Secure Astra Data Store

Astra Data Store

NetApp
June 16, 2022

Table of Contents

Secure Astra Data Store

Manage security certificates

Astra Data Store uses Mutual Transport Layer Security (mTLS) encryption between the cluster's software components. Each Astra Data Store cluster has a self-signed root CA certificate (`astrads-cert-root`) and an intermediate CA certificate (`astrads-cert-<cluster_name>`). These certificates are managed by the Astra Data Store operator; the operator automatically renews each certificate 7 days before their expiration date. You can also revoke the certificates manually.

Revoke a certificate

If an Astra Data Store controller, node, or CA certificate is compromised, you can revoke it by deleting its mTLS secret. When you do this, the Astra Data Store operator automatically issues a new certificate. You can revoke an Astra Data Store certificate at any time.

If you revoke a CA certificate, this will revoke any certificates signed by that CA.

Steps

1. Log in to the controller node in the Astra Data Store cluster.
2. List the existing certificates on the system. For example:

```
kubectl get secrets -n astrads-system | grep astrads-cert
```

   The output should be similar to the following:

```
astrads-cert-astrads-cluster-controller
kubernetes.io/tls     4      6d6h
astrads-cert-astrads-cluster-f23d158
kubernetes.io/tls     4      6d6h
astrads-cert-astrads-ds-dms-astrads-cluster-f23d158
kubernetes.io/tls     4      6d6h
astrads-cert-astrads-ds-support-astrads-cluster-f23d158
kubernetes.io/tls     4      6d6h
astrads-cert-astrads-support-astrads-cluster-f23d158
kubernetes.io/tls     4      6d6h
astrads-cert-root
kubernetes.io/tls     4      6d6h
astrads-cert-sti-net-com
kubernetes.io/tls     5      6d6h
```

3. In the output, note the name of the certificate you need to revoke.
4. Use the `kubectl` utility to revoke the certificate, replacing `CERTIFICATE_NAME` with the name of the certificate. For example:

```
kubectl delete secret CERTIFICATE_NAME -n astrads-system
```

Result

The existing certificate is revoked, and a new certificate is automatically generated in its place.

Manage external keys

You can use one or more external key management servers to secure the keys that the cluster uses to access encrypted data. An external key management server is a third-party system in your storage environment that serves keys to nodes using the Key Management Interoperability Protocol (KMIP).

Astra Data Store enables Software Encryption at Rest (SEAR) with an internal key provider by default when the Astra Data Store cluster is created.

Managing keys involves the following custom resource definitions (CRDs):

- **AstraDSKeyProvider**: Configures an external KMIP server, which could be a cluster of servers.
- **AstraDSSEARKeyRotate**: Gets a new key encryption key from the key provider and provides it to Astra Data Store.

You can perform the following tasks related to external key management:

- Set up external key management
- Check the software encryption at rest status
- Change external to internal key management
- Rotate keys for security

Set up external key management

Setting up external key management in Astra Data Store uses `kubectl astrads` commands.

What you'll need

You will need an SSL certificate on the cluster or KMIP server that enables you to set up external keys, for example, by using OpenSSL.

Steps

1. Prepare the certificate for the key provider client. Include the client certificate, client private key, and trust CA bundles.

   You'll prepare the SSL certificate on the cluster or KMIP server that allows you to set up external keys, for example, by using OpenSSL.

2. Log in to one of the nodes in the Astra Data Store cluster.
3. Configure the key provider for the Astra Data Store cluster by entering the following kubectl extension command:

```
kubectl-astrads key-provider certs --key key.pem
--client-cert client_cert.pem --ca-cert server_ca.pem
--hostnames=<kmip_server_ip> <key_provider_cr_name>
--namespace astrads-system --cluster <ads_cluster_name>
```

Example

The following example configures an external key provider named "hashicorp" for ADS Cluster "astrads-cluster-f23d158".

```
kubectl-astrads key-provider certs --key key.pem
--client-cert client_cert.pem --ca-cert server_ca.pem
--hostnames=10.235.nnn.nnn hashicorp
--namespace astrads-system --cluster astrads-cluster-f23d158
```

1. Configure the Astra Data Store cluster to use an external key manager for SEAR via the AstraDSCluster CR. Display the help.

   ```
   kubectl-astrads clusters sears -h
   ```

   Response:

```
Configure SEARS in AstraDS cluster

Usage:
  astrads clusters sears [flags]

Flags:
  -d, --duration string   Duration for key rotation (default "2160h")
  -h, --help              help for sears


Global Flags:
      --ads-cluster-name string        Name of the ADS Cluster
      --ads-cluster-namespace string   Namespace of the ADS Cluster
      ...
```

Example

The following command configures the Astra Data Store cluster to use `AstraDSKeyProvider hashicorp` as the key manager of SEAR. The command also uses the key rotate time, which has the default value of 90 days (2160 hours).

```
kubectl-astrads clusters sears -d 500h hashicorp
--ads-cluster-name=astrads-cluster-f23d158
--ads-cluster-namespace=astrads-system
```

Check the software encryption at rest status

You can check the configuration of the software encryption at rest.

Step

1. Inspect the AstraDSCluster CR.

```
Name:         astrads-cluster-f23d158
Namespace:    astrads-system
Labels:       <none>
Annotations:  <none>
API Version:  astrads.netapp.io/v1beta1
Kind:         AstraDSCluster
...
Spec:
...
  Software Encryption At Rest:
    Ads Key Provider:     hashicorp
    Key Rotation Period:  500h0m0s
...
Status:
...
  Software Encryption At Rest Status:
    Key Active Time:       2022-05-16T15:53:47Z
    Key Provider Name:     hashicorp
    Key Provider UUID:     ccfc2b0b-dd98-5ca4-b778-99debef83550
    Key UUID:              nnnnnnnn-nnnn-nnnn-nnnn-nnnnnnnnnnnn
```

Change external to internal key management

If you are currently using an external key manager, you can change it to an internal key manager.

Steps

1. Change the AstraDSCluster CR by removing the SoftwareEncryptionAtRest configuration.
2. (Optional) Delete the previous AstraDSKeyProvider and associated secret.

> The previous key provider and secret will not be removed automatically.

Rotate keys for security

Key rotation enhances security. By default, Astra Data Store rotates keys automatically every 90 days. You can change the default setting. Additionally, you can rotate keys on demand when you want.

Configure automatic key rotation

1. Update the AstraDSSEARKeyRotate parameter in the CRD.

```
kubectl patch astradscluster astrads-cluster-f23d158
-n astrads-system
--type=merge -p '{"spec": {"softwareEncryptionAtRest": {
"keyRotationPeriod": "3000h"}}}'
```

Configure on-demand key rotation

1. Create an AstraDSSEARKeyRotateRequest CR to rotate keys.

```
cat << EOF | kubectl apply -f  -
apiVersion: astrads.netapp.io/v1beta1
kind: AstraDSSEARKeyRotateRequest
metadata:
  name: manual
  namespace: astrads-system
spec:
  cluster: astrads-cluster-f23d158
EOF
```