MidPoint Advanced Customization

MID-102

revision 4.0-LTS-B

Course Goals

- Use advanced resource features such as account uniqueness, dependencies, provisioning scripts and delayed delete
- Understand the generic synchronization and configure synchronization and provisioning of objects other than accounts
- Configure role request and approvals processes

Course Goals (2)

- Configure advanced security features such as, password hashing, account activation, password reset and segregation of duties
- Authorization reprise and enhancements
- Enhance object templates with advanced features including unique user name generation
- Configure and use basic reports

Course Goals (3)

- Use bulk actions for scripted data modification in midPoint repository and provisioning to other systems
- Modify midPoint look&feel

Course Map

Module 1

MidPoint Deployment Fundamentals Training Review

Module 2

Advanced Resource Features

Module 3

Generic Synchronization

Module 4

Role Request And Approval Processes

Module 5

Advanced Security Features

Module 6

Advanced midPoint Features

Course Map (2)

Module 7	Module 8	Module 9
Reporting Basics	Bulk Actions	GUI Customization

Note

- This is a **sample** of our training materials
- Please contact **sales@evolveum.com** to order a real training course session

Module 5: Advanced Security Features

Passwords in midPoint

- User object – password and history (`credentials/password/*`)
- Resource configuration (property name varies)
- Shadow (only in memory, not saved to repository unless the operation is pending)

Password Storage

- Passwords are encrypted by default
- Password history is hashed by default (since 3.6)
- Encryption key stored on disk, not in repository
  - `$midpoint.home/keystore.jceks` file
- Default random encryption key generated upon first midPoint initialization

Password Storage - Encryption

- Reversible encryption (symmetric key)
- Used to connect to resources
- Password can be passed to resources (outbound)
- Can be used in expressions:

```
midpoint.getPlaintextUserPassword()
```

Password Hashing

- Improves security by using hashed password for Users in midPoint
- Security Policy with hashing storage set as global
  - Applied from *now* (does not touch existing passwords!)
- Self-service still possible for users
- System Configuration (`accountActivationNotifier`)
- Account Activation Procedure

Password Hashing Algorithm

Algorithm	PBKDF2 with HMAC SHA512
Key size (bits)	256
Work factor (iterations)	10 000
Salt size (bits)	32

Security Policy with Hashing

```
<securityPolicy oid="076eabee-332d-11e8-8087-f3c9c7e9809d"
xmlns='http://midpoint.evolveum.com/xml/ns/public/common/common-3'>
  <name>ExAmPLE Security Policy with Password Hashing</name>
  <credentials>
    <password>
      <storageMethod>
        <storageType>hashing</storageType>
      </storageMethod>
      <maxAge>P180D</maxAge>
      <lockoutMaxFailedAttempts>3</lockoutMaxFailedAttempts>
<lockoutFailedAttemptsDuration>PT3M</lockoutFailedAttemptsDuration>
      <lockoutDuration>PT15M</lockoutDuration>
      <valuePolicyRef oid="10000000-9999-9999-0000-a000f2000002"/>
    </password>
  </credentials>
</securityPolicy>
```

System Configuration – Infrastructure

```
<systemConfiguration . . .>
. . .

 <infrastructure>
  <defaultHostname>http://192.168.56.20:8080/midpoint</defaultHostname>
 </infrastructure>

 <notificationConfiguration>
   <handler>
     <accountActivationNotifier>
. . .<!-- see next slide -->
     </accountActivationNotifier>
 </handler>
. . .
 </notificationConfiguration>
. . .
</systemConfiguration>
```

System Configuration – AccountActivationNotifier

```
<accountActivationNotifier>
  <subjectExpression><script>
    <code>return "[IDM] Activate your account(s)"</code>
  </script></subjectExpression>
  <recipientExpression>
  <runAsRef oid="00000000-0000-0000-0000-000000000002"/>
    <script>
      <code>
requesteeEmail = requestee?.getEmailAddress()
if (basic.isEmpty(requesteeEmail)) {
  managers = midpoint.getManagersOidsExceptUser(requestee)
  if (!basic.isEmpty(managers)) {
    m = midpoint.getUserByOid(managers[0])
    return m?.getEmailAddress()
  }
} else return requesteeEmail
      </code>
    </script>
  </recipientExpression>
  <transport>mail</transport>
  <confirmationMethod>link</confirmationMethod>
</accountActivationNotifier>
```

Synchronization from Source with Hashing

- Synchronization from source generates random password (inbound mapping)
- MidPoint can access the password during the operation (in memory)
- Target accounts can be created and initialized without an explicit account activation

Self-Service Password Change with Hashing

- Even with password hashing, self-service is possible
- User will set the new password, which is known during the operation
- Target accounts passwords can be modified without an explicit account activation

Account Activation Procedure

- Creating account *after* the user is created requires account activation
- Account is created:
  - With either generated or empty password (we can't access the user password)
  - Shadow: `lifecycleState=proposed`
- Notification is sent to the user (with activation link)
  - The same link can be used to activate several accounts

Account Activation Procedure (2)

- User clicks the link and enters the midPoint password:
  - Authentication
  - Password provisioning (midPoint knows the password now)
  - Shadow is updated: `lifecycleState=active`

Lab 5-1: Password Hashing and Account Activation

Lab 5-2: Password Validation with checkExpression

Lab 5-3: E-mail-based Password Reset

Lab 5-4 (BONUS): Segregation of Duties Using Archetypes/Metaroles

Lab 5-5 (BONUS): Maximum Assignees with Policy Rule

Module Summary

- Password storage: encryption vs. hashing
- Account activation process
- Password reset using e-mail
- Disallowed password values using expressions
- Authorizations
- Policy Rules usage for non-approval scenarios

End of Module 5

Conclusion

Questions and Answers

Follow Evolveum

- **Web**: www.evolveum.com
- **Blogs**: https://evolveum.com/blog/
- **Twitter**: https://twitter.com/evolveum
- **FB**: https://www.facebook.com/evolveum