Password policies

password, security, configuration

The types of policies - validation and generation - are determined by enumeration PasswordPolicyTypeEnum. The policies marked as the GENERATE type can also become the type determined by enumeration PasswordPolicyGenerateTypeEnum, this is a generation type (random, passphrase).

Standard policies

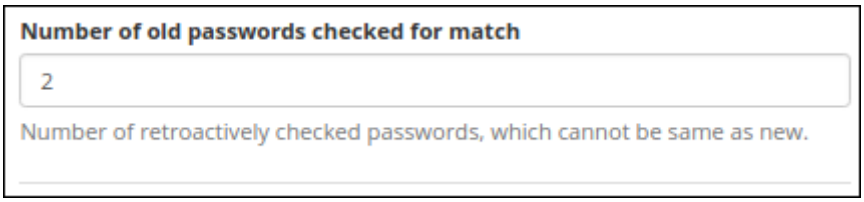
Standard policy for validation

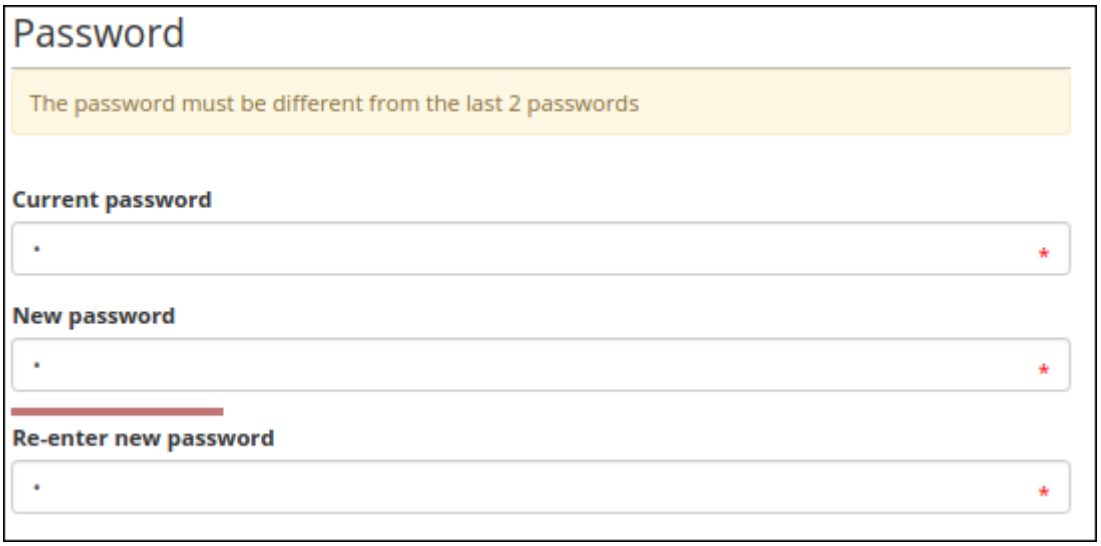
If there is a standard validation policy *(in the code we talk about default policy)* in the CzechIdM system, it is used for password validation against the CzechIdM system and for all the other systems with no determined policy. In the service IdmPasswordPolicyService there are three methods of validation, all of them accepting IdmPasswordValidationDto as a parameter containing new password, the IdmPassword object, and the identity to which the password belongs to. Another parameters of the `validate` methods is a list or one object itself IdmPasswordPolicy. There is also a possibility of validation only with the object IdmPasswordValidationDto, standard policy will be used for its validation, if applicable.

The identity is also part of the object IdmPassword but this object does not necessarily need to exist, like e.g. synchronization with the system.

If the standard validation policy does not exist, the validation of new passwords is *turned off* and the password is always valid, also the attribute "valid till" will not be filled.

Standard validation policy allow set how many password can't be same as passwords before. The feature calls **maxHistorySimilar**. Check with old password is done with new password (during password change) and all old password is in BCrypt hash. This check is done during password policy validation and works for all users. For users with `APP_ADMIN` permission is the check skip.





The standard validation policy also allow set **maxUnsuccessfulAttempts** this feature block user after try unsuccessful login. User will be block for seconds. This time period (seconds) is defined in attribute **blockLoginTime**. After user for firsttime reached unsuccessful attempts is blocked for seconds defined in **blockLoginTime**. **But** after user will continued with unsuccessful login is after second reach blocked for the time period multiplied by 2. After third reached is user blocked for time period multiplied by 3 and etc.

After successful login is attributes that stores actual unsuccessful attempts and block time period cleared. This is usefful feature for helpdesk or administrator that wants change password user and clear the block for login. **This is allow only when old password isn't required.**

From public password change **isn't possible** change password during user has blocked login. Same rule is for user itself. When user has blocked login and on another computer is loged in into CzechIdM password change isn't possible.

Standard policy for password generation

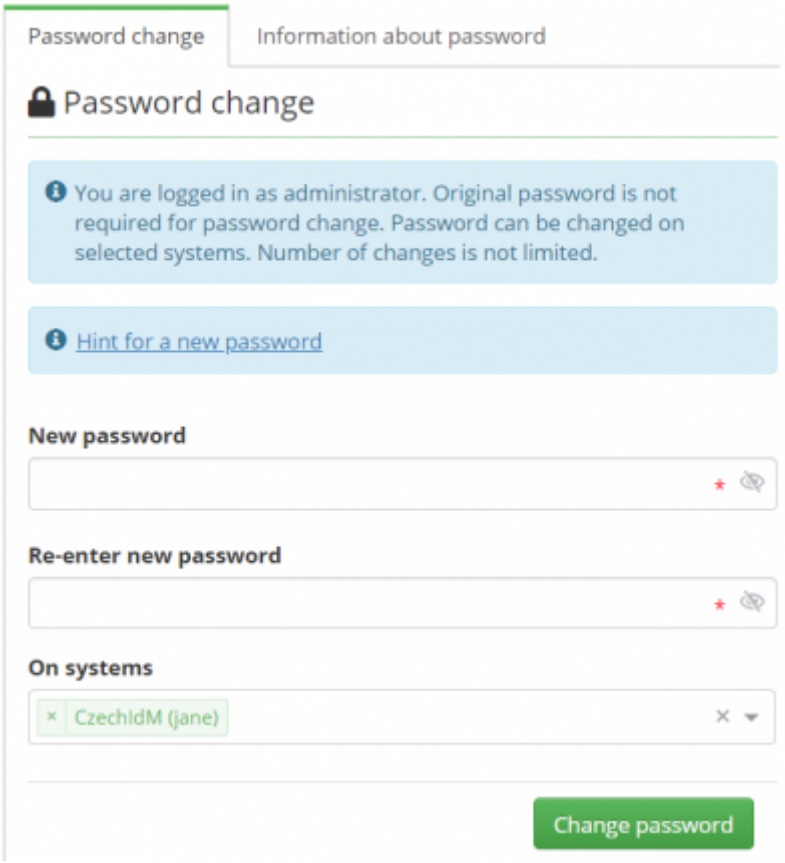
For generation, there are methods in IdmPasswordPolicyService available, marked as `generate`. As a parameter, the methods accept the object IdmPasswordPolicy, this policy **must** be of the GENERATE type. Or, as the case may be, a method without parameters may be used. In this case, standard generation policy is used; if it doesn´t exist, a random password of 8 characters is generated.

Policy <-> System link

Every system may have two policies - one for generation, and one for validation. If any of the two policies is missing, it is replaced by the standard policy of the CzechIdM system.

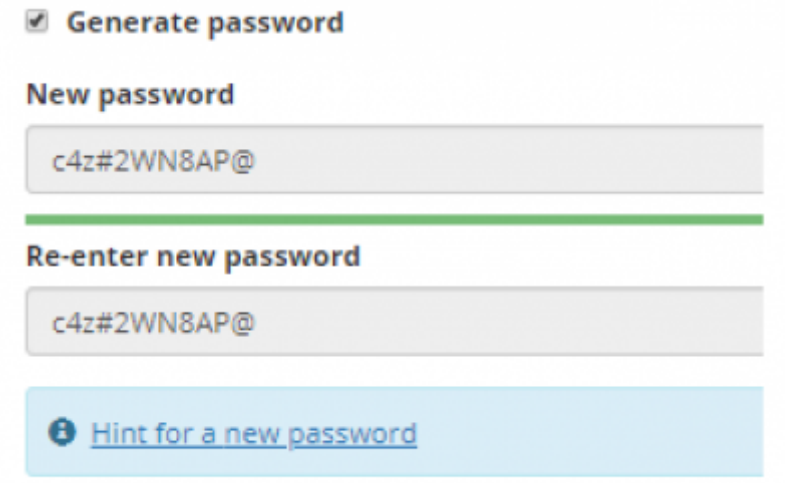
When creating a new account in the system, a new password is generated using the generation policy. After the account is successfully created, the password is sent in a notification to the user (by SMS, e-mail, ...). Sending of this notification can be disabled by disabling the appropriate processor, i.e. setting the application property `idm.sec.acc.processor.provisioning-send-notification-processor.enabled=false`.

When changing the password for the systems, including the CzechIdM system, their validation policies are used, in case of absence of the validation policy for a system, the standard validation policy is used *(if there is no standard validation policy, the password is always valid - see above)*.



Password generation

The CzechIdM system offers two possibilities of password generation: random and passphrase. Password generation takes places when creating a new user on the CzechIdg system. The password can be generated again using checkbox.



Random

A random password is generated depending on the minimum number of characters set up in the policy. Password generation takes place against characters bases which are part of every policy. Forbidden characters are removed from these bases and then a password with a minimum number of characters defined by each of the rules is generated (minimum number of numbers, minimum number of special characters, etc.). Then the password is completed by different characters up to the minimum number of characters, and finally with a random number of characters ranging between the minimum and the maximum number of characters. After being generated, the whole password is jumbled up. All this process takes place in the class PasswordGenerator. The PasswordGenerator **does not accept** the password policy as a parameter, but the interface PasswordGenerate.

Password policies that use prefix and suffix will be generated like random generation, but at the end will be added prefix/suffix. Both strings are optional. Random string is generated from settings and after generated is added prefix/suffix ⇒ generated string must pass all password policy settings. **Final password may not pass** maximum length and another generation validations.

Passphrase

The second option is to generate the password using passphrase. Again, the password generation is done by PasswordGenerator. For the purposes of passphrase generation, a dictionary which can be found in diceware_cs.csv is necessary.

Examples of passphrase generation containing 5 words

```
masitou karibska kartaga dotovany cipy
nesnadny odvijela cedicova zapolici projel
bunicito tesnenim udelaji zatahnut lenoseni
agend civilnim udrzbou kopati odvsiven
```

Validation

When changing the password for one or more systems, the password is validated for all the found policies. If only one policy for more systems is used, only one validation takes place. When complying with all the validation rules, the password is changed on all the marked systems. In case of non-compliance, validation message will be displayed - see further below.

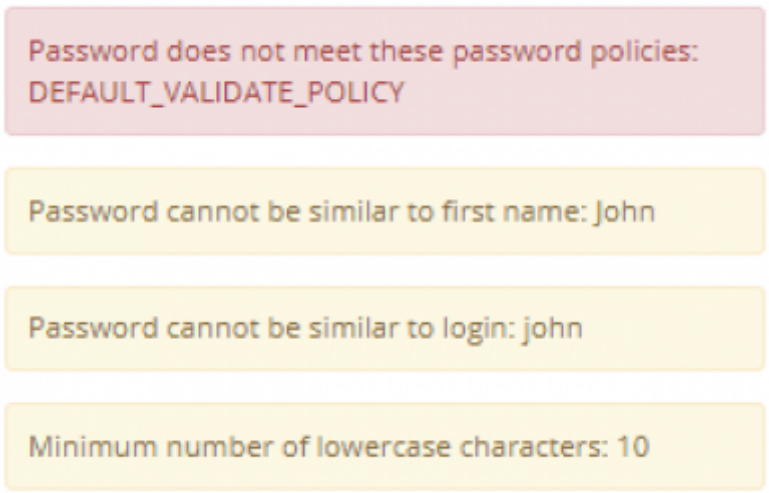
The password is validated through policies and it always needs to comply with the highest (maximum) requirements, i. e. minimum number of characters, maximum number of characters, minimum number of numbers, minimum number of upper-case characters, minimum number of special characters. In case the requirements of two policies turn out to be contradictory (e.g. the minimum number of characters of one policy is lower than the maximum number of characters of another policy), and so **the password WILL NEVER BE marked as valid**, then it is the task of the CzechIdM system administrator to remedy the situation.

The change of password can also be impossible due to minimum validity of the password - this piece

of information is verified when validating the password.

Visualization of validation messages

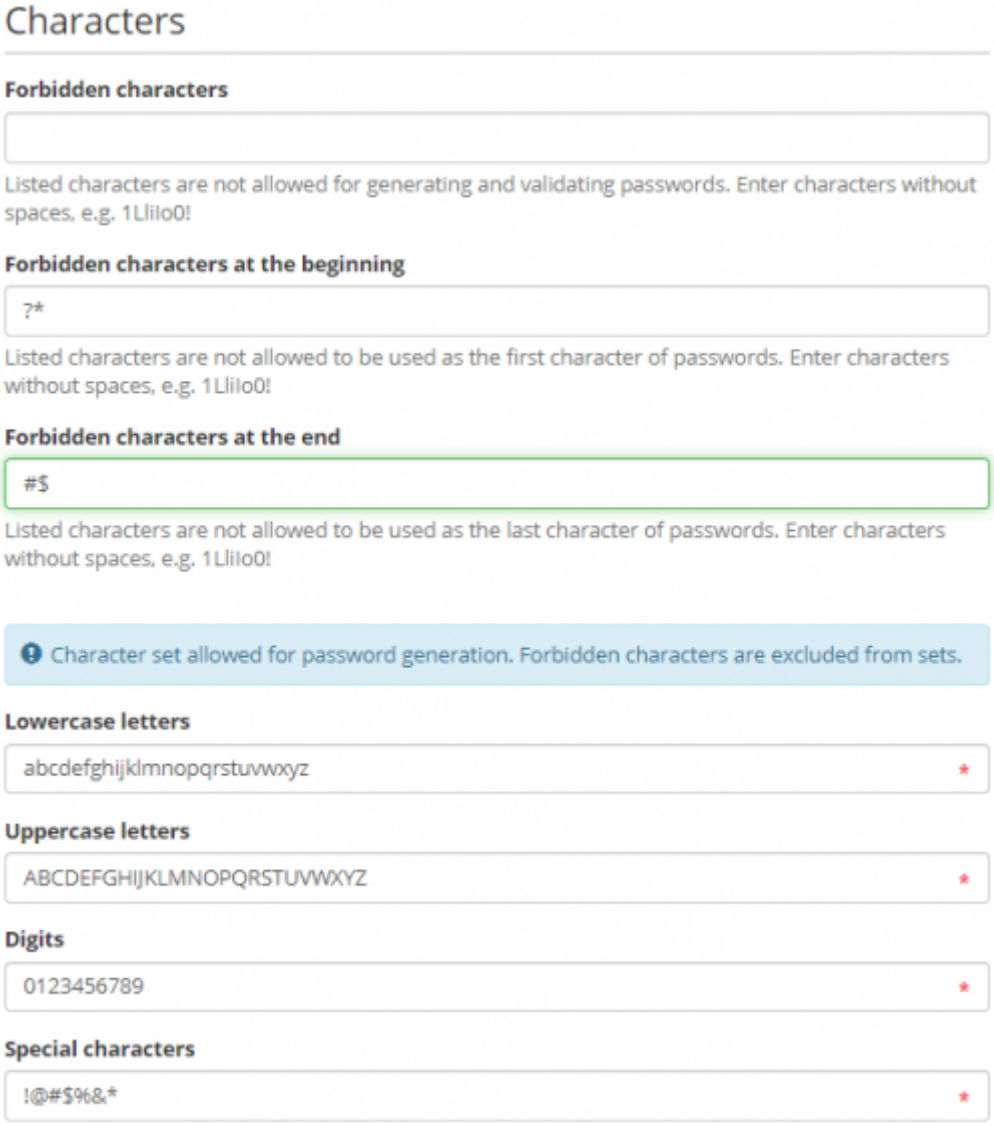
In case of non-compliance with one or more policies, the validation message will be displayed, which contains the information for which policies the password turned out to be invalid and why.



The display of the validation message is ensured by the component ValidationMessage. The component displays only the validation messages related to validation processing. The detailed procedure of performance of the individual messages can be found in DefaultIdmPasswordPolicyService the validate method and the variable Map<String, Object> errors.

Password policies, Signs

As for password generation every policy enables to set up character bases from which the password will be generated. For both types of password policies (generation/validation), it is also possible to define forbidden characters for the whole password and sets of characters, which password must not start or end with. When being validated/generated, the password is tested for presence of these forbidden characters. In case of password validation the user is informed about using such characters in the password. When a password is generated, forbidden characters are omitted during the process of generation.



The standard values of the character sets are not saved to the policies but they are gathered from the class PasswordGenerator (static variables).

Password policies, advanced password control

User attributes not allowed in password

If **enhanced control** of the validation password policy is enabled (see picture below), user can specify **User attributes not allowed in password** to which password must not resemble. Otherwise the password is not possible to be used. Motivation for this feature is to be able to set more strict rules for password policy, which are sometimes enforced by target systems (e.g. MS Active Directory). The manner of the similarity assessment is inspired by password complexity requirements by Microsoft MS password requirements. It suggests to split attributes into individual substrings by delimiters, if present, and then check the password for substring presence. As delimiters are considered following characters inside quotation marks `", . - – _ £"` and `white space characters` such as spaces, tabulators etc. If some substrings are shorter then 3 characters, they are omitted from validation. Validation is case insensitive and also insensitive to accents/diacritic.

IdM supports check of following attributes:

* **Email** * **Username** * **First name** * **Last name** * **Personal number** * **Titles before name** * **Titles after name**

Not all attributes are treated the same way. They can be divided into 3 groups. according to the approach to the assessment of similarity.

1) Email is tested whether it is contained in the password as a whole. Example: for email: j.doe@provider.com passphrases: XYZj.doe@provider.com, j.doe@provider.comXXX etc. are assessed as invalid. On the other hand passwords: jdoe, doe@provider etc. are valid.

2) Titles after/before name have, beside some exceptions, similar form containing several characters ended by period sign (e.g. Prof. MUDr.). When titles are processed, period sign is removed and created string is split by delimiters. This approach prevents from falling apart of titles containing delimiter somewhere in the middle (e.g. Ph.D.).

3) User name, First/Last name and Personal number are treated according to the above mentioned general procedure. E.g. person with name "Erin M. Hagens" is first split into "Erin" "M" "Hagens" substrings. Then "Erin", "Hagens" are checked for presence in the password. "M" is not considered because it is shorter than 3 characters. Passwords such as Hagens1234, ErinIsGreat etc. are invalid.

Mandatory rules

The main part of the advanced check of the password is the option to set up which of the rules are mandatory, and which are not. The control mechanism is as follows:

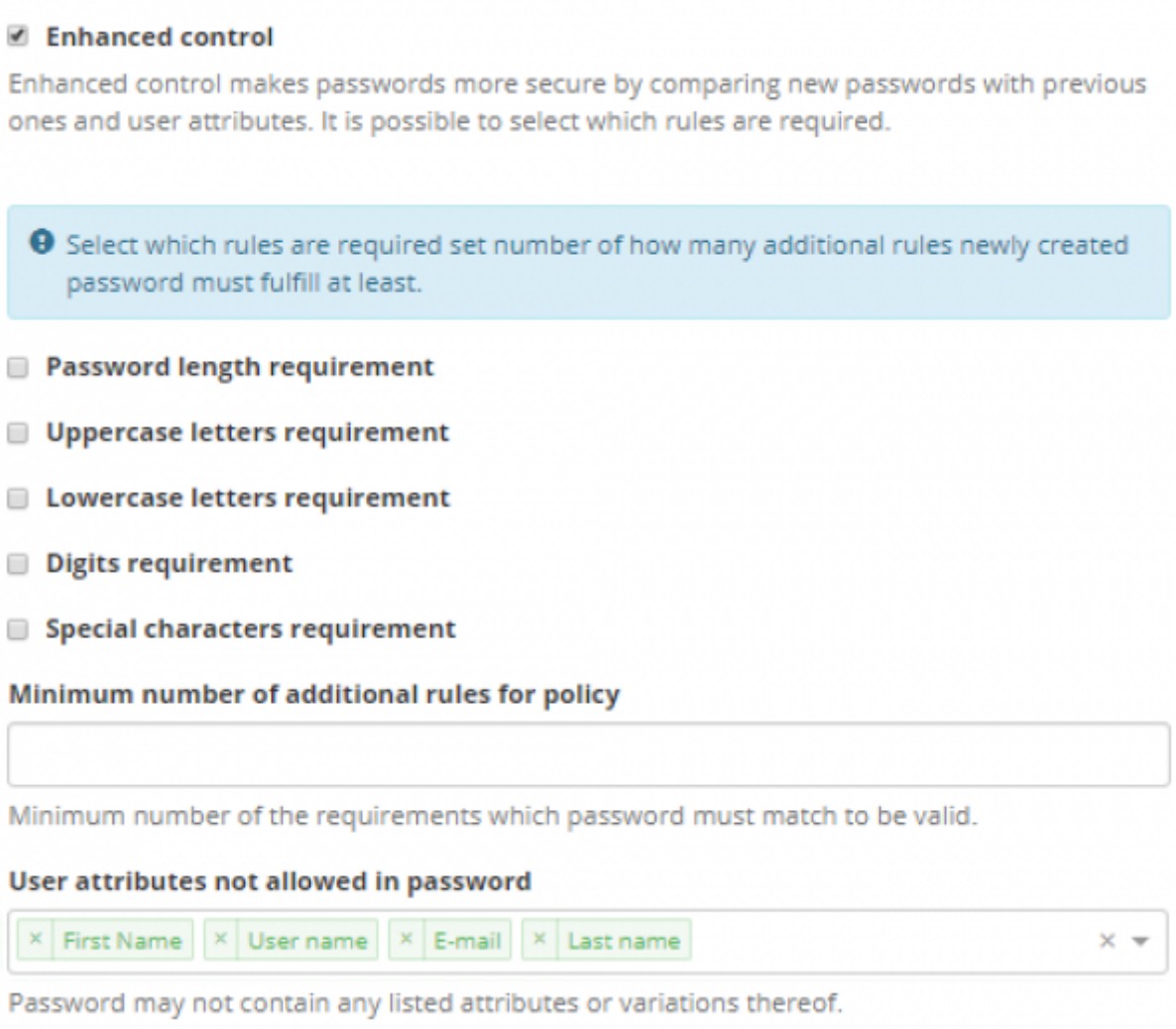
To each of the rules (see the rules list below), you may assign the feature saying whether the rule is mandatory or not. If the rule is marked as mandatory, it must be always satisfied. In case the rule is not mandatory, the number of satisfied optional rules must be higher or equal to the minimum feature of the rules to comply with the policy. For a better understanding, you can read the description of the following UseCase, including the settings.

UseCase

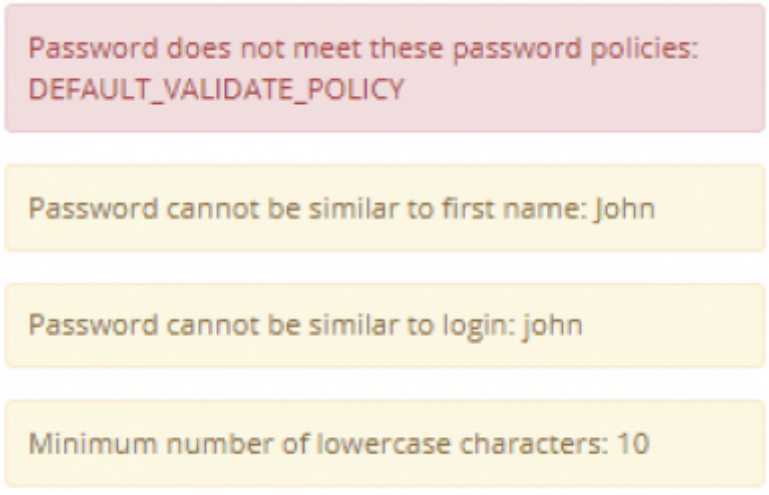
The administrator of the CzechIdM system has set up that the passwords must always include at least one number, the number of password characters amounting exactly to 8 characters. Then it must contain one special character, or two upper-case characters - it doesn´t matter which one of the two rules will be met.

Basic policy settings:

To satisfy the requirement of the password having at least one special character, OR two upper-case characters, an advanced check with the following settings will be used:



The validation messages with an unmet advanced check of the password looks as follows:



List of rules

- minimum and maximum number of the password,
- minimum number of upper-case characters,
- minimum number of lower-case characters,
- minimum number of numbers,

- minimum number of special characters,
- maximum password age (in days)
- minimum number of days for password validity, before password can be changed again
- number of old passwords checked for match (number of password changes they has to happen to allow password reuse)
- login blocking time (seconds)
- maximum number of unsuccessful login attempts (if exceeded next login attempt is blocked for set time)

Passwords General

In the CzechIdM system, there are no passwords saved in plain text or a decipherable form. The object IdmPassword is assigned to the identities, the link exists from IdmPassword to IdmIdentity, but not the other way around. Not every identity necessarily needs to have a password.

The object IdmPassword has the following attributes:

- **password (string)** - a password in an indecipherable form saved by the BCrypt algorithm, including salt,
- **identity (IdmIdentity)** - the identity to which the object belongs
- **validTill (datetime)** - date until which the password is valid, the date has been taken over from the policy
- **validFrom (datetime)** - date from which the password is valid - used mainly for validation with minimum number of days, before password can be changed again
- **mustChange (boolean)** - in case this feature has been set up, after logging in, the user is redirected to the password change Until the password is not changed, access to the CzechIdM system is not allowed.

Password and block/unblock login

Default password policy allow block login for user for time period after several unsuccessful attempts. Both options is defined by default validate policy. After block is to password set **blockLoginDate** with current date plus block interval. When is user in this interval isn't possible login and change password from public page.

Administrator or user that has permission for change users passsword without required old password can reset **blockLoginDate** and **unsuccessfulAttempts** with force password change or password reset (password reset is possible only when you have password reset module).

Password and minimum number of days validation

Minimum number of days validation says, when password can be changed again. It prevents to workaround validation of reusing old passwords - e.g. when user want to have the same "favorite" password, when **three old password** are checked and minimum number of days is set to **one**, then user will be able to use his "favorite" password after three days and he need to change password every day to something else in the meantime. This validation is **not evaluated**:

- When user is enforced to change password after next login, e.g. after initial password was created by the system,
- when password is changed by someone else, e.g. by his supervisor or by helpdesk,
- when password is changed by administrator,
- when password was changed by someone else, then next change attempt by user is not checked too.

This validation is **evaluated**:

- When user changes his own password (except user is administrator),
- when password is changed by password filter (except password filter configured token owner is administrator).

Password length

After using Bcrypt, the question is what the maximum number of password characters is. When using passphrase, you get even more than 255 characters.

Password with 300 characters: the password creation and the log-in were successful. The generated hash always had the same number of characters, including salt; e.g. ($2a$12$ZWIqBjjZrONY2Nw5V8VyOesVw2/naVq.N0oR/CAxabozI5KONxScK), so there **is no** dependence on the number of characters of the password and the hash.

If password generator is used and no maximum password length is specified (no minimum and maximum length is defined), then password with **12 chars** will be generated (constant).

Public password change

If user want change password by public change password (REST endpoint: **/public/identities/{backendId}/password-change**) is **necessary** know old password. In this moment **is not required** that identity has password in our system (identity is get by synchronization and object idm_password for this identity don't exist). After public password change is old password check by all authentication chain see AuthenticationManager and **password will be changed on all system including IdM.**

Public password change is also validate by password policies via PasswordPolicyService (validation by policies including default policy and system policies).

Future development

- dictionary of weak words,
- nice generation of random passwords.