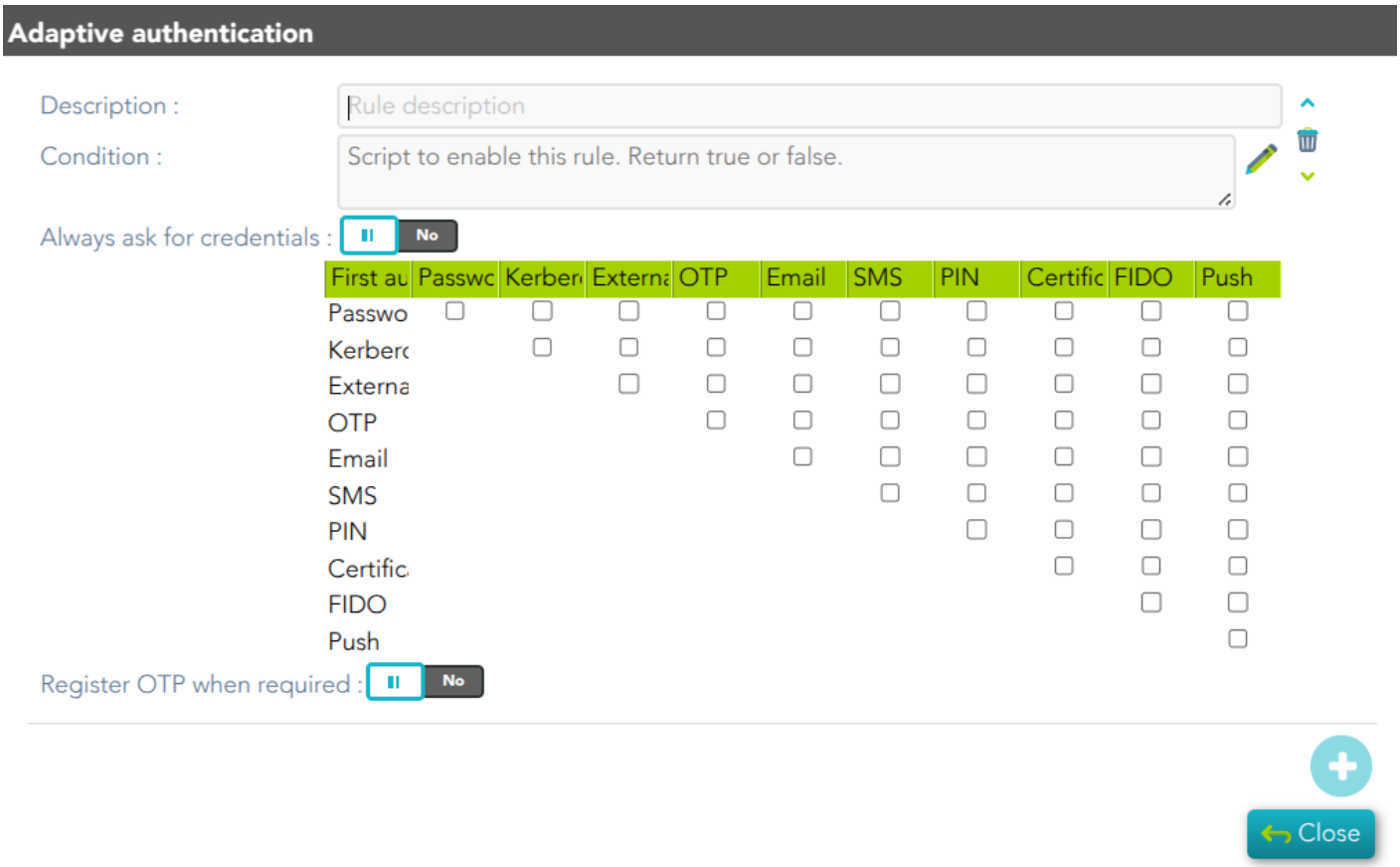
Condition for Adaptive authentication

Introduction

Adaptive authentication is designed to improve the security of online accounts by adding an additional layer of protection against unauthorized access.

When the authentication is being defined, Soffid allows you to add some **adaptive authentications** in addition to the Authentication methods. Those adaptive authentications will be evaluated, and when the result of the condition will be true, the rule will be enabled.

Screen overview



Standard attributes

Description

Description to identify the rule

Condition

Allows you to write a script validation, with the result true or false. To develop the script you can use some vars defined to that:

There are some available vars to create the condition:

- **currentAuthenticationMethod**: return tue current authentication method. Check the Authentication methods available.
- **dayOfWeek**: number of the day of the week (1-7 where 1 is Sunday and 7 is Saturday).
- **daysSinceLastLogon**: number of days (integer).
- **daysSinceLastLogonByMethod**: return a hashmap with the logon method and the number of days (integer). Check the Authentication methods available.
- **daysSinceLastLogonFromSameHost**: number of days (integer).
- **deviceCertificate**: is or is not a device certificate (boolean).

- **displacement**: distance in kilometers (Double).
- **displacementSpeed**: distance in kilometers since last login attempt (Double).
- **failuresForSameIp**: integer value to determine the number of failures.
- **failuresForSameUser**: integer value to determine the number of failures.
- **failuresRatio**: value between 0 and 1.
- **geoInformation**: it is an object that contains:
  - ip (string)
  - date (Date)
  - country (string)
  - countryDivision1 (string)
  - countryDivision2 (string)
  - city (string)
  - latitude (Double)
  - longitude (Double)
  - accuracy (Double): accuracy in kilometers.
  - domain (string)
  - isp (string)
  - userType (string)
  - anonymous (Double): what is the probability that they use an anonymizer?:
    - > 0.5 => anonymizer
    - < 0.5 => end user
- **hasCertificate**: the user has or does not have a certificate (boolean).
- **hasFidoToken**: the user has or does not have a Fido token (boolean).
- **hasOtp**: the user has or does not have an OTP (boolean).
- **hasOtpHotp**: the user has or does not have an OTP based on events (boolean).
- **hasOtpMail**: the user has or does not have an email OTP(boolean).
- **hasOtpPin**: the user has or does not have a Pin OTP (boolean).
- **hasOtpSms**: the user has or does not have an SMS OTP (boolean).
- **hasOtpTotp**: the user has or does not have an OTP based on time (boolean).
- **hasPushToken**: the user has or does not have a Push Token (boolean).
- **hasToken**: the user has or does not have a Token (boolean).
- **hour**: integer value between 0 and 23.
- **identityProvider**: string value with the name of the identity provider.
- **ipAddress**: string with the IP address.
- **isEsso**: return true if the login is done through ESSO.
- **minute**: integer value between 0 and 59.
- **newDevice**: boolean value (true or false). It validates if the connection is from a new device.
- **remoteHost**: it is a host object.
- **sameCountry**: boolean value (true or false). It validates if the connection is from the same country as the last user connection.
- **serviceProvider**: string value with the name of the service provider.
- **sourceCountry**: sting value to identify the country. It uses the first two ISO characters.
- **user**: it is a user object.

Authentication methods available:

- **P**: Password
- **K**: Kerberos
- **E**: Broker
- **O**: OTP
- **M**: Email
- **S**: SMS
- **I**: PIN
- **C**: Certificate
- **F**: Finger print
- **Z**: Push

Matrix

To define the authentication methods that will be required to successfully authenticate the user. Each row indicates the first authentication method, and each column indicates the second factor to use.

Always ask for credentials

Use this option to force the request for credentials when the user meets the defined condition.

Register OTP when required

Use this option to allow the user to register the OTP the first time he/she attempts to log in when the user meets the defined condition.

Actions

Apply changes	Allows you to save the data of a new adaptive authentication or to update the data of the previously created adaptive authentucation.
Add	Allows you to add a new adaptive authentication. When you click the add button (+) Soffid will display new fields to fill in. For each adaptive authentication you must fulfill the description, the condition to evaluate and the matrix which will be enable when the condition will be true. Then you must click on the "Apply changes" button to save the data.
Delete	Allows you to remove one by one the adaptive authentication defined. You must click on the trash icon the account of the proper rule. Then you must click the "Apply changes" button to save the data.

Up	Allows you to reorder (up) the defined adaptive authentication.
Down	Allows you to reorder (down) the defined adaptive authentication.

Examples

Rule 1

```
failuresRatio > 0.8
```

Rule 2

```
(daysSinceLastLogon > 10) && (ipAddress.startsWith("192.168.")
```

Rule 3

```
((dayOfWeek == 7) || (dayOfWeek == 1)) &&  (user!=null && "<USER_NAME>".equals(user.userName))
```

Rule 4

```
"ES".equals(sourceCountry) || ipAddress.startsWith("192.168.")
```

Rule 5

```
isEsso
```

Rule 6

```
if (daysSinceLastLogonByMethod["PO"] == null || daysSinceLastLogonByMethod["PO"] > 30)
  return true;
```

Rule 7

```
"YOUR_SERVICE_PROVIDER".equals(serviceProvider)
```

Rule 8

```
"P".equals(currentAuthenticationMethod)
```

Rule 9

```
if (user!=null)
  log.info("**** User  " + user.userName);
user!=null && "Ada".equals(user.userName);
```

Revision #33
Created 16 September 2021 14:13:18
Updated 7 April 2025 13:59:12