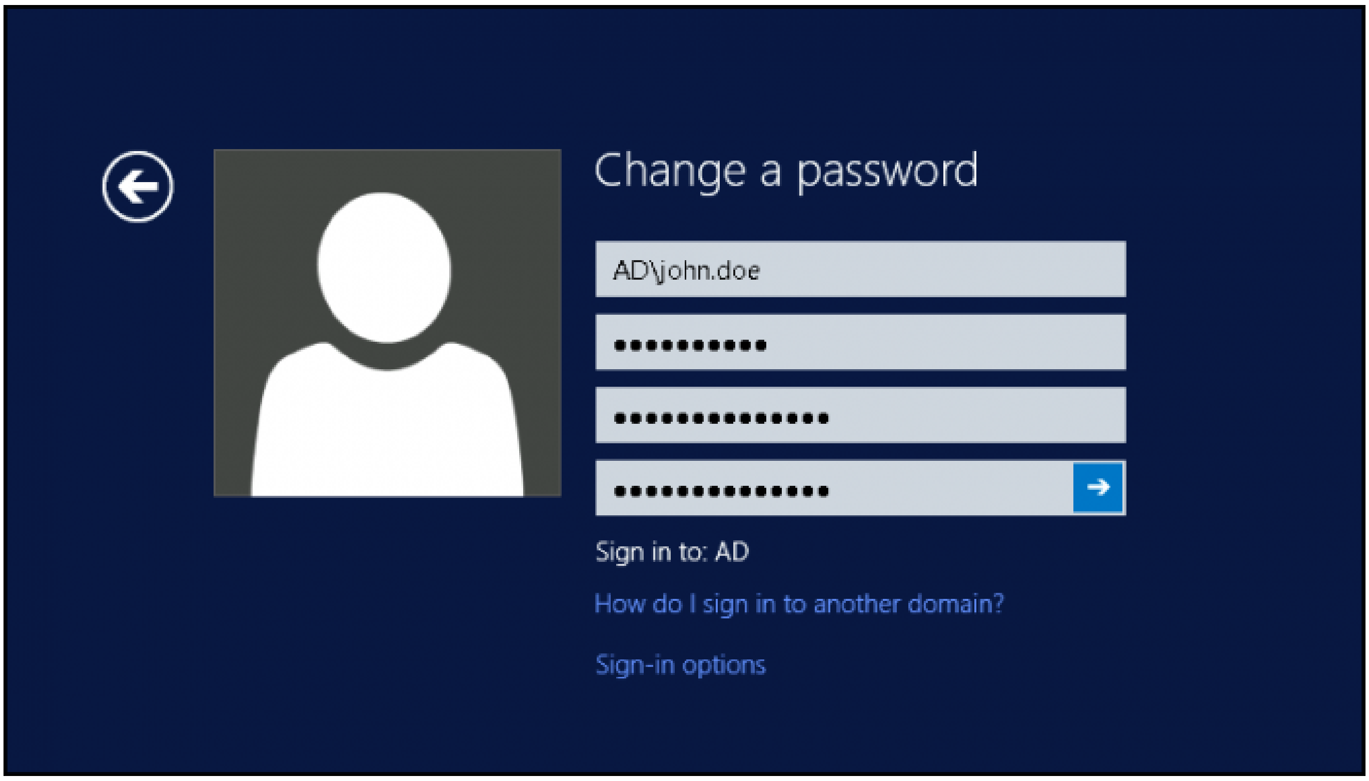
Password synchronization

synchronization, password, filter, passwordfilter, echo, echos, uniform, password, system, systems, change, passwords, CTRL+ALT+DELETE, CTRL, ALT, DELETE

Password synchronization allows users **very simply** change password to **all connected system** just from theirs workstations → **only one password trough all system**.



CzechIdM provides feature **synchronize passwords from system**. IdM receives request for password synchronization from external system (for example Active Directory, Open Ldap). After IdM successful validates request for password synchronization. Password will be distributes into connected systems via classic password change behavior.

Setup the password synchronization is easy - administrator musts just activate the feature **password synchronization** in IdM and setup external system for sending password change requests to IdM. Then users can simply initialized password change form on his own workstation and start changing the passwords by standard behavior. For example Windows stations has standard shortcut CTRL+ALT+DELETE for initialize password change form. External system sends request for password change into IdM and IdM will take care of the rest of the password change process.

IdM process the password change request and **distributes password to all next connected system**. The result is that user just change the password once via his local workstation and **password is same trough all connected systems**. User uses only **ONE password**.

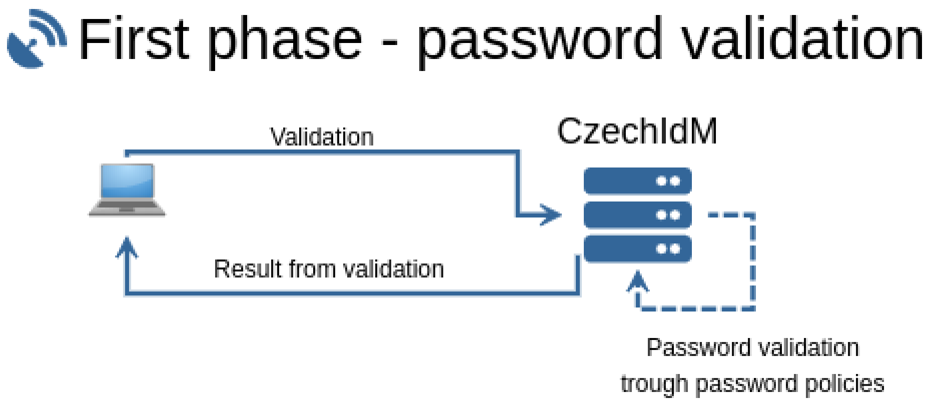
Users also don't need change password via some special form or change password on every system that they use separately. Just one simple change trough own local workstation is enough.

Workstation based on Active Directory uses password synchronization via password filter more about this component can be found there.

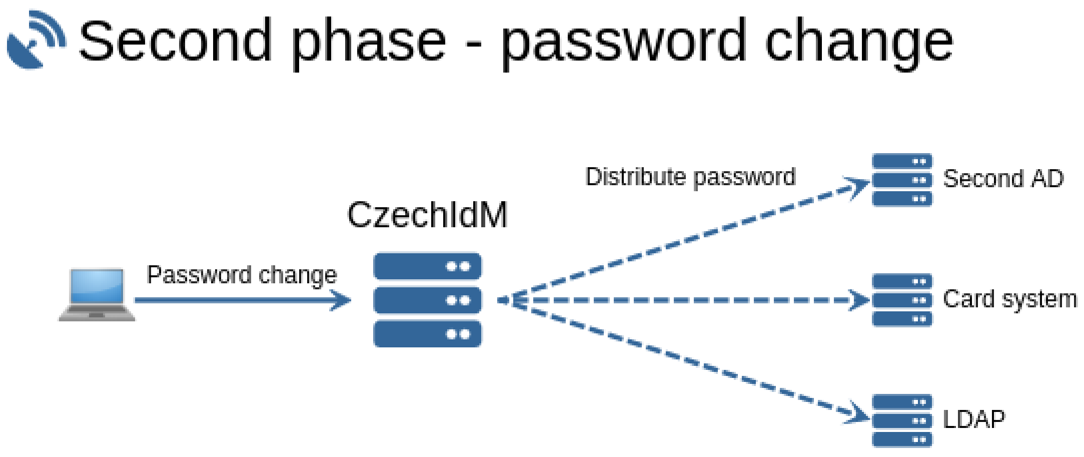
Password synchronization works in two phases. First phase is password **validation** and the second is **password change** in IdM and distribution password to next system. IdM receives password trough **HTTPS** protocol and REST calling.

Passwords are **never saved in plain text** in IdM.

Phases in password synchronization


First phase - password validation
Validation
CzechIdM
Result from validation
Password validation trough password policies

In first phase is synchronized password validated trough defined password policies in IdM. In IdM has every connected system specific password policy including IdM application. Through these policies will be password validated and the result will be send back to user. When password doesn't meet only one password policy the password will not be changed even on users local workstation. Password must be successfully validated trough given password policies and only then can be changed. Validation can be repeated.



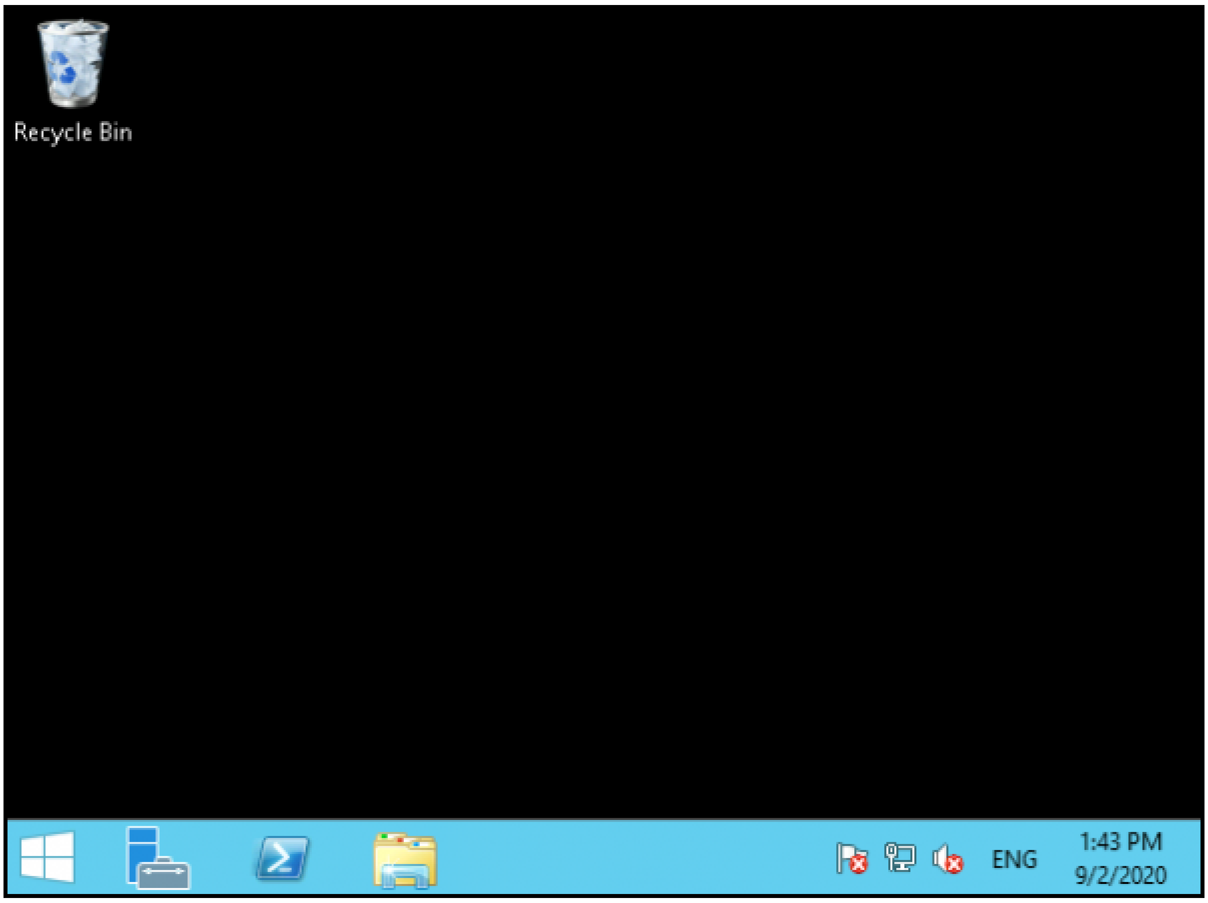
After success password validation continues password synchronization with password change. Password will be for first changed in IdM and all another connected systems. Basically the password change has same process as standard password changed that can be done in IdM by basic password change form. Before this phase must be password successfully validated, otherwise isn't possible change the password.

Each phase has own log format and it is very easy for check and control password synchronization with own log control mechanism, or just simple via log agenda in IdM. Simple example of parsed logs:

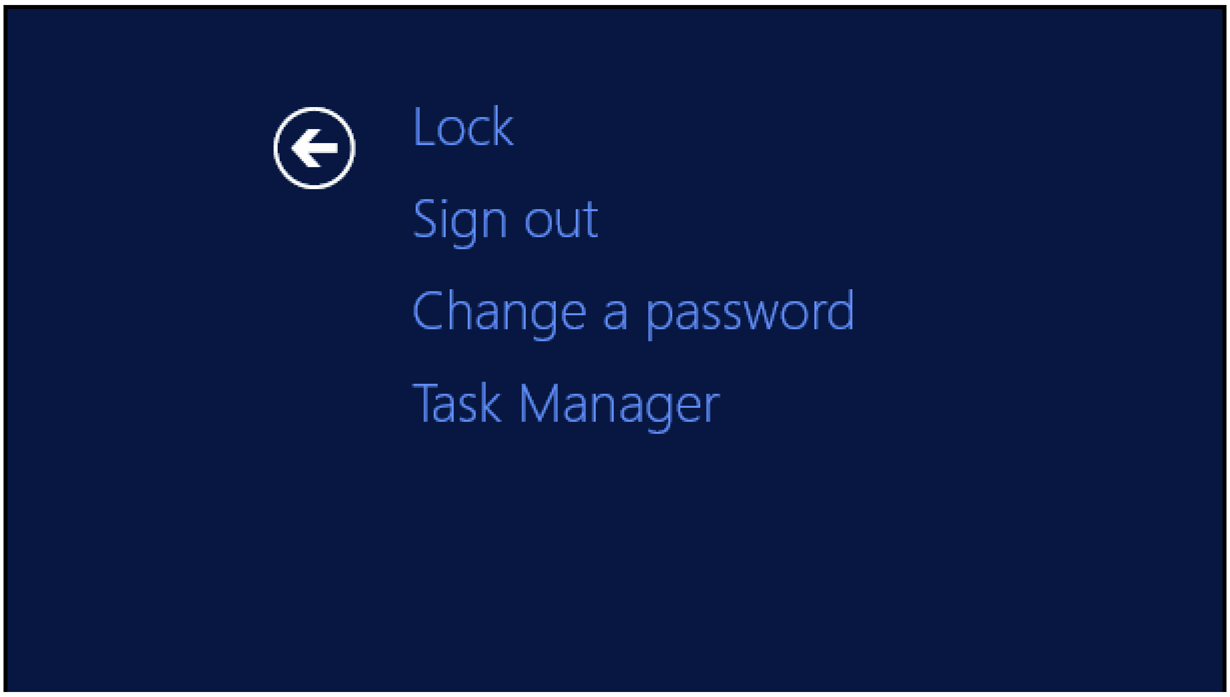
```
INFO validate : Validation request pass! For identity [john.doe] and system code [AD users]. Log identifier: [1454118233]. Password filter version: [1.0.0].
INFO change : Change request from resource [AD users] for identity identifier [john.doe] starting. Log identifier: [3621046325]. Password filter version: [1.0.0].
INFO change : Password change will be processed. For identity [john.doe] and system [AD users] was found these accounts for change [john.doe,j.doe,doe,john_doe]. Log identifier: [3621046325]. Password filter version: [1.0.0].
```

Step by step password synchronization from Active Directory

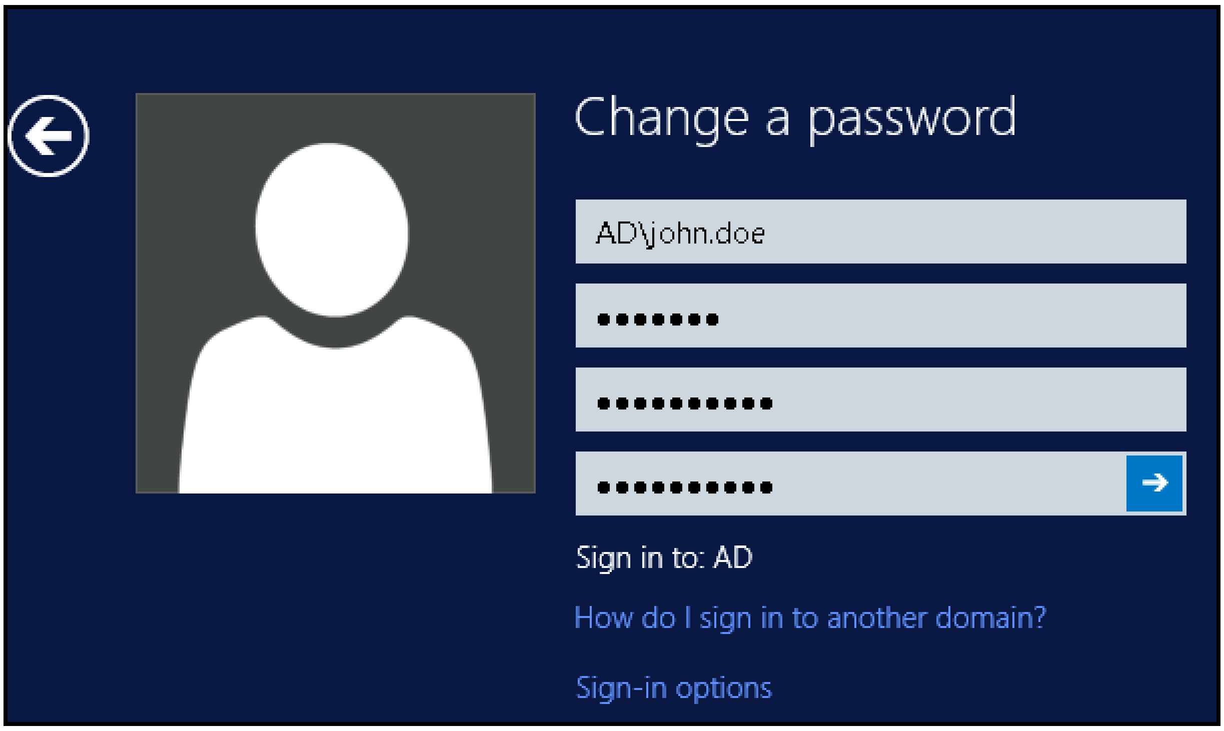
User wants change password on his own workstation and press CTRL+ALT+DELETE for initialize password change process:



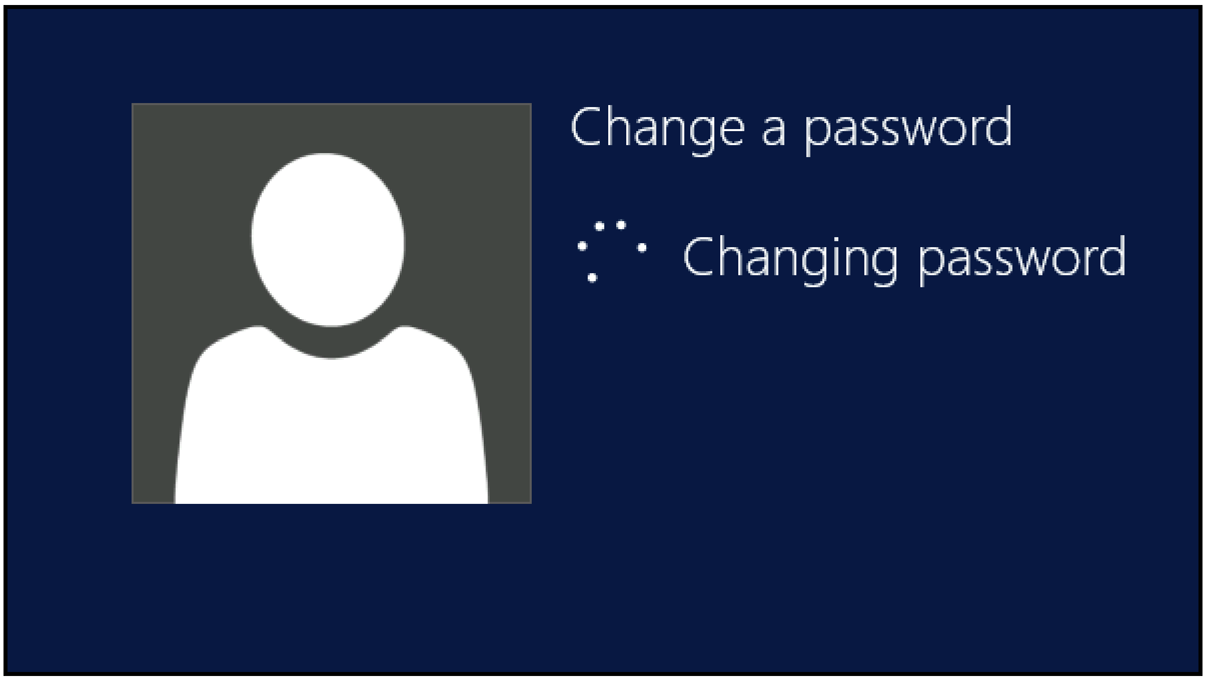
After user press CTRL+ALT+DELETE Windows shows context menu that allow change password by option "Change a password":



On password change form fill old password and new password and then confirm the password change by enter:

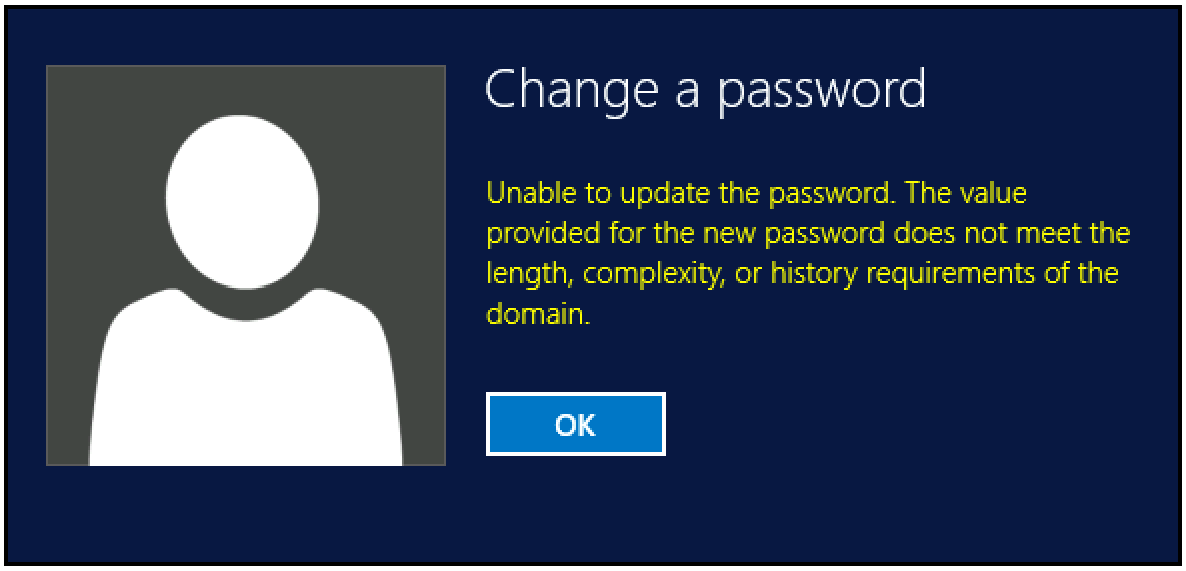


After user press enter starts whole process described in above. Validation phase and then change phase. User doesn't about anything on background and some phases:



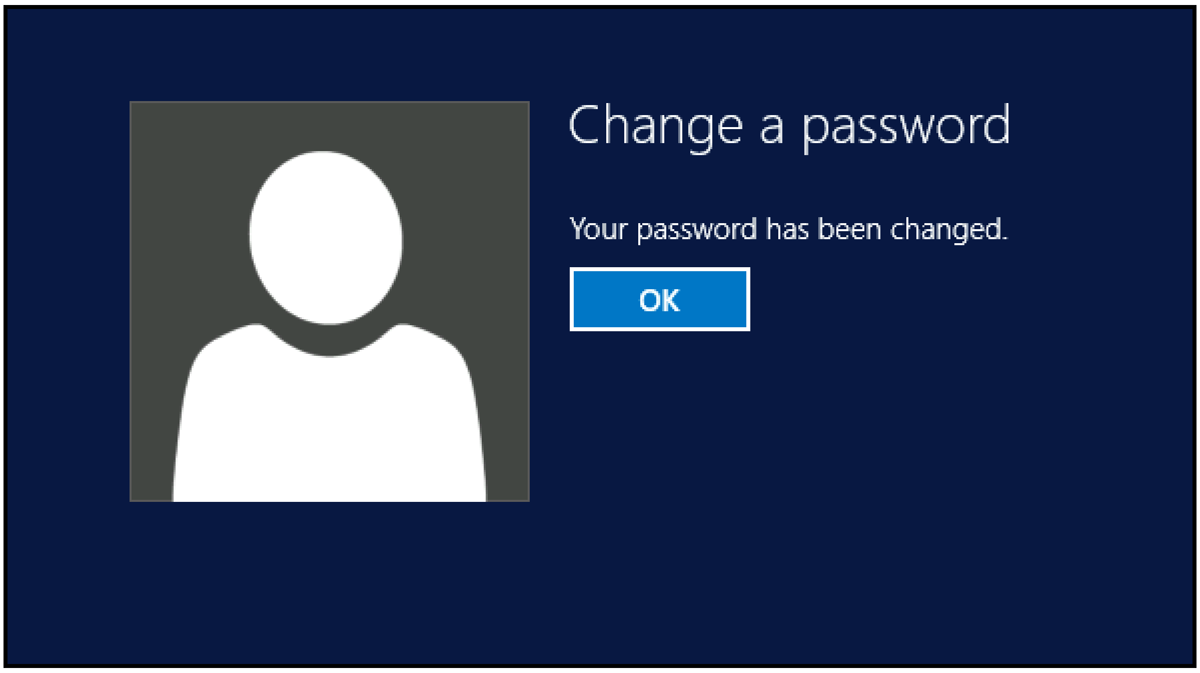
If new password doesn't meet with policies defined in AD or inIdM Windows show to users standard info message about not valid password:

```
Unable to update the password. The value provided for the new password does
not meet the length, complexity, or history requirements of the domain
```


Change a password

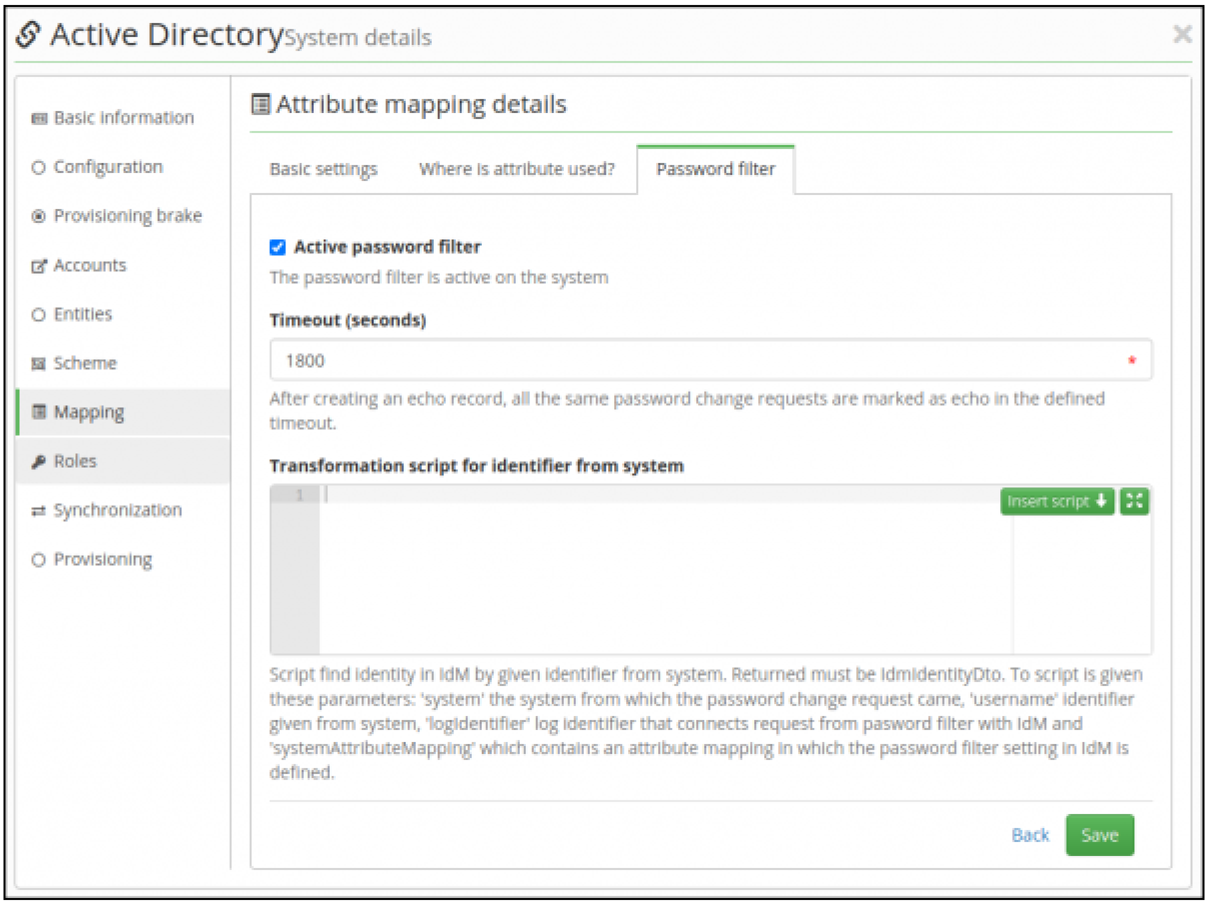
Unable to update the password. The value provided for the new password does not meet the length, complexity, or history requirements of the domain.

After successful password change Windows shows success result:


Change a password

Your password has been changed.

How it works in detail?



Active password filter
The password filter is active on the system

Timeout (seconds)
1800
After creating an echo record, all the same password change requests are marked as echo in the defined timeout.

Transformation script for identifier from system

Script find identity in IdM by given identifier from system. Returned must be IdmIdentityDto. To script is given these parameters: 'system' the system from which the password change request came, 'username' identifier given from system, 'logIdentifier' log identifier that connects request from pasword filter with IdM and 'systemAttributeMapping' which contains an attribute mapping in which the password filter setting in IdM is defined.

Password synchronization can be setup **only for system that has provisioning mapping** for identities and mapped **password** (standard attribute name: __PASSWORD__) in system atribute mapping. Configuration for password filter will be available after checkbox **Attribute with password** will be checked.

Password synchronization has two phase as we said above. Each phase has own REST endpoint. These REST endpoints must be called with **POST** method and each endpoint has **own permission**. First endpoint provides validation and the second provides password change.

CzechIdM has two REST endpoints that is called by password filter implementation. The first endpoint **validate must be called** before the second endpoint **change**. Call directly **change** REST endpoint isn't allowed and **error will be thrown**.

The **change** REST endpoint doesn't validate password again. Calling first **validation** REST endpoint is required.

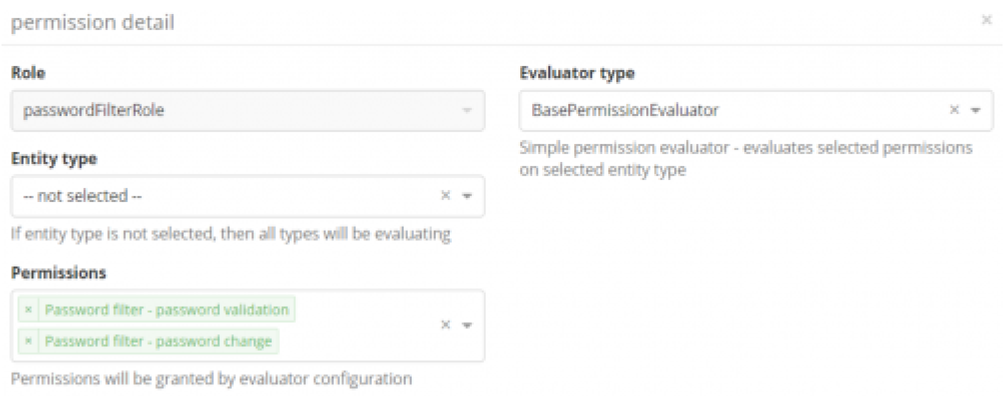
Both IdM REST endpoints receive following parameters from password filter:

- `password` - new password changed on system,
- `username` - for the username is new password changed,
- `resource` - unique identifier system where is password changed,
- `logIdentifier` - identifier that is printed into all logs - identifier helps connect request from password filter with IdM process,
- `version` - version of password filter.

Parameters `logIdentifier` and `version` is part for **every log record** written into application log. Both these parameter is not required. Parameters `password`, `username` and `resource` is **required**.

Both REST endpoints has new permissions `SYSTEM\_PASSWORDFILTERVALIDATE` for validation and for change `SYSTEM\_PASSWORDFILTERCHANGE`. Identity that password filter used for calling these REST endpoint must have both these permissions. For authentication can be used standard **basic** authentication or **CIDMST** token. Recommended is use **CIDMST** token with long expiration date. It's recommend create new identity with these permissions.

Permission settings for password filter


permission detail

Role: passwordFilterRole

Evaluator type: BasePermissionEvaluator

Simple permission evaluator - evaluates selected permissions on selected entity type

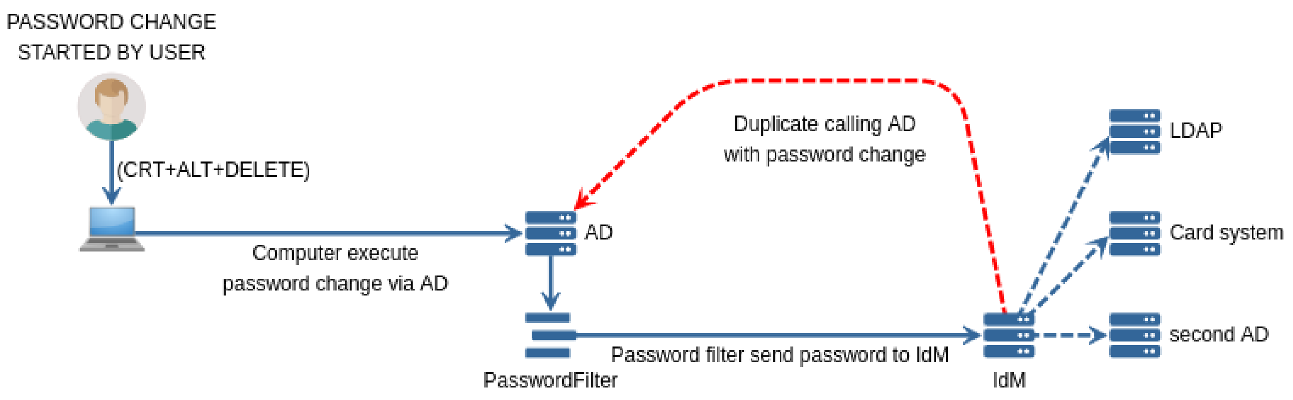
Entity type: -- not selected --

If entity type is not selected, then all types will be evaluating

Permissions: Password filter - password validation; Password filter - password change

Permissions will be granted by evaluator configuration

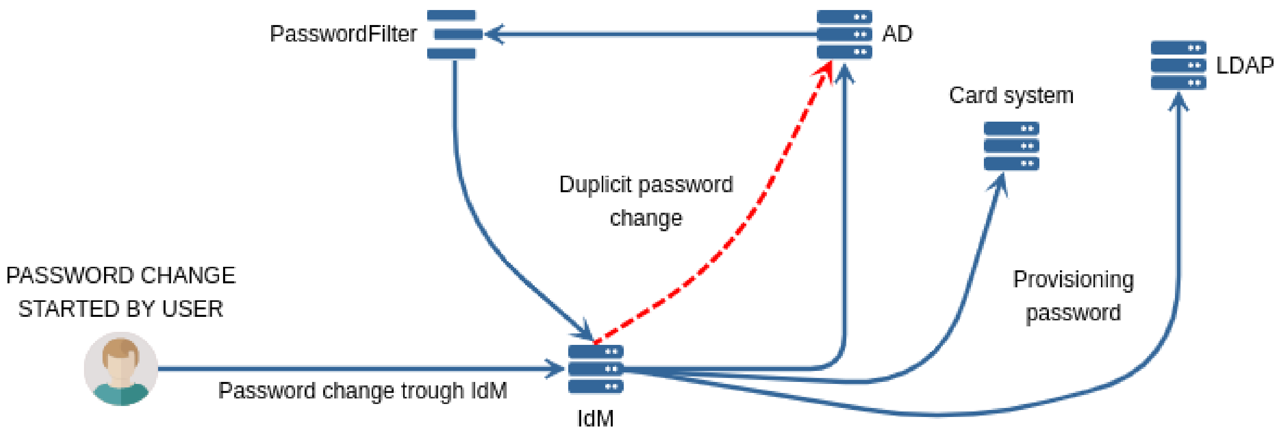
Why we want check echos?


PASSWORD CHANGE STARTED BY USER
(CRT+ALT+DELETE)
Computer execute password change via AD
AD
Duplicate calling AD with password change
PasswordFilter
Password filter send password to IdM
IdM
LDAP
Card system
second AD

Without checking echos is very likely that password filter call IdM, IdM will process password and provisioning to all another system, **but including** the system from which was password change executed. The password filter on the system again catch this second password change and whole password change process is executed repeatably.

As you see on picture at left side. The **user initialize** password change on his **own computer**. The computer process the password **change trough AD**. AD **send password to IdM** via password filter and IdM check echo's and then process password into next systems *(eq LDAP, Card system, AD's, …)*. If IdM doesn't check echo the password will be sent againt to AD from which was password change executed for first time - red line. The AD doesn't know about password initializer and will process password again against via password filter to IdM and whole password change can cycles to infinity.

When IdM check echo the process marked as red line will never happen.



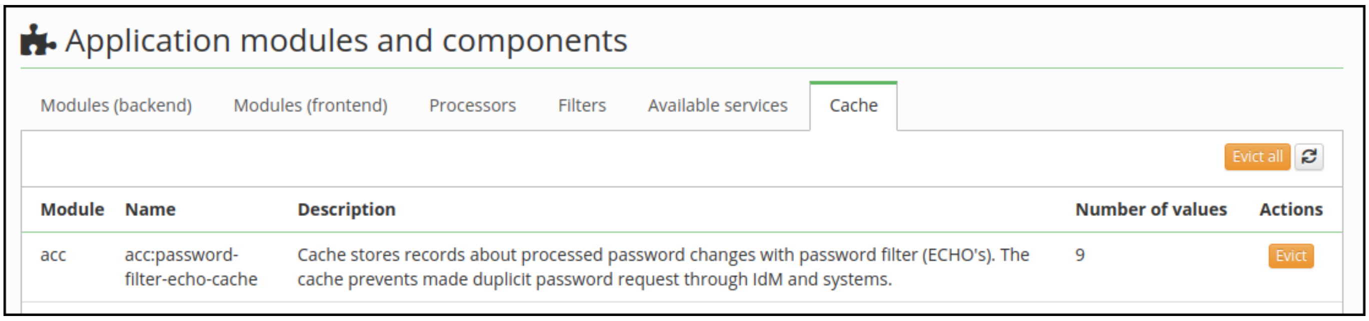
The second image (right image) present password change **executed by user from IdM**. Password change can be executed by **public change form**, **classic form** change on user detail, **password reset**, **password generation**, even **create user with password**. For all these cases **IdM creates echo record** and it is not possible for a cycle to occur.

After user executed password change. IdM will **create** and process **provisioning operation** with password to all systems that has mapped password. For systems with password filter will be also **create echo** records and then will be processed classic provisioning with password. AD and other system with password *(eq: LDAP, card system, …)* **receives request** with password change and process. AD **process the password** change request via **password filer**. AD's password filter doesn't know about who sended the request and password filter **send request again to IdM**. IdM will **check echo** for the given system - echo for AD exists and IdM will not process password again against trough AD - red line.

If IdM does not check the echo. The process marked as red line will be processed and the cycle occurs.

Echo and flags

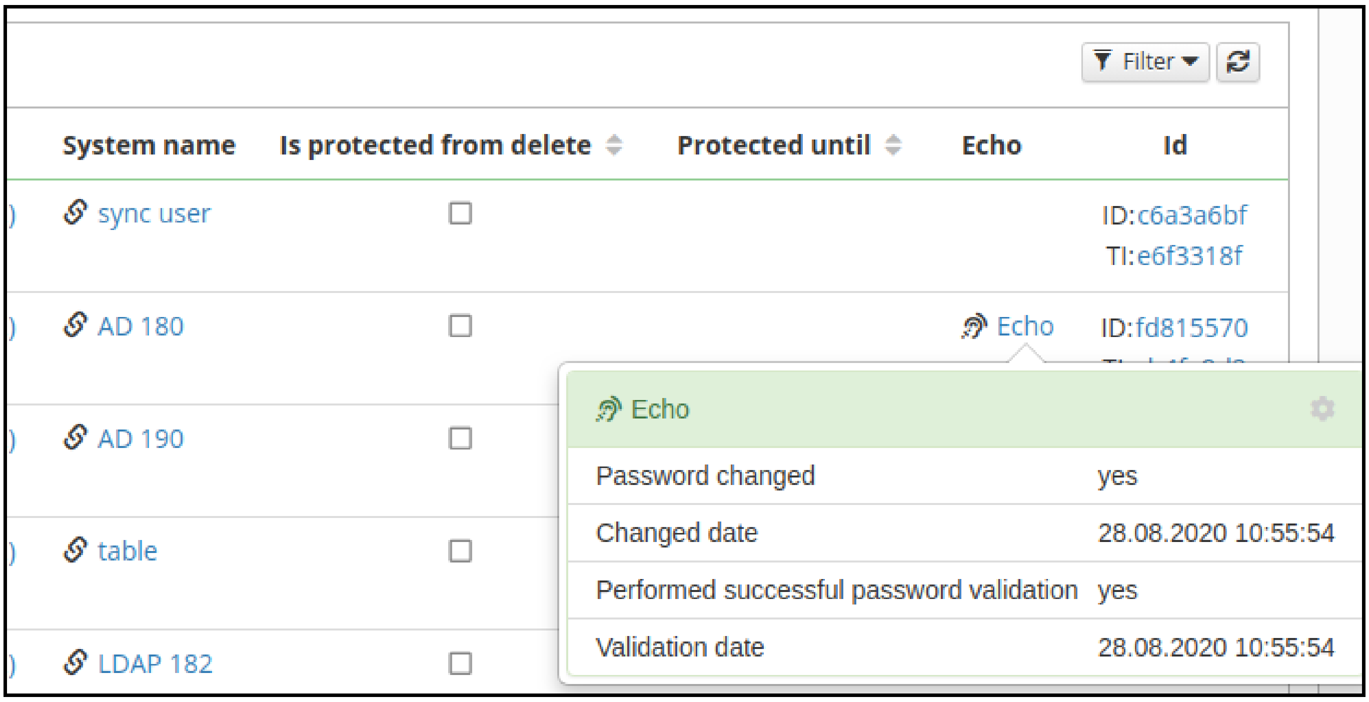
All echo records are stored in distributed cache in IdM. Echo record **can be evicted** by application configuration (Settings→Modules→Cache). In same agenda is also shown how many records are now in cache. Maximum time that will be echo stored is **12 hours**. **Longer period for echo storation cannot be set!**



Application modules and components

Module	Name	Description	Number of values	Actions
acc	acc:password-filter-echo-cache	Cache stores records about processed password changes with password filter (ECHO's). The cache prevents made duplicit password request through IdM and systems.	9	Evict

Echo record are persisted in cache - with application restart may be all echos deleted - it depend on cache configuration.



System name	Is protected from delete	Protected until	Echo	Id
sync user	☐			ID:c6a3a6bf TI:e6f3318f
AD 180	☐		Echo	ID:fd815570
AD 190	☐			
table	☐			
LDAP 182	☐			

Echo

Password changed	yes
Changed date	28.08.2020 10:55:54
Performed successful password validation	yes
Validation date	28.08.2020 10:55:54

Global agenda for all echo doesn't exists. But on user detail in section Accounts can be founded record for each user.

Echo column will be shown only when logged user has permission for `Show system information`. The permission is set by profile configuration for each user separately.

In modal windows with echo detail are show 4 rows with these information:

- **Password changed** (boolean) - flag/check if password was **successfully** changed,
- **Changed date** (date) - date when was password **successfully** changed,
- **Performed successful password validation** (boolean) - flag if password validation was success or not,
- **Validation date** (date) - validation date.

Echo record will be created after password filter call first validation REST endpoint. Validation set flag **Performed successful password validation**. The result value depends on the result from validation. **For calling change REST endpoint is required successfully validation by validate REST endpoint.**

Password validation request

There is step by step behavior processed by endpoint **VALIDATE** (*eq: http://localhost/idm/api/v1/systems/password-filter/validate/*):

Validation must pass trough all password policies! Only one failed check trough password policy, will return failed result to password filter.

1. **find correct system** (SysSystemDto) by parameter `resource`,
    1. if system cannot be found **exception will be throw** (404 - PASSWORD_FILTER_SYSTEM_NOT_FOUND),
2. **find mapped attribute** that contains configuration for password filter,
    1. if attribute cannot be found or has bad configuration **exception will be throw** (404 - PASSWORD_FILTER_DEFINITION_NOT_FOUND),
3. **find identity** for given parameter `username`,
    1. if identity for whom the password is being validated cannot be found in IdM **exception will be throw** (404 - PASSWORD_FILTER_IDENTITY_NOT_FOUND),
    2. for more information about find specific identity see this section
4. **check if exists uniform password definition**
    1. uniform password definition unite all another password policies by systems,
    2. on the systems will be also changed password
    3. for system with password filter will be also set echo record,
5. **check echo record** for the system from which the password validation request came. Checking echo for validation has following steps:
    1. **echo doesn't exists** □ - password will be processed
    2. **echo record exists** - password in echo record **isn't same** □ - password validation **will be** processed
    3. **echo record exists** - password in echo record **is same** □ - password validation **will not be** processed - to password filter will be returned password is valid
    4. **echo record is already expired** □ - password will be processed
    5. **echo record was already changed and validated** □ - skip all another process - to password filter will be returned password is valid,
    6. **echo record was already only checked and someone call second validation** □ - password will be processed
6. **check if exists uniform password with IdM system** (change in IdM),
7. create **final password policy set** that may include default password policy,
8. **process validation**,
9. when only one password policy **failed** - **set/create echos** for all managed accounts **with failed check flag**,
10. **successfully** validate password trough password policies - **set/create echos** for all managed accounts with **valid check flag**.

Password change request

Second endpoint **CHANGE** has following behavior (*eq: http://localhost/idm/api/v1/systems/password-filter/change/*):

First steps for finding system, attribute and identity are same as validation

- **find correct system** (SysSystemDto) by parameter `resource`,

1. if system cannot be found **exception will be throw** (404 - PASSWORD_FILTER_SYSTEM_NOT_FOUND),

- **find mapped attribute** that contains configuration for password filter,

1. if attribute cannot be found or has bad configuration **exception will be throw** (404 - PASSWORD_FILTER_DEFINITION_NOT_FOUND),

- **find identity** for given parameter `username`,

1. if identity cannot be found **exception will be throw** (404 - PASSWORD_FILTER_IDENTITY_NOT_FOUND),
2. for more information about find specific identity see this section

- **check if exists uniform password definition**

1. uniform password definition unite all another password policies by systems,
2. on the systems will be also changed password
3. for system with password filter will be also set echo record,

- **check echo record** for the system from which the password change request came. Checking echo for change request has following steps:

1. **echo doesn't exists** - exception will be throw (403 - PASSWORD_FILTER_NOT_VALID_CHANGE_REQUEST),
2. **echo record exists** - password in echo record **isn't same** - exception will be throw (403 - PASSWORD_FILTER_NOT_VALID_CHANGE_REQUEST),
3. **echo record exists** - password in echo record **is same** - password change request continues,
4. **password wan't successfully validate** - exception will be throw (403 - PASSWORD_FILTER_NOT_VALID_CHANGE_REQUEST),

- prepare final account set for password change request, - set echos for all managed accounts (accounts for systems where is active password filter), - process password change event trough IdM (for more information see section bellow).

PASSWORD event

Endpoint change publish after all check new event **PASSWORD** for entity IdmIdentityDto. **The whole event including provisioning is synchronous now**. Classic **PASSWORD** event provides password validation, but event published by password filter has all validations skipped by property `IdentityProcessor.SKIP\_PASSWORD\_VALIDATION`.

One part of **PASSWORD** event also provides unite by uniform password system. The uniform password behavior is not wanted because cycles can arise. The whole event **PASSWORD** contains second parameter `PasswordFilterManager.EXCLUDED\_SYSTEM` that skip password change on some specific system.

If you use password reset module and setup the configuration correctly - add the event types `PASSWORD\_RESET` and `PASSWORD\_GENERATE` the echo system and uniform password will works even for password reset module.

Before password filter publish **PASSWORD** event is created echos for managed system. But processor IdentityPasswordProvisioningProcessor (acc module) can also remove these records after provisioning with password will not be executed successfully.

When provisioning with password ended with another operation state than **executed** from echo record will be removed flag for change.

Find correct identity by username from system

The script used to transform the username **must be of type** `Transform from a system`!

From external system IdM receives **user identifier** - `username` parameter. If for the identifier exists equal account (UID) the owner of the account will be used as owner of the password change request. If doesn't exists equal account, IdM checks if exists identity (username) with the given **user identifier**.

But some external system has own system identifier. For these cases exists **transformation script** that allows to find correct owner of password change request. It is required returned identity otherwise exception will be thrown. The **script has to be** of the `Transformation from a system` type, **another script types will not work**!