Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'adcli.8' command

$ man adcli.8

ADCLI(8) System Commands ADCLI(8)

NAME

adcli - Tool for performing actions on an Active Directory domain

SYNOPSIS

adcli info domain.example.com

adcli join domain.example.com

adcli update

adcli testjoin

adcli create-user [--domain=domain.example.com] user

adcli delete-user [--domain=domain.example.com] user

adcli passwd-user [--domain=domain.example.com] user

adcli create-group [--domain=domain.example.com] user

adcli delete-group [--domain=domain.example.com] user

adcli add-member [--domain=domain.example.com] group
user or computer...

adcli remove-member [--domain=domain.example.com] group user...

adcli preset-computer [--domain=domain.example.com] computer...

adcli reset-computer [--domain=domain.example.com] computer

adcli delete-computer [--domain=domain.example.com] computer

adcli show-computer [--domain=domain.example.com] computer

adcli create-msa [--domain=domain.example.com]

GENERAL OVERVIEW

adcli is a command line tool that can perform actions in an Active

Directory domain. Among other things it can be used to join a computer to a domain.

See the various sub commands below. The following global options can be used:

-D, --domain=domain

The domain to connect to. If a domain is not specified, then the domain part of the local computer's host name is used.

-R, --domain-realm=REALM

Kerberos realm for the domain. If not specified, then the upper cased domain name is used.

-S, --domain-controller=server

Connect to a specific domain controller. If not specified, then an appropriate domain controller is automatically discovered.

--use-ldaps

Connect to the domain controller with LDAPS. By default the LDAP port is used and SASL GSS-SPNEGO or GSSAPI is used for authentication and to establish encryption. This should satisfy all requirements set on the server side and LDAPS should only be used if the LDAP port is not accessible due to firewalls or other reasons.

Please note that the place where CA certificates can be found to validate the AD DC certificates must be configured in the OpenLDAP configuration file, e.g. /etc/openldap/ldap.conf. As an alternative it can be specified with the help of an environment variable, e.g.

```
$ LDAPTLS_CACERT=/path/to/ad_dc_ca_cert.pem adcli join --use-ldaps -D domain.example.com
...
```

Please see ldap.conf(5) for details.

-C

Use the default Kerberos credential cache to authenticate with the domain.

--login-ccache[=ccache_name]

Use the specified Kerberos credential cache to authenticate with

the domain. If no credential cache is specified, the default
Kerberos credential cache will be used. Credential caches of type
FILE can be given with the path to the file. For other credential
cache types, e.g. DIR, KEYRING or KCM, the type must be specified
explicitly together with a suitable identifier.
Please note that since the ccache_name is optional the =(equal)
sign is mandatory. If = is missing the parameter is treated as
optionless extra argument. How this is handled depends on the
specific sub-command.

-U, --login-user=User
Use the specified user account to authenticate with the domain. If
not specified, then the name 'Administrator' will be used.

--no-password
Don't show prompts for or read a password from input.

-W, --prompt-password
Prompt for a password if necessary. This is the default.

--stdin-password
Read a password from stdin input instead of prompting for a
password.

-v, --verbose
Run in verbose mode with debug output.

QUERYING DOMAIN INFORMATION

adcli info displays discovered information about an Active Directory
domain or an Active Directory domain controller.

$ adcli info domain.example.com
...
$ adcli info --domain-controller=dc.domain.example.com
...

adcli info will output as much information as it can about the domain.
The information is designed to be both machine and human readable. The
command will exit with a non-zero exit code if the domain does not
exist or cannot be reached.
To show domain info for a specific domain controller use the

--domain-controller option to specify which domain controller to query.

Use the --verbose option to show details of how the domain is discovered and queried. Many of the global options, in particular authentication options, are not usable with the adcli info command.

JOINING THE LOCAL MACHINE TO A DOMAIN

adcli join creates a computer account in the domain for the local machine, and sets up a keytab for the machine. It does not configure an authentication service (such as sssd).

```
$ adcli join domain.example.com
Password for Administrator:
```

In addition to the global options, you can specify the following options to control how this operation is done.

-N, --computer-name=computer

The short non-dotted name of the computer account that will be created in the domain. If not specified, then the first portion of the --host-fqdn is used.

-O, --domain-ou=OU=xxx

The full distinguished name of the OU in which to create the computer account. If not specified, then the computer account will be created in a default location.

-H, --host-fqdn=host

Override the local machine's fully qualified domain name. If not specified, the local machine's hostname will be retrieved via gethostname(). If gethostname() only returns a short name getaddrinfo() with the AI_CANONNAME hint is called to expand the name to a fully qualified domain name.

-K, --host-keytab=/path/to/keytab

Specify the path to the host keytab where host credentials will be written after a successful join operation. If not specified, the default location will be used, usually /etc/krb5.keytab.

--login-type={computer|user}

Specify the type of authentication that will be performed before creating the machine account in the domain. If set to 'computer',

then the computer must already have a preset account in the domain.

If not specified and none of the other --login-xxx arguments have been specified, then will try both 'computer' and 'user' authentication.

--os-name=name

Set the operating system name on the computer account. The default depends on where adcli was built, but is usually something like 'linux-gnu'.

--os-service-pack=pack

Set the operating system service pack on the computer account. Not set by default.

--os-version=version

Set the operating system version on the computer account. Not set by default.

--description=description

Set the description attribute on the computer account. Not set by default.

--service-name=service

Additional service name for a Kerberos principal to be created on the computer account. This option may be specified multiple times.

--user-principal=host/name@REALM

Set the userPrincipalName field of the computer account to this Kerberos principal. If you omit the value for this option, then a principal will be set in the form of host/host.example.com@REALM

--one-time-password

Specify a one time password for a preset computer account. This is equivalent to using --login-type=computer and providing a password as input.

--trusted-for-delegation=yes|no|true|false

Set or unset the TRUSTED_FOR_DELEGATION flag in the userAccountControl attribute to allow or not allow that Kerberos tickets can be forwarded to the host.

--dont-expire-password=yes|no|true|false

Set or unset the DONT_EXPIRE_PASSWORD flag in the userAccountControl attribute to indicate if the machine account password should expire or not. By default adcli will set this flag while joining the domain which corresponds to the default behavior of Windows clients.

Please note that if the password will expire (--dont-expire-password=false) a renewal mechanism has to be enabled on the client to not loose the connectivity to AD if the password expires.

--add-service-principal=service/hostname

Add a service principal name. In contrast to the --service-name the hostname part can be specified as well in case the service should be accessible with a different host name as well.

--setattr=name=value

Add the LDAP attribute name with the given value to the new LDAP host object. This option can be used multiple times to add multiple different attributes. Multi-value attributes are currently not supported.

Please note that the account used to join the domain must have the required privileges to add the given attributes. Some attributes might have constraints with respect to syntax and allowed values which must be met as well. Attributes managed by other adcli options cannot be set with this option.

--show-details

After a successful join print out information about join operation. This is output in a format that should be both human and machine readable.

--show-password

After a successful join print out the computer machine account password. This is output in a format that should be both human and machine readable.

--add-samba-data

After a successful join add the domain SID and the machine account

password to the Samba specific databases by calling Samba's net utility.

Please note that Samba's net requires some settings in smb.conf to create the database entries correctly. Most important here is currently the workgroup option, see smb.conf(5) for details.

--samba-data-tool=/path/to/net

If Samba's net cannot be found at /usr/bin/net, this option can be used to specific an alternative location with the help of an absolute path.

--ldap-passwd

Use LDAP add/mod operations to set the machine account password instead of Kerberos. This might help in some situations where Kerberos fails or is unreliable. But please note that 'Change password' or 'Reset password' permissions or similar might be needed to make the LDAP operation work. Additionally there will be no read-only domain controller (RODC) support as there is with Kerberos.

If supported on the AD side the msDS-supportedEncryptionTypes attribute will be set as well. Either the current value or the default list of AD's supported encryption types filtered by the permitted encryption types of the client's Kerberos configuration are written.

UPDATING THE MACHINE ACCOUNT PASSWORD AND OTHER ATTRIBUTES

adcli update updates the password of the computer account on the domain controller for the local machine, write the new keys to the keytab and removes older keys. It keeps the previous key on purpose because AD will need some time to replicate the new key to all DCs hence the previous key might still be used.

$ adcli update

If used with a credential cache, other attributes of the computer account can be changed as well if the principal has sufficient privileges.

$ kinit Administrator

$ adcli update --login-ccache=/tmp/krbcc_123

In addition to the global options, you can specify the following options to control how this operation is done.

-N, --computer-name=computer

The short non-dotted name of the computer account that will be created in the domain. If not specified, it will be retrieved from the keytab entries.

-H, --host-fqdn=host

The local machine's fully qualified domain name. If not specified, the local machine's hostname will be retrieved from the keytab entries.

-K, --host-keytab=/path/to/keytab

Specify the path to the host keytab where current host credentials are stored and the new ones will be written to. If not specified, the default location will be used, usually /etc/krb5.keytab.

--os-name=name

Set the operating system name on the computer account. Not set by default.

--os-service-pack=pack

Set the operating system service pack on the computer account. Not set by default.

--os-version=version

Set the operating system version on the computer account. Not set by default.

--description=description

Set the description attribute on the computer account. Not set by default.

--service-name=service

Additional service name for a Kerberos principal to be created on the computer account. This option may be specified multiple times.

--user-principal=host/name@REALM

Set the userPrincipalName field of the computer account to this Kerberos principal.

--computer-password-lifetime=lifetime

Only update the password of the computer account if it is older than the lifetime given in days. By default the password is updated if it is older than 30 days.

--trusted-for-delegation=yes|no|true|false

Set or unset the TRUSTED_FOR_DELEGATION flag in the userAccountControl attribute to allow or not allow that Kerberos tickets can be forwarded to the host.

--dont-expire-password=yes|no|true|false

Set or unset the DONT_EXPIRE_PASSWORD flag in the userAccountControl attribute to indicate if the machine account password should expire or not. By default adcli will set this flag while joining the domain which corresponds to the default behavior of Windows clients.

Please note that if the password will expire (--dont-expire-password=false) a renewal mechanism has to be enabled on the client to not loose the connectivity to AD if the password expires.

--account-disable=yes|no|true|false

Set or unset the ACCOUNTDISABLE flag in the userAccountControl attribute to disable or enable the computer account.

--add-service-principal=service/hostname

Add a service principal name. In contrast to the --service-name the hostname part can be specified as well in case the service should be accessible with a different host name as well.

--remove-service-principal=service/hostname

Remove a service principal name from the keytab and the AD host object.

--setattr=name=value

Add the LDAP attribute name with the given value to the LDAP host object. This option can be used multiple times to add multiple different attributes. Multi-value attributes are currently not supported.

Please note that the account used to update the host object must

have the required privileges to modify the given attributes. Some attributes might have constraints with respect to syntax and allowed values which must be met as well. Attributes managed by other adcli options cannot be set with this option.

--delattr=name

Remove the LDAP attribute name from the LDAP host object. This option can be used multiple times to remove multiple different attributes.

Please note that the account used to update the host object must have the required privileges to delete the given attributes.

Attributes managed by other adcli options cannot be removed.

--show-details

After a successful join print out information about join operation. This is output in a format that should be both human and machine readable.

--add-samba-data

After a successful join add the domain SID and the machine account password to the Samba specific databases by calling Samba's net utility.

Please note that Samba's net requires some settings in smb.conf to create the database entries correctly. Most important here is currently the workgroup option, see smb.conf(5) for details.

Note that if the machine account password is not older than 30 days, you have to pass --computer-password-lifetime=0 to force the update.

--samba-data-tool=/path/to/net

If Samba's net cannot be found at /usr/bin/net, this option can be used to specific an alternative location with the help of an absolute path.

--ldap-passwd

Use LDAP add/mod operations to set the machine account password instead of Kerberos. This might help in some situations where Kerberos fails or is unreliable. But please note that 'Change

password' or 'Rest password' permissions or similar might be needed to make the LDAP operation work. Additionally there will be no read-only domain controller (RODC) support as there is with Kerberos.

If supported on the AD side the msDS-supportedEncryptionTypes attribute will be set as well. Either the current value or the default list of AD's supported encryption types filtered by the permitted encryption types of the client's Kerberos configuration are written.

TESTING IF THE MACHINE ACCOUNT PASSWORD IS VALID

adcli testjoin uses the current credentials in the keytab and tries to authenticate with the machine account to the AD domain. If this works the machine account password and the join are still valid. If it fails the machine account password or the whole machine account have to be refreshed with adcli join or adcli update.

```
$ adcli testjoin
```

Only the global options not related to authentication are available, additionally you can specify the following options to control how this operation is done.

-K, --host-keytab=/path/to/keytab

Specify the path to the host keytab where current host credentials are stored and the new ones will be written to. If not specified, the default location will be used, usually /etc/krb5.keytab.

CREATING A USER

adcli create-user creates a new user account in the domain.

```
$ adcli create-user Fry --domain=domain.example.com \
    --display-name="Philip J. Fry" --mail=fry@domain.example.com
```

In addition to the global options, you can specify the following options to control how the user is created.

--display-name="Name"

Set the displayName attribute of the new created user account.

-O, --domain-ou=OU=xxx

The full distinguished name of the OU in which to create the user account. If not specified, then the computer account will be

created in a default location.

--mail=email@domain.com

Set the mail attribute of the new created user account. This attribute may be specified multiple times.

--unix-home=/home/user

Set the unixHomeDirectory attribute of the new created user account, which should be an absolute path to the user's home directory.

--unix-gid=111

Set the gidNumber attribute of the new created user account, which should be the user's numeric primary group id.

--unix-shell=/bin/shell

Set the loginShell attribute of the new created user account, which should be a path to a valid shell.

--unix-uid=111

Set the uidNumber attribute of the new created user account, which should be the user's numeric primary user id.

--nis-domain=nis_domain

Set the msSFU30NisDomain attribute of the new created user account, which should be the user's NIS domain is the NIS/YP service of Active Directory's Services for Unix (SFU) are used. This is needed to let the 'UNIX attributes' tab of older Active Directory versions show the set UNIX specific attributes. If not specified adcli will try to determine the NIS domain automatically if needed.

DELETING A USER

adcli delete-user deletes a user account from the domain.

$ adcli delete-user Fry --domain=domain.example.com

The various global options can be used.

(RE)SETTING THE PASSWORD OF A USER WITH AN ADMINISTRATIVE ACCOUNT

adcli passwd-user sets or resets the password of user account. The administrative account used for this operation must have privileges to set a password.

$ adcli passwd-user Fry --domain=domain.example.com

The various global options can be used.

CREATING A GROUP

adcli create-group creates a new group in the domain.

```
$ adcli create-group Pilots --domain=domain.example.com \
    --description="Group for all pilots"
```

In addition to the global options, you can specify the following options to control how the group is created.

--description="text"

Set the description attribute of the new created group.

-O, --domain-ou=OU=xxx

The full distinguished name of the OU in which to create the group. If not specified, then the group will be created in a default location.

DELETING A GROUP

adcli delete-group deletes a group from the domain.

```
$ adcli delete-group Pilots --domain=domain.example.com
```

The various global options can be used.

ADDING A MEMBER TO A GROUP

adcli add-member adds one or more users to a group in the domain. The group is specified first, and then the various users or computers to be added. You must use dollar sign for computer account (computername$)

```
$ adcli add-member --domain=domain.example.com Pilots Leela Scruffy
$ adcli add-member --domain=domain.example.com servers srv-smb$
```

The various global options can be used.

REMOVING A MEMBER FROM A GROUP

adcli remove-member removes a user from a group in the domain. The group is specified first, and then the various users to be removed.

```
$ adcli remove-member --domain=domain.example.com Pilots Scruffy
```

The various global options can be used.

PRESET COMPUTER ACCOUNTS

adcli preset-computer pre-creates one or more computer accounts in the domain for machines to later use when joining the domain. By doing this machines can join using a one time password or automatically without a

password.

$ adcli preset-computer --domain=domain.example.com \

host1.example.com host2

Password for Administrator:

If the computer names specified contain dots, then they are treated as fully qualified host names, otherwise they are treated as short computer names. The computer accounts must not already exist.

In addition to the global options, you can specify the following options to control how this operation is done.

-O, --domain-ou=OU=xxx

The full distinguished name of the OU in which to create the computer accounts. If not specified, then the computer account will be created in a default location.

--one-time-password

Specify a one time password to use when presetting the computer accounts. If not specified, then a default password will be used, which allows for later automatic joins.

--os-name=name

Set the operating system name on the computer account. The default depends on where adcli was built, but is usually something like 'linux-gnu'.

--os-service-pack=pack

Set the operating system service pack on the computer account. Not set by default.

--os-version=version

Set the operating system version on the computer account. Not set by default.

--service-name=service

Additional service name for a Kerberos principal to be created on the computer account. This option may be specified multiple times.

--user-principal

Set the userPrincipalName field of the computer account to this Kerberos principal in the form of host/host.example.com@REALM

RESET COMPUTER ACCOUNT

adcli reset-computer resets a computer account in the domain. If the appropriate machine is currently joined to the domain, then its membership will be broken. The account must already exist.

$ adcli reset-computer --domain=domain.example.com host2

If the computer names specified contain dots, then they are treated as fully qualified host names, otherwise they are treated as short computer names.

In addition to the global options, you can specify the following options to control how this operation is done.

--login-type={computer|user}

Specify the type of authentication that will be performed before creating the machine account in the domain. If set to 'computer', then the computer must already have a preset account in the domain. If not specified and none of the other --login-xxx arguments have been specified, then will try both 'computer' and 'user' authentication.

DELETE COMPUTER ACCOUNT

adcli delete-computer deletes a computer account in the domain. The account must already exist.

$ adcli delete-computer --domain=domain.example.com host2

Password for Administrator:

If the computer name contains a dot, then it is treated as fully qualified host name, otherwise it is treated as short computer name.

If no computer name is specified, then the host name of the computer adcli is running on is used, as returned by gethostname().

The various global options can be used.

SHOW COMPUTER ACCOUNT ATTRIBUTES

adcli show-computer show the computer account attributes stored in AD. The account must already exist.

$ adcli show-computer --domain=domain.example.com host2

Password for Administrator:

If the computer name contains a dot, then it is treated as fully

qualified host name, otherwise it is treated as short computer name.

If no computer name is specified, then the host name of the computer adcli is running on is used, as returned by gethostname().

The various global options can be used.

CREATE A MANAGED SERVICE ACCOUNT

adcli create-msa creates a managed service account (MSA) in the given Active Directory domain. This is useful if a computer should not fully join the Active Directory domain but LDAP access is needed. A typical use case is that the computer is already joined an Active Directory domain and needs access to another Active Directory domain in the same or a trusted forest where the host credentials from the joined Active Directory domain are not valid, e.g. there is only a one-way trust.

```
$ adcli create-msa --domain=domain.example.com
Password for Administrator:
```

The managed service account, as maintained by adcli, cannot have additional service principals names (SPNs) associated with it. An SPN is defined within the context of a Kerberos service which is tied to a machine account in Active Directory. Since a machine can be joined to a single Active Directory domain, managed service account in a different Active Directory domain will not have the SPNs that otherwise are part of another Active Directory domain's machine.

Since it is expected that a client will most probably join to the Active Directory domain matching its DNS domain the managed service account will be needed for a different Active Directory domain and as a result the Active Directory domain name is a mandatory option. If called with no other options adcli create-msa will use the short hostname with an additional random suffix as computer name to avoid name collisions.

LDAP attribute sAMAccountName has a limit of 20 characters. However, machine account's NetBIOS name must be at most 16 characters long, including a trailing '$' sign. Since it is not expected that the managed service accounts created by adcli will be used on the NetBIOS level the remaining 4 characters can be used to add uniqueness. Managed

service account names will have a suffix of 3 random characters from number and upper- and lowercase ASCII ranges appended to the chosen short host name, using '!' as a separator. For a host with the shortname 'myhost', a managed service account will have a common name (CN attribute) 'myhost!A2c' and a NetBIOS name (sAMAccountName attribute) will be 'myhost!A2c$'. A corresponding Kerberos principal in the Active Directory domain where the managed service account was created would be 'myhost!A2c$@DOMAIN.EXAMPLE.COM'.

A keytab for the managed service account is stored into a file specified with -K option. If it is not specified, the file is named after the default keytab file, with lowercase Active Directory domain of the managed service account as a suffix. On most systems it would be /etc/krb5.keytab with a suffix of 'domain.example.com', e.g. /etc/krb5.keytab.domain.example.com.

adcli create-msa can be called multiple times to reset the password of the managed service account. To identify the right account with the random component in the name the corresponding principal is read from the keytab. If the keytab got deleted adcli will try to identify an existing managed service account with the help of the fully-qualified name, if this fails a new managed service account will be created.

The managed service account password can be updated with

    $ adcli update --domain=domain.example.com --host-keytab=/etc/krb5.keytab.domain.example.com

and the managed service account can be deleted with

    $ adcli delete-computer --domain=domain.example.com 'myhost!A2c'

In addition to the global options, you can specify the following options to control how this operation is done.

-N, --computer-name=computer

    The short non-dotted name of the managed service account that will be created in the Active Directory domain. The long option name --computer-name is kept to underline the similarity with the same option of the other sub-commands. If not specified, then the first portion of the --host-fqdn or its default is used with a random suffix.

-O, --domain-ou=OU=xxx

The full distinguished name of the OU in which to create the managed service account. If not specified, then the managed service account will be created in a default location.

-H, --host-fqdn=host

Override the local machine's fully qualified DNS domain name. If not specified, the local machine's hostname will be retrieved via gethostname(). If gethostname() only returns a short name getaddrinfo() with the AI_CANONNAME hint is called to expand the name to a fully qualified DNS domain name.

-K, --host-keytab=/path/to/keytab

Specify the path to the host keytab where credentials of the managed service account will be written after a successful creation. If not specified, the default location will be used, usually /etc/krb5.keytab with the lower-cased Active Directory domain name added as a suffix e.g. /etc/krb5.keytab.domain.example.com.

--show-details

After a successful creation print out information about the created object. This is output in a format that should be both human and machine readable.

--show-password

After a successful creation print out the managed service account password. This is output in a format that should be both human and machine readable.

DELEGATED PERMISSIONS

It is common practice in AD to not use an account from the Domain Administrators group to join a machine to a domain but use a dedicated account which only has permissions to join a machine to one or more OUs in the Active Directory tree. Giving the needed permissions to a single account or a group in Active Directory is called Delegation. A typical example on how to configured Delegation can be found in the Delegation section of the blog post Who can add workstation to the domain[1].

When using an account with delegated permissions with adcli basically the same applies as well. However some aspects are explained here in a bit more details to better illustrate different concepts of Active Directory and to make it more easy to debug permissions issues during the join. Please note that the following is not specific to adcli but applies to all applications which would like to modify certain properties or objects in Active Directory with an account with limited permissions.

First, as said in the blog post it is sufficient to have "Create computer object" permissions to join a computer to a domain. But this would only work as expected if the computer object does not exist in Active Directory before the join. Because only when a new object is created Active Directory does not apply additional permission checks on the attributes of the new computer object. This means the delegated user can add any kind of attribute with any value to a new computer object also long as they meet general constraints like e.g. that the attribute must be defined in the schema and is allowed in a objectclass of the object, the value must match the syntax defined in the schema or that the sAMAccountName must be unique in the domain.

If you want to use the account with delegated permission to remove computer objects in Active Directory (adcli delete-computer) you should of course make sure that the account has "Delete computer object" permissions.

If the computer object already exists the "Create computer object" permission does not apply anymore since now an existing object must be modified. Now permissions on the individual attributes are needed. e.g. "Read and write Account Restrictions" or "Reset Password". For some attributes Active Directory has two types of permissions the plain "Read and Write" permissions and the "Validated Write" permissions. For the latter case there are two specific permissions relevant for adcli, namely

? Validated write to DNS host name

? Validated write to service principal name

Details about the validation of the values can be found in the "Validated Writes" section of [MS-ADTS], especially dNSHostName[2] and servicePrincipalName[3]. To cut it short for "Validated write to DNS host name" the domain part of the fully-qualified hostname must either match the domain name of the domain you want to join to or must be listed in the msDS-AllowedDNSSuffixes attribute. And for "Validated write to service principal name" the hostname part of the service principal name must match the name stored in dNSHostName or some other attributes which are not handled by adcli. This also means that dNSHostName cannot be empty or only contain a short name if the service principal name should contain a fully-qualified name.

To summarize, if you only have validated write permissions you should make sure the domain part of the hostname matches the domain you want to join or use the --host-fqdn with a matching name.

The plain read write permissions do not run additional validations but the attribute values must still be in agreement with the general constraints mentioned above. If the computer object already exists adcli might need the following permissions which are also needed by Windows clients to modify existing attributes:

? Reset Password

? Read and write Account Restrictions

? Read and (validated) write to DNS host name

? Read and (validated) write to service principal name

additionally adcli needs

? Read and write msDS-supportedEncryptionTypes

This is added for security reasons to avoid that Active Directory stores Kerberos keys with (potentially weaker) encryption types than the client supports since Active Directory is often configured to still support older (weaker) encryption types for compatibility reasons.

All other attributes are only set or modified on demand, i.e. adcli must be called with an option the would set or modify the given attribute. In the following the attributes adcli can modify together with the required permissions are listed:

? userPrincipalName

  ? Read/Write userPrincipal Name

? msDS-supportedEncryptionTypes

  ? Read/Write msDS-SupportedEncryptionTypes

? dNSHostName

  ? Read/Write dNSHostName

  ? Read and write DNS host name attributes

  ? Validated write to DNS host name

? servicePrincipalName

  ? Read/Write servicePrincipalName

  ? Validated write to service principal name

? operatingSystem

  ? Read/Write Operating System

? operatingSystemVersion

  ? Read/Write Operating System Version

? operatingSystemServicePack

  ? Read/Write operatingSystemServicePack

? userAccountControl

  ? Read/Write userAccountControl

? description

  ? Read/Write Description

For the management of users and groups (adcli create-user, adcli delete-user, adcli create-group, adcli delete-group) the same applies only for different types of objects, i.e. users and groups. Since currently adcli only supports the creation and the removal of user and group objects it is sufficient to have the "Create/Delete User objects" and "Create/Delete Group objects" permissions.

If you want to manage group members as well (adcli add-member, adcli remove-member) "Read/Write Members" permissions are needed as well.

Depending on the version of Active Directory the "Delegation of Control Wizard" might offer some shortcuts for common task like e.g.

? Create, delete and manage user accounts

? Create, delete and manage groups

? Modify the membership of a group

The first 2 shortcuts will provided full access to user and group objects which, as explained above, is more than currently is needed. After using those shortcut it is a good idea to verify in the "Security" tab in the "Properties" of the related Active Directory container that the assigned permissions meet the expectations.

BUGS

Please send bug reports to either the distribution bug tracker or the upstream bug tracker at https://bugs.freedesktop.org/enter_bug.cgi?product=realmd&component=adcli

SEE ALSO

realmd(8), net(8), sssd(8)

Further details available in the realmd online documentation at http://www.freedesktop.org/software/realmd/

NOTES

1. Who can add workstation to the domain
   https://docs.microsoft.com/en-us/archive/blogs/dubaisec/who-can-add-workstation-to-the-domain
2. dNSHostName
   https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-adts/5c578b15-d619-408d-ba17-380714b89fd1
3. servicePrincipalName
   https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-adts/28ca4eca-0e0b-4666-9175-a37ccb8edada