



# Get started with external key management

Element Software

NetApp  
June 11, 2021

This PDF was generated from [https://docs.netapp.com/us-en/element-software/storage/task\\_system\\_manage\\_key\\_set\\_up\\_external\\_key\\_management.html](https://docs.netapp.com/us-en/element-software/storage/task_system_manage_key_set_up_external_key_management.html) on June 11, 2021. Always check [docs.netapp.com](https://docs.netapp.com) for the latest.

# Table of Contents

- Get started with external key management ..... 1
  - Set up external key management ..... 1
  - Rekey software encryption at rest master key ..... 2
  - Recover inaccessible or invalid authentication keys ..... 5
  - External key management API commands ..... 5

# Get started with external key management

External key management (EKM) provides secure Authentication Key (AK) management in conjunction with an off-cluster external key server (EKS). The AKs are used to lock and unlock Self Encrypting Drives (SEDs) when [encryption at rest](#) is enabled on the cluster. The EKS provides secure generation and storage of the AKs. The cluster utilizes the Key Management Interoperability Protocol (KMIP), an OASIS defined standard protocol, to communicate with the EKS.



Only software encryption at rest is available for SolidFire Enterprise SDS clusters.

- [Set up external management](#)
- [Rekey software encryption at rest master key](#)
- [Recover inaccessible or invalid authentication keys](#)
- [External key management API commands](#)

## Find more information

- [CreateCluster API that can be used to enable software encryption at rest](#)
- [NetApp SolidFire Resources Page](#)
- [Documentation for earlier versions of NetApp SolidFire and Element products](#)

## Set up external key management

You can follow these steps and use the Element API methods listed to set up your external key management feature.

### What you'll need

- If you are setting up external key management in combination with software encryption at rest, you have enabled software encryption at rest using the [CreateCluster](#) method on a new cluster that does not contain volumes.

### Steps

1. Establish a trust relationship with the External Key Server (EKS).
  - a. Create a public/private key pair for the Element cluster that is used to establish a trust relationship with the key server by calling the following API method: [CreatePublicPrivateKeyPair](#)
  - b. Get the certificate sign request (CSR) which the Certification Authority needs to sign. The CSR enables the key server to verify that the Element cluster that will be accessing the keys is authenticated as the Element cluster. Call the following API method: [GetClientCertificateSignRequest](#)
  - c. Use the EKS/Certificate Authority to sign the retrieved CSR. See third-party documentation for more information.
2. Create a server and provider on the cluster to communicate with the EKS. A key provider defines where a key should be obtained, and a server defines the specific attributes of the EKS that will be communicated with.
  - a. Create a key provider where the key server details will reside by calling the following API method:

## CreateKeyProviderKmpip

- b. Create a key server providing the signed certificate and the public key certificate of the Certification Authority by calling the following API methods: [CreateKeyServerKmpip](#) [TestKeyServerKmpip](#)

If the test fails, verify your server connectivity and configuration. Then repeat the test.

- c. Add the key server into the key provider container by calling the following API methods: [AddKeyServerToProviderKmpip](#) [TestKeyProviderKmpip](#)

If the test fails, verify your server connectivity and configuration. Then repeat the test.

3. Do one of the following as a next step for encryption at rest:

- a. (For hardware encryption at rest) Enable [hardware encryption at rest](#) by providing the ID of the key provider that contains the key server used for storing the keys by calling the [EnableEncryptionAtRest](#) API method.



You must enable encryption at rest via the [API](#). Enabling encryption at rest using the existing Element UI button will cause the feature to revert to using internally generated keys.

- b. (For software encryption at rest) In order for [software encryption at rest](#) to utilize the newly created key provider, pass the key provider ID to the [RekeySoftwareEncryptionAtRestMasterKey](#) API method.

## Find more information

- [Enable and disable encryption for a cluster](#)
- [NetApp SolidFire Resources Page](#)
- [Documentation for earlier versions of NetApp SolidFire and Element products](#)

## Rekey software encryption at rest master key

You can use the Element API to rekey an existing key. This process creates a new replacement master key for your external key management server. Master keys are always replaced by new master keys and never duplicated or overwritten.

You might need to rekey as part of one of the following procedures:

- Create a new key as part of a change from internal key management to external key management.
- Create a new key as a reaction to or as protection against a security-related event.



This process is asynchronous and returns a response before the rekey operation is complete. You can use the [GetAsyncResult](#) method to poll the system to see when the process has completed.

### What you'll need

- You have enabled software encryption at rest using the [CreateCluster](#) method on a new cluster that does not contain volumes and has no I/O. Use [GetSoftwareEncryptionatRestInfo](#) to confirm that the state is `enabled` before proceeding.
- You have [established a trust relationship](#) between the SolidFire cluster and an External Key Server (EKS).

Run the [TestKeyProviderK mip](#) method to verify that a connection to the key provider is established.

### Steps

1. Run the [ListKeyProvidersK mip](#) command and copy the key provider ID (`keyProviderID`).
2. Run the [RekeySoftwareEncryptionAtRestMasterKey](#) with the `keyManagementType` parameter as `external` and `keyProviderID` as the ID number of the key provider from the previous step:

```
{
  "method": "rekeysoftwareencryptionatrestmasterkey",
  "params": {
    "keyManagementType": "external",
    "keyProviderID": "<ID number>"
  }
}
```

3. Copy the `asyncHandle` value from the [RekeySoftwareEncryptionAtRestMasterKey](#) command response.
4. Run the [GetAsyncResult](#) command with the `asyncHandle` value from the previous step to confirm the change in configuration. From the command response, you should see that the older master key configuration has been updated with new key information. Copy the new key provider ID for use in a later step.

```

{
  "id": null,
  "result": {
    "createTime": "2021-01-01T22:29:18Z",
    "lastUpdateTime": "2021-01-01T22:45:51Z",
    "result": {
      "keyToDecommission": {
        "keyID": "<value>",
        "keyManagementType": "internal"
      },
      "newKey": {
        "keyID": "<value>",
        "keyManagementType": "external",
        "keyProviderID": <value>
      },
      "operation": "Rekeying Master Key. Master Key management being transferred from Internal Key Management to External Key Management with keyProviderID=<value>",
      "state": "Ready"
    },
    "resultType": "RekeySoftwareEncryptionAtRestMasterKey",
    "status": "complete"
  }
}

```

5. Run the `GetSoftwareEncryptionatRestInfo` command to confirm that new key details, including the `keyProviderID`, have been updated.

```

{
  "id": null,
  "result": {
    "masterKeyInfo": {
      "keyCreatedTime": "2021-01-01T22:29:18Z",
      "keyID": "<updated value>",
      "keyManagementType": "external",
      "keyProviderID": <value>
    },
    "rekeyMasterKeyAsyncResultID": <value>
  },
  "status": "enabled",
  "version": 1
}

```

**Find more information**

- [Manage storage with the Element API](#)
- [NetApp SolidFire Resources Page](#)
- [Documentation for earlier versions of NetApp SolidFire and Element products](#)

## Recover inaccessible or invalid authentication keys

Occasionally, an error can occur that requires user intervention. In the event of an error, a cluster fault (referred to as a cluster fault code) will be generated. The two most likely cases are described here.

### **The cluster is unable to unlock the drives due to a KmipServerFault cluster fault.**

This can occur when the cluster first boots up and the key server is inaccessible or the required key is unavailable.

1. Follow the recovery steps in the cluster fault codes (if any).

### **A sliceServiceUnhealthy fault might be set because the metadata drives have been marked as failed and placed into the "Available" state.**

Steps to clear:

1. Add the drives again.
2. After 3 to 4 minutes, check that the `sliceServiceUnhealthy` fault has cleared.

See [cluster fault codes](#) for more information.

## External key management API commands

List of all of the APIs available for managing and configuring EKM.

Used for establishing a trust relationship between the cluster and external customer-owned servers:

- `CreatePublicPrivateKeyPair`
- `GetClientCertificateSignRequest`

Used for defining the specific details of external customer-owned servers:

- `CreateKeyServerKmip`
- `ModifyKeyServerKmip`
- `DeleteKeyServerKmip`
- `GetKeyServerKmip`
- `ListKeyServersKmip`
- `TestKeyServerKmip`

Used for creating and maintaining key providers which manage external key servers:

- `CreateKeyProviderKmip`

- DeleteKeyProviderKmip
- AddKeyServerToProviderKmip
- RemoveKeyServerFromProviderKmip
- GetKeyProviderKmip
- ListKeyProvidersKmip
- RekeySoftwareEncryptionAtRestMasterKey
- TestKeyProviderKmip

For information about the API methods, see [API reference information](#).



## Copyright Information

Copyright © 2021 NetApp, Inc. All rights reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means-graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system- without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).

## Trademark Information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.