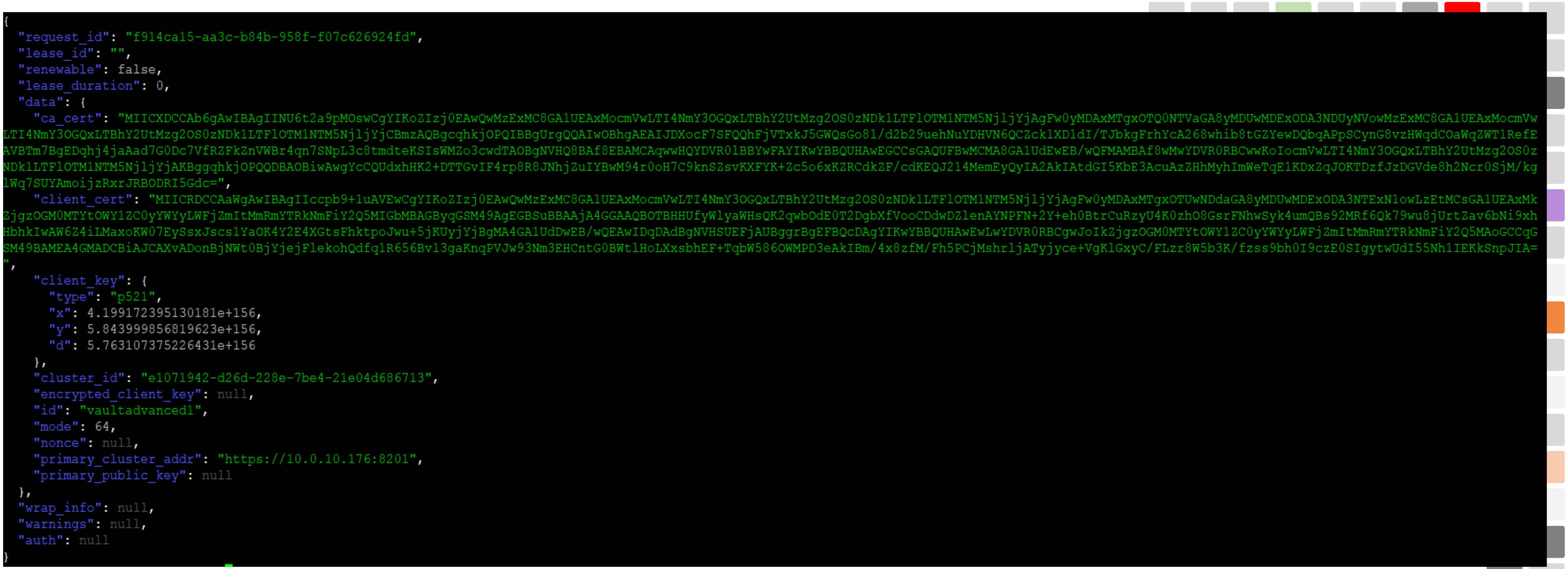
Multiple people should “have eyes” on the secondary token once it’s been issued until it is submitted to the secondary cluster.

Once the token is successfully used, it is useless.

The secondary token includes information such as:

- The redirect address of the primary cluster
- The client certificate and CA certificate

Secondary Token



```
{
  "request_id": "f914ca15-aa3c-b84b-958f-f07c626924fd",
  "lease_id": "",
  "renewable": false,
  "lease_duration": 0,
  "data": {
    "ca_cert": "MIICXDCCAb6gAwIBAgIINU6t2a9pMOswCgYIKoZIzj0EAwQwMzExMC8GA1UEAxMocmVwLTI4NmY3OGQxLTBhY2UtMzg2OS0zNDk1LTFlOTM1NTM5NjljYjAgFw0yMDAxMTgxOTQ0NTVaGA8yMDUwMDExODA3NDUyNVowMzExMC8GA1UEAxMocmVw
LTI4NmY3OGQxLTBhY2UtMzg2OS0zNDk1LTFlOTM1NTM5NjljYjCBmzAQBgcqhkjOPQIBBgUrgQQAIwOBhgAEAIJDXocF7SFQQhFjVTxkJ5GWQsGo8l/d2b29uehNuYDHVN6QCZck1XD1dI/TJbkgFrhYcA268whib8tGZYewDQbqAPpSCynG8vzHWqdCOaWqZWT1RefE
AVBTm7BgEDqhj4jaAad7G0Dc7VfRZFkZnVWBr4qn7SNpL3c8tmdteKSIsWMZo3cwdTAOBgNVHQ8BAf8EBAMCAqwwHQYDVR0lBBYwFAYIKwYBBQUHAwEGCCsGAQUFBwMCMA8GA1UdEwEB/wQFMAMBAf8wMwYDVR0RBCwwKoIocmVwLTI4NmY3OGQxLTBhY2UtMzg2OS0z
NDk1LTFlOTM1NTM5NjljYjAKBggqhkjOPQQDBAOBiwAwgYcCQUdxhHK2+DTTGvIF4rp8R8JNhjZuIYBwM94r0oH7C9knSZsvKXFYK+Zc5o6xKZRCdkZF/cdKEQJ214MemEyQyIA2AkIAtdGI5KbE3AcuAzZHhMyhImWeTqE1KDxZqJOKTDzfJzDGVde8h2Ncr0SjM/kg
1Wq7SUYAmoijzRxrJRBODRI5Gdc=",
    "client_cert": "MIICRDCCAaWgAwIBAgIIccpb9+1uAVEwCgYIKoZIzj0EAwQwMzExMC8GA1UEAxMocmVwLTI4NmY3OGQxLTBhY2UtMzg2OS0zNDk1LTFlOTM1NTM5NjljYjAgFw0yMDAxMTgxOTUwNDdaGA8yMDUwMDExODA3NTExN1owLzEtMCsGA1UEAxMk
ZjgzOGM0MTYtOWY1ZC0yYWYyLWFjZmItMmRmYTRkNmFiY2Q5MIGbMBAGByqGSM49AgEGBSuBBAAjA4GGAAQBOTBHHUfyWlyaWHsQK2qwbOdE0T2DgbXfVooCDdwDZlenAYNPFN+2Y+eh0BtrCuRzyU4K0zhO8GsrFNhwSyk4umQBs92MRf6Qk79wu8jUrtZav6bNi9xh
HbhkIwAW6Z4iLMaxoKW07EySsxJscs1YaOK4Y2E4XGtsFhktpoJwu+5jKUyjYjBgMA4GA1UdDwEB/wQEAwIDqDAdBgNVHSUEFjAUBggrBgEFBQcDAgYIKwYBBQUHAwEwLwYDVR0RBCgwJoIkZjgzOGM0MTYtOWY1ZC0yYWYyLWFjZmItMmRmYTRkNmFiY2Q5MAoGCCqG
SM49BAMEA4GMADCBiAJCAXvADonBjNWt0BjYjejFlekohQdfq1R656Bv13gaKnqPVJw93Nm3EHCntG0BWtlHoLXxsbhEF+TqbW586OWMPD3eAkIBm/4x8zfM/Fh5PCjMshrljATyjyce+VgKlGxyC/FLzr8W5b3K/fzss9bh0I9czE0SIgytwUdI55Nh1IEKkSnpJIA=
",
    "client_key": {
      "type": "p521",
      "x": 4.199172395130181e+156,
      "y": 5.843999856819623e+156,
      "d": 5.763107375226431e+156
    },
    "cluster_id": "e1071942-d26d-228e-7be4-21e04d686713",
    "encrypted_client_key": null,
    "id": "vaultadvanced1",
    "mode": 64,
    "nonce": null,
    "primary_cluster_addr": "https://10.0.10.176:8201",
    "primary_public_key": null
  },
  "wrap_info": null,
  "warnings": null,
  "auth": null
}
```

How Communication Works

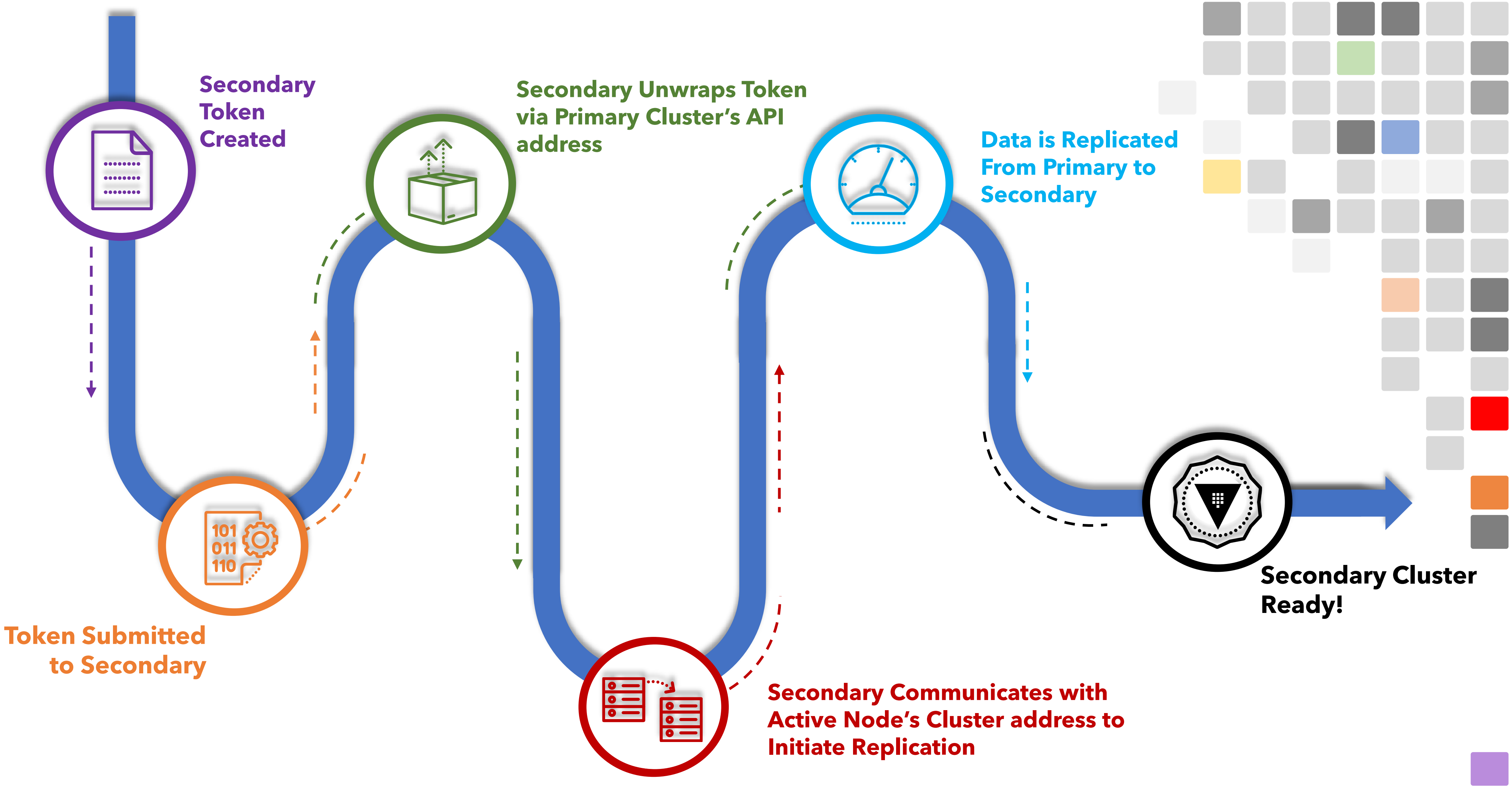
There are two ports that are important to Vault:

- tcp/8200
  - API traffic
  - Defined by the api_addr flag
- tcp/8201
  - Vault server-to-server communication, request forwarding, replication traffic
  - Define by the cluster_addr flag

For 8201, Vault creates a mutual TLS connection between the nodes using self-signed certificates and keys – *NOT the same TLS configured for the listener*

If Vault sits behind a load balancer which is terminating TLS, it will break the mutual TLS between the nodes

How Communication Works



Secondary Token Created

Token Submitted to Secondary

Secondary Unwraps Token via Primary Cluster's API address

Secondary Communicates with Active Node's Cluster address to Initiate Replication

Data is Replicated From Primary to Secondary

Secondary Cluster Ready!

How to Configure

1 Activate Performance Replication

```
$ vault write -f sys/replication/performance/primary/enable
```

2 Create the Secondary Token

```
$ vault write sys/replication/performance/primary/secondary-token id=<id>
```

3 Activate the Secondary Cluster

```
$ vault write sys/replication/performance/secondary/enable token=<token>
```

Monitoring Replication

1 Check Status of Replication

```
$ vault read -format=json sys/replication/status
```

```
$ vault read -format=json sys/replication/performance/status
```

Performance Replication Only

```
$ vault read -format=json sys/replication/dr/status
```

DR Replication Only

2 Use Vault Telemetry

- logshipper.streamWALs.missing_guard
- logshipper.streamWALs.guard_found
- replication.fetchRemoteKeys
- replication.merkleDiff
- replication.merkleSync

- vault.wal_persistwals
- vault.wal_flushready
- wal.gc.total
- wal.gc.deleted

Demo

Vault Replication

- Create secondary token
- Configure secondary cluster
- Monitor the Status of Replication

Recap

Vault Replication

- Setting the Stage
- Replication Architecture
- Replication Requirements
- Secondary Token
- How Communication Works
- How to Configure Replication
- Monitoring Replication
- Vault Lab Environment