Encryption and Keys

Introduction

Encryption is used in midPoint to protect sensitive parts of the database such as passwords. Such information is stored in the database in an encrypted form. Therefore anyone with an access to the database (or database backups) cannot access information that is protected by encryption. MidPoint transparently encrypts and decrypts the information as needed.

Keystore

The encryption key is **not** stored in the database (storing encrypted data at the same place as key would be really meaningless). The key is stored in standard Java JCE keystore that is located in midPoint home directory by default. It can be achieved by `keytool` utility.

See Keystore Configuration page for more information about midPoint keystore.

Managing The Keys

Encryption and Decryption Keys

MidPoint can work with several keys at once. It is using a single **encryption** key. This key is used to encrypt any new data that needs to be encrypted. There may be also many **descryption** keys. These are used to decrypt data that were encrypted previously. Each encrypted piece of data contain identification of a key that was used to encrypt it therefore midPoint can decrypt them even if the default encryption key is changed. This works as long as the original key remains in the keystore.

This approach provides an ability to change keys gradually. The procedure usually consists of following steps:

1. Add new key, make it a default encryption key (see below)
2. Keep the old key in the keystore
3. The protected data are usually aging. Passwords are expiring and are being changed. New values will be encrypted with the new key, the old key is gradually phased out of usage.
4. When the old key is no longer used you may remove it from the keystore.

Initial Key

First start of midPoint generates a initial encryption key. However it generates a short encryption key that is suitable both for use by export-limited and full-strength cryptography modules which is 128-bit AES key by default. Therefore is full-strength JCE extension was installed it is recommended to change the encryption key to a full-strength key.

Adding New Encryption Key

To add a new encryption key follow this procedure:

First stop web container. E.g. to stop tomcat in Linux:

```
/etc/init.d/tomcat stop
```

Generate new key with a `keytool` command:

```
keytool -genseckey -alias strong -keystore /var/opt/midpoint/keystore.jceks -storetype jceks -storepass
changeit -keyalg AES -keysize 256 -keypass midpoint
```

That command will create a new 256-bit AES key with alias `strong`. Now reconfigure midPoint to use the new key. Edit the `config.xml` file to change encryption key alias. Also make sure a strong algorithm is specified:

/var/opt/midpoint/config.xml

```
...
        <keystore>
            <keyStorePath>${midpoint.home}/keystore.jceks</keyStorePath>
            <keyStorePassword>changeit</keyStorePassword>
            <encryptionKeyAlias>strong</encryptionKeyAlias>
            <xmlCipher>http://www.w3.org/2001/04/xmlenc#aes256-cbc</xmlCipher>
        </keystore>
...
```

Start web container again:

```
/etc/init.d/tomcat start
```

Migrating Keys Between Servers

```
keytool -importkeystore -srckeystore keystore-old.jceks -srcstoretype jceks -destkeystore /var/opt/midpoint
/keystore.jceks -deststoretype jceks -srcstorepass changeit -deststorepass changeit -srcalias default -
destalias oldkey -srckeypass midpoint -destkeypass midpoint
```