SIEMENS

User Management Component 2.9.2

UMX User Manual

Contents

09/2020
A5E50361210-AA

Guidelines

This manual contains notes of varying importance that should be read with care; i.e.:

Important:

Highlights key information on handling the product, the product itself or to a particular part of the documentation.

Note: Provides supplementary information regarding handling the product, the product itself or a specific part of the documentation.

Trademarks

All names identified by ® are registered trademarks of Siemens AG.

The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Security information

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, systems, machines and networks. In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial security concept. Siemens' products and solutions only form one element of such a concept.

Customer is responsible to prevent unauthorized access to its plants, systems, machines and networks. Systems, machines and components should only be connected to the enterprise network or the internet if and to the extent necessary and with appropriate security measures (e.g. use of firewalls and network segmentation) in place.

Additionally, Siemens' guidance on appropriate security measures should be taken into account. For more information about industrial security, please visit https://www.siemens.com/industrialsecurity.

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends to apply product updates as soon as available and to always use the latest product versions. Use of product versions that are no longer supported, and failure to apply latest updates may increase customer's exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed under https://www.siemens.com/industrialsecurity.

Siemens AG
Digital Industries
Postfach 48 48
90026 NÜRNBERG
GERMANY

A5E50361210-AA
20200903_45788

Contents

1 UMX Overview

The **umx** utility can be used to manage users, groups, roles and account policies of the User Management Component (UMC). According to the selected options (switches) and the related parameters, the utility allows you to execute different commands. When describing the commands, we do not provide an explicit description of the switches when they identify the parameter they precede.

This utility is distributed with UMC and it is installed in the subdirectory \BIN (for example C:\Program Files\Siemens\UserManagement\BIN) and must be executed from a command prompt within this directory.

In addition, the execution of a **umx** command can be performed interactively, see interactive command.

Important:

As **umx** is a command line utility, if you want to insert a parameter with blank spaces you have to enclose it between double quotes.

Users Allowed to Execute umx Commands

The execution of a **umx** command can be performed according to two login modes:

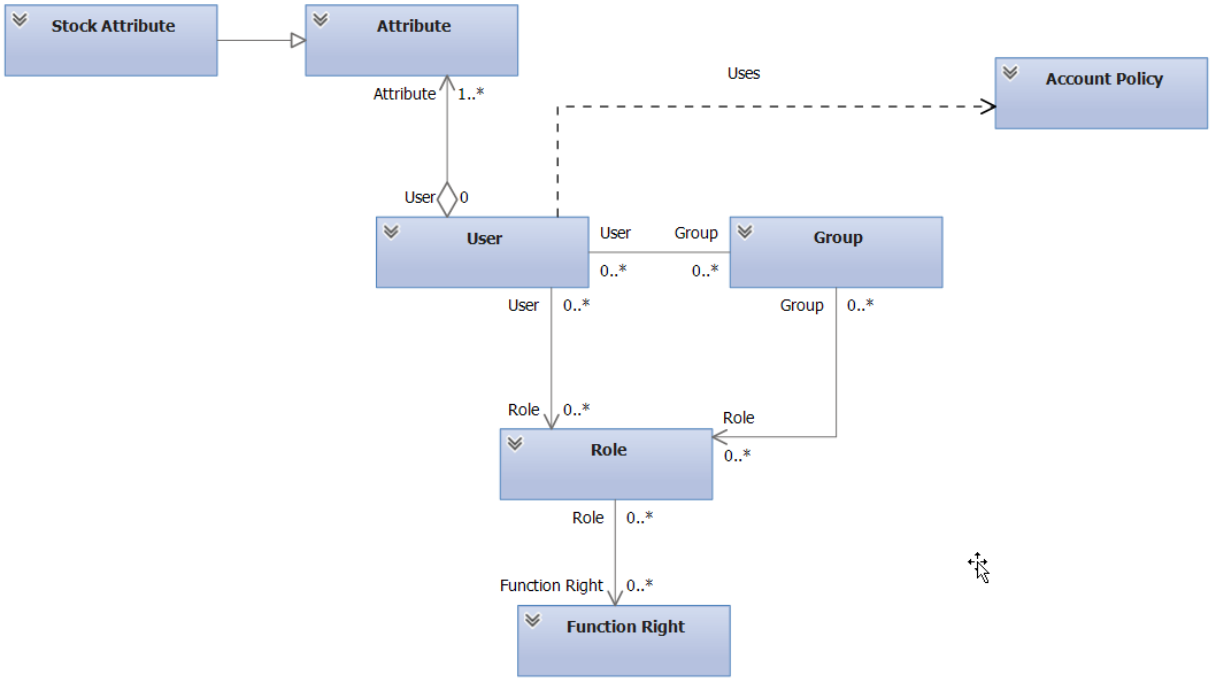
- **current user**: the user that performs the UMC command is the Windows user which has launched the windows command prompt;
- **specified user**: the user that performs the UMC command is explicitly inserted in the command line; -x switch has to be inserted as first parameter when launching the command; username and password have to be inserted as second and third parameter.

Regardless of the login mode which is used the execution of commands is limited according to the function rights of the user, for example:

- if the User Manager user owns the **UM_ADMIN** function right, they can execute all the **umx** commands;
- users who own the **UM_VIEW** function right can only execute commands which access the database with read only access rights;
- users who own the **UM_VIEW** function right plus the User Manager function right that is required for the specific of the action can execute the relative action.

See User Manager Function Rights for a list of UM function rights.

The following diagram presents the UMC object model whose understanding helps in using the **umx** utility.


Stock Attribute
Attribute
Attribute
1..*
Uses
Account Policy
User
0
User
Group
Group
0..*
0..*
User
0..*
Group
0..*
Role
0..*
Role
0..*
Role
0..*
Function Right
0..*
Function Right

2 Concepts You Need to Know About

The following concepts are the basics you need to know before you start configuring UMC:

- UM User
- User Manager Group
- User Manager Role
- User Manager Function Rights

2.1 User Manager User

A User Manager user (UM user in what follows) is a user in the User Manager Component database, identified by a user name. Note that UM users are different entities with respect to Windows users, which are defined at operating system level.

Custom attributes can be associated with UM users. Example of custom attributes are common user properties such as phone number, department, and so on.
To apply Secure Application Data Support (SADS), access to encrypted application data can be granted to authorized users to allow them to decrypt it using specific Subject Keys.

UM User Types

You can distinguish three types of UM users:

- **users created from scratch** in UMC or created via csv file;
- **Windows local users** that are imported into UMC (via umx): in this case the user name follows the pattern <*machineName*>\<*localUserName*>;
- **Active Directory users** that are imported into UMC (via umx or via Web UI): in this case the user name follows the pattern <*ADdomainName*>\<*ADuserName*>.

UM User Passwords

Users created within UMC have also an associated password. Empty passwords are not allowed. Users imported from Windows authenticate against Windows and do not have a UMC password. Imported Windows local users authenticate **only** locally against Windows on the machine where they are present. They can be used **only** for configuration purposes, for instance to be associated with a Windows service running on the machine.

Offline Users

When you create a UMC user you can flag the user as *offline*. UMC provisioning service checks if the offline user exists in Active Directory:

- if the user is present, user data are synchronized and the user becomes online,
- otherwise the user remains offline.

Important:

Users created as *offline* are enabled by design: they can therefore perform the actions allowed by their function rights.

The user name of offline users must follow the AD pattern *<domainName>\<ADuserName>*. They do not have a UMC password, as they cannot authenticate until they become online. The User Security Identifier (SID, see Microsoft Documentation on Security Identifiers for more details) property is set to a default value (S-1-0-0) that is synchronized with the actual AD value by the UMC provisioning service.

Users are also flagged *offline* if they are deleted from AD. In this case users are permanently deleted from UMC database after an amount of time that can be configured (default is12 hours). See the additional provisioning configuration in the *User Management Component Installation Manual* for more details.

User Limits

Description	Maximum
Number of groups assigned to a user	50
Number of roles assigned to a user	50

2.2 User Manager Group

A User Manager group (UM group in what follows) is a container of users and is identified by a name. Note that UM groups are different entities with respect to Windows groups that are defined at operating system level.

To apply Secure Application Data Support (SADS), access to encrypted application data can be granted to authorized groups to allow them to decrypt it using specific Subject Keys.

UM Group Types

There are two types of UM groups:

- **groups created from scratch** in UMC or created via csv file;
- **Active Directory groups** that are imported into UMC (via umx or via Web UI).

Offline Groups

When creating a UMC group, you can flag the group as *offline*. UMC provisioning service checks if the offline group exists in Active Directory:

- if the group is present, group data are synchronized, the AD users members of the groups are imported into UMC and the group becomes online,
- otherwise the group remains offline.

The group name of offline users must follow the AD pattern <*ADdomainName*>\<*ADgroupName*>. The UMC provisioning service searches for the AD group by its Common Name (CN).

If required, the description field of the created group can be used to configure how the UMC provisioning service must query the AD group and import its users into UMC. In this case the description must follow the pattern:

{{Q=<ldap query>

where {{Q= is a fixed prefix and <ldap query> is the query to be applied. The group name in this case can be <*ADdomainName*>\<*GroupName*>, where GroupName can be chosen by the user.

Group Limits

Description	Maximum
number of groups assigned to a user	50
number of roles assigned to a group	50
number of users bound to a group	1000

2.3 User Manager Role

A User Manager role (UM role in what follows) groups a set of function rights. Function rights are the capabilities to perform operations. They are associated with roles so that the set of UM users having a specific UM role is allowed to perform the set of operations associated with it. UM roles can be associated with UM users or with UM groups so that all the users belonging to such groups inherit the UM role function rights. UM roles are used to define the function rights within UMC, for instance, to define whether a user can configure UMC or not.

The following roles are automatically created by the system while configuring UMC:

- **Administrator**: built-in "root" role, can perform any operation. The user that has this role is a root user that can perform any operation. This role cannot be associated with any group. It can be associated with a user if the user performing the association has in turn the **Administrator** role. The **Administrator** role cannot be deleted. Only users having the **Administrator** role can modify other users having this role.
- **UMC Admin**: can manage users, groups and all the other UMC entities.
- **UMC Viewer**: can access the user management configuration without making modifications.

2.4 User Manager Function Rights

Function rights are the capabilities to perform operations. They are associated with roles so that the set of UM users having a specific UM role is allowed to perform the set of operations associated with it. The following table contains a list of UM Function Rights:

Name	Description
UM_ADMIN	Allows you to display the UMC database data and to configure the UMC database, that is to create users, groups and so on, to import and export data via file, to register UMC station clients. This function right allows you to execute all **umx** commands.
UM_VIEW	Allows you to display the UMC database data related to users, groups, roles and account policies.
UM_RESETPWD	The user can reset the password of another user. The user must also have associated the **UM_VIEW** function right.
UM_UNLOCKUSR	The user can unlock any other user. The user must also have associated the **UM_VIEW** function right.
UM_ATTACH	The user can attach a machine to a UM domain, the machine is promoted to the *UM agent role*.
UM_JOIN	The user can promote a machine to a *UM server role*. If the machine is not yet attached to the UM domain, it is attached. This function right incorporates the **UM_ATTACH** function right.
UM_RESETJOIN	The user can downgrade a machine from the UM ring server or UM server role to the UM agent role.
UM_IMPORT	The user can import the UM Configuration via package. The user must also have associated the **UM_VIEW** function right.
UM_EXPORT	The user can export the UM Configuration into a package. The user must also have associated the **UM_VIEW** function right.
UM_BACKUP	The user can back up the UM Configuration (Full backup). *This function right is not used, as the functionality controlled by it has not yet been implemented.*
UM_EXPORTCK	The user can export Claim Key. *This function right is not used, as the functionality controlled by it has not yet been implemented.*
UM_EXPORTDK	The user can export Domain Key. *This function right is not used, as the functionality controlled by it has not yet been implemented.*
UM_RA	Login from Remote Authentication. *This function right is not used, as the functionality controlled by it has not yet been implemented.*
UM_RINGMNG	The user can promote a machine to a *UM ring server role*. If the machine is not yet attached to the UM domain, it is attached.
UM_ADSYNC	The user can perform the background AD provisioning synchronization.
UM_VIEWELG	The user can display event logging data. The user must also have associated the **UM_VIEW** function right.
UM_CLAIMAUTH	The user can create an identity from a valid claim.

Name	Description
UM_REGCLIENT	The user can register UMC station clients.

3 How to Display UMX Information

The following command can be used to display information on UMX commands.

- Help

3.1 Help

This command displays the first level of help or the details of the different commands depending on the input parameters.

Syntax

```
umx -h [command]
```

Parameters

- *command* can be one of the command categories displayed launching umx -h; examples are *create*, *update*, etc.

4 How to Create UMC Objects

The following commands can be used to create UMC objects such as users, groups and roles.

- Create User
- Create Group
- Create Role

4.1 Create User

This command creates a new UM user setting the necessary parameters. To enable the user parameter *override lock*, you have to create the user and then perform a user update.

Syntax

Create users

```
umx [-x commandUserName commandUserPassword] -c -u name -p password [-f
fullName] [-m paramMustPwd] [-C paramCanPwd] [-l paramLock] [-e paramEnabled]
```

Create offline users

Users created as *offline* are enabled by design.

```
umx [-x commandUserName commandUserPassword] -c -u name -off
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *name* is the string representing the user name, only alphanumeric characters are allowed.
- *fullName* is the string representing the user full name, for instance the pair Name and Surname.

- *password* is the password associated to the user. Empty passwords are not allowed. The password specified for the user via this command may not comply with password policies, to enable a check passwords see Set Password Check.
- *paramMustPwd* allows the values 0 or 1. If set to 1, the user is forced to change the password at first login, if set to 0 the user is not forced to change the password at first login. Default value is 0.
- *paramCanPwd* allows the values 0 or 1. If set to 1 the user can change the password, if set to 0 the user cannot change the password. Default value is 0.
- *paramLock* allows the values 0 or 1. If set to 1 the user is locked and cannot perform any action. If set to 0 the user is unlocked and can perform the actions according to the associated function rights. The user can be locked by the system when attempting to login several times with a wrong password, the number of allowed attempts is established by the security policies. Default value is 0.
- *paramEnabled* allows the values 0 or 1. If set to 0 the user cannot perform any action, if set to 1 the user is enabled and can perform the actions according to the associated function rights. Default value is 0.

Parameter Behavior

The following table presents the different behaviors of the application depending on the values of the ***paramMustPwd*** and ***paramCanPwd*** parameters.

***paramMustPwd* value**	***paramCanPwd* value**	**Behavior**
0	0	The user cannot change the password.
0	1	The user can change the password whenever he/she wants.
1	0	The user must change the password at first login. Afterwards, he/she cannot change the password anymore.
1	1	The user must change the password at first login. Afterwards, he/she can change the password whenever he/she wants.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.
-off	If specified, the user is created as an offline user. For more details on offline users see User Manager User.

Examples

```
umx -c -u myUser -f "Peter Brown" -p default123 -m 1 -C 1 -l 0 -e 1
```

The command above creates the user myUser with full name Peter Brown, password default123. The user have to change the password at first login, is unlocked and enabled.

```
umx -c -u DOM\userOFF -off
```

The command above creates the offline user DOM\userOFF, with all the flags set to 0.

```
umx -c -u userOn -p a
```

The command above creates the offline user userOn, password a and with all the flags set to 0.

4.2 Create Group

This command creates a new UM group.

Syntax

```
umx [-x commandUserName commandUserPassword] -c -g name -d description [-off]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *name* is the string representing the group name. Only alphanumeric characters are allowed.
- *description* is the string containing a short description of the group. This field is optional if the group is created offline.

If the group is created offline the description can contain an ldap query to be used by UMC provisioning service to find the AD group and populate the UMC group with its users. See User Manager Group for details about offline groups and how to configure the import. An example of usage can be found below.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.
-d	If the description start with {{Q= the rest of the string is the ldap query for the group.
-off	If specified, the group is created as an offline group. For more details on offline groups see User Manager Group.

Examples

create an offline group by query ldap

```
umx -x manager manager -c -g UMC_domain\Group_test -d {{Q=distinguishedname=
cn=colors,ou=other_ou1,dc=umc_domain,dc=net -off
```

In the example above an offline group with the name UMC_domain\Group_test is created. T he group created will be associated with the users of the AD group obtained with the execution of the query:

&(objectCategory=group)(***distinguishedname=cn=colors,ou=other_ou1,dc=umc_domain,dc=net***)
.

It is recommended that you use this mode only if the result of the query returns a single group like in the case of searching for the group by ***distinguished name***.

4.3 Create Role

This command creates a new UM role. The number of roles present in the system cannot exceed 36200.

In addition a database constraint on the role identifiers exists. In case you get an error message that no more role identifiers are available, to create new roles, you have first to purge the existing one with the corresponding **umconf** command. See the *UMCONF User Manual* for more details.

Syntax

```
umx [-x commandUserName commandUserPassword] -c -r name -d description
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *name* is the string representing the role name, only alphanumeric characters are allowed.
- *description* is the string containing a short description of the role.

Switches

Switch	Description

-x	The command is executed by the user given as input parameter.

5 How to Update UMC Objects

The following commands can be used to update UMC objects such as users, user attributes or groups.

- Update User
- Update Group
- Update User Alias
- Update User Attribute

5.1 Update User

This command updates an existing UM user. Note that at least one of the UM user properties must be updated.

When editing user which have been imported into UMC from active directory or windows, see Limitations on imported users.

Syntax

```
umx [-x commandUserName commandUserPassword] -U -u user [-s (use username
instead of userId)] [-e expirationDate]  [-ae alertOnExDays]  [-p pwdDays]
[-ap alertOnPwdExDays]   [-al autologoffMinutes]  [-wa
warningOnAutologoffMinutes] [-la language] [-da dataLanguage] [-fu fullname]
[-co comment] [-em emailAddress] [-o paramOverrideLock] [-canchangepsw 0|1]
[-mustchangepsw 0|1]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *user* represents the user name if the switch –s is present, represents the user internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record;
- *expirationDate* is the account expiration date in Unix time format.
- *alertOnExDays* is the number of days from which a warning appears to the user notifying him/her about the user expiration.
- *pwdDays* is the number of days for which the password is valid, max 1828 days.
- *alertOnPwdExDays* is the number of days from which a warning appears to the user notifying him/her about the password expiration.

- *autologoffMinutes* is the number of minutes that must elapse before a user is automatically logged off from the system (session-based).
- *warningOnAutologoffMinutes* is the number of minutes that must elapse before a warning appears to a user to inform that he/she will be logged off from the system (session-based).
- *language* is the user language and has the format <langcode>-<countrycode> (for example *en-GB*) where:
  – langcode is the language code according to the ISO 639 standard; we accept both two-letter codes (ISO 639-1) and three-letter codes (ISO 639-2);
  – countrycode is the country code according to the ISO 3166 standard.
- *dataLanguage* is the language the language in which are displayed the user data, see above for the language.
- *fullname* is the user Full Name (include in doublequotes if it contains spaces, e.g. -fu "Full Name")
- *comment* is the user comment (include in doublequotes if it contains spaces, e.g. -co "This User is Used Only For Test")
- *emailAddress* is the user email address
- *paramOverrideLock* allows the values 0 or 1. If set to 1 the user cannot be locked. If set to 0 the user can be locked. For instance, the user can be locked by the system when attempting to login several times with a wrong password, the number of allowed attempts is established by the security global account policies. In case of value set to 1, even though the user attempts to login several times with a wrong password, he/she is not locked.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.
-s	After the "-u" switch the user name must be entered instead of the numeric user Id.

Update Limitations on Imported Users

User which have been imported from Active direct and Windows local users can be updated from the UMX but the following limitations apply:

UMX Parameter	UMC Web Name	AD Name	UMC users	Active Directory	Windows Local
expirationDate	User Expiration Date	N/A	✓	✗	✗
alertOnExDays	Alert when user is about to expire	N/A	✓	✗	✗
pwdDays	Password Duration (days)	N/A	✓	✗	✗
alertOnPwdExDays	Alert when password is about to expire	N/A	✓	✗	✗

UMX Parameter	UMC Web Name	AD Name	UMC users	Active Directory	Windows Local
language	Language	N/A	✓	✓	✓
dataLanguage	Data Language	N/A	✓	✓	✓
fullname	Full Name	Common Name	✓	✗	✓
comment	Comment	Description	✓	✗	✓
N/A	Initials	Initials	✓	✗	✗
emailAddress	Email 1	Email	✓	✗	✓
paramOverrideLock	Override Lock Policy on invalid credentials	N/A	✓	✗	✗
UMC Attributes and their values	UMC Attributes and their values	N/A	✓	✓	✓
User Alias	User Alias	N/A	✓	✓	✓

5.2 Update Group

This command updates an existing UM group. Note that at least one of the UM group properties must be updated.

Syntax

```
umx [-x commandUserName commandUserPassword] -U(pdate) -g(roup) <Id|name>
[-s (use name instead of Id)][-d(escription) <Description>][-o(verride user
lock) <0|1>] [-sadsstatus   <1(empty)|2(sync)>]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *group* represents the group ID or name, if the switch –s is present, represents the user internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record;
- *description* is the string containing a short description of the group.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.
-s	After the "-u" switch the user name must be entered instead of the numeric user Id.

5.3 Update User Alias

This command modifies the alias of the specified user.

Syntax

```
umx [-x commandUserName commandUserPassword] -U -a -v aliasName -u user [-s
(use username instead of userId)]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *aliasName* is a string representing the alias name;
- *user* represents the user name if the switch –s is present, represents the user internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record; to retrieve the user identifier see List Entity Details.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

5.4 Update User Attribute

This command updates the value of an existing attribute of a UM user. The user attribute **USER_USNCHANGED** is used by UMC provisioning for UMC AD users. If you set the value to 0 the user fields are synchronized with AD fields.

Syntax

```
umx [-x commandUserName commandUserPassword] -U -A attributeName -v
attributeNewValue -u userId
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *attributeName* is a string representing the attribute name;
- *attributeNewValue* is the attribute value;
- *userId* is a positive number representing the internal identifier of the record representing the UM user to which the given attribute and its value are associated; to retrieve the user identifier see List Entity Details.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

6 How to Retrieve Information about UMC Objects

The following commands can be used to retrieve information about UMC objects such as users, groups, roles or AT records.

- List Entity Details
- List Event Log Records

6.1 List Entity Details

This command lists the details of the selected entity (user, group, role). It can be used to retrieve the entity identifier from the entity name.

Syntax

```
umx [-x commandUserName commandUserPassword] -i {-u user [-s]| -g group [-s]
| -r role [-s]}
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *user* represents the user name if the switch –s is present, represents the user internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record;
- *group* represents the group name if the switch –s is present, represents the group internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record;
- *role* represents the role name if the switch –s is present, represents the role internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

-s	If the switch –s is present, objects are identified by their name. If the switch –s is not present, objects are identified by their internal identifier. The identifier is a positive number univocally identifying the record.

6.2 List Event Log Records

This command lists the event log records related to an input date, for the entire day starting from 00: 01. The limit is set to 1001 records per day. to retrieve all the records for a day you have to use the switch -f.

Syntax

```
umx [-x commandUserName commandUserPassword] -i -at time [-f ]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *time* is the date in Unix time format of the day for which you wish to the event log records. The string **now** represents the current date. As an example, the Unix time 1460939793 corresponds to ISO 8601: 2016-04-18T00:36:33Z.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.
-f	Forces the dump of all the records of the specified day to a file with name <unixtime>.dat which is saved in the location from which UMX is being launched.

Example

Two command examples follow.

```
umx –i –at now

umx –i –at 1450259593

umx –i –at 1460279589 –f
```

in the last example umx create a file with name 1460279589.dat with all records about Sun, 10 Apr 2016 09:13:09 GMT

An example of the output follows.

```
---- AT Records 1 ----
AT Record {
  "timestamp": "2015-12-1613:44:23.0+0100",
  "source": "",
  "username": "SWQA\\itre0043",
  "action": "login error",
  "value": {
    "result":4
             }
}
Time taken: 0.02s
```

7 How to Display Lists of UMC or Windows Objects

The following commands can be used to display a list of UMC or Windows objects such as users, groups, roles, function rights or account policies.

- List Entities
- Count Entities

7.1 List Entities

This command lists the set of entities, their details stored in the database depending on the selected switch. For the users also the attributes and their values are displayed.

Syntax

```
umx [-x commandUserName commandUserPassword] -l {-u [-v] | -wu | -du
searchStringUsers | -g | -dg searchStringGroups | -r | -f | -a | -d
domainName | -xc}
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *searchStringUsers* filters the Active Directory users to be listed, the "*" wildcard can be used, the search field is the user name.
- *searchStringGroups* filters the Active Directory groups to be listed, the "*" wildcard can be used, the search field is the group name.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.
-u	Displays the list of UM users.
-v	If this switch is present, more user details are displayed.
-wu	Displays the list of Windows local users.

-du	Displays the list of Active Directory users belonging to the domain to which the local machine is joined filtered by the search string. The first field displayed is used to import purposes. The domain name has to be specified with *-d domainName* switch.
-g	Displays the list of UM groups.
-dg	Displays the list of Active Directory groups belonging to the domain to which the local machine is joined filtered by the search string. The first field displayed is used to import purposes. The domain name has to be specified with *-d domainName* switch.
-r	Displays the list of UM roles.
-f	Displays the list of UM function rights.
-a	Displays the list of UM account policies. Note that the user associated to the provisioning service is stored as an account policy and is displayed in this list.
-d	Displays the list of Windows domains.
-xc	Displays the names and the fingerprints of the registered station clients.

Example

umx -l -du ross*

The command above displays the list of Active Directory users belonging to the domain to which the local machine is joined whose user name starts with the string "ross".

7.2 Count Entities

This command lists the number of the different main entities stored in the database: number of UM users, number of UM roles, number of UM groups and number of function rights.

Syntax

```
umx [-x commandUserName commandUserPassword] -k
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

8 How to Delete UMC Objects

The following command can be used to delete UMC objects such as users, groups or roles.

- Delete Entity

8.1 Delete Entity

This command deletes the entity given in input.

Syntax

```
umx [-x commandUserName commandUserPassword] -d {-u user | -g group | -r
role | -a} [-f][-s]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *user* represents the user name if the switch –s is present, represents the user internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record;
- *group* represents the group name if the switch –s is present; it represents the group internal identifier if the switch –s is not present. The identifier is a positive number univocally identifying the record;
- *role* represents the role name if the switch –s is present, represents the role internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record;

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.
-a	All the UM users, groups and roles are deleted except for the UM user launching the command, the UM administrator and the Administrator role.

-s	If the switch –s is present, objects are identified by their name. If the switch –s is not present, objects are identified by their internal identifier. The identifier is a positive number univocally identifying the record.
-f	Forces the deletion of the user launching the command.

9 How to Perform Binding / Unbinding Commands

The following commands can be used to perform binding or unbinding operations on UMC objects such as adding/removing a user to a group/role, adding/removing user attributes or adding/removing function rights to a role.

Binding Commands

Unbinding Commands

9.1 Add Attribute to User

This command adds an attribute and its value to a UM user.

Syntax

```
umx [-x commandUserName commandUserPassword] -a -A attributeName -v
attributeValue -u user [-s (use username instead of userId)]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *attributeName* is a string representing the attribute name;
- *attributeValue* is the attribute value;
- *user* represents the user name if the switch –s is present, represents the user internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record; to retrieve the user identifier see List Entity Details.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

9.2 Add Attribute to User - Size

This command is only for testing purposes. It adds an attribute of a predefined size to a UM user. A default value is given to the attribute.

Syntax

```
umx [-x commandUserName commandUserPassword] -a –A attributeName -S size -u
user [-s (use username instead of userId)]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *attributeName* is a string representing the attribute name;
- *size* is the attribute size in bytes;
- *user* represents the user name if the switch –s is present, represents the user internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record; to retrieve the user identifier see List Entity Details .

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

9.3 Add a Set of Attributes to User

This command is only for testing purposes. It adds a set of attributes of a given size to a UM user. A default value is given to the attributes.

Syntax

```
umx [-x commandUserName commandUserPassword] -a –A namePrefix -n number –S
size –u user [-s (use username instead of userId)]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *namePrefix* is the prefix used for the name of the set of attributes;
- *number* is the number of attributes to add;
- *size* is the attribute size in bytes;
- *user* represents the user name if the switch –s is present, represents the user internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record; to retrieve the user identifier see List Entity Details .

Example

umx -a -A testAtt –n 10 –s 20 -u myUser
The command above creates ten attributes for the user myUser. The attributes are named testAtt1, testAtt2, and so on, the size of each attribute is 20 bytes.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

9.4 Add Alias to User

This command adds an alias to a UM user. Only one alias per user is supported.

Syntax

```
umx [-x commandUserName commandUserPassword] -a -a -v aliasName -u user [-s
(use username instead of userId)]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *aliasName* is a string representing the alias name;
- *user* represents the user name if the switch –s is present, represents the user internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record; to retrieve the user identifier see List Entity Details.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

9.5 Assign Group/Role to User

This command adds a user to a group or assigns a role to a user.

Syntax

```
umx [-x commandUserName commandUserPassword] -a -u user {-g group| -r role}
[-s]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *user* represents the user name if the switch –s is present, represents the user internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record;
- *group* represents the group name if the switch –s is present; it represents the group internal identifier if the switch –s is not present. The identifier is a positive number univocally identifying the record;
- *role* represents the role name if the switch –s is present, represents the role internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record;

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.
-s	If the switch –s is present, objects are identified by their name. If the switch –s is not present, objects are identified by their internal identifier. The identifier is a positive number univocally identifying the record.

9.6 Assign Role to Group

This command assigns a role to a group.

Syntax

```
umx [-x commandUserName commandUserPassword] -a -g group -r role [-s]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *group* represents the group name if the switch –s is present; it represents the group internal identifier if the switch –s is not present. The identifier is a positive number univocally identifying the record;

- *role* represents the role name if the switch –s is present, represents the role internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record;

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.
-s	If the switch –s is present, objects are identified by their name. If the switch –s is not present, objects are identified by their internal identifier. The identifier is a positive number univocally identifying the record.

9.7 Assign Function Right to Role

This command assigns a function right to a role.

Syntax

```
umx [-x commandUserName commandUserPassword] -a –f functionRightName -r role
[-s]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *functionRightName* is the name of a UM function right. For the list of the UM function rights see User Manager Function Rights.
- *role* represents the role name if the switch –s is present, represents the role internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record;

Switches

Switch	Description
-x	The command is executed by the user given as input parameter..
-s	If the switch –s is present, objects are identified by their name. If the switch –s is not present, objects are identified by their internal identifier. The identifier is a positive number univocally identifying the record.

9.8 Remove User from Group/Role

This command removes a UM user from a group or deletes the association of a UM role to a UM user.

Syntax

```
umx [-x commandUserName commandUserPassword] -R -u user {-g group | -r role}
[-s]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *user* represents the user name if the switch –s is present, represents the user internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record;
- *group* represents the group name if the switch –s is present; it represents the group internal identifier if the switch –s is not present. The identifier is a positive number univocally identifying the record;
- *role* represents the role name if the switch –s is present, represents the role internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record;

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.
-s	If the switch –s is present, objects are identified by their name. If the switch –s is not present, objects are identified by their internal identifier. The identifier is a positive number univocally identifying the record.

9.9 Remove Role from Group

This command deletes the association of a role to a group.

Syntax

```
umx [-x commandUserName commandUserPassword] -R -g group -r role [-s]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *group* represents the group name if the switch –s is present; it represents the group internal identifier if the switch –s is not present. The identifier is a positive number univocally identifying the record;
- *role* represents the role name if the switch –s is present, represents the role internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record;

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.
-s	If the switch –s is present, objects are identified by their name. If the switch –s is not present, objects are identified by their internal identifier. The identifier is a positive number univocally identifying the record.

9.10 Remove Attribute from User

This command removes an attribute and its value from a user.

Syntax

```
umx [-x commandUserName commandUserPassword] -R -A attributeName -u userId
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;

- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *attributeName* is a string representing the attribute name;
- *userId* is a positive number representing the internal identifier of the record representing the UM user to which the given attribute and its value are associated; to retrieve the user identifier see List Entity Details .

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

9.11 Remove Alias from User

This command removes the alias from a user.

Syntax

```
umx [-x commandUserName commandUserPassword] -R -a -u userId [-s (use
username instead of userId)]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *userId* is a positive number representing the internal identifier of the record representing the UM user to which the given attribute and its value are associated; to retrieve the user identifier, see List Entity Details.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

9.12 Remove Function Right from Role

This command deletes the association of a function right to a role.

Syntax

```
umx [-x commandUserName commandUserPassword] -R -f functionRightName -r role
[-s]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *functionRightName* is the name of a UM function right. For the list of UM function rights see User Manager Function Rights .
- *role* represents the role name if the switch –s is present, represents the role internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record;

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.
-s	If the switch –s is present, objects are identified by their name. If the switch –s is not present, objects are identified by their internal identifier. The identifier is a positive number univocally identifying the record.

10 How to Import / Export UMC Users and Groups

The following commands can be used to perform import or export operations such as importing\ exporting users/groups from/to a csv file, importing Windows local users, importing Active Directory users/groups or importing\exporting a UMC package to/from a file.

Import Commands

Export Commands

10.1 Import Entities from File

This command imports a set of users or groups into the UMC database. If the user/group is an existing user/group in the UMC database, the corresponding record is updated, if the user/group does not exist it is inserted.

The format of the file to import users/groups can be **csv** (Comma Separated Values) or **json**. The first row of the csv file must contain the column names, the following lines must contain the corresponding values **semicolon** separated.

CAUTION:

The order of the column of the file must be as they are listed.

Users Schema

CSV Format

The table below lists the record names and descriptions for the import of users. The insertion of a user whose group does not exist fails. As a consequence, before importing the users, you have to import their groups.

Name	Description
Name	User name.
Password	User password. This field is empty in case of export.
Full Name	Full user name.
Groups	List of group names to which the user belongs separated by ",". Example: *group1, group2,group3*. If the group does not exist binding is not performed; no error is returned.
Email	User email.
Language	User language.
Data Language	User data language.
Status	A bit mask representing the following flags: • **USER_IS_ENABLED** • **USER_IS_LOCKED** • **USER_IS_IMPORTED**: indicates that the user has been imported from AD. Note that this information is significant only in case of export. • **USER_HAS_EXPIRATION_DATE** To set a flag to true the character "x" has to be inserted in the corresponding position, otherwise the character "-" has to be inserted. Example: x-x denotes a user which is enabled, is not locked, is not imported and has an expiration date. The expiration date is stored in a user property, if the flag **USER_HAS_EXPIRATION_DATE** is set to false, the stored value is ignored.
Mobile	Mobile phone number.
Phone	Phone number.
First Name	First name.
Last Name	Last name.
Initials	Initials.
Comment	Comment.

Name	Description
Policy	A bit mask representing following flags: • **USER_MUST_CHANGE_PASSWORD** • **USER_CAN_CHANGE_PASSWORD** • **USER_HAS_PASSWORD_EXPIRATION** • **USER_HAS_ALARM_BEFORE_PASSWORD_EXPIRATION** To set a flag to true the character "x" has to be inserted in the corresponding position, otherwise the character "-" has to be inserted. Example: xx- denotes a user that must have the password at first login, can change the password, the password never expires and no alarm is displayed for password expiration. The password expiration days are stored in a user property, if the flag **USER_HAS_PASSWORD_EXPIRATION** is set to false, the stored value is ignored.
Expiration Date	User expiration date.
Password Expiration Days	An integer representing the password expiration days.
Alarm Password Expiration Days	An integer representing the alert in days before the password expiration.

CAUTION:

Note that Active Directory (AD) users cannot be imported into UMC via csv file. In case you perform this operation, the newly-created UM users are not linked to AD. To import AD users you have to use the UMC Web UI or the appropriate umx command.

JSON Format

The following schema is used to import/export users.

Read only fields: these fields will be returned during the import/export operation and ignored during import.

In case of creation, the following fields are mandatory: **name**, **full name, comment**

During the import operation the **name** is read only and used to identify the object to modify. The **id** can be used as an alternative.

```
{
  "users": [
    {
      "result": 0, //read only
      "id": 1735, //read only
```

```
      "name": "Mario Rossi",
      "password" : "thepassword",
      "objver": 2,
      "fullname": "full name",
      "firstname": "Mario",
      "lastname": "Rossi",
      "initials": "M.R.",
      "groups": [ ],
      "roles": [ "UMC Admin" ],
      "alias": [ ],
      "attributes": [
        {
          "name": "attr_name",
          "value": "attr_value"
        }
      ],
      "canchange": 1,
      "mustchange": 0,
      "locked": 0,
      "override_lock_policy": 0,
      "offline": 0,
      "comment": "",
      "datalanguage": "und",
      "language": "und",
      "autologoff": 0,
      "email1": "",
      "email2": "",
      "email3": "",
      "enabled": 1,
      "expirationdate": "",
      "expired": 0,
      "imported": 0,//read only
      "importedfromad": 0,//read only
      "importedfromgroup": 0,//read only
      "otp_enabled": 0,//read only
      "passwordexpirationdays": 60,
      "alertbeforepasswordexpirationdays": 0,
      "alertsbeforeexpirationdate": "",
      "phone": "",
      "mobile": "",
      "sid": "",
      "timebeforeautologoff": 0
    }
  ]
}
```

Important:

Alias cannot be imported using UMC 2.5.

Use hash for password transport

In order to reduce the risk to password disclosure, it is possible to use a slated hash instead of the plain text password.

Example:

```
 "password" :  { "hash" : "vHZsbfcdnw8aPL0Tcc6fW5lBYitRLGusm/
uMcXcOKbmn8nTWZIEk6YZ3M9VFfV4Oqrx2X4CMNaIulEzWOKNfT3Q8ed3zVFAtCIXgmGiECug=",
//encoded base64
                "salt" : "ZnVvY28=",//salt encoded as base64 - max 32 byte
                "algorithm" : "PBKDF2-HMAC-SHA512", //algorithm used for
hashing (PBKDF2-HMAC-SHA512 unique value supported)
         "encoding" : "utf16le",// encoding for the password before hashing
(allowed values: utf8 - utf16le)
                "iteration" : 10000, //min accepted 0 - max 100000
                "length" : 80 //min 32 bytes - max 256  bytes }
```

Hash is generated hashing the password (without null terminator) using **PBKDF2 (HMAC-SHA512).**

Important:

- The password cannot be exported in any case (plain or hashed).
- recommended values:
  - salt 32 bytes
  - length 128 bytes
  - iterations 1000
- Usage of different parameters may lead to security (lower) or performance issues (higher).

Groups Schema

CSV Format

The table below lists the record names and descriptions for the import of groups.

Name	Description
Name	Group name
Description	Group description

JSON Format

The following schema is used to import/export users.
Read only fields: these fields will be returned during the import/export operation and ignored during import.
In case of creation the following fields are mandatory: **name**
During the import the **name** is read only and used to identify the object to modify, the **id** can be used as alternative.

```
{
    "groups": [
      {
        "result": 0, //read only
        "id": 389, //read only
        "name": "groupname",
        "objver": 1,
        "description": "description",
        "users": [ "MarioRossi" ],
        "roles": [ "UMC Admin" ],
        "imported": 0,
        "lastSync": 0,
        "offline": 0,
        "sid": "",
        "syncStatus": "SYNC_IGNORE"
      }
    ]
  }
```

Role Schema

CSV Format

Important:

CSV format not available.

JSON Format

The following schema is used to import/export roles.
Read only fields: these fields will be returned during the import/export operation and ignored during import.
In case of creation, the following fields are mandatory: **name**
During the import operation the **name** is read only and used to identify the object to modify. The **id** can be used as alternative.

```
{
  "roles":{
    "description":"Who can join a server to the ring",
    "name":"Joiner",
    "rights":["UM_JOIN","UM_VIEW"]
  }
}
```

Rights are defined in the following table:

Function rights are the capabilities to perform operations. They are associated with roles so that the set of UM users having a specific UM role is allowed to perform the set of operations associated with it. The following table contains a list of UM Function Rights:

Name	Description
UM_ADMIN	Allows you to display the UMC database data and to configure the UMC database, that is to create users, groups and so on, to import and export data via file, to register UMC station clients. This function right allows you to execute all **umx** commands.
UM_VIEW	Allows you to display the UMC database data related to users, groups, roles and account policies.
UM_RESETPWD	The user can reset the password of another user. The user must also have associated the **UM_VIEW** function right.
UM_UNLOCKUSR	The user can unlock any other user. The user must also have associated the **UM_VIEW** function right.
UM_ATTACH	The user can attach a machine to a UM domain, the machine is promoted to the *UM agent role*.
UM_JOIN	The user can promote a machine to a *UM server role*. If the machine is not yet attached to the UM domain, it is attached. This function right incorporates the **UM_ATTACH** function right.
UM_RESETJOIN	The user can downgrade a machine from the UM ring server or UM server role to the UM agent role.
UM_IMPORT	The user can import the UM Configuration via package. The user must also have associated the **UM_VIEW** function right.
UM_EXPORT	The user can export the UM Configuration into a package. The user must also have associated the **UM_VIEW** function right.
UM_BACKUP	The user can back up the UM Configuration (Full backup). *This function right is not used, as the functionality controlled by it has not yet been implemented.*
UM_EXPORTCK	The user can export Claim Key. *This function right is not used, as the functionality controlled by it has not yet been implemented.*
UM_EXPORTDK	The user can export Domain Key. *This function right is not used, as the functionality controlled by it has not yet been implemented.*
UM_RA	Login from Remote Authentication. *This function right is not used, as the functionality controlled by it has not yet been implemented.*

Name	Description
UM_RINGMNG	The user can promote a machine to a *UM ring server role*. If the machine is not yet attached to the UM domain, it is attached.
UM_ADSYNC	The user can perform the background AD provisioning synchronization.
UM_VIEWELG	The user can display event logging data. The user must also have associated the **UM_VIEW** function right.
UM_CLAIMAUTH	The user can create an identity from a valid claim.
UM_REGCLIENT	The user can register UMC station clients.

Account Policies Schema

CSV Format

 CSV format not available.

JSON Format

The following schema is used to import/export account policies.
Read only fields: these fields will be returned during the import/export operation and ignored during import.

```
{
    "accpol": {
      "globalAccountPolicies": {
        "enableLockAfterNumberOfAttempts": 1,
        "enablePasswordCheck": 0,
        "enablePasswordHistoryByDays": 1,
        "enablePasswordHistoryByNumber": 0,
        "maxLoginErrors": 5,
        "minDaysBeforePasswordReuse": 120,
        "numberOfPasswordBeforeReuse": 5,
        "passwordAging": 60,
        "passwordMaxLength": 120,
        "passwordMinAlphaChar": 2,
        "passwordMinLength": 8,
        "passwordMinLowerChar": 1,
        "passwordMinNumericChar": 1,
        "passwordMinOtherChar": 0,
        "passwordMinUpperChar": 1
      },
      "systemPolicies": {
        "pki": {
          "authmode": 10,
```

```
            "filter": "test",
            "issuer": "anothertest"
          },
          "sads": { "enableAkp": 0 }
        }
}
```

A list of rules to manage special characters follows:

- *semicolon*: if a record value contains a semicolomn, it must be delimited by quotation marks. For example: suppose that we want to insert the full name as Brown;Peter, we should have in the csv file "Brown;Peter"
- *quotation mark*: If a record value contains the quotation mark, it must delimited by quotation marks and any quotation mark must be preceded by a quotation mark. For example: if we want to insert the value "Peter" we should insert in the csv file the value ""Peter""
- *comma:* list of values are separated by commas; if one of the values contains comma character, any value must be delimited by quotation marks. For example: to insert group,1 and group,2 associated to a user we should insert in the csv file "group,1","group,2". If one of the listed values contains a quotation character, this character must be preceded by three quotation mark.

Syntax

```
umx [-x commandUserName commandUserPassword] -I {-u |-g |-all} -f fileName
-t format -noroot -fout outputfileName
```

Parameters

- *filename* is the name of the csv file, for instance *myFile.csv.*
- *format* is the format of the file: 0 for **csv**, 1 for **json.**

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.
-u	The data of the csv or json files is imported as UM users.
-g	The data of the csv or json files is imported as UM groups.
-all	The whole configuration is imported. Only supported with the json format.
-f	File name.

-t	File format that is: • 0 for **csv** • 1 for **json**
-noroot	If this flag is enabled, the password of the root user is not modified.
-fout	Result file name.

For the **-all** option, the json format is a concatenation of the sections group, users, roles and policies.

10.2 Import Windows Local Users or Virtual User Accounts

This command imports the given Windows user into UMC users; optionally, an existing role can be assigned to the imported user. The user name of the UMC imported user follows the pattern *<machineName>\<localUserName>*. In case of Virtual Service Account, the user name of the UMC imported user is *<machineName>\$VUA$<service name>.* In case of IIS APPPOOL identity, the user name of the UMC imported user is *<machineName>\$IIS$<apppool identity name>.*

The command can only be used to import a local machine user on a UM Server or UM Ring Server, if you need to import a Windows Local User on an Agent, see, Importing a Windows Local User on an Agent in the *UMC Installation Manual*. To import Active Directory users you have to use the UMC Web UI or the appropriate **umx** command.

This command allows one also to import built-in Windows local users. In the following table we report the user name parameter corresponding to the built-in Windows local user.

User	User name parameter
Local System	"NT AUTHORITY\System"
Local Service	"NT AUTHORITY\LOCAL SERVICE"
Network Service	"NT AUTHORITY\NETWORK SERVICE"
Virtual Service Account	"NT SERVICE\<SERVICE NAME>"
IIS APPPool identity	"IIS APPPOOL\<application pool name>"

CAUTION:

Imported Windows local users should be used **only** for configuration purposes, for instance to be associated to a Windows service running on the machine. Their authentication is demanded to the underlying operating system.

Syntax

```
umx [-x commandUserName commandUserPassword] -I -u -w userName [-r role]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *userName* is the string representing the user name;
- *role* represents the role name, it must exist in UMC.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.
-w	Indicates that the user to be imported is a Windows user.

10.3 Import Active Directory Users

This command imports a given Active Directory user (Windows domain user) into UMC users. The user name of the UMC imported user follows the pattern *<ADdomainName>\<ADuserName>*. Imported Active Directory users authenticate against Windows. The import is based on a search index that is returned by the umx command List Entities, thus a precondition is to execute the List Entities command first with a search criteria that matches the user you want to import. The command based on a search index works only in interactive mode and cannot be used in scripts. Alternatively, you can import a set of Active Directory users according to an input search criteria. Optionally an existing role can be assigned to the imported users.

Syntax

```
umx [-x commandUserName commandUserPassword] -I -du -s searchString -d
domainName [-r role] [-f]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *searchString* filters the Active Directory users to be listed, the "*" wildcard can be used, the search is perfomed in the following Active Directory fields: user name (sAMAccountName), user full name (displayName), and common name (cn).

- *domainName* is the domain from which the user(s) will be imported.
- *role* is the existing role that will be assigned to the imported user(s).

Switches

Switch	Description
-f	Forces the creation of multiple users.

10.4 Import Active Directory Groups

This command imports a set of Active Directory groups into UMC groups according to an input search criteria. The group name of the UMC imported groups follows the pattern *<ADdomainName>*\ *<ADgroupName>*. All the users belonging to the group are imported and authenticate against Windows. Optionally an existing role can be assigned to the imported groups.

Syntax

```
umx [-x commandUserName commandUserPassword] -I -dg -s searchString -d
domainName [-r role] [-f]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *searchString* filters the Active Directory groups to be listed, the "*" wildcard can be used, the search field is the group name (cn).
- *domainName* is the name (without extension) of the domain to which the group belongs.
- *role* is the existing role that will be assigned to the imported user(s).

Switches

Switch	Description
-f	Forces the creation of multiple groups.

Note: In case it is required to import an Active Directory group not by its Common Name (CN), the group must be created offline and the description can be used for configuring the import criteria. See User Manager Group for details.

Note: By default, nested groups of the imported group are not imported. It is possible to enable the import of users belonging to nested groups, so that the users of nested groups are imported and bound to the parent group. See the *UMC installation Manual* (Appendix, Additional Provisioning Configurations) for how to enable it.

10.5 Import Package - UMC Fully Configured

This command imports UMC user, groups and roles from an input UMC package and merges the package data with the current configuration.

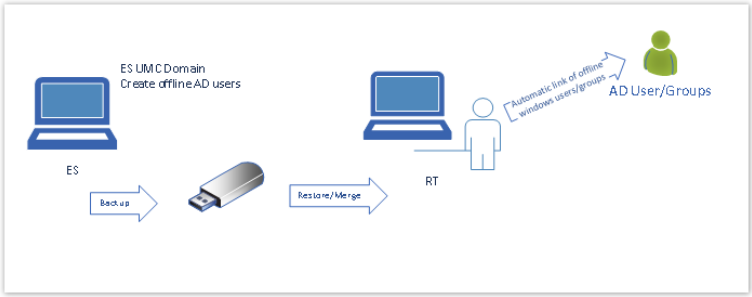
UMC package is a UMC proprietary format, zipped and encrypted. You will be prompted to insert a password for the decryption that has to be the same of the one used in the export package command.

Prerequisites

- The user performing the command must have the function right **UM_ADMIN** or both the function right **UM_VIEW** and **UM_IMPORT**.
- UMC has been configured. If UMC is only installed and not configured, to import a package you have to use the corresponding umconf command. For more details see the *UMCONF User Manual*.
- This command can be run only on the priority ring server.

Merge Behavior

The following image provides a visual description of the process.



In the engineering station (a UM ring server) you can create users, groups and roles. When importing the configuration package on the target system, the data in the target system are updated or overwritten according to the imported data. The engineering data in the package overwrites all the engineering data of the target system, except:

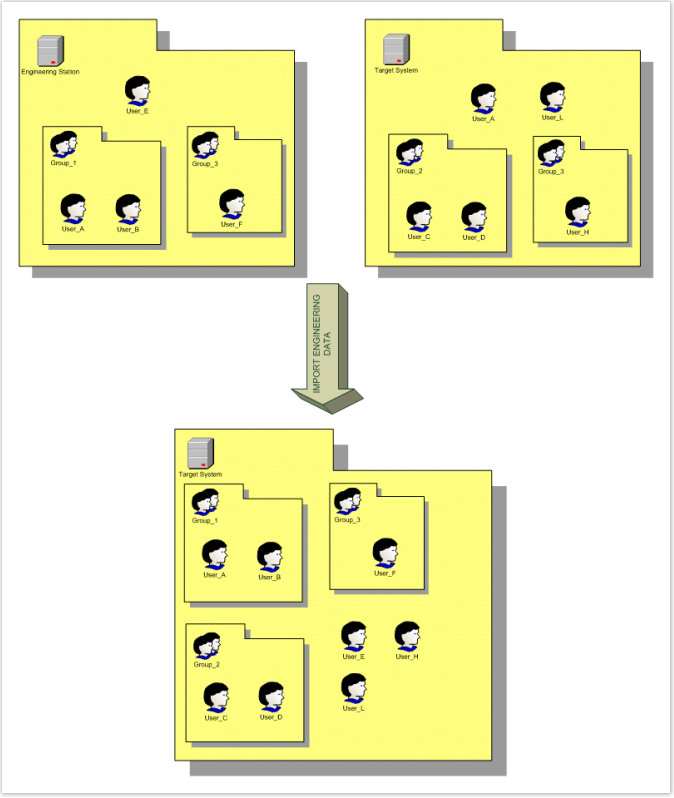
- Passwords: the passwords of target system are maintained.

- Object identifiers: if an object with a different name and the same identifier exists, a new identifier is assigned to the imported object.

The entities are merged as follows:

- **Users:** The users on the source machine are added to the target machine along with all of their attributes and properties. When a user is present on both the source and the target machine (same name), the details of the user on the target machine are overwritten but their password is maintained.
- **Groups:**The groups present on the source machine are added to the target machine along with all of their roles and members. When a group is present on both the source and the target machine (same name), the details of the groups, including its members and roles are overwritten with the values present in the source.
- **Roles:**The roles present on the source machine are added to those present on the target machine with all the their functional rights. When a role is present on both the source and target machine the (same name), the functional rights and configurations of the role is overwritten with the values present in the source.
- **Windows Local users:** If Windows local users are imported into an engineering station and then they are imported via package into a target system, their username changes from *<engineeringStationName>\<localUserName>*, to *<targetMachineName>\<localUserName>*.
- **Offline users and groups:** can be created in the engineering station. After the import package operation in the target system, when the UMC provisioning service runs, if a user/group is found in AD with the same name, object data are aligned and the object becomes online. For the groups, all the AD users belonging to the group are imported into UMC.

The following image presents an example of the import of the UMC configuration of an engineering station into a target system.



Syntax

```
umx  [-x commandUserName commandUserPassword] -Ip -f file -p password
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *file* is the path and name of the file to be imported, for instance C:\temp\myPackage;
- *password* is the archive password.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

10.6 Export Entities to File

This command exports the UM users or UM groups to a csv (Comma Separated Values) file. The format of the file is the same of the import one (see import command). Passwords, Attributes and Aliases are not exported. The flag **USER_IS_IMPORTED** of the **Status** field indicates that the user has been imported from AD.

Syntax

```
umx [-x commandUserName commandUserPassword] -E {-u | -g |-all} -f fileName
-t format
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *filename* is the name of the csv file, for instance *myFile.csv*

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.
-u	The data of the csv or json files are exported as UM users.
-g	The data of the csv or json files are exported as UM groups.
-all	The whole configuration is exported. Only supported with the json format.
-f	File name.
-t	File format that is: • 0 for **csv** • 1 for **json**

10.7 Export Package

This command exports UMC user, groups and roles data to a UMC package. UMC package is a UMC proprietary format, zipped and encrypted. The exported package is the input of the import package command. The user performing the command must have the function right **UM_ADMIN** or both the function right **UM_VIEW** and **UM_EXPORT.**

For more information on the import/export package usage see the *Standalone Engineering Station Scenario* in the *User Management Component Installation Manual*.

Syntax

```
umx  [-x commandUserName commandUserPassword] -Ep -f file -p password
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *file* is the path and name of the exported file, for instance C:\temp\myPackage. The directory where the file will be located has to be previously created in the file system.
- *password* is the encryption password for the archive.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

10.8 Update via Package

UMX provides an additional update via package command. this can be used if you want to update the engineering station with production data, you can export production data into a package and subsequently perform an update package command in the engineering station using the exported package. The effects of the update via package operation are:

- All the data of the engineering station are overwritten or deleted. The only exceptions are user passwords: if a user exists both in the production machine and in the engineering station, the password of the engineering station is maintained.
- The Administrator user of the engineering station is maintained.

This command can be run only on the priority ring server.

UMC package is a UMC proprietary format, zipped and encrypted. You will be prompted to insert a password for the decryption that has to be the same of the one used in the export package command.

The user performing the command must have the function right **UM_IMPORT**.

For more information on the import/export/update package usage see the *Standalone Engineering Station Scenario* in the *User Management Component Installation Manual*.

Syntax

```
umx  [-x commandUserName commandUserPassword] -Up -f file [-p password]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *file* is the path and name of the file to be imported, for instance C:\temp\myPackage;
- *password* is the archive password. If not provided, the user will be prompted to insert the password.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

10.9 Import Active Directory Group by LDAP Query

This command create an offline group and associates users after executing the query in the description. The group name of the UMC imported group follows the pattern <ADdomainName>\<GroupName>. All users belonging to the group identified by the query are imported. This is a way to import a group and its users via an ldap query partially written in the group description.

Syntax

```
umx [-x commandUserName commandUserPassword] -c -g <ADdomainName>\<GroupName>
 -d {{Q=<ldap query> -off
```

Note

F or more information, see Create Group.

11 How to Execute Administrative Commands

The following commands can be used to perform administrative operations, such as setting passwords, enabling/disabling/unlocking a user, displaying the connection status or disabling the safe mode.

- Set User Password
- Enable User
- Disable User
- Unlock User
- Disable Safe Mode
- Show Status
- Get Domain Identifier
- Get Domain Name
- Generate or Reset a Secret Key

11.1 Set User Password

This command changes the user password. The new password must not necessarily conform to the global account policies. The user performing the command must have the function right **UM_ADMIN** or both the function rights **UM_VIEW** and **UM_RESETPSW**. Empty passwords are not allowed.

Syntax

```
umx [-x commandUserName commandUserPassword] -setpwd userName newPassword
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *userName* is the string representing the user name
- *newPassword* the new password, the password specified may not meet the global password constraints. To enable a check on the password see Set Password Check.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

11.2 Enable User

This command enables the user.

Syntax

```
umx [-x commandUserName commandUserPassword] -enableusr userName
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *userName* is the string representing the user name.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

11.3 Disable User

This command disables the user.

Syntax

```
umx [-x commandUserName commandUserPassword] -disableusr userName
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *userName* is the string representing the user name.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

11.4 Unlock User

This command unlocks the user.

Syntax

```
umx [-x commandUserName commandUserPassword] -unlockusr userName
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *userName* is the string representing the user name.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

11.5 Disable Safe Mode

This command disables the safe mode. The UM ring server on which safe mode is disabled becomes the master ring server and is able to perform write operations on the UMC database. Consider that some operations on the UMC system configuration are not allowed in this case, e. g. modifying the whitelist (see *UMCONF User Manual* for more details). For more information on the UMC machine roles see also the overview of the *User Management Component Installation Manual*.

Syntax

```
umx  [-x commandUserName commandUserPassword] –disablesafe
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

11.6 Show Status

This command provides the results of the UMC health check as output.

Note: In order to retrieve the status of UMC you must specify a user who is a Windows user with Administrative rights or elevated user if User Account Control (UAC) is enabled.

Info	Description
UMC Health Status	The status of UMC server. Example: *All UMC servers are running*.
UMC Communication Status	The status of UMC Secure Communication Services. Example: *All UMC communication are running*.

Info	Description
Machine role	The possible values are: • ring • server • agent Example: *Machine role is ring*
Claim Key	(Not for the agent) A claim key is present or not present.
Ticket Key	(Not for the agent) A ticket key is present or not present.
UMC databases	(Not for the agent) UMC databases are present or not present.
Discovery status	(Not for the agent) Details on the connection between server and client. Example: *Discovery status is connected* The possible values are: • connected • standalone (not used) • no configuration found • not initialized • generic error
Workstation status	(Not for the agent) Example: *Workstation status is master* The possible values are: • master: the machine is a master UM ring server. • online: the machine is a UM ring server (not master), or a UM server. • remote_master_is_in_safe_mode: the machine is a UM server connected to a master in safe mode. • initializing: the machine is an initialization phase. • degraded: the machine is a UM server not connected to any UM ring server. • unconnected: not connected. • segregated: the machine is a segregated server. • error: generic error.
Ring master	The name of the ring master. Example: *Now Ring master is : vm-umc1,* this field also specifies if the ring server is in safe mode: *vm-umc1 in safe mode.*
Authentication server	The connected authentication server. Example: *Authentication server is vm-umc1*
Network certificate	Example with certificate found: present and it expires in 3649 days. Example with missing certificate : not present.
Machine certificate	Example with certificate found: present and it expires in 3649 days. Example with missing certificate : not present. Example with expired certificate: present and it expires in 0 days.

CAUTION: Certificate renewal

Network and Machine certificates can be automatically renewed when close to the expiration date: see the Appendix in the *UMC Installation Manual* for details about the certificate renewal procedure.

Syntax

```
umx  [-x commandUserName commandUserPassword] -status
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

Examples

Example #1: the following example shows the output of the health check performed on the ring server.

```
UMC Health Check information.
UMC Health Status : All UMC servers are running.
UMC Communication Status : All UMC communication are running.
Machine role is ring
Claim Key : present
Ticket Key : present
UMC databases : present
Discovery status is connected
Workstation status is master
Now Ring master is : vm-umc1
Authentication server is vm-umc1
Network certificate: present and it expires in 3649 days
Machine certificate: present and it expires in 3649 days
Time taken: 0.13s
```

Example #2: the following example shows the output of the health check performed on the agent.

```
UMC Health Check information.
UMC Health Status : All UMC servers are running.
UMC Communication Status : All UMC communication are running.
Machine role is agent
Now Ring master is : vm-umc1
Authentication server is vm-umc1
Network certificate: present and it expires in 3649 days
Machine certificate: present and it expires in 3649 days
Time taken: 11.46s
```

11.7 Get Domain Identifier

Retrieves the ID of the current domain.

Syntax

```
umx [-x commandUserName commandUserPassword] -getdomainid
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

11.8 Get Domain Name

Retrieves the name of the current domain.

Syntax

```
umx [-x commandUserName commandUserPassword] -getdomainname
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

11.9 Generate or Reset a Secret Key

This command generates the Secret Key for the user specified if not already present, and if present resets the secret key. This command allows you to set the secret key for the Administrator user.

Syntax

```
umx [-x UserName UserPassword] -resettotp
```

Parameters

- The Username of the user for whom you are setting or resetting the secret key.
- The Password of the user for whom you are setting or resetting the secret key.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

11.10 Change User Language

This command set the user Language of the currently UMX logged user

Syntax

```
umx [-x commandUserName commandUserPassword] -changeUserLang language
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *language* is the user language and has the format <langcode>-<countrycode> (for example en-GB) where:◦langcode is the language code according to the ISO 639 standard; we accept both two-letter codes (ISO 639-1) and three-letter codes (ISO 639-2);
  ◦countrycode is the country code according to the ISO 3166 standard.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

11.11 Change User Data Language

This command set the user data Language of the currently UMX logged user

Syntax

```
umx [-x commandUserName commandUserPassword] -changeDataLang datalanguage
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *datalanguage* is the data language and has the format <langcode>-<countrycode> (for example en-GB) where:◦langcode is the language code according to the ISO 639 standard; we accept both two-letter codes (ISO 639-1) and three-letter codes (ISO 639-2);
  ◦countrycode is the country code according to the ISO 3166 standard.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

11.12 Show User Properties

This command shows the properties of the currently UMX logged user

Syntax

```
umx [-x commandUserName commandUserPassword] -showUserProperties
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

12 How to Manage Account Policies

These commands can be used to modify the account policies related to the following operations:

- Manage Passwords
- Associate UMC User to Provisioning Service
- Restore account policy default values
- Set Default PKI Rule
- Reset Default PKI Rule
- Manage Secure Application Data Support (SADS)
- Set Password Check
- Reset Password Check

12.1 Modify Account Policies - Passwords

This command modifies the account policies related to password management. If you enable the password history by days, the password history by number of password is disabled and vice versa.

Syntax

```
umx  [-x commandUserName commandUserPassword] -AP
     [-pwdMinLen minLen]
     [-pwdMaxLen maxLen]
     [-pwdMinLowChar minLC]
     [-pwdMinUpChar minUC]
     [-pwdMinAlphaChar minAlphaC]
     [-pwdMinNumChar minNumC]
     [-pwdMinOtherChar minOC]
     [-enablePwdHistoryByDays flag -pwdMinDaysBeforePwdReuse daysReuse]
     [-enablePwdHistoryByNumPwd flag -pwdMinNumPwdBeforePwdReuse numPwd]
     [-enableLockAfterNumAttempts flag -maxLoginErrors numErrors]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *minLen* is the minum number of characters allowed in passwords. If you set this value to 0, this check is disabled. Empty passwords are not allowed.

- *maxLen* is the maximum number of characters allowed in passwords. If you set this value to 0, this check is disabled. Empty passwords are not allowed.
- *minLC* is the minum number of lower case characters allowed in passwords.
- *minUC* is the minum number of upper case characters allowed in passwords.
- *minAlphaC* is the minum number of letters allowed in passwords.
- *minNumC* is the minum number of numbers allowed in passwords.
- *minOC* is the minum number of special characters (not numbers or letters) allowed in passwords.
- *flag*, for the description of this parameter refer to the switch that precedes it in the command above, 0 corresponds to false and 1 corresponds to true;
- *daysReuse* is the number of days before you can reuse the same password;
- *numPwd* is the number of passwords before you can reuse the same password;
- *numErrors* is the number of errors in typing the password after which the user is locked. The lock of the user depends also on the value of the user parameter *paramOverrideLock* (see Update User).

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

12.2 Modify Account Policies - Associate UMC User with Provisioning Service

This command associates a UMC user identified by the parameter *name* with the UM service **UPService.exe**. This user must have a role with the associated function right **UM_ADSYNC**. This role is not automatically created and has to be created before associating this user. This user is stored as an account policy. It is mandatory to restart the UM service **UPService.exe** after executing the command.

Important:

This configuration is not mandatory. The UMC user associated by default with the provisioning service is the UMC administrator. We strongly recommend to perform this configuration to harden your system following the least privilege principle.

Syntax

```
umx  [-x commandUserName commandUserPassword] -AP -provisioningDefaultUser
user [-s]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *user* represents the user name if the switch –s is present, represents the user internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record;

Switches

Switch	Description
-s	If the switch –s is present, objects are identified by their name. If the switch –s is not present, objects are identified by their internal identifier. The identifier is a positive number univocally identifying the record.

12.3 Modify Account Policies - Restore

This command modifies the account policies restoring them to their default values. Default values are:

- minum number of characters: 8
- maximum number of characters: 120
- minum number of lower case characters: 1
- minum number of upper case characters: 1
- minum number of letters: 2.
- minum number of numbers: 1.
- minum number of special characters (not numbers or letters): 0
- password history*:*
  - enable Password History by Days: true
  - number of days before you can reuse the same password: 120
  - enable Password History by Number of Passwords: false
  - number of passwords before you can reuse the same password: 5
- Enable user lock after *n* wrong login attempts: true
- number of errors in typing the password after which the user is locked: 5

Syntax

```
umx  [-x commandUserName commandUserPassword] -AP -restore
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

12.4 Modify Account Policies - Set Default PKI Rule

This command sets a default PKI rule to be used in smart card authentication. A PKI rule establishes, for a given issuer, which is the authorization mode that can be used and the corresponding filter. Please refer to the smart card authentication configuration in the *User Management Component Installation Manual* for more details and the example.

Syntax

```
umx -AP -setdefaultpki -issuer issuerName -authmode authModeValue [-filter
filterValue]
```

Parameters

- *issuerName* is the name of the issuer of the certificate, by now this value is not used;
- *authModeValue* represents the different types of allowed authentication modes:
  – 2: Authentication using filter on Subject;
  – 3: Alias Authentication using filter on Subject
  – 4: Authenticate using CN;
  – 5: Alias Authentication using CN;
  – 10: Authentication using filter on Alternate Subject;
  – 11: Alias Authentication using filter on Alternate Subject.
- *filterValue* is the regular expression representing the filter.

12.5 Modify Account Policies - Reset Default PKI Rule

This command resets the default PKI rule to be used in smart card authentication. A PKI rule establishes, for a given issuer, which is the authorization mode that can be used and the corresponding filter.

Syntax

```
umx -AP -resetdefaultpki
```

12.6 Modify Account Policies - Secure Application Data Support

This command modifies the account policy related to Secure Application Data Support (SADS).

Syntax

```
umx -AP {-setakp | -resetakp}
```

Switches

Switch	Description
-setakp	Enables Secure Application Data Support for users and groups.
-resetakp	Disables Secure Application Data Support for users and groups.

12.7 Modify Account Policies - Set Password Check

This command enables a check on passwords set by the administrator, both when a user is created and updated. The check verifies that passwords which do not meet global account policies cannot be set. Note that password reuse policies are not applied.

Syntax

```
umx -AP -setpswcheck
```

12.8 Modify Account Policies - Reset Password Check

This command disables a check on passwords set by the administrator so that passwords which do not meet global account policies cannot be set.

Syntax

```
umx -AP -resetpswcheck
```

13 How to Execute Commands in Interactive Mode

The following command can be used to run UMX in interactive mode.

- Interactive Mode
- Enable Notifications

13.1 Interactive Mode

This command launches the **umx** interactive mode. Umx process is always active and the single commands can be launched directly without being preceded by the **umx** string.

The following commands can be launched only in interactive mode:

- **-q**, to exit the interactive mode;
- **-lockdb**, to perform a write lock on the UMC database.
- **-unlockdb**, to remove the write lock on the UMC database; the only possible command when DB is unlocked is the lockdb.

Syntax

```
umx -interactive
```

Important: Database Lock

When **umx** is launched in interactive mode, a write lock on the database is performed. Thus, no other process/application can modify the database entities. Read only access is allowed.

13.2 Enable Notifications

This command is only for testing purposes. It enables the notifications on the server connection status and on the create/update/delete of the following database objects: users, groups, roles and account policies.

Syntax

```
umx [-x commandUserName commandUserPassword] -N
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

14 How to Execute Authentication Commands

The following command can be used to perform the authentication of UMC users.

- Test Authentication
- Generate Ticket
- Change User Password

14.1 Test Authentication

This command performs an authentication versus UMC with the credentials provided in input. If the switch -win is present, the authentication of the current Windows logged in user is performed. If the switch -dsso is present, the authentication of the user currently logged on dSSO (desktop single sign on) is performed. The command output shows whether the authentication succeeds or fails and whether the authenticated user owns or not the function right **UM_ADMIN**.

Syntax

```
umx -t userName userPassword [-totp]
umx -t -win [-totp]
umx -t -dsso [-totp]
```

Parameters

- *userName* represents the user name of the user to be authenticated;
- *userPassword* represents the user password.

Switches

Switch	Description
-win	Authenticates the current Windows logged in user.
-dsso	Authenticates the current dSSO logged in user.
-totp	Displays the secret key.

14.2 Generate Ticket

This command, given a user name and an associated password, generates a ticket that is used for authentication instead of the password. Ticket duration (in seconds) has to be specified; after the duration, the ticket expires and cannot be used to login anymore.

Syntax

```
umx  [-x commandUserName commandUserPassword] -T userName password duration
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *userName* represents the user name;
- *password* represents the user password;
- *duration* represents the ticket duration in seconds.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

14.3 Change User Password

This command changes the user password knowing the old password. The new password must conform to the global account policies. Empty passwords are not allowed. This command can be performed also on a UM server in degraded mode (see the *User Management Component Installation Manual* for more details). When the UM server degraded mode ends, the reconciliation between passwords at database level is performed and the most recent is kept.

Syntax

```
umx [-x commandUserName commandUserPassword] –changepwd userName oldPassword
newPassword
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *userName* is the string representing the user name;
- *oldPassword* the old password;
- *newPassword* the new password.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

15 How to Manage Secure Application Data Support

The following commands can be used to manage data encryption and decryption for users and groups to apply Secure Application Data Support (SADS).

- Enable Encryption
- Encrypt Keys
- Decrypt Keys

SADS capabilities at application level can be enabled via umx or Web UI by modifying an account policy. For what concerns the subject level, this can only be done via umx.

15.1 Enable Encryption

This command enables encryption capabilities on subjects (users or groups).

Syntax

```
umx [-x commandUserName commandUserPassword] -SK -e {-u user |-g group} [-s]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *user* represents the user name if the switch –s is present, represents the user internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record; to retrieve the user identifier see List Entity Details.
- *group* represents the group name if the switch –s is present, represents the group internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.

-s	If the switch –s is present, objects are identified by their name. If the switch –s is not present, objects are identified by their internal identifier. The identifier is a positive number univocally identifying the record.

15.2 Encrypt Keys

This command encrypts an application key (AK) using the user or group subject key (SK).

Syntax

```
umx [-x commandUserName commandUserPassword] -AK -e -k Key {-u user |-g
group} [-s] [-f]
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *Key* is the application key to be used to encrypt application data.
- *user* represents the user name if the switch –s is present, represents the user internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record; to retrieve the user identifier see List Entity Details.
- *group* represents the group name if the switch –s is present, represents the group internal identifier if the switch –s is not present, the identifier is a positive number univocally identifying the record.

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.
-s	If the switch –s is present, objects are identified by their name. If the switch –s is not present, objects are identified by their internal identifier. The identifier is a positive number univocally identifying the record.
-f	The encrypted key is saved into UMX clipboard file.

15.3 Decrypt Keys

This command decrypts an application key (AK) using the user or group's subject key (SK). The encrypted Application Key can be decrypted only if it has previously been encrypted with a Subject Key of the current authenticated user or the groups he is a member of.

SADS offers offline authentication, which means the user can authenticate even if it is not possible to connect any UM server. This only applies if the user has performed at least one decryption operation, the decryption of EAKs should be possible according to the subject keys available for that user. In order to use this functionality the subject keys used for decryption are copied on the local cache.

Syntax

```
umx [-x commandUserName commandUserPassword] -AK -d {-k EncryptedKey|-f}
```

Parameters

- *commandUserName* is the string representing the name of the user that is executing the command;
- *commandUserPassword* is the string representing the password of the user that is executing the command;
- *EncryptedKey* is the key to be decrypted for the user or group according to their data access control configuration;

Switches

Switch	Description
-x	The command is executed by the user given as input parameter.
-f	The UMX clipboard file is taken as input encrypted key.

16 Error Codes

Value	Description
0	Success
-1	Command syntax error

In all the other cases, the command returns the last error code (decimal format) returned by the UMC APIs invoked during the command execution. See UMC APIs Error Codes for more details.

Example #1

Consider the following script to display the information of the user identified by 66 where no user has assigned the identifier 66 in the UMC database:

```
umx  -i -u 66
echo %errorlevel%
```

No deletion is performed and **umx** returns the decimal number **273** that corresponds to the following error code in UMC APIs Error Codes.

SL_OBJ_DOES_NOT_EXIST	0x111	273	The UMC object does not exist or has not yet been saved into the UMC database.

16.1 UMC APIs Error Codes

All the UMC APIs return a boolean value or an object handle. If the API is successful, the returned boolean value is true or the object handle is well formed; otherwise the returned boolean value is false, or null is returned instead of the object handle. If the API fails an error code can be retrieved calling the **SL_GetLastError** method. **SL_RESULT** defines the type of error. In what follows we list the possible error codes.

Generic Errors

Name	Hexadecimal Value	Decimal Value	Description
SL_SUCCESS	0X00	0	No errors have occurred.
SL_GENERROR	0X01	1	Generic error.
SL_BAD_HANDLE	0x114	276	Internal error for invalid handle.
SL_NOSESSION	0X30	48	The Web session is expired.

Authentication Errors

Name	Hexadecimal Value	Decimal Value	Description
SL_USERLOCKED	0X02	2	The user for whom you want to perform the authentication is locked.
SL_USERDISABLED	0X03	3	The user for whom you want to perform the authentication is disabled.
SL_WRONGUSERNAMEPASSWORD	0X04	4	During the authentication phase, the user name or password are incorrect.
SL_PASSWORDPOLICYVIOLATION	0X05	5	Password policy violation (determined by UMC account policies). For a detailed list of Account Policies, see *User Management Component API SDK Developer Manual*.
SL_USERMUSTCHANGEPASSWORD	0X06	6	The user password must be changed.
SL_PASSWORDEXPIRED	0X07	7	The user password is expired.
SL_FAILED	0X0A	10	Generic operation failed.
SL_ALREADYLOCKED	0X0B	11	The UMC object is already locked.
SL_COMMERR	0X0C	12	Transmission/Communication error.
SL_NOTIMPL	0X10	16	Returned if a not implemented method is invoked.
SL_CHANGEPSWDISABLE	0X19	25	The user cannot change the password.
SL_USERUNKNOWN	0X20	32	The user is not present in the system.
SL_USERNEVEREXPIRE	0X21	33	The user never expires.
SL_TICKETEXPIRED	0X22	34	The authentication ticket is expired.
SL_USER_EXPIRED	0x27	39	The user is expired.
SL_PSWMINLEN_ERR	0x120	288	The account policy related to the minimal password length has been violated.
SL_PSW_CHANGE_FAIL	0X154	340	Password change failure.

Name	Hexadecimal Value	Decimal Value	Description
SL_INVALID_NONCE	0x166	358	Login failed: invalid token. This event may occur if you try to access the login page directly from the URL or if you leave the login page open.
SL_WEAK_AUTH	0x167	359	Login failed: access not allowed using weak authentication method.

CRUD Operation Errors

Name	Hexadecimal Value	Decimal Value	Description
SL_ALREADYEXIST	0x0D	13	The UMC object already exists.
SL_LOCK_NEEDED	0x23	35	A lock is needed to complete the operation.
SL_NOT_LOCKED	0x24	36	The UMC object is not locked so you cannot unlock it.
SL_OBJVERMISMATCH	0X31	49	A UMC object has been simultaneously modified by two Web UI instances and an object version mismatch has been detected.
SL_INVALID_OPERATION	0x103	259	The operation cannot be performed on the selected object.
SL_OBJ_DOES_NOT_EXIST	0x111	273	The UMC object does not exist or has not yet been saved into the UMC database.
SL_OBJECT_LOCKED_IN_DATABASE	0X153	339	The UMC object is already locked.
SL_FAIL_NOTAMASTER	0x160	352	An attempt has been made to modify the UMC database on a machine that is not a master.

Name	Hexadecimal Value	Decimal Value	Description
SL_FAIL_BINDING_ADMIN_ROLE	0x161	353	An attempt has been made to assign the Administrator role to a group or the user who performed the association, either a UMX user or a Web UI user, does not have the Administrator role.
SL_OBJ_OFFLINE	0x0F	15	The user/group for which you want to perform an operation is offline and the operation is not allowed for offline objects.
SL_INVALID_NAME_FOR_OFFLINE_OBJ	0x165	357	The offline user/group that you are creating does not follow the pattern *<domainName>\<objName>*.
SL_INVALID_SID	0x5C	92	Invalid User Security Identifier (SID). See Microsoft Documentation on Security Identifiers for more details.

Provider Operation Errors

Name	Hexadecimal Value	Decimal Value	Description
SL_INVALID_PROVIDER	0x100	256	Operation not provided by this provider.
SL_INVALID_HANDLE	0x101	257	An invalid handle was passed as parameter.
SL_ERROR_LOADING_PROVIDER	0x102	258	An error occurred when loading the provider.

Internal or Parameter Errors

Name	Hexadecimal Value	Decimal Value	Description
SL_INVALID_PARAMETERS	0x104	261	The method has an incorrect parameter.
SL_MEMORY_ERROR	0x105	262	Memory allocation error.
SL_INITIALIZATION_ERROR	0x106	263	Initialization error.
SL_INVALID_LOCK_OPTION	0x108	264	The lock option has not been defined.

Name	Hexadecimal Value	Decimal Value	Description
SL_INVALID_PROPERTY	0x109	265	The property has not been defined for the object.
SL_INVALID_CULTURE	0x17B	379	Invalid language

File Errors

Name	Hexadecimal Value	Decimal Value	Description
SL_ACCESS_FILE_ERROR	0x112	274	Access file error.
SL_UNKNOWN_FILE_FORMAT	0x113	275	Unknown file format.
SL_FILE_NOT_FOUND	0x50	80	File not found.
SL_PATH_NOT_FOUND	0x51	81	Path not found.
SL_FILE_CREATION_FAIL	0x52	82	Error during file creation.
SL_PATH_CREATION_FAIL	0x53	83	Error during path creation.
SL_INVALID_PATH	0x54	84	Invalid path.

Function Rights Errors

Name	Hexadecimal Value	Decimal Value	Description
SL_RESOURCE_NOT_FOUND	0x150	336	The user does not have the correct function right to perform the requested operation. This error has the same meaning as the SL_MISSING_FUNCTION_RIGHT error.
SL_INVALID_RESOURCE	0x151	337	The function right does not exist.
SL_MISSING_FUNCTION_RIGHT	0x152	338	The user does not have the correct function right to perform the requested operation. This error has the same meaning as the SL_RESOURCE_NOT_FOUND error.

Service Layer Errors

Name	Hexadecimal Value	Decimal Value	Description
SL_CLAIM_EXPIRED	0X155	341	The claim is expired.
SL_CLAIM_INVALID	0X156	342	The claim is invalid.
SL_JSON_ERROR	0X157	343	The .json file is not well formed.
SL_MKTKT_FAILURE	0X158	344	The "make ticket" operation failed.
SL_ABORTED	0x159	345	Operation aborted.

Package Errors

Name	Hexadecimal Value	Decimal Value	Description
SL_PACKAGE_CREATION_FAIL	0x55	85	Package creation failed.
SL_PACKAGE_COMPRESSION_FAIL	0x56	86	Package compression failed.
SL_PACKAGE_UNCOMPRESSION_FAIL	0x57	87	Package decompression failed.
SL_PACKAGE_ENCRYPTION_FAIL	0x58	88	Package encryption failed.
SL_PACKAGE_DECRYPTION_FAIL	0x59	89	Package decryption failed.
SL_PACKAGE_RESTORE_FAIL	0x5A	90	Package restore failed.
SL_PACKAGE_WRONG_PASSWORD	0x5B	91	Wrong password for the package.

Database Errors

Name	Hexadecimal Value	Decimal Value	Description
SL_DBFILE_ACCESS_DENIED	0X32	50	The user cannot access a UMC database file.
SL_DBFILE_ERROR	0X33	51	Generic UMC database file error.
SL_DBFILE_OUT_OF_SPACE	0X34	52	A UMC database file is full.
SL_TOO_MANY_GROUPS	0X36	102	Too many groups assigned to a user.
SL_TOO_MANY_ROLES	0X37	103	Too many roles assigned to a user or group.

Name	Hexadecimal Value	Decimal Value	Description
SL_TOO_MANY_USERS	0X38	104	Too many users assigned to a group.
SL_ROLEIDS_OUT_OF_SPACE	0X35	53	No more role IDs are available in the role database file. A purge of the roles is needed.

User Alias Errors

Name	Hexadecimal Value	Decimal Value	Description
SL_INVALID_USER_ALIAS	0x5E	94	Invalid user alias name.
SL_USER_ALIAS_ALREADY_EXIST	0x5F	95	User alias already exists.
SL_BAD_PKI_FILTER_NAME	0x115	277	Invalid filter name or filter name not present when authmode = SL_PKI_FILTER_MASK.

Secure Application Data Support (SADS) Errors

Name	Hexadecimal Value	Decimal Value	Description
SL_INVALID_DOMAIN_NAME	0x60	96	Invalid domain name.
SL_NOT_CURRENT_DOMAIN	0x61	97	Input domain name is not the current domain.
SL_INVALID_KEY	0x70	112	Invalid key.
SL_KEY_GENERATION_FAIL	0x71	113	Error during key generation.
SL_KEY_ENCRYPTION_FAIL	0x72	114	Error during key encryption.
SL_KEY_DECRYPTION_FAIL	0x73	115	Error during key decryption.
SL_KEY_NOT_FOUND	0x74	116	Key not found.
SL_KEY_ENCRYPTION_NOT_ENABLED	0x75	117	Application key protection (global policies) not enabled.
SL_MAX_NUM_KEY	0x76	118	The maximum number of allowed keys has been reached.
SL_KEY_DECRYPTION_NO_ID_FOUND	0x77	119	No SUID of the identity has been found in EAK array.
SL_SADS_VERSION_ERROR	0x78	120	Wrong SADS version.

Name	Hexadecimal Value	Decimal Value	Description
SL_WRONG_IDENTITY	0x79	121	Ticket authentication error while decrypting a key.
SL_EAK_BAD_FORMAT	0x80	128	Bad format of the encryption application object.
SL_SUBJECT_NOT_ENABLED	0x81	129	Encryption not enabled for the specified subject.
SL_SUBJECT_KEY_OBSOLETE	0x82	130	The decryption has been executed using an obsolete key.

17 Parameter Sizes

The following table lists the sizes for the main UMC database fields.

API Property Name	API Object	Web UI Display Name	UMX Parameter	Size in Chars
SL_USER_NAME	SLOBJ_USER	User Name	*name*	100
SL_USER_PASSWORD	SLOBJ_USER	Password	*password*	120
SL_USER_FULLNAME	SLOBJ_USER	Full Name	*fullName*	250
SL_GROUP_NAME	SLOBJ_GROUP	Group Name	*name*	100
SL_GROUP_DESCRIPTION	SLOBJ_GROUP	Description	*description*	260
SL_ROLE_NAME	SLOBJ_ROLE	Role Name	*name*	255
SL_ROLE_DESCRIPTION	SLOBJ_ROLE	Description	*description*	40
SL_ATTRIBUTE_NAME	SLOBJ_ATTRIBUTE	Attribute Name	*attribute name*	80