CEPHX CONFIG REFERENCE

The cephx protocol is enabled by default. Cryptographic authentication has some computational costs, though they should generally be quite low. If the network environment connecting your client and server hosts is very safe and you cannot afford authentication, you can turn it off. **This is not generally recommended**.

> **Note:** If you disable authentication, you are at risk of a man-in-the-middle attack altering your client/server messages, which could lead to disastrous security effects.

For creating users, see User Management. For details on the architecture of Cephx, see Architecture - High Availability Authentication.

DEPLOYMENT SCENARIOS

There are two main scenarios for deploying a Ceph cluster, which impact how you initially configure Cephx. Most first time Ceph users use `ceph-deploy` to create a cluster (easiest). For clusters using other deployment tools (e.g., Chef, Juju, Puppet, etc.), you will need to use the manual procedures or configure your deployment tool to bootstrap your monitor(s).

CEPH-DEPLOY

When you deploy a cluster with `ceph-deploy`, you do not have to bootstrap the monitor manually or create the `client.admin` user or keyring. The steps you execute in the Storage Cluster Quick Start will invoke `ceph-deploy` to do that for you.

When you execute `ceph-deploy new {initial-monitor(s)}`, Ceph will create a monitor keyring for you (only used to bootstrap monitors), and it will generate an initial Ceph configuration file for you, which contains the following authentication settings, indicating that Ceph enables authentication by default:

```
auth_cluster_required = cephx
auth_service_required = cephx
auth_client_required = cephx
```

When you execute `ceph-deploy mon create-initial`, Ceph will bootstrap the initial monitor(s), retrieve a `ceph.client.admin.keyring` file containing the key for the `client.admin` user. Additionally, it will also retrieve keyrings that give `ceph-deploy` and `ceph-volume` utilities the ability to prepare and activate OSDs and metadata servers.

When you execute `ceph-deploy admin {node-name}` (**note:** Ceph must be installed first), you are pushing a Ceph configuration file and the `ceph.client.admin.keyring` to the `/etc/ceph` directory of the node. You will be able to execute Ceph administrative functions as `root` on the command line of that node.

MANUAL DEPLOYMENT

When you deploy a cluster manually, you have to bootstrap the monitor manually and create the `client.admin` user and keyring. To bootstrap monitors, follow the steps in Monitor Bootstrapping. The steps for monitor bootstrapping are the logical steps you must perform when using third party deployment tools like Chef, Puppet, Juju, etc.

ENABLING/DISABLING CEPHX

Enabling Cephx requires that you have deployed keys for your monitors, OSDs and metadata servers. If you are simply toggling Cephx on / off, you do not have to repeat the bootstrapping procedures.

ENABLING CEPHX

When cephx is enabled, Ceph will look for the keyring in the default search path, which includes `/etc/ceph/$cluster.$name.keyring`. You can override this location by adding a `keyring` option in the `[global]` section of your Ceph configuration file, but this is not recommended.

Execute the following procedures to enable cephx on a cluster with authentication disabled. If you (or your deployment utility) have already generated the keys, you may skip the steps related to generating keys.

1. Create a `client.admin` key, and save a copy of the key for your client host:

```
ceph auth get-or-create client.admin mon 'allow *' mds 'allow *' mgr 'allow *' osd 'allow *' -
```

Warning: This will clobber any existing `/etc/ceph/client.admin.keyring` file. Do not perform this step if a deployment tool has already done it for you. Be careful!

2. Create a keyring for your monitor cluster and generate a monitor secret key.

```
ceph-authtool --create-keyring /tmp/ceph.mon.keyring --gen-key -n mon. --cap mon 'allow *'
```

3. Copy the monitor keyring into a `ceph.mon.keyring` file in every monitor's `mon data` directory. For example, to copy it to `mon.a` in cluster ceph, use the following:

```
cp /tmp/ceph.mon.keyring /var/lib/ceph/mon/ceph-a/keyring
```

4. Generate a secret key for every MGR, where `{$id}` is the MGR letter:

```
ceph auth get-or-create mgr.{$id} mon 'allow profile mgr' mds 'allow *' osd 'allow *' -o /var/
```

5. Generate a secret key for every OSD, where `{$id}` is the OSD number:

```
ceph auth get-or-create osd.{$id} mon 'allow rwx' osd 'allow *' -o /var/lib/ceph/osd/ceph-{$id
```

6. Generate a secret key for every MDS, where `{$id}` is the MDS letter:

```
ceph auth get-or-create mds.{$id} mon 'allow rwx' osd 'allow *' mds 'allow *' mgr 'allow profi
```

7. Enable cephx authentication by setting the following options in the `[global]` section of your Ceph configuration file:

```
auth cluster required = cephx
auth service required = cephx
auth client required = cephx
```

8. Start or restart the Ceph cluster. See Operating a Cluster for details.

For details on bootstrapping a monitor manually, see Manual Deployment.

DISABLING CEPHX

The following procedure describes how to disable Cephx. If your cluster environment is relatively safe, you can offset the computation expense of running authentication. **We do not recommend it.** However, it may be easier during setup and/or troubleshooting to temporarily disable authentication.

1. Disable cephx authentication by setting the following options in the `[global]` section of your Ceph configuration file:

```
auth cluster required = none
auth service required = none
auth client required = none
```

2. Start or restart the Ceph cluster. See Operating a Cluster for details.

CONFIGURATION SETTINGS

ENABLEMENT

`auth cluster required`

Description: If enabled, the Ceph Storage Cluster daemons (i.e., `ceph-mon`, `ceph-osd`, `ceph-mds` and `ceph-mgr`) must authenticate with each other. Valid settings are `cephx` or `none`.

Type: String
Required: No
Default: cephx.

`auth service required`

Description: If enabled, the Ceph Storage Cluster daemons require Ceph Clients to authenticate with the Ceph Storage Cluster in order to access Ceph services. Valid settings are cephx or none.
Type: String
Required: No
Default: cephx.

`auth client required`

Description: If enabled, the Ceph Client requires the Ceph Storage Cluster to authenticate with the Ceph Client. Valid settings are cephx or none.
Type: String
Required: No
Default: cephx.

KEYS

When you run Ceph with authentication enabled, ceph administrative commands and Ceph Clients require authentication keys to access the Ceph Storage Cluster.

The most common way to provide these keys to the ceph administrative commands and clients is to include a Ceph keyring under the `/etc/ceph` directory. For Cuttlefish and later releases using `ceph-deploy`, the filename is usually `ceph.client.admin.keyring` (or `$cluster.client.admin.keyring`). If you include the keyring under the `/etc/ceph` directory, you don't need to specify a `keyring` entry in your Ceph configuration file.

We recommend copying the Ceph Storage Cluster's keyring file to nodes where you will run administrative commands, because it contains the `client.admin` key.

You may use `ceph-deploy admin` to perform this task. See Create an Admin Host for details. To perform this step manually, execute the following:

```
sudo scp {user}@{ceph-cluster-host}:/etc/ceph/ceph.client.admin.keyring /etc/ceph/ceph.client.admir
```

Tip: Ensure the `ceph.keyring` file has appropriate permissions set (e.g., `chmod 644`) on your client machine.

You may specify the key itself in the Ceph configuration file using the key setting (not recommended), or a path to a keyfile using the `keyfile` setting.

`keyring`

Description: The path to the keyring file.
Type: String
Required: No
Default: `/etc/ceph/$cluster.$name.keyring,/etc/ceph/$cluster.keyring,/etc/ceph/keyring,/etc/ceph/keyring.bin`

`keyfile`

Description: The path to a key file (i.e,. a file containing only the key).
Type: String
Required: No
Default: None

`key`

Description: The key (i.e., the text string of the key itself). Not recommended.
Type: String
Required: No
Default: None

DAEMON KEYRINGS

Administrative users or deployment tools (e.g., `ceph-deploy`) may generate daemon keyrings in the same way as generating user keyrings. By default, Ceph stores daemons keyrings inside their data directory. The default keyring locations, and the capabilities necessary for the daemon to function, are shown below.

`ceph-mon`

- **Location:** `$mon_data/keyring`
- **Capabilities:** `mon 'allow *'`

`ceph-osd`

- **Location:** `$osd_data/keyring`
- **Capabilities:** `mgr 'allow profile osd' mon 'allow profile osd' osd 'allow *'`

`ceph-mds`

- **Location:** `$mds_data/keyring`
- **Capabilities:** `mds 'allow' mgr 'allow profile mds' mon 'allow profile mds' osd 'allow rwx'`

`ceph-mgr`

- **Location:** `$mgr_data/keyring`
- **Capabilities:** `mon 'allow profile mgr' mds 'allow *' osd 'allow *'`

`radosgw`

- **Location:** `$rgw_data/keyring`
- **Capabilities:** `mon 'allow rwx' osd 'allow rwx'`

Note: The monitor keyring (i.e., `mon.`) contains a key but no capabilities, and is not part of the cluster `auth` database.

The daemon data directory locations default to directories of the form:

```
/var/lib/ceph/$type/$cluster-$id
```

For example, `osd.12` would be:

```
/var/lib/ceph/osd/ceph-12
```

You can override these locations, but it is not recommended.

SIGNATURES

In Ceph Bobtail and subsequent versions, we prefer that Ceph authenticate all ongoing messages between the entities using the session key set up for that initial authentication. However, Argonaut and earlier Ceph daemons do not know how to perform ongoing message authentication. To maintain backward compatibility (e.g., running both Botbail and Argonaut daemons in the same cluster), message signing is **off** by default. If you are running Bobtail or later daemons exclusively, configure Ceph to require signatures.

Like other parts of Ceph authentication, Ceph provides fine-grained control so you can enable/disable signatures for service messages between the client and Ceph, and you can enable/disable signatures for messages between Ceph daemons.

`cephx require signatures`

- **Description:** If set to `true`, Ceph requires signatures on all message traffic between the Ceph Client and the Ceph Storage Cluster, and between daemons comprising the Ceph Storage Cluster.
- **Type:** Boolean
- **Required:** No
- **Default:** `false`

`cephx cluster require signatures`

- **Description:** If set to `true`, Ceph requires signatures on all message traffic between Ceph daemons comprising the Ceph Storage Cluster.
- **Type:** Boolean
- **Required:** No
- **Default:** `false`

`cephx service require signatures`

Description: If set to `true`, Ceph requires signatures on all message traffic between Ceph Clients and the Ceph Storage Cluster.
Type: Boolean
Required: No
Default: `false`

`cephx sign messages`

Description: If the Ceph version supports message signing, Ceph will sign all messages so they cannot be spoofed.
Type: Boolean
Default: `true`

TIME TO LIVE

`auth service ticket ttl`

Description: When the Ceph Storage Cluster sends a Ceph Client a ticket for authentication, the Ceph Storage Cluster assigns the ticket a time to live.
Type: Double
Default: `60*60`

BACKWARD COMPATIBILITY

For Cuttlefish and earlier releases, see Cephx.

In Ceph Argonaut v0.48 and earlier versions, if you enable cephx authentication, Ceph only authenticates the initial communication between the client and daemon; Ceph does not authenticate the subsequent messages they send to each other, which has security implications. In Ceph Bobtail and subsequent versions, Ceph authenticates all ongoing messages between the entities using the session key set up for that initial authentication.

We identified a backward compatibility issue between Argonaut v0.48 (and prior versions) and Bobtail (and subsequent versions). During testing, if you attempted to use Argonaut (and earlier) daemons with Bobtail (and later) daemons, the Argonaut daemons did not know how to perform ongoing message authentication, while the Bobtail versions of the daemons insist on authenticating message traffic subsequent to the initial request/response–making it impossible for Argonaut (and prior) daemons to interoperate with Bobtail (and subsequent) daemons.

We have addressed this potential problem by providing a means for Argonaut (and prior) systems to interact with Bobtail (and subsequent) systems. Here's how it works: by default, the newer systems will not insist on seeing signatures from older systems that do not know how to perform them, but will simply accept such messages without authenticating them. This new default behavior provides the advantage of allowing two different releases to interact. **We do not recommend this as a long term solution**. Allowing newer daemons to forgo ongoing authentication has the unfortunate security effect that an attacker with control of some of your machines or some access to your network can disable session security simply by claiming to be unable to sign messages.

Note: Even if you don't actually run any old versions of Ceph, the attacker may be able to force some messages to be accepted unsigned in the default scenario. While running Cephx with the default scenario, Ceph still authenticates the initial communication, but you lose desirable session security.

If you know that you are not running older versions of Ceph, or you are willing to accept that old servers and new servers will not be able to interoperate, you can eliminate this security risk. If you do so, any Ceph system that is new enough to support session authentication and that has Cephx enabled will reject unsigned messages. To preclude new servers from interacting with old servers, include the following in the `[global]` section of your Ceph configuration file directly below the line that specifies the use of Cephx for authentication:

```
cephx require signatures = true    ; everywhere possible
```

You can also selectively require signatures for cluster internal communications only, separate from client-facing service:

```
cephx cluster require signatures = true    ; for cluster-internal communication
cephx service require signatures = true    ; for client-facing service
```

An option to make a client require signatures from the cluster is not yet implemented.

We recommend migrating all daemons to the newer versions and enabling the foregoing flag at the nearest practical time so that you may avail yourself of the enhanced authentication.

Note: Ceph kernel modules do not support signatures yet.