Authorship: https://www.linkedin.com/in/sedoykin

Teamcenter & OneLogin MFA

Briefly: Here, we will try to integrate two-factor authentication based on OneLogin with Teamcenter.

Contents

Authorship: https://www.linkedin.com/in/sedoykin
Disclaimer:
No guarantees or responsibilities are provided. You perform all actions at your own risk!

Introduction

Here we will try to implement two-factor authentication based on OneLogin for Teamcenter. Guys, be prepared that this document might not work in your specific conditions, as there are too many uncertainties. However, I hope that it will provide the right directions for your research.

Authorship: https://www.linkedin.com/in/sedoykin
Disclaimer:
No guarantees or responsibilities are provided. You perform all actions at your own risk!

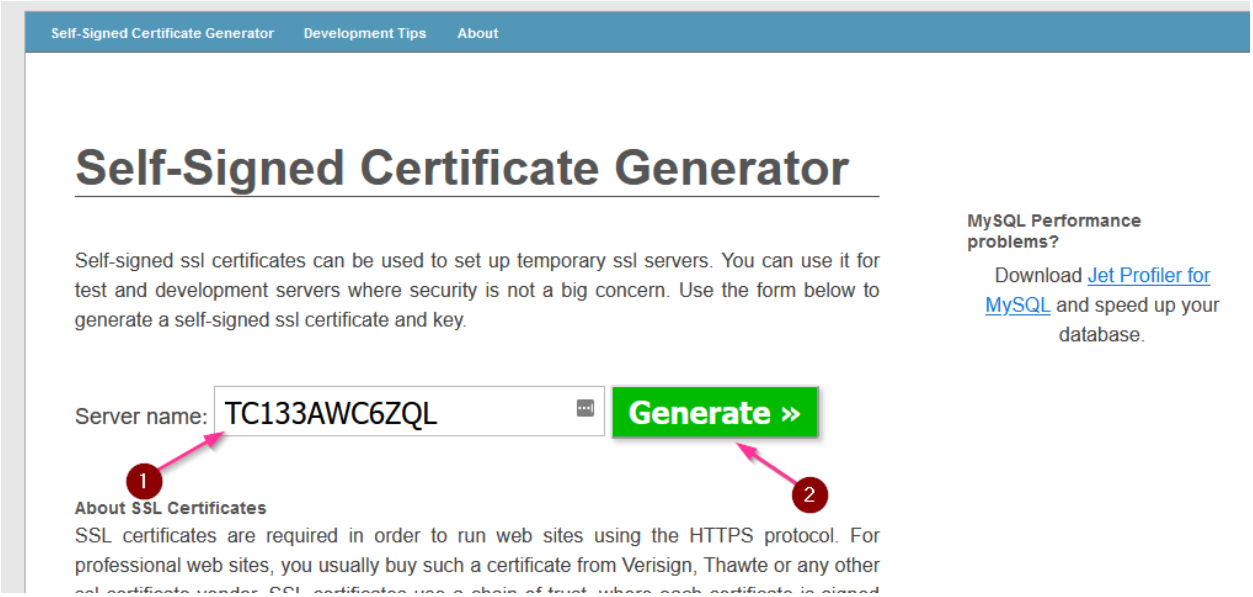
Prerequisites

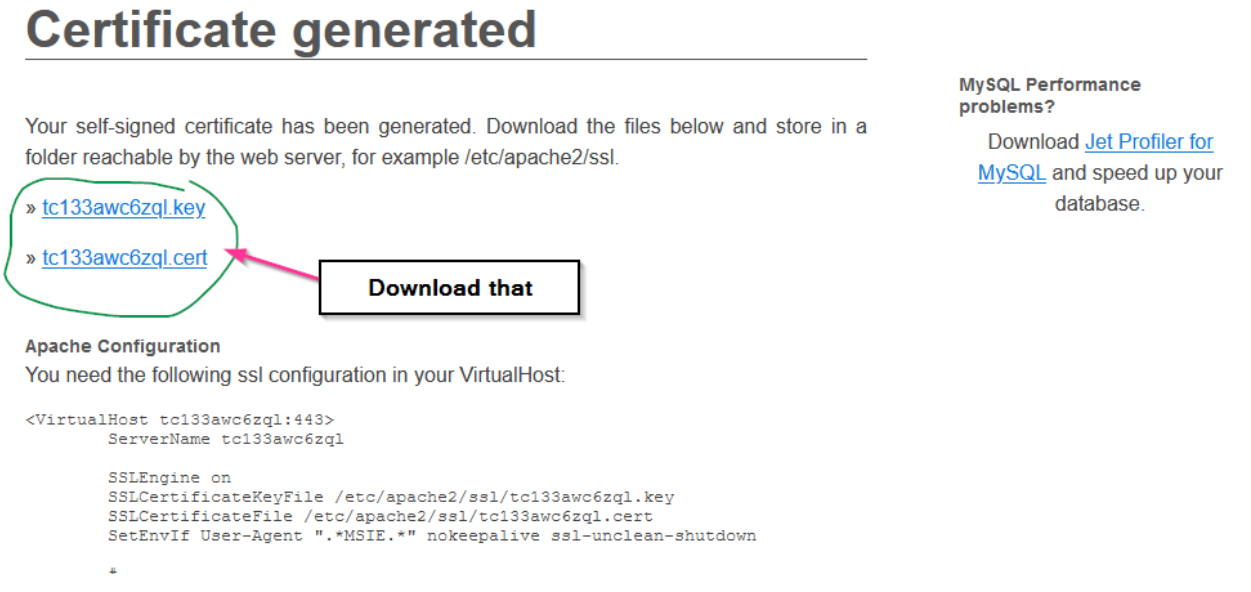
- Teamcenter 13.3 (TcSS 13.3) + AWC 6
- Tomcat 9.0+
- Single server for all of TC components
- Hostname: *TC133AWC6ZQL
- All software installed in C:\Siemens
- OneLogin (onelogin.com) dev account:
    - *sdistcrus-dev.onelogin.com
- Will be used OpenID (OIDC) as technology for provide authorize data
- We will not use LDAP catalog on-premise!

Authorship: https://www.linkedin.com/in/sedoykin
Disclaimer:
No guarantees or responsibilities are provided. You perform all actions at your own risk!

Steps on the Teamcenter Infrastructure Side

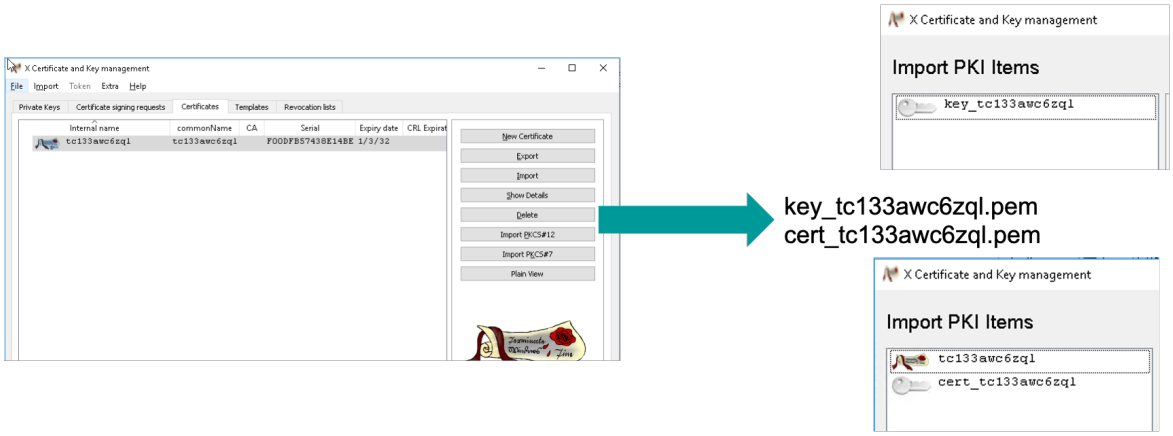
1. **Generating self-signed certificates** - **https://www.selfsignedcertificate.com/**





Authorship: https://www.linkedin.com/in/sedoykin
Disclaimer:
No guarantees or responsibilities are provided. You perform all actions at your own risk!

2. **Convert key and cert to PEM format**
    a. Download https://hohnstaedt.de/xca
    b. Via XCA application, convert cert and key to PEM files



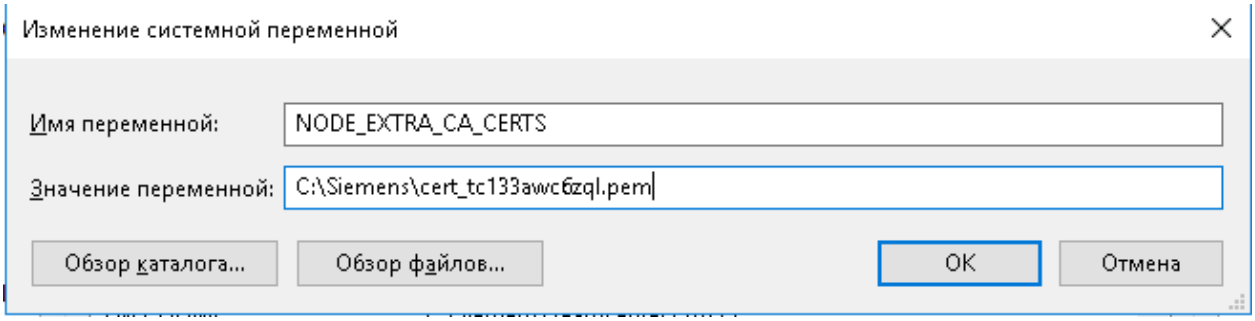
3. **Move the:**

    key_tc133awc6zql.pem

    cert_tc133awc6zql.pem

    files to C:\Siemens and c:\Siemens\Tomcat9SSO\conf\

Create the next system variable (it needs for AWC)



4. **Configuring AWC for access via HTTPS**

   Open **config.json** (c:\Siemens\Teamcenter13\microservices\gateway-1.6.0**config.json**)

```
},
"sso": {
    "tcSSOAppID": "AWCSSO",
    "tcSSOURL": "https://TC133AWC6ZQL:8001/LoginService",
    "tcSSORedirectMethod": "GET",
    "tcSSOLogoutURL": "",
    "proxyServerUrl": "",
    "queryParams": ""
},
"fms": {
    "bootstrapFSCURLs": [  ],

},
"csurf": {
    "cookie": true
},
"csurfExclusions": [
    "/AWSSOLogin",
    "/graphql",
    "/LoginService/sso_login_callback",
    "siemensdisrus.okta.com",
    "/siemensdisrus.okta.com"
],
"userAgentWhiteList": [
    "Apache-HttpClient",
```

5. **Configuring Tomcat for deploying TcSS components**

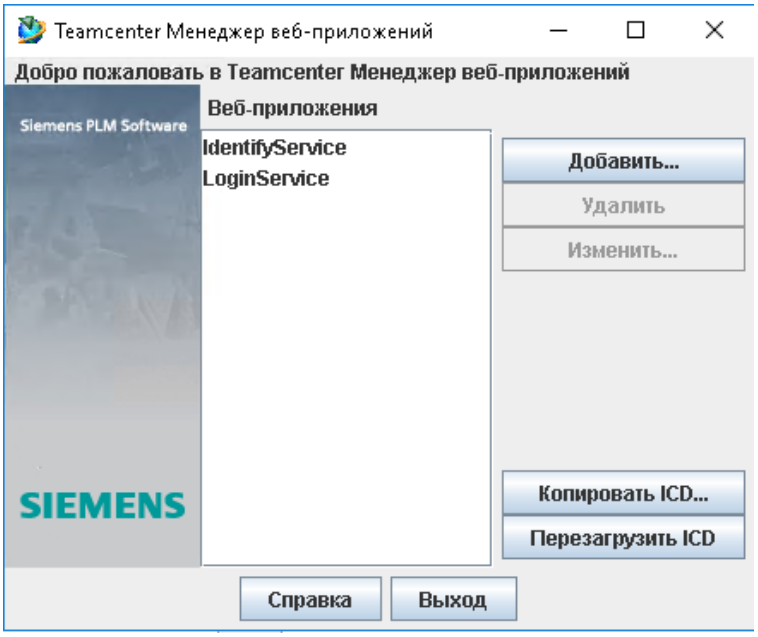
   Open server.xml (c:\Siemens\Tomcat9SSO\conf**server.xml**)

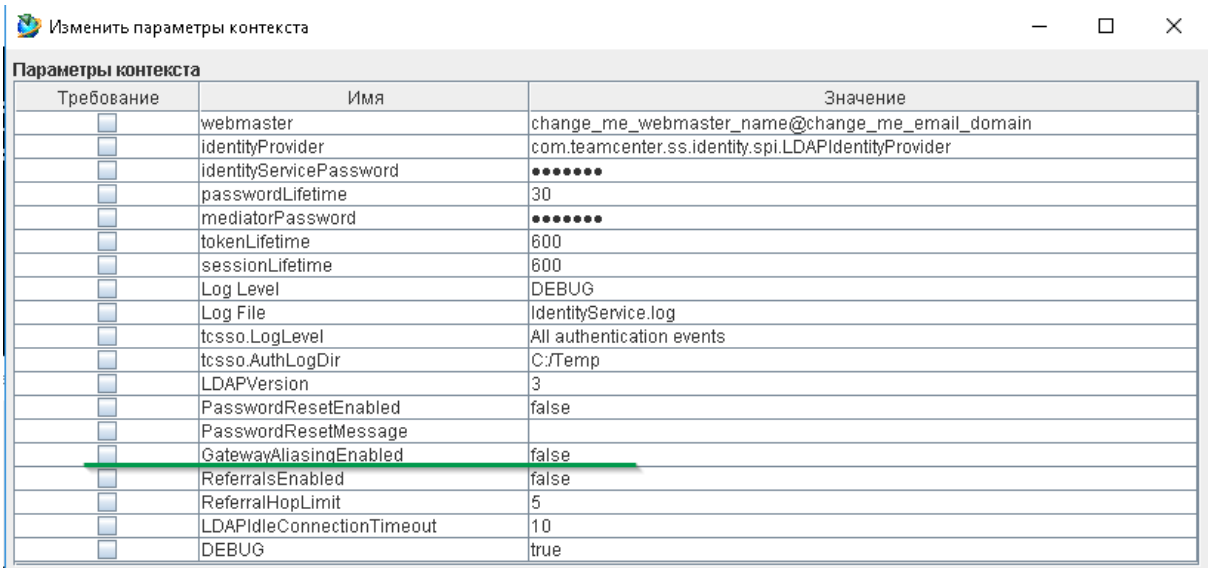
```
<Connector port="8002" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
```

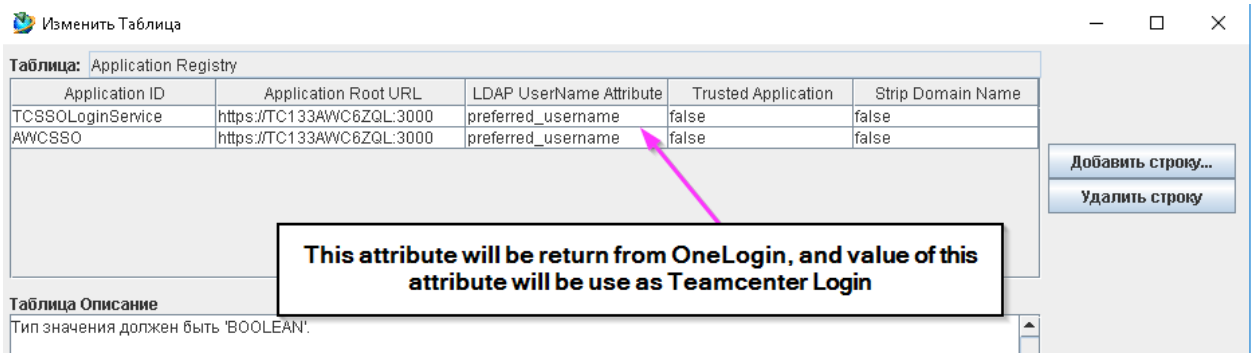
```
<Connector
      protocol="org.apache.coyote.http11.Http11NioProtocol"
      port="8001" maxThreads="200"
      scheme="https" secure="true" SSLEnabled="true"
      SSLCertificateFile="conf/cert_tc133awc6zql.pem"
      SSLCertificateKeyFile="conf/key_tc133awc6zql.pem"
      SSLVerifyClient="optional" SSLProtocol="TLSv1+TLSv1.1+TLSv1.2" useIPVHosts="true" address="0.0.0.0"/>
```

6. **Creating Login and Identify Services**

   Ohhh, for details go to: https://docs.sw.siemens.com/en-US/product/282219420/doc/PL20210421143201885.tss00001/html/xid373993

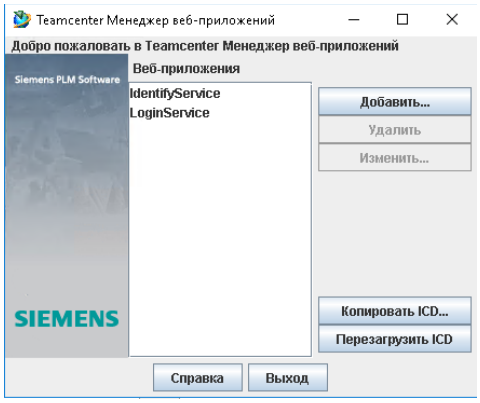


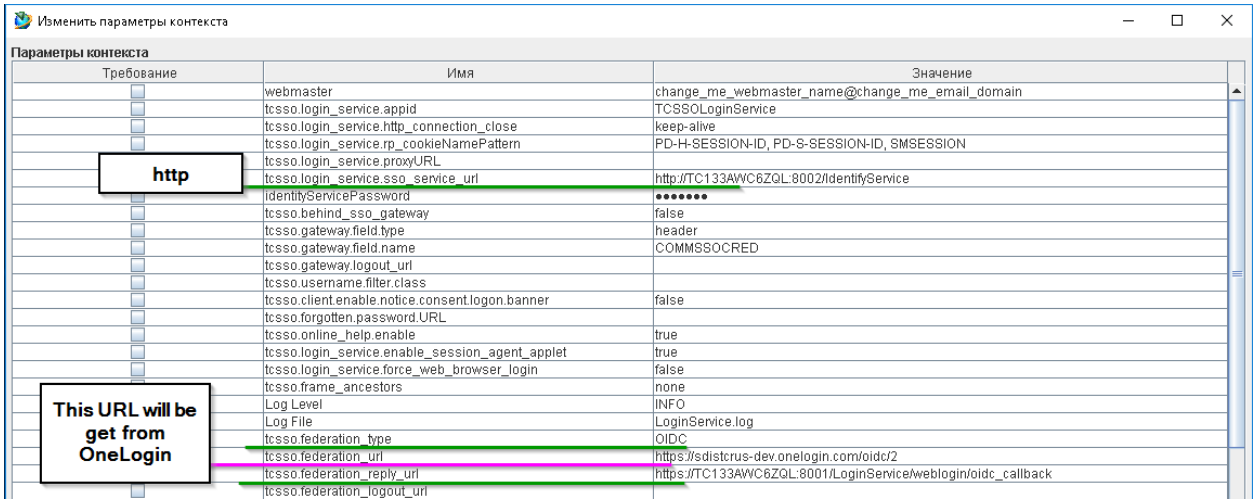


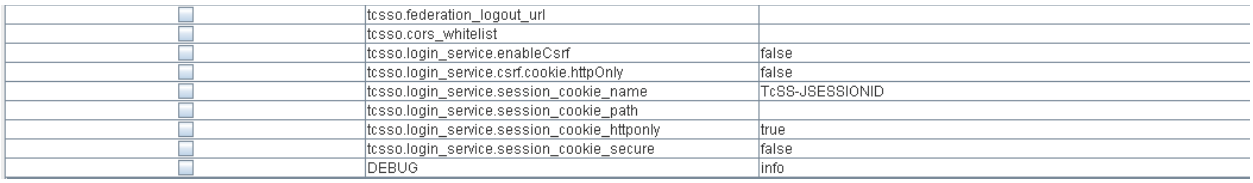


Authorship: https://www.linkedin.com/in/sedoykin
Disclaimer:
No guarantees or responsibilities are provided. You perform all actions at your own risk!

7. Creating Login and Identify Services







	tcsso.federation_logout_url	
	tcsso.cors_whitelist	
	tcsso.login_service.enableCsrf	false
	tcsso.login_service.csrf.cookie.httpOnly	false
	tcsso.login_service.session_cookie_name	TcSS-JSESSIONID
	tcsso.login_service.session_cookie_path	
	tcsso.login_service.session_cookie_httponly	true
	tcsso.login_service.session_cookie_secure	false
	DEBUG	info

8. **Configuring federation.properties (1)**
   Open **federation.properties**(c:\Siemens\SSOSrv\LoginService\webapp_root\WEB-INF\classes**federation.properties**)

```
# OIDC properties
#

# This value is obtained from the OpenID Provider when TcSS is registered
# as an OIDC Client
tcsso.oidc.client_id=b3[redacted]97

# Only required if tcsso.oidc.client_auth_method=client_secret_basic
# This value is obtained from the OpenID Provider when TcSS is registered
# as an OIDC Client
# WARNING: Sensitive value
tcsso.oidc.client_secret=6bc[redacted]ff487

# The value of the authorization_endpoint from the OpenID Provider
tcsso.oidc.auth_endpoint=https://sdistcrus-dev.onelogin.com/oidc/2/auth
# The value of the token_endpoint from the OpenID Provider
tcsso.oidc.token_endpoint=https://sdistcrus-dev.onelogin.com/oidc/2/token
# The value of the jwks_uri from the OpenID Provider
tcsso.oidc.jwks_endpoint=https://sdistcrus-dev.onelogin.com/oidc/2/certs

# Specifies the name of the OIDC ID Token claim that will be used as the
# Teamcenter SSO User ID.
# Note: The value of the tcsso.oidc.scope above will affect which claims are
# returned in the ID Token. Make sure the correct scope(s) are configured to
# include this desired claim
tcsso.oidc.userid_claim=preferred_username

# The value of the scope that will be included in the OIDC Authentication Request
tcsso.oidc.scope=openid profile

# Specifies the algorithm that will be used to verify the signature of the OIDC ID Tc
tcsso.oidc.token_sig_alg=RS256

# Specifies the value of the OIDC display parameter sent in the Authentication Request - page default
tcsso.oidc.display=page
```

These data will be get from OneLogin

This attribute will be return from OneLogin, and it uses in IdentifyService (Application Registry)

openid - is mandatory scope, profile - scope has preferred_username attribute

9. **Configuring federation.properties (2)**

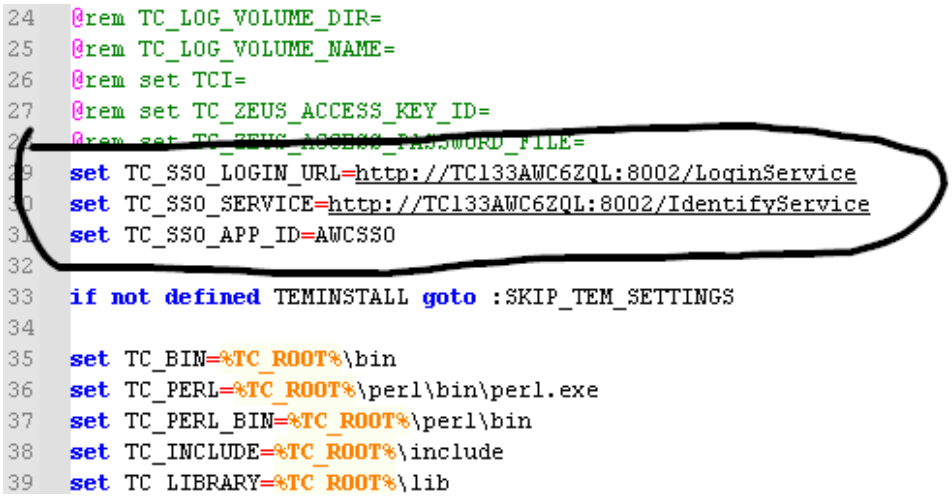
```
# OIDC properties - Advanced Configurations
#

# The Authentication method that will be used when sending a Token Request to the
# OpenID Provider
# Currently supported methods:
# - client_secret_basic
# - private_key_jwt (This method requires that the following tcsso.oidc.signing..
#   properties be set and that TcSS is registered at the OpenID Provider to use
#   private_key_jwt for Token Endpoint authentication)
# Note that the OpenID Provider may not support all the above methods
tcsso.oidc.client_auth_method=client_secret_basic

# The following four tcsso.oidc.signing.. properties are only required if the
# private_key_jwt Token Endpoint authentication method is selected above.
#
# A JKS formatted keystore containing the public key that should be passed to the
# OpenID Provider to validate the signed JWT sent with a Token Request as well as
# the corresponding private key that we will use to sign the JWT.
tcsso.oidc.signing_jks_file=</secure/path/oidc_signing_key.jks>
# WARNING: Sensitive value
tcsso.oidc.signing_jks_file_pwd=<password>
# The name of the private key in the keystore used to sign the JWT
tcsso.oidc.signing_private_key_name=<key name>
# WARNING: Sensitive value
tcsso.oidc.signing_private_key_pwd=<password>

# If private_key_jwt Client Authentication is selected, specifies the signing
# algorithm used to sign the jwt sent to the OpenID Provider
tcsso.oidc.jwt.sig_alg=RS256
# If private_key_jwt Client Authentication is selected, specifies the expiration
# time (in seconds) of the jwt sent to the OpenID Provider
tcsso.oidc.jwt.expiration=60
```

10. **Configuring federation.properties (3)**

```
158
159  #
160  # The following four tcsso.oidc.encryption.. properties are only required if ID
161  # Token encryption is supported and enabled at the OpenID Provider.
162  #
163  # A JKS formatted keystore containing the public key that should be passed to the
164  # OpenID Provider to encrypt an OIDC ID Token, and the corresponding private key
165  # that we will use to decrypt the OIDC ID Token.
166  tcsso.oidc.encryption_jks_file=</secure/path/oidc_encrypt_key.jks>
167  # WARNING: Sensitive value
168  tcsso.oidc.encryption_jks_file_pwd=<password>
169  # The name of the private key in the keystore used to decrypt the OIDC ID Token
170  tcsso.oidc.encryption_private_key_name=<key name>
171  # WARNING: Sensitive value
172  tcsso.oidc.encryption_private_key_pwd=<password>
173
174  # If true, a jwks_uri will be exposed at the <LoginService>/certs endpoint.
175  # Public keys that are configured above will be converted to JWK format and
176  # returned from that endpoint.
177  tcsso.oidc.enable_jwks_uri=false
```

11. **Configuring Teamcenter core**

    Open **tc_profilevars.bat** (C:\Siemens\tcdata**tc_profilevars.bat**)



```
@rem TC_LOG_VOLUME_DIR=
@rem TC_LOG_VOLUME_NAME=
@rem set TCI=
@rem set TC_ZEUS_ACCESS_KEY_ID=
@rem set TC_ZEUS_ACCESS_KEY_PASSWORD_FILE=
set TC_SSO_LOGIN_URL=http://TC133AWC6ZQL:8002/LoginService
set TC_SSO_SERVICE=http://TC133AWC6ZQL:8002/IdentifyService
set TC_SSO_APP_ID=AWCSSO

if not defined TEMINSTALL goto :SKIP_TEM_SETTINGS

set TC_BIN=%TC_ROOT%\bin
set TC_PERL=%TC_ROOT%\perl\bin\perl.exe
set TC_PERL_BIN=%TC_ROOT%\perl\bin
set TC_INCLUDE=%TC_ROOT%\include
set TC_LIBRARY=%TC_ROOT%\lib
```

12. **Deploying TcSS services**

    Copy these files:

    c:\Siemens\SSOSrv\IdentifyService\deployment**IdentifyService.war**

    c:\Siemens\SSOSrv\LoginService\deployment**LoginService.war**

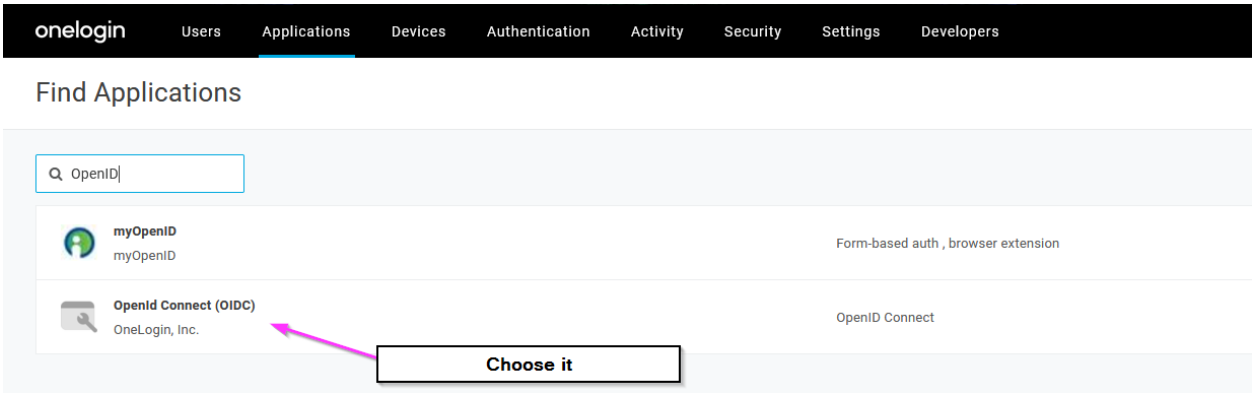
    to:

    c:\Siemens\Tomcat9SSO**webapps**

Authorship: https://www.linkedin.com/in/sedoykin
Disclaimer:
No guarantees or responsibilities are provided. You perform all actions at your own risk!

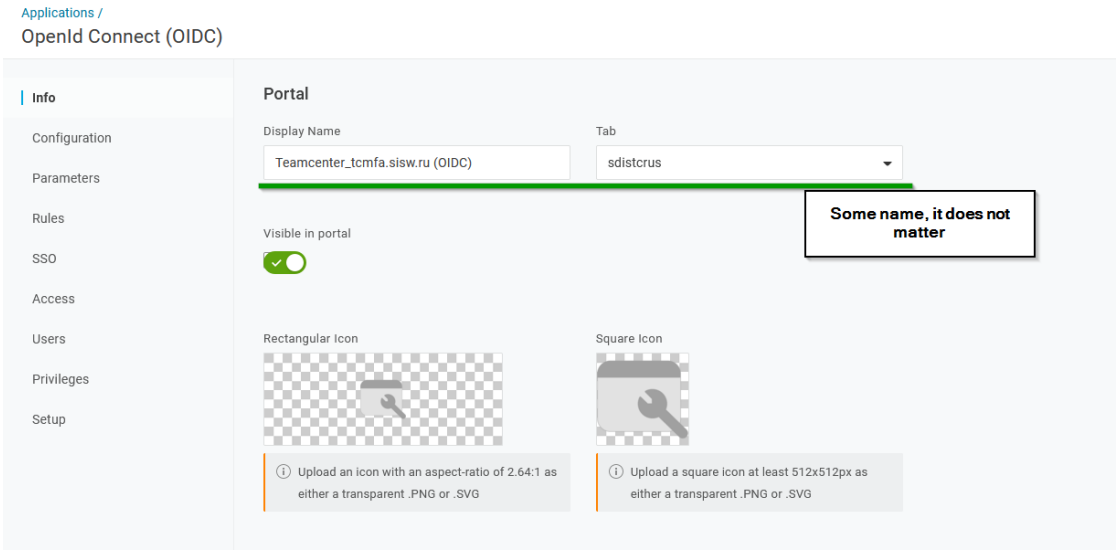
Steps on the OneLogin service side

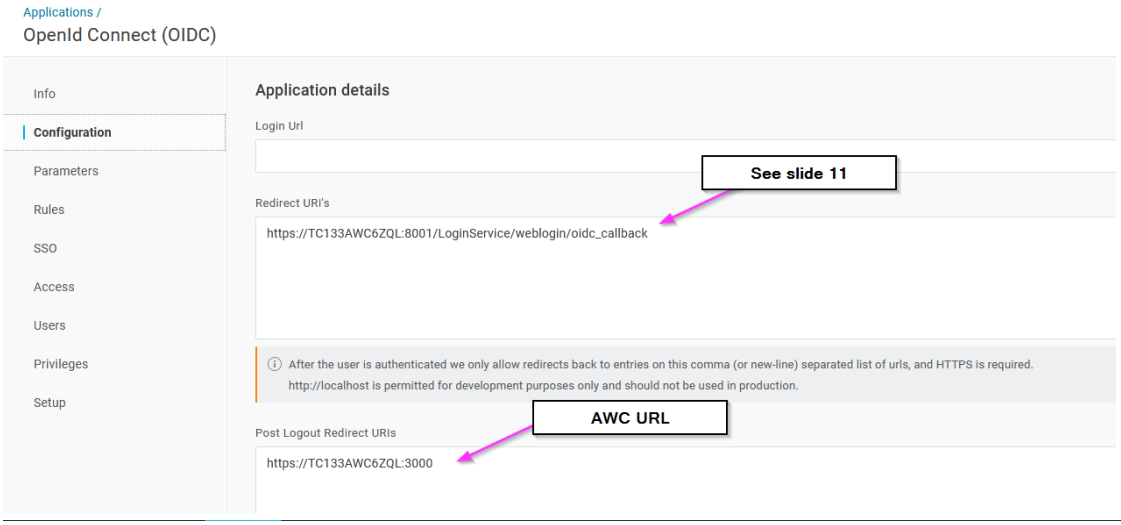
1. **You should have dev account, it is free for developing purpose** after get it you should create new OpenID integration application

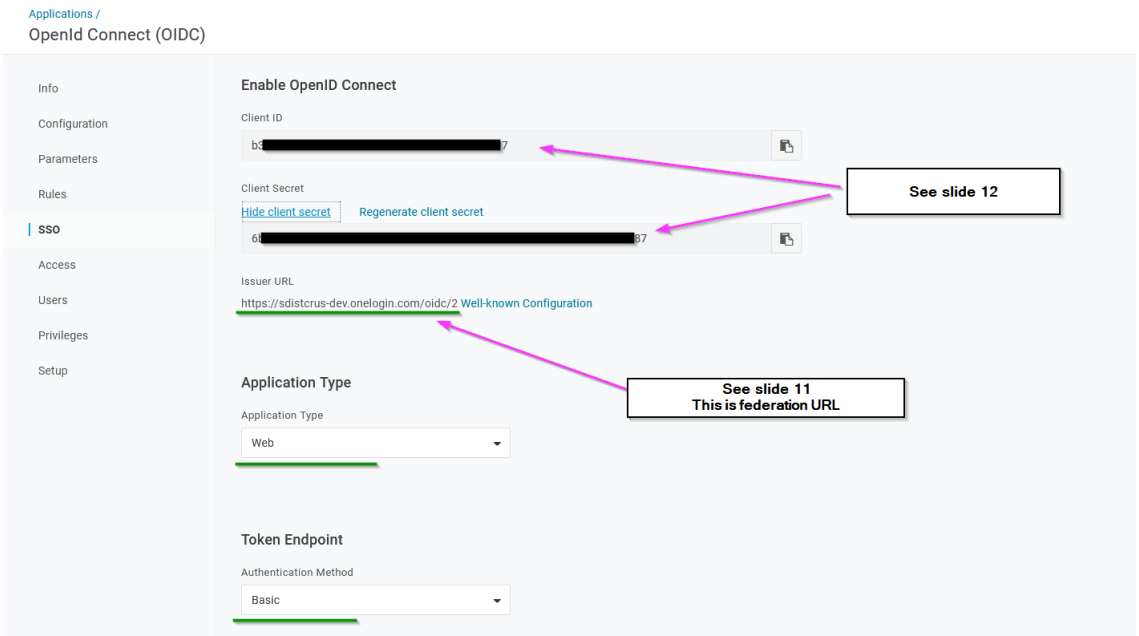


Authorship: https://www.linkedin.com/in/sedoykin
Disclaimer:
No guarantees or responsibilities are provided. You perform all actions at your own risk!

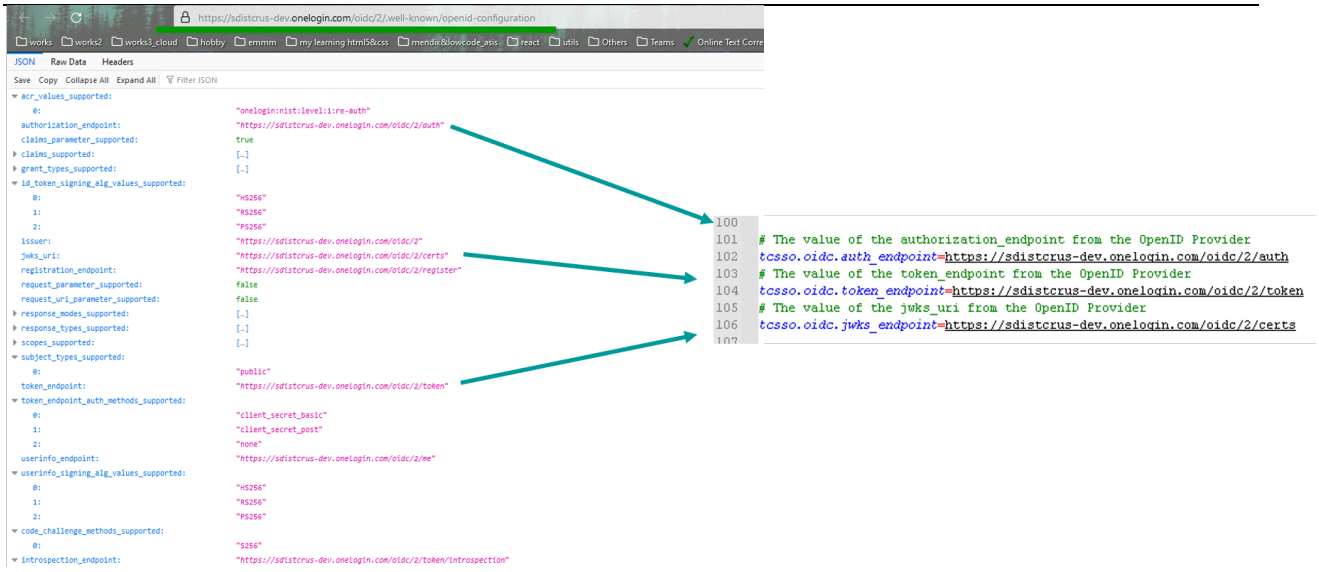
2. Configuring OpenId Connect (OIDC)





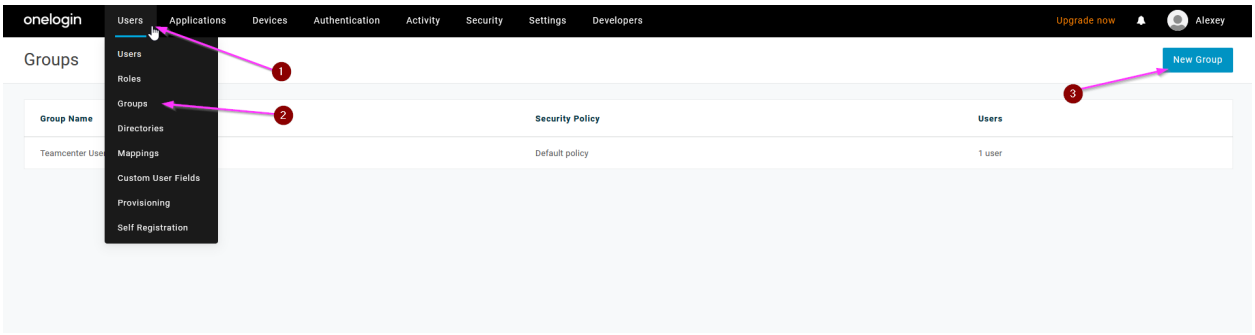


Authorship: https://www.linkedin.com/in/sedoykin
Disclaimer:
No guarantees or responsibilities are provided. You perform all actions at your own risk!



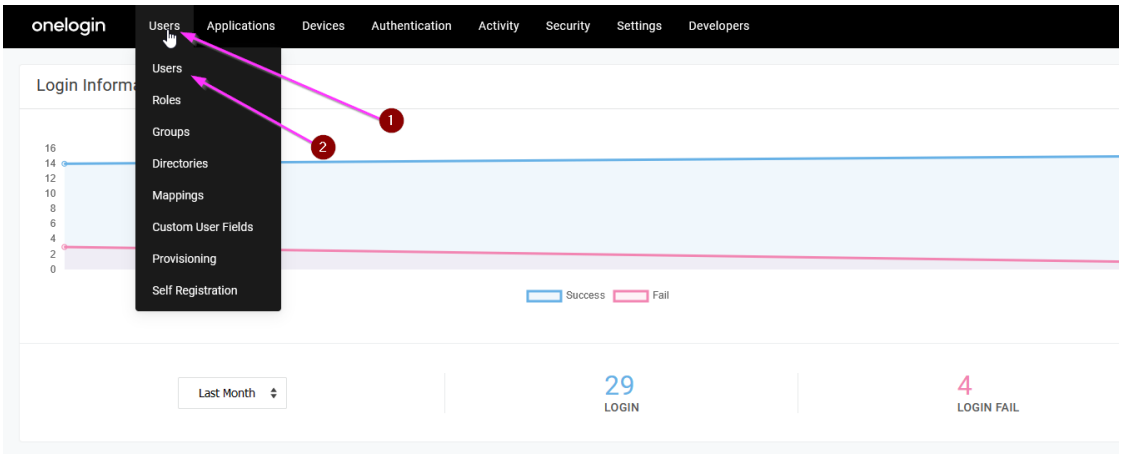
Authorship: https://www.linkedin.com/in/sedoykin
Disclaimer:
No guarantees or responsibilities are provided. You perform all actions at your own risk!

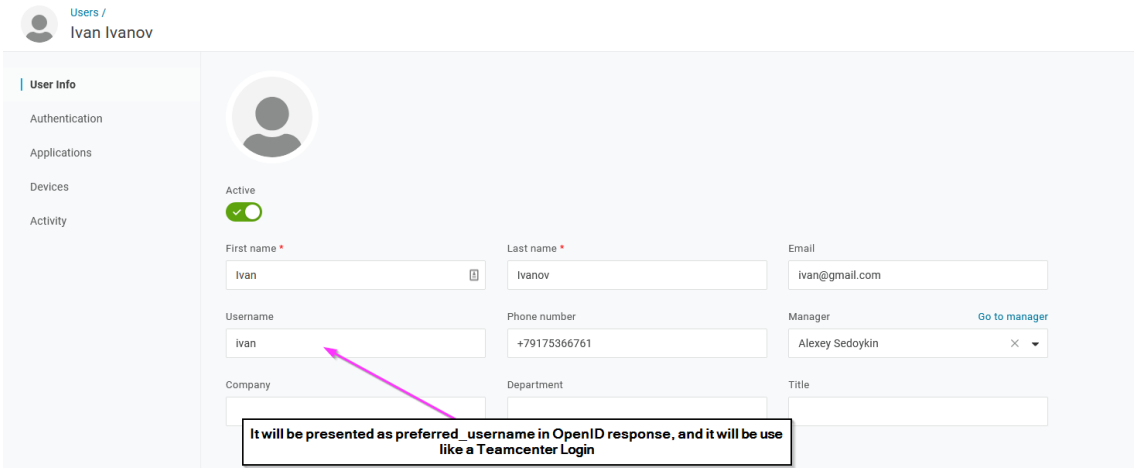
3. **Creating User Group in OneLogin account**

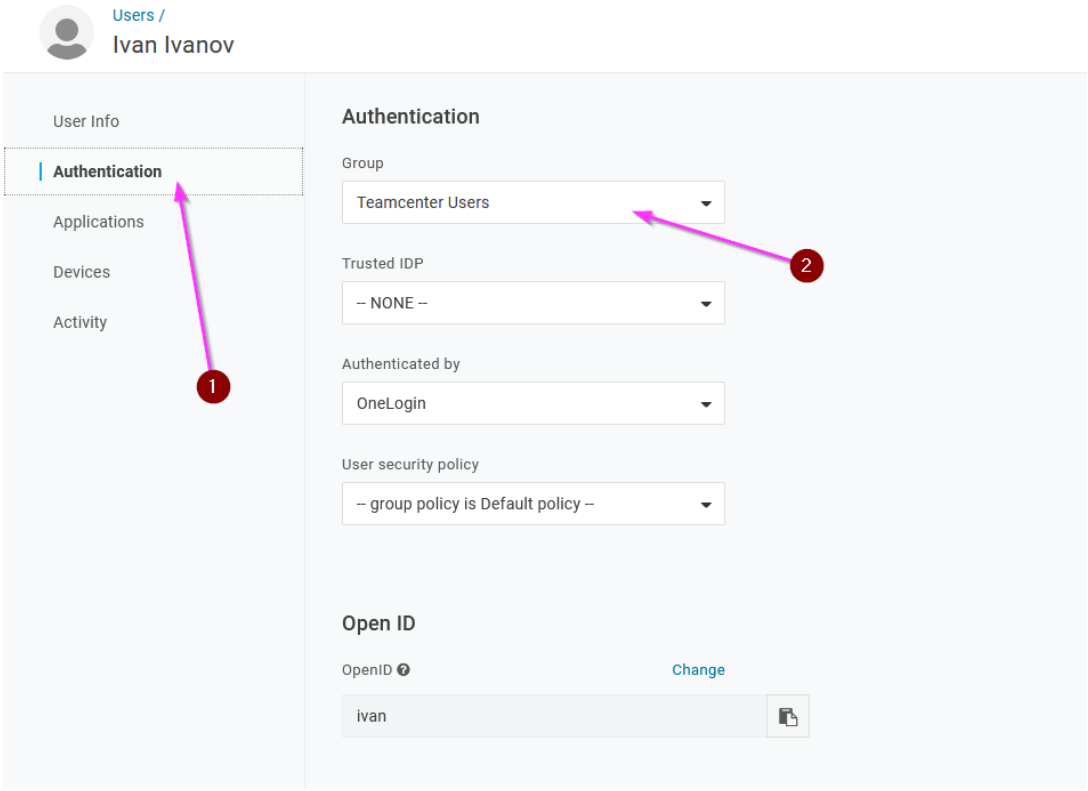


Authorship: https://www.linkedin.com/in/sedoykin
Disclaimer:
No guarantees or responsibilities are provided. You perform all actions at your own risk!

4. **Adding and configuring users at OneLogin account**



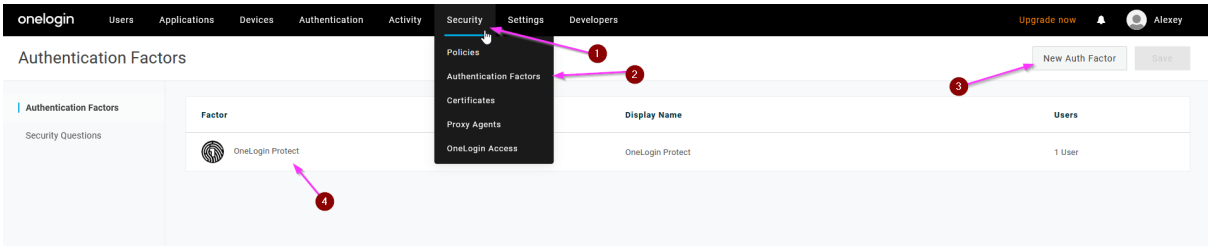


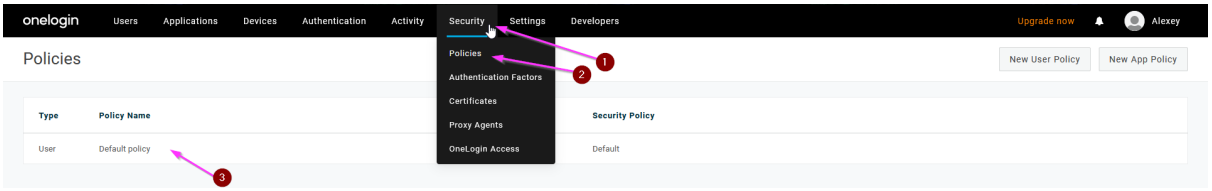


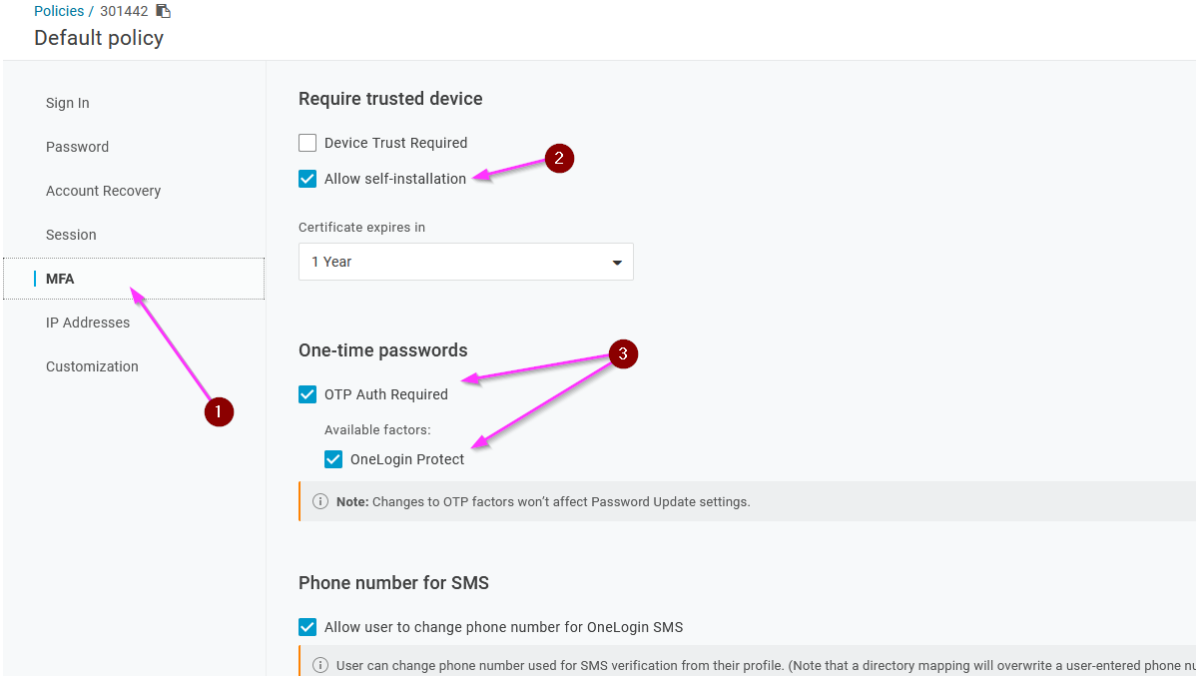
Authorship: https://www.linkedin.com/in/sedoykin
Disclaimer:
No guarantees or responsibilities are provided. You perform all actions at your own risk!

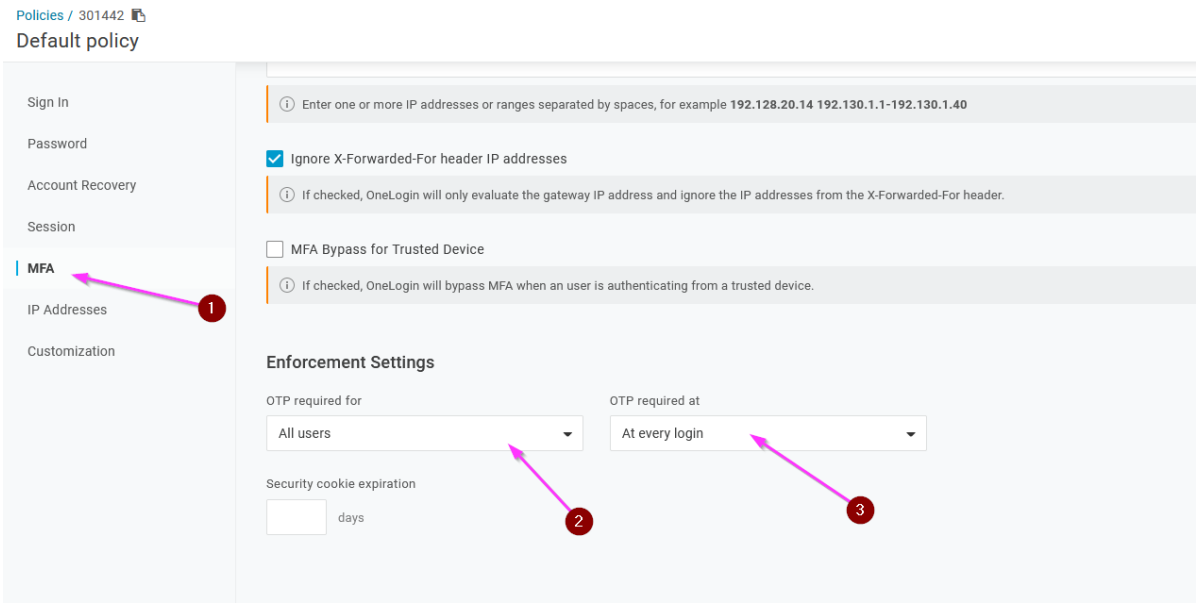
Authorship: https://www.linkedin.com/in/sedoykin
Disclaimer:
No guarantees or responsibilities are provided. You perform all actions at your own risk!

5. Adding MFA feature



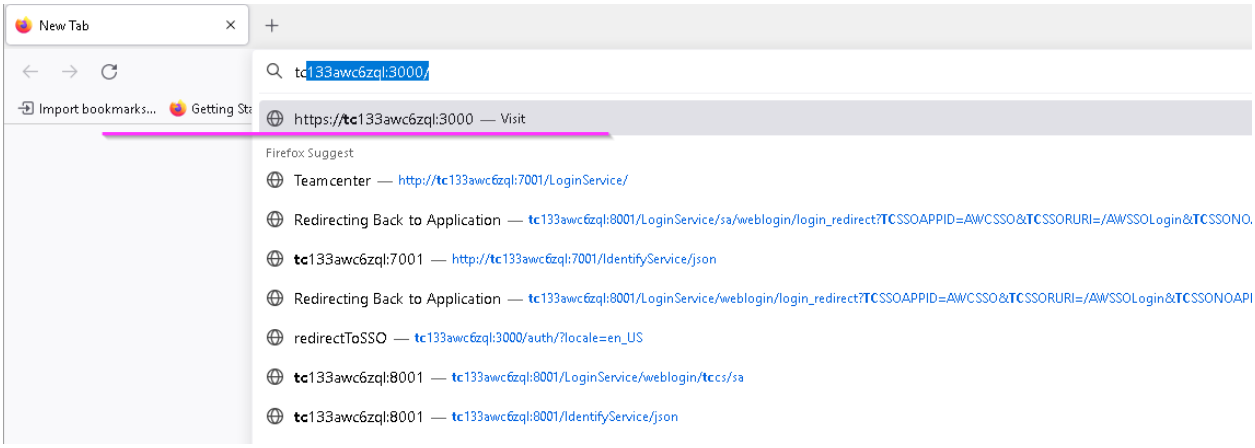


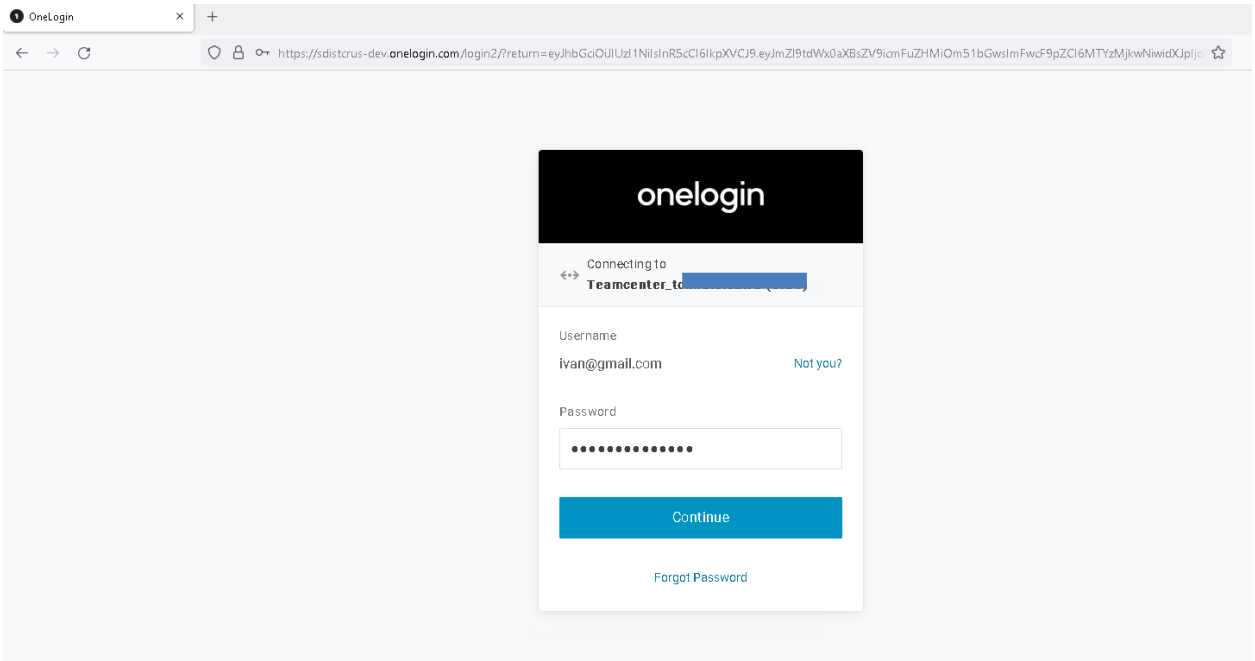


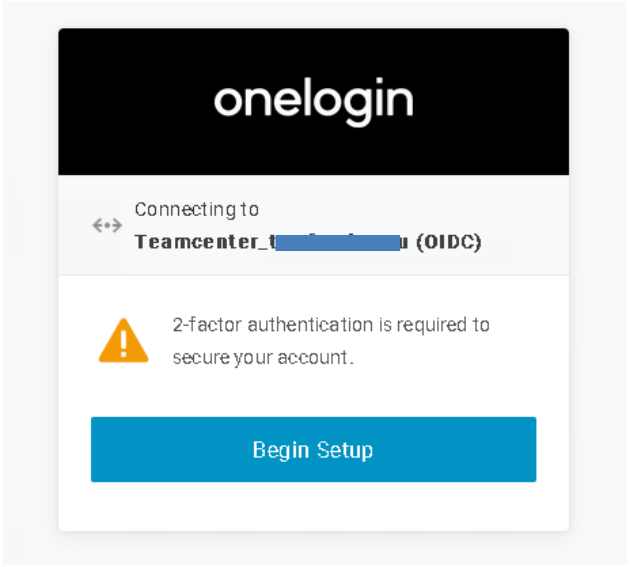


Authorship: https://www.linkedin.com/in/sedoykin
Disclaimer:
No guarantees or responsibilities are provided. You perform all actions at your own risk!

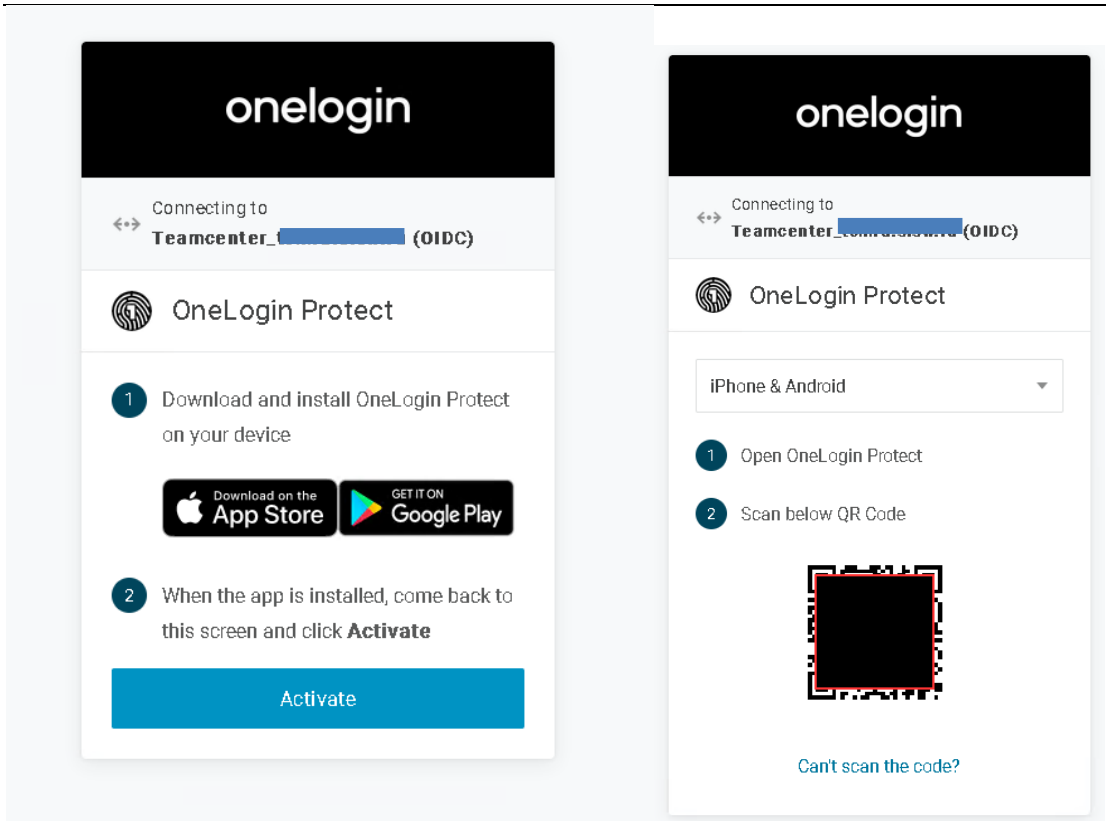
How can I check all this?!

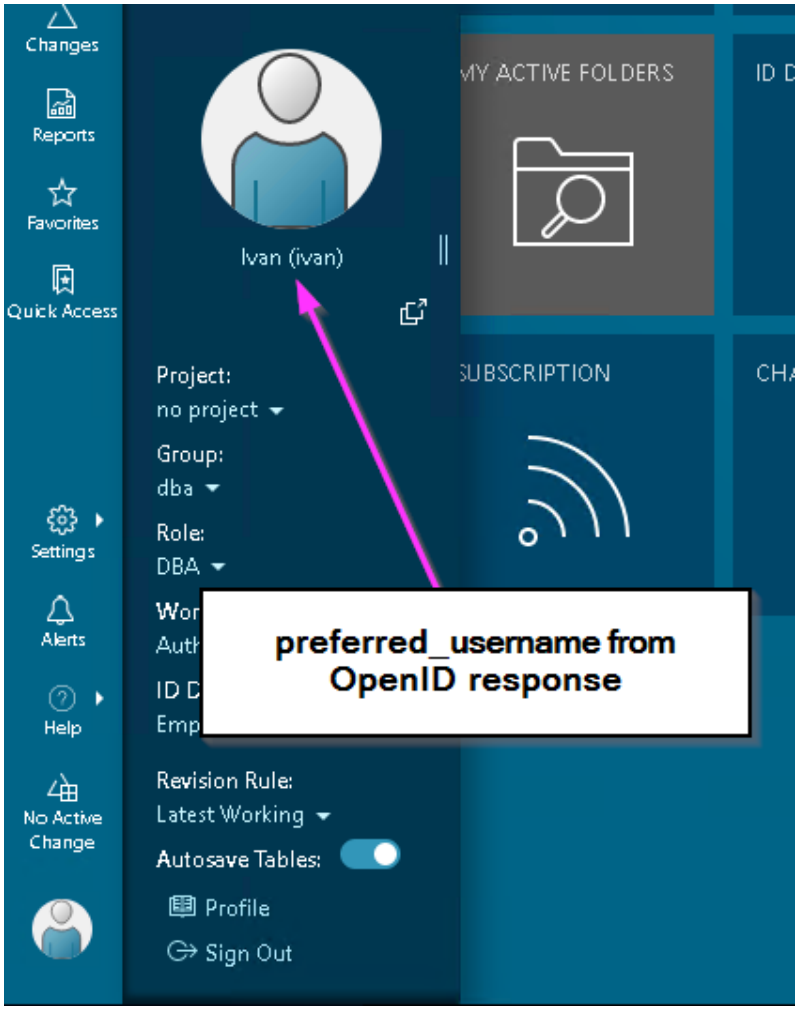






Authorship: https://www.linkedin.com/in/sedoykin
Disclaimer:
No guarantees or responsibilities are provided. You perform all actions at your own risk!





THAT'S ALL, gOOd LUCK!