K-193041

Forvaltning og kjøp av elektrisk kraft

Vedlegg F – ADFS – Avinor Federation Guidelines



AVINOR – FEDERATION GUIDELINES

Version 1.3.2

1 Document Description....5
1.1 Purpose of the document....5
1.2 Scope....5
1.3 Target Audience....5
1.4 Related Documents....5
2 Terms And Abbreviations....6
3 General Description....7
4 Federation Guidelines....8
4.1 Usernames....8
4.2 Preferred Security Token Service (STS)....8
4.3 Azure Active Directory – Entra ID....9
4.3.1 Conditional Access....9
4.3.2 App registrations and authentications flows....10
4.3.3 Cross Broker SSO....11
4.3.4 Azure AD B2B Guest Users....11
4.3.5 Azure Application Gallery....11
4.4 ADFS – Active Directory Federation Service....12
4.4.1 ADFS Preferred Primary Solution....12
4.4.2 ADFS Preferred Secondary Solution....13
4.5 Federation Protocols....14
4.6 Provisioning of identities....15
4.6.1 Supported Protocols for provisioning of identities....15
4.6.2 Preferred Primary Solution of Identities Provisioning....16
4.6.3 Preferred Secondary Solution of Identities Provisioning....20
4.7 Usage of Authorization Claim....22
5 Protocol Requirements....23
5.1 Requirement to OpenIDConnect Implementation....23
5.1.1 OpenIDConnect Metadata Endpoint....23
5.1.2 OpenIDConnect Security Requirements....23
5.1.3 id_token....23
5.1.4 access_token....24
5.2 Requirement to “SAML 2.0 / WS-FED” Implementation....25
5.2.1 Usage of IdP-Initiated Sign On....25
5.2.2 FederationMetadata.xml....26
5.3 Single Logout Protocol (SLO)....28
5.4 Token Signing Certificate....28
5.5 Authentication Libraries....29

5.5.1 Microsoft Authentication Library (MSAL) 29
5.5.2 Azure Active Directory Authentication Library (ADAL) 29
6 PKI Security Guidelines 30
6.1 Certificate Requirements 30
6.2 PKI Secure Channel Protocols 30
6.3 PKI Security Test 31
6.4 PKI Cipher Suites 32
6.4.1 TLS 1.3 Ciphers Suites 33
6.4.2 TLS 1.2 Ciphers Suites 33
7 Appendix - ADFS FederationMetaData 34
7.1 Sample Code - Update Web.Config ADFS FederationMetaData 34
7.2 Sample Code - Update SharePoint ADFS FederationMetaData 37
7.3 Sample Code – Update .NET App ADFS FederationMetaData 40
8 Appendix – PKI 41
8.1 Sample Code – Configure Cipher Suites for Microsoft Windows 41
9 Appendix – Identities provisioning 44
9.1 Appendix – SCIM API 44
9.1.1 Get All Schemas 44
9.1.2 Get Schema Detail 46
9.1.3 Create User 61
9.1.4 Read User 62
9.1.5 Update User Attribute 63
9.1.6 Update User 64
9.1.7 Bulk Operation 66
9.1.8 Search User(s) 68
9.1.9 Get All Groups 70
9.1.10 Get Group Detail 71
9.1.11 Add User to Group 72
9.1.12 Remove User from Group 73

1 DOCUMENT DESCRIPTION

1.1 Purpose of the document

The purpose of this document is to describe Avinor's federation guidelines that shall be followed when establishing federation with 3rd party organizations.

1.2 Scope

The scope is to describe what kind of federation that Avinor supports.

1.3 Target Audience

This document is targeted to 3rd party organizations whom need insight in Avinor's federation guidelines.

1.4 Related Documents

A list of related documents and information is provided below.

- ADFS Design Document.

2 TERMS AND ABBREVIATIONS

Terms and abbreviations used throughout this document are described in the following table:

Term / Abbreviation	Description
Active Directory Federation Service (ADFS)	https://msdn.microsoft.com/en-us/library/bb897402.aspx
Microsoft Entra ID (Azure Active Directory)	https://learn.microsoft.com/en-us/entra/identity/
Home Realm Discovery (HRD)	Home Realm Discovery (HRD) is a feature where the end user gets to choose which Authentication Provider they want to authenticate against from the available Claims Provider Trusts; e.g. users from a partner organization would choose their company's authentication server from the list. The user would then be redirected to the relevant Identity Provider to authenticate.
Identity Provider (IdP)	Identity provider refers to the directory service holding the end users identities/user accounts used by the "Security Token Service" to validate user identities.
Multi Factor Authentication (MFA)	https://azure.microsoft.com/en-us/documentation/articles/multi-factor-authentication/
Security Token Service (STS)	This is the service component that build, signs and issues Security Tokens according to "SAML 2.0", "WS-Fed" and "OAUTH" protocols as part of a claims-based identity system.
Microsoft Intune Endpoint Management	https://learn.microsoft.com/en-us/mem/intune/
Device Registration	https://learn.microsoft.com/en-us/windows-server/identity/ad-fs/technical-reference/device-registration-technical-reference

Table 1.

3 GENERAL DESCRIPTION

Avinor has implemented a standardized claims based “Security Token” authentication and authorization framework for web applications, supporting protocols “OpenIDConnect”, “SAML 2.0”, “WS-FED” security tokens issued either by “Microsoft Entra ID (formerly known as Azure Active Directory)” or “Active Directory Federation Service (ADFS)”.

Avinor has implemented hybrid Azure Active Directory with on-premises Active Directory Domain Services infrastructure with federated accounts through ADFS. This means that Avinor supports both Entra ID and ADFS for authentication of applications depending on required functionality.

ADFS and Entra ID acts as “Security Token Service (STS)” while “Active Directory” acts as “Identity Provider (IdP)” This framework also includes “Federated Accounts” in “Microsoft Azure Active Directory”.

This framework shall be used between Avinor, Cloud Services and 3rd party organizations to establish “Single Sign On” for Avinor’s Active Directory domain joined computers as well as provide access to “Registered Devices (Workplace Join)” and “Unregistered Devices” by utilizing “Azure Multifactor Authentication (MFA)”

4 FEDERATION GUIDELINES

This document is written to ensure that Avinor, together with 3rd party organizations, utilizes federation according to best practices. The document also describes which federation solutions Avinor supports and those that are unsupported. Each implementation intending to utilize federation must follow these guidelines.

4.1 Usernames

Avinor uses the “User-Principal-Name (UPN)” claim token attribute (http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn) as the unique username identifier during authentication. The UPN contains the value of the end user primary e-mail address according to the standard format (https://tools.ietf.org/html/rfc5322#section-3.4.1); e.g. anders.horgen@avinor.no. Avinor policy states that the end user e-mail address shall be used in the UPN claim as the unique user ID for all federated applications. This is extremely important to follow when the federated application is using “Home Realm Discovery (HRD)” service.

4.2 Preferred Security Token Service (STS)

Both “Microsoft Entra ID” and “Active Directory Federation Services (ADFS)” acts as “Security Token Service” in Avinor. Since Avinor operates “Microsoft Entra ID” and “Active Directory Federation Services (ADFS)” with on-Premises Active Directory in a hybrid environment, it is important to select which STS to bed used based on required functionality.

Preference	STS	Protocols	Guest Users	MSAL	Cross Broker SSO
Preferred	Entra ID / Azure AD	OIDC, SAML2	Yes	Yes	Yes
Secondary	ADFS	OIDC, SAML2, WS-FED	No	Yes	No

Figure 1.

The table documents Avinor’s preferred STS and supported functionality.

4.3 Azure Active Directory – Entra ID

Avinor has implemented hybrid Azure Active Directory (AAD) with federated accounts to the on-premises Active Directory infrastructure. AAD is the preferred “Security Token Service” and the Identity Provider (IdP) by using the OpenIDConnect protocol in Avinor.

This design enables Avinor to:

- Support modern authentication to various application architectures, such as (but not limited to) traditional web application, native apps to mobile devices, rest API etc.
- Support various authentication methods such as traditional username / password, biometric, FIDO 2.0, certificate etc.
- Secure and harden access to applications with conditional access.
- Collaborate between applications API’s for sharing data.
- Collaborate with external partners (B2B guest users).
- Establish single sign on to corporate devices (PC and mobile devices).

The guideline in this document is based on best security practices and it is crucial that all integration to AAD follows these guidelines.

4.3.1 Conditional Access

Conditional access polices is a security mechanism to control various signals of whom is granted access to the application. Such signals can be application, users and their location, devices and their managed state, Realtime risk etc. By combining these signals, a policy can be created that block, allow access, or requires multifactor authentication before access is granted.

Avinor requires that it should be possible define dedicated conditional access policy for the particular app that is registered in Azure AD. This will require that the application is using a front-end and back-end architecture, where the API of the application can be protected with conditional access.

Applications registered in Azure AD that can’t define dedicated Condition Access policy, but only works with the default “All Cloud Apps” application signal for defining conditional access policy is not allowed.

Read more here:
https://learn.microsoft.com/en-us/entra/identity/conditional-access/plan-conditional-access

4.3.2 App registrations and authentications flows

All app registrations towards “Azure Active Directory / Entra ID” shall follow Microsoft best practices and recommendations. Avinor may add extra requirements, and this will be documented in this guide that must be always followed.

Microsoft has developed checklist according to best practices that shall be followed:
https://learn.microsoft.com/en-us/entra/identity-platform/identity-platform-integration-checklist

4.3.2.1 Auth Code Flow

Avinor prefers and recommends using Authorization code flow which enables client applications to obtain authorized access to protected web APIs. By combining this flow with “Proof Key for Code Exchange (PKCE)” with “OpenIDConnect (OIDC)”, it is possible issue ID tokens and access tokens for these application types:

- Single-page web application (SPA).
- Standard (server based) web application.
- Desktop Applications.
- Mobile Applications (iOS / Android).

Avinor highly recommends usage of the MSAL library to ensure that all authentication is incompliance to Azure AD / Entra ID best practices. It handles all the authentication scenarios and capabilities that Azure AD / Entra ID supports. Developing your own authentication library can be a complex and time-consuming task, and it may not be the most efficient approach in most cases to be incompliance as Azure AD evolves over time. See reference v2 libraries link for further details.

Read more here:

Microsoft identity platform and OAuth 2.0 authorization code flow
https://learn.microsoft.com/en-us/entra/identity-platform/v2-oauth2-auth-code-flow

Microsoft identity platform authentication libraries
https://learn.microsoft.com/en-us/entra/identity-platform/reference-v2-libraries

4.3.2.2 Implicit Grant Flow

Implicit grant flow is no longer suitable authentication method due to that modern browser are planning to phase out support for third party cookies and cross domain cookies. This will cause that application will break when they attempt to get a new token during the SSO capabilities of this flow.

Microsoft recommends that all developers shall prefer the auth code flow instead. Avinor requires usage of Auth Code flow and do not support implicit grant flow in its infrastructure.

Read more here:

Microsoft identity platform and OAuth 2.0 implicit grant flow
https://learn.microsoft.com/en-us/entra/identity-platform/v2-oauth2-implicit-grant-flow

How to handle third-party cookie blocking in browsers
https://learn.microsoft.com/en-us/entra/identity-platform/reference-third-party-cookies-spas

4.3.3 Cross Broker SSO

The Microsoft Authentication Library (MSAL) supports Single Sign-on (SSO) between iOS/iPadOS native apps and the Microsoft Authenticator App for mobile devices that is registered in Azure Active Directory / Entra ID.

This allows the native iOS/iPadOS to read the refresh tokens written by the Microsoft Authenticator keychain store and exchanging them for access token silently. The end user authenticates inside the Authenticator app as part of the user experience.

Avinor requires the use of Cross Broker SSO for all native iOS / iPadOS apps developed for Avinor.

Read more here:
https://learn.microsoft.com/en-us/entra/identity-platform/single-sign-on-macos-ios

https://learn.microsoft.com/en-us/entra/identity-platform/single-sign-on-macos-ios#sso-through-authentication-broker-on-ios

4.3.4 Azure AD B2B Guest Users

Avinor support collaboration with external guest users (B2B) for registered Azure AD tenant's domains where NDA agreement is established between Avinor and the external collaboration company.

The external guest user object in Azure AD is either created by invitation from the application itself via the Microsoft Graph Invitation API or by self-service via Microsoft Access Package.

It is required that the application supports the special UPN format for B2B guest users:
jonny.good_domain.com#EXT#domain.onmicrosoft.com

It is also required that application supports synchronization of guest users object with the SCIM protocol.

4.3.5 Azure Application Gallery

Avinor has implemented "Microsoft Azure Active Directory" with "Federated Accounts".
If the 3rd party web application is registered in the "Azure Application Gallery", it can be viewed as an alternative federated integration as long as the end user don't need to access the web application via the "Azure Application Portal", but rather directly by using the web application (URL) address directly.

4.4 ADFS – Active Directory Federation Service

4.4.1 ADFS Preferred Primary Solution

Preferred federation integration is according to best practices, which involves establishing “Relying Party Trust” (also sometimes referred to as “Claims Provider Trust” / “Service Provider Initiated Sign-on”) between the vendor web application and Avinor’s “Security Token Service”.

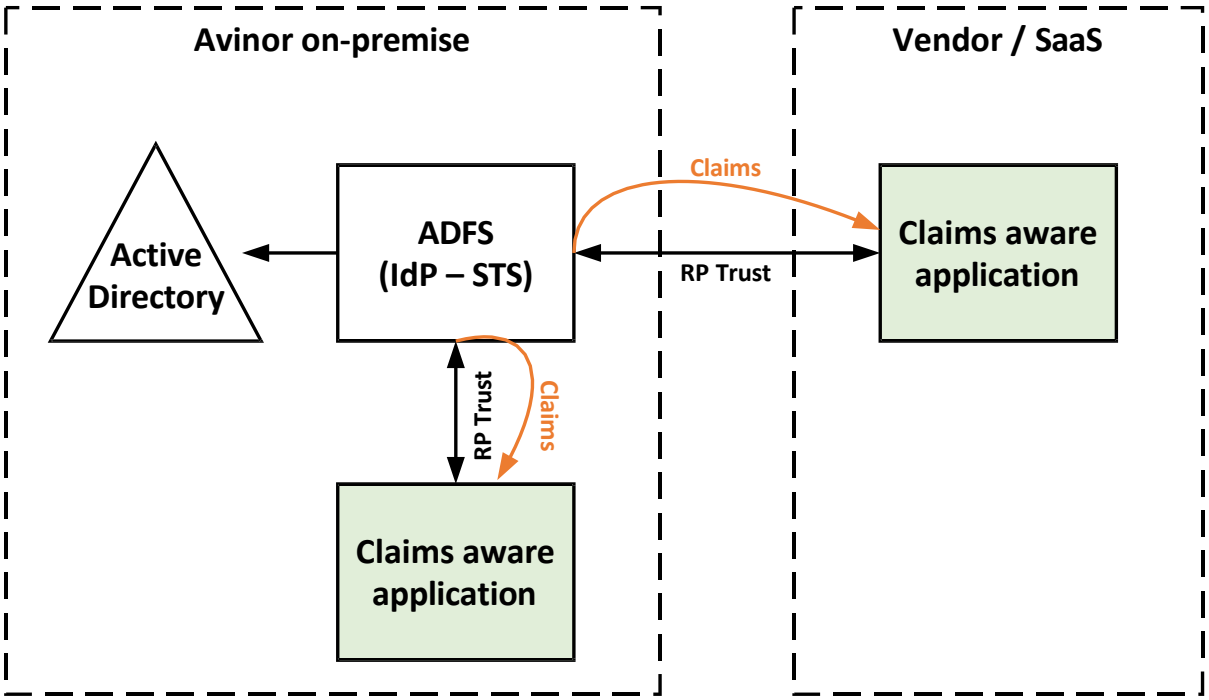


Figure 2.

This design requires that the web application is configured with a unique “Fully Qualified Domain (FQDN)” DNS host header (URL) address dedicated for Avinor. The dedicated FQDN can’t be used by any other customers that the vendor may have, as validation of identities is carried out by Avinor ADFS. Avinor can in this design re-federate this application to Avinor’s partner organizations that support federation.

The table below explains valid URL configurations.

Example URL	Comment
https://applikasjonsnavn.avinor.no	Preferred configuration, shall be selected.
https://avinor.applikasjonsnavn.com	Allowed only if preferred URL can’t be used.
https://www.applikasjonsnavn.com/Avinor	Invalid configuration

Table 2.

4.4.2 ADFS Preferred Secondary Solution

Preferred primary solution cannot be used when the 3rd party organization rely on its own "Security Token Service (STS)" that the web application must use. This is applicable when the web application uses a shared DNS host header (URL) address among their partners where Avinor can't be granted dedicated DNS host header (URL) address to access the web application. Examples of such web services is e.g. "Microsoft Office 365".

To achieve this design, the 3rd party organization's "Security Token Service" shall be registered as a "Relying Party Trust" at Avinor's ADFS (STS). Both parties must exchange "FederationMetaData.xml" over HTTPS URL.

The 3rd party web application is configured as a "Relying Party Trust" at the 3rd party "STS".
This design requires implementation of "Home Realm Discovery (HRD)" service at the 3rd party organization's "STS", whereas the vendor's "STS" redirects all authentication to Avinor's "Security Token Service" for validation and authentication of Avinor user identities. The end users must provide their e-mail address during the "Home Realm Discovery" process. This provides "Single Sign On" as long as the "HRD" cookie exists or for the duration of the lifetime of the "HRD" cookie. The vendor must inform Avinor of the lifetime of the "HRD" cookie. Single Sign On will not be available during the first logon since the "HRD" process must be carried out first to have the "HRD" cookie issued.

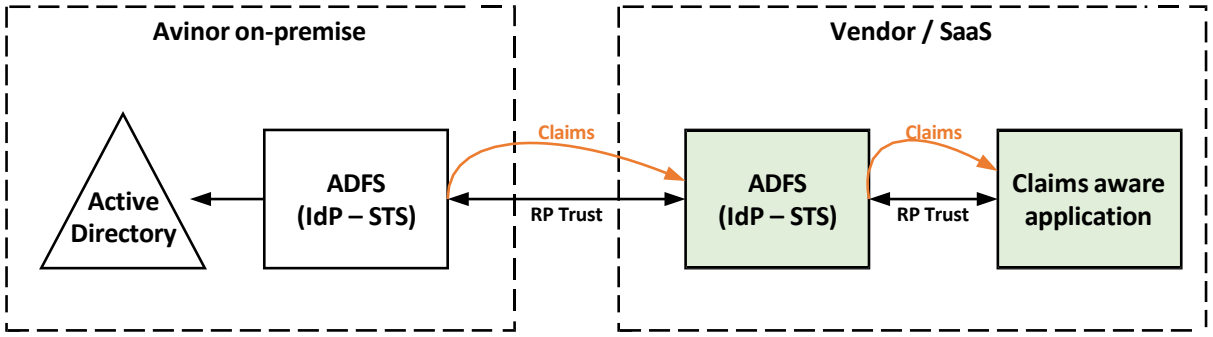


Figure 3.

4.5 Federation Protocols

Avinor supports following claims-based security token protocols:

- OpenIDConnect
- SAML 2.0
- WS-FED

Above protocols is based on open standards and shall be implemented according to best practices. The technical requirement for each protocol is described later in this document.

Each protocol has various purposes, possibilities and its use must be selected carefully accordingly to the application purpose, application availability and Avinor's strategy for federation.

Modern Authentication

Avinor's strategy is to utilize "Modern Authentication" as preferred authentication mechanism to support and allow authentication for native iOS and Android applications including standard web applications. To achieve this goal, the "OpenIDConnect" protocol must be selected as the other legacy protocol do not support this.

Legacy Authentication

If implementing standard web application, "SAML 2.0" and "WS-FED" protocol can be used as the authentication protocol to ADFS. Be aware of that these protocols are not supported to be used as authentications protocols for native iOS and Android applications, hence OpenIDConnect should be selected.

Below table describes how these protocols can be used for and its preferred implementation order.

Protocol	Purpose	Auth. Type	Preferred order
OpenIDConnect	Web Application (Web API) Native Application Server Application	Modern Authentication	Primary preferred protocol
SAML 2.0	Web Application	Legacy Auth.	Can be used, reaching EOL
WS-FED	Web Application	Legacy Auth.	Can be used, reaching EOL

Table 3.

4.6 Provisioning of identities

Federation itself do not provide provisioning of user identities between the 3rd party organization and Avinor, hence it will be necessary to provision user identities by establishing synchronization between Avinor and the 3rd party organization.

Avinor prefers that such synchronization is carried out between the parties' directory services via Avinor's "Identity Management (IDM)" solution. Alternatively, provisioning may occur between Avinor "Azure Active Directory" and the 3rd party organizations directory service.

If the preferred provisioning method is to utilize Avinor's "Identity Management (IDM)" solution, the 3rd party organization implicitly authorize Avinor to manage its own identities. This is achieved by the 3rd party providing web services that supports operations like "Create User", "Modify User", "Update User", "Disable User", "Delete User" etc. Details of such integration must be agreed upon on a case-by-case basis.

The 3rd party organization must provide an API that supports the complete management of their user lifecycle (Create, Update, Disable, Delete). Avinor's IDM solution supports a wide range of protocols (REST, SOAP, LDAP, SQL etc.) for the provisioning and management of user identities to both on-premises and Cloud-based applications. The API must be protected with authentication mechanism to ensure security.

Integration method	Preferred Order
Avinor IDM – SCIM / REST	Preferred Primary Solution
Avinor IDM – SOAP	Preferred Secondary Solution
Azure Active Directory App Gallery	Optional Solution

Table 4.

4.6.1 Supported Protocols for provisioning of identities

Avinor's IDM system is capable of supporting a rich set protocols for integrating user management to 3rd party solutions. Some of the protocols have limited functionality and does not match well with best practices for managing user identities. The table below shows the list of protocols that Avinor supports.

Protocol	Supported Configuration	
	On-Premise	Cloud
Web Services SCIM	Yes	Yes
Web Services REST	Yes	Yes
Web Services SOAP	Yes	Yes
LDAP / ADSI	Yes	No
SQL	Yes	No
File (CSV, LDIFF etc.) / FTP / SMTP exchange	No	No

Table 5.

Although the system is able to integrate using a range of other protocols (FTP, SMTP, File, etc), these are not supported as they have limited integration functionalities for proper automation.

4.6.2 Preferred Primary Solution of Identities Provisioning

The System for Cross-Domain Identity Management (SCIM) specification is a widely used standard for managing user identities. It is designed for simplifying the management of user identities in cloud-based applications and services. This section explains an overview of the SCIM protocol. More details around this specification is also described here:

- RFC 7643 – SCIM Core Schema https://tools.ietf.org/html/rfc7643
- RFC 7644 – SCIM Protocol https://tools.ietf.org/html/rfc7644

The SCIM object model uses Resource as the common denominator that SCIM objects derives from.

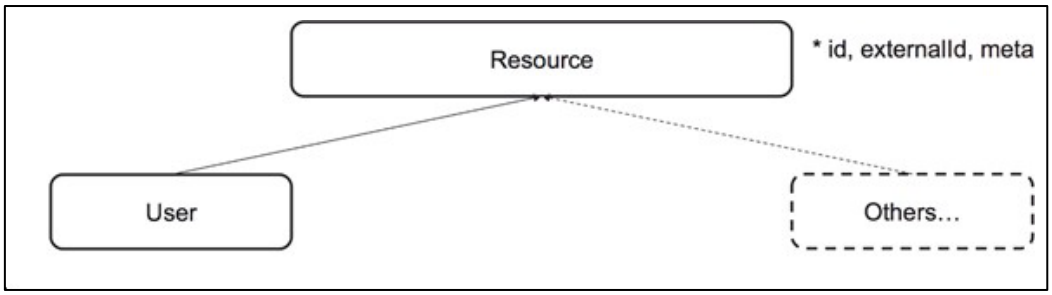


Figure 4.

4.6.2.1 Schema

The schema endpoint is used to retrieve information about SCIM resources. SCIM Schema endpoint is one of the SCIM Discovery Mechanisms and allows introspection on SCIM Resources.

Method	High Level Example (See Appendix 9 for details)	Required
READ	GET https:// democompany.com/{version}/Schemas	Required

Table 6.

4.6.2.2 Common Attributes

A resource's common attributes are those attributes that are part of every SCIM resource, regardless of the value of the "schemas" attribute present in a JSON body. These attributes are not defined in any schema but SHALL be assumed to be present in every resource, regardless of the value of the "schemas" attribute. The following common attributes are included in each SCIM resource:

4.6.2.2.1 ID

A unique identifier for a SCIM resource as defined by the service provider. Each representation of the resource MUST include a non-empty "id" value. This identifier MUST be unique across the SCIM service provider's entire set of resources. It MUST be a stable, non-reassignable identifier that does not change when the same resource is returned in subsequent requests. The value of the "id" attribute is always issued by the service provider and MUST NOT be specified by the client. The string "bulkId" is a reserved keyword and MUST NOT be used within any unique identifier value. The attribute characteristics are "caseExact" as "true", a mutability of "readOnly", and a "returned" characteristic of "always".

4.6.2.2.2 EXTERNALID

A String that is an identifier for the resource as defined by the provisioning client. The "externalId" may simplify identification of a resource between the provisioning client and the service provider by allowing the client to use a filter to locate the resource with an identifier from the provisioning domain, obviating the need to store a local mapping between the provisioning domain's identifier of the resource and the identifier used by the service provider. Each resource MAY include a non-empty "externalId" value. The value of the "externalId" attribute is always issued by the provisioning client and MUST NOT be specified by the service provider. The service provider MUST always interpret the externalId as scoped to the provisioning domain. While the server does not enforce uniqueness, it is assumed that the value's uniqueness is controlled by the client setting the value. This attribute has "caseExact" as "true" and a mutability of "readWrite". This attribute is OPTIONAL.

4.6.2.2.3 META

A complex attribute containing resource metadata. All "meta" sub-attributes are assigned by the service provider (have a "mutability" of "readOnly"), and all of these sub-attributes have a "returned" characteristic of "default". This attribute SHALL be ignored when provided by clients. "meta" contains the following sub-attributes:

- **resourceType**: The name of the resource type of the resource. This attribute has a mutability of "readOnly" and "caseExact" as "true".

- **created**: The "DateTime" that the resource was added to the service provider. This attribute MUST be a DateTime.

- **lastModified**: The most recent DateTime that the details of this resource were updated at the service provider. If this resource has never been modified since its initial creation, the value MUST be the same as the value of "created".

- **location**: The URI of the resource being returned. This value MUST be the same as the "Content-Location" HTTP response header

- **version**: The version of the resource being returned. This attribute has "caseExact" as "true". Service provider support for this attribute is optional and subject to the service provider's support for versioning.

4.6.2.3 User Resource

SCIM provides a resource type for "User" resources. The core schema or "User" is identified using the following schema URI: "urn:ietf:params:scim:schemas:core:2.0:User"

User Format Example

The example below represents the user data as SCIM object using JSON format. The attributes are not limited to what is shown in the example and may be of different types (Simple, Complex, and Multivalued) depending on the target systems requirements

--- Sample JSON Start ---

```
{
  "schemas": ["urn:ietf:params:scim:schemas:core:2.0:User"],
  "id":"23123",
  "externalId":"GMOLA",
  "meta":{
    "resourceType": "User",
    "created":"2018-09-01T12:21:11.714Z",
    "lastModified":"2018-09-01T12:21:11.714Z",
    "location":"https://democompany.com/v2/Users/23123"
  },
  "name":{
    "familyName": "Nordmann",
    "givenName": "Ola"
  },
  "userName": "GMOLA",
  "active": true,
  "phoneNumbers":[
    {
      "value":"+4712345678",
      "type":"work"
    }
  ],
  "emails":[
    {
      "value":"ola.nordmann@avinor.no",
      "type":"work",
      "primary": true
    }
  ],
  "groups" : [
    {
      "value" : "e9e30dba-f08f-4109-8486-d5c6a331660a",
      "$ref": "https://democompany.com/v2/Groups/e9e30dba-f08f-4109-8486-d5c6a331660a",
      "display": "Supervisor"
    },
    {
      "value" : "c3a26dd3-27a0-4dec-a2ac-ce211e105f97",
      "$ref": "https://democompany.com/v2/Groups/c3a26dd3-27a0-4dec-a2ac-ce211e105f97",
      "display": "Sales"
    }
    ]
}
```

--- Sample JSON Stop ---

Table 7.

User Operations

For performing manipulation on user identities, SCIM provides a REST API that supports these operations (the appendix describes each SCIM operation in detail):

Method	High Level Example (See Appendix 9 for details)	Required
CREATE	POST https://democompany.com/{version}/{user}	Required
READ	GET https:// democompany.com/{version}/{user}/{id}	Required
REPLACE	PUT https:// democompany.com/{version}/{user}/{id}	Required
DELETE	DELETE https:// democompany.com/{version}/{user}/{id}	Required
SEARCH	GET https:// democompany.com/{version}/{user}?filter={attribute} &sortBy={attributeName}&sortOrder={ascending|descending}	Required
BULK	POST https:// democompany.com/{version}/Bulk	Optional

Table 8.

4.6.2.4 Group Resource

The Group resource is used to represent group-based or role-based access control models. It is intended that the semantics of group membership, and any behavior or authorization granted as a result of membership, are defined by the service provider.

Group Operations

Method	High Level Example (See Appendix 9 for details)	Required
CREATE	POST https://democompany.com/{version}/Groups	Optional
GET ALL	GET https:// democompany.com/{version}/Groups	Required
GET	GET https:// democompany.com/{version}/Groups/{id}	Required
SEARCH	GET https://democompany.com/{version}/Groups?filter={attribute}	Required
ADD MEMBER	PATCH https:// democompany.com/{version}/Groups/{id}	Required
REMOVE MEMBER	PATCH https:// democompany.com/{version}/Groups/{id}	Required

Table 9.

4.6.3 Preferred Secondary Solution of Identities Provisioning

In cases where the preferred solution is not available from the 3rd party provider, SOAP based Web Services may be used for cloud-based applications. The Web Service must support Avinor's IDM system by providing operations for managing the complete user identity lifecycle processes (Create, Update, Disable, Delete, and Search). The SOAP API must be agreed upon by both parties and be maintained by 3rd party provider.

The following shows the minimum set of operations for user management that must be supported:

- Read / Search
- Create
- Update
- Enable / Disable / Delete

WSDL User Type Example

--- WSDL Sample Code Start ---

```
<wsdl:types>
 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:tns="http://www.democompany.com/webservices/model/user"
xmlns:ns1="http://webservices.democompany.com/userService" attributeFormDefault="unqualified"
elementFormDefault="unqualified"
targetNamespace="http://www.democompany.com/webservices/model/user">
 <xs:import namespace="http://webservices.democompany.com/userService"/>
  <xs:complexType name="User">
   <xs:sequence>
    <xs:element name="UserId" type="xs:string"/>
    <xs:element name="UserName" type="xs:string"/>
    <xs:element minOccurs="0" name="FirstName" type="xs:string"/>
    <xs:element minOccurs="0" name="LastName" type="xs:string"/>
    <xs:element minOccurs="0" name="Email" type="xs:string"/>
    <xs:element minOccurs="0" name="MobilePhone" type="xs:string"/>
    <xs:element minOccurs="0" name="WorkPhone" type="xs:string"/>
    <xs:element minOccurs="0" name="Status" type="xs:boolean"/>
   </xs:sequence>
  </xs:complexType>
 </xs:schema>
</wsdl:types>
```

--- WSDL Sample Code Stop ---

Table 10.

WSL User Operations Example

--- WSL Sample Code Start ---

```
<wsdl:portType name="UserService">
 <wsdl:operation name="createUser">
  <wsdl:input message="tns:createUser" name="createUser"> </wsdl:input>
  <wsdl:output message="tns:createUserResponse" name="createUserResponse"> </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="updateUser">
  <wsdl:input message="tns:updateUser" name="updateUser"> </wsdl:input>
  <wsdl:output message="tns:updateUserResponse" name="updateUserResponse"> </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="createUser">
  <wsdl:input message="tns:createUser" name="createUser"> </wsdl:input>
  <wsdl:output message="tns:createUserResponse" name="createUserResponse"> </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="enableUser">
```

```
  <wsdl:input message="tns:enableUser" name="enableUser"> </wsdl:input>
  <wsdl:output message="tns:enableUserResponse" name="enableUserResponse"> </wsdl:output>
 </wsdl:operation>
</wsdl:portType>
```

--- WSL Sample Code Stop ---

Table 11.

4.6.3.1 Optional API for On-Premises Solutions

In cases where 3rd party cannot provide neither SCIM, REST nor SOAP API, the following options may be considered for on-premises applications. The APIs implemented on top of these protocols must be maintained and supported by the 3rd party provider. In call cases, a clearly defined API must be designed and agreed upon prior to setting up the integration.

- LDAP/ADSI
- SQL

LDAP

Integration over the LDAP protocol may be considered for applications that use an LDAP directory as user metadata store and supports standard LDAP operations for creating and manipulating user identities. If the application has an internal user database that synchronizes with LDAP, then the preferred way of integration is to use an API to connect directly with the user store through SQL API to avoid delay and inconsistencies between the user data stores (database, LDAP).

SQL

Avinor IDM also supports SQL API for on-premises applications. The 3rd party must provide and support a complete API for management of the entire user lifecycle. This can be achieved through access to Tables, Views, Stored Procedures, PL/SQL, etc.

4.7 Usage of Authorization Claim

The 3rd party organizations shall describe how and if the application can utilize claims issued by Avinor "Security Token Service" to regulate functionality inside the application.

There is a desire to investigate the possibility to regulate access to functionality for "mobile devices", "managed devices" and "non-managed devices" inside of a given application.

Avinor has implemented Microsoft Intune together with "Azure Device Registration Service".
This allows Avinor to differentiate access to web applications and native applications based on whether or not the mobile device is "Device registered (Workplace Join)" enrolled and/or Microsoft Intune enrolled. Azure Active Directory (AAD) and Active Directory Federation Service (ADFS) can differentiate these devices by two specific security tokens, as described in the table below.

Service	Claim	Claim Value
Microsoft Intune	IsManaged	True / False
Microsoft Intune	IsCompliant	True / False
Workplace Join	IsRegisteredUser	True / False

Table 12.

By utilizing these three claims, functionality may be regulated and thus improve on the security as shown in the usage scenarios below:

Example 1:
The application can store "offline" information on the mobile device.

If the mobile device is registered/enrolled in "Microsoft Intune" and "Workplace Join",
then the end user mobile device is authorized to keep an offline copy of the content of the application on the mobile device.

Mobile devices that are not enrolled, may only access application content that is available online and might require usage of "Multi Factor Authentication" when authenticating.

Example 2:
The application is used to sign and validate company invoices.

If the mobile device is registered/enrolled in "Microsoft Intune" and "Workplace Join",
then the end user mobile device is authorized to validate and sign the invoice and process it for payment if the device is compliant. Mobile devices that are not enrolled, cannot process the invoice for payment by validating and signing the invoice, but might only be authorized to review the invoice. It's possible to allow validation of the invoice, but in this scenario we'd require "Multi Factor Authentication" (using e.g. "SMS Text Message Pin Code") for each such actions carried out inside the application.

The two examples above explain how the two claims (IsManaged, IsComplaint and IsRegisteredUser) may be used to authorize access to various functionality inside the application.

The 3rd party vendor should describe the possibilities for Avinor to govern functionality based on the claims and also describe other possible claims that may provide similar ways to govern the functionality of the application.

5 PROTOCOL REQUIREMENTS

5.1 Requirement to OpenIDConnect Implementation

5.1.1 OpenIDConnect Metadata Endpoint

According to best practice when implementing OpenIDConnect protocol and establishing a trust between the web application and Avinor "Security Token Service (STS)", the web application shall utilize all metadata endpoints provided by AAD or ADFS "OpenIDConnect Well-Known configuration" metadata. The application shall read the OpenIDConnect Metadata Endpoint and configure its application accordingly. It is not allowed to hardcode endpoints directly in the application.

5.1.2 OpenIDConnect Security Requirements

To ensure secure and safe implementation of OpenIDConnect,
it is important that some validation is carried out before authentication can take place at the application level. Receiving an "**id_token**" / "**access_token**" without validating the token's signature during authentication is forbidden. The application must validate the token's signature from the Secrity Token Service (STS) and verify that the claims in the token is accordingly to as per app's requirement.

5.1.3 id_token

After validating the "**id_token**", several claims require validation:

- Validate the "**nonce**" claim to prevent token replay attacks.
  Its value should be what you specified in the sign-in request.

- Validate the "**aud**" claim to ensure that the "**id_token**" was issued for your app.
  Its value should be the application ID of your app.

- Validate the "**iat**" and "**exp**" claims to ensure that the "**id_token**" has not expired.
  If expired, the user must re-authenticate to the Security Token Service (STS) and gain a new "**id_token**" that is valid.

- Validate the "**iss**" claim to ensure that the correct IdP has issued the token.
  Its value should be the value of AAD or ADFS OpenIDConnect issuer endpoint

There are also several more validations that shall be performed according to the detail description of the "OpenID Connect Core Spec". Depending of scenario, there might be other validation that shall be carried out:

- Ensure that the user has proper authorization / privileges.
  Presenting the application GUI and displaying no data is not allowed if the user is not authorized even if a valid "**id_token**" may exist. The user shall then be presented with "no access" GUI.
- Ensuring that a certain strength of authentication has occurred, such as Multi-Factor Authentication

5.1.4 access_token

After validating the "**access_token**", several claims require validation:

- Validate "**alg**" claim to ensure that correct hash algorithm from STS is being used.
  Its value should be the same hash algorithm to STS Token Signing Certificate

- Validate "**`x5t`**" to ensure correct public key of the token signing certificate has been used.
  Use the "**`jwks_uri`**" property in "OpenIDConnect Metadata Endpoint" to locate correct endpoint for locating current used token signing certificated.

- Validate the "**nonce**" claim to prevent token replay attacks.
  Its value should be what you specified in the sign-in request.

- Validate the "**aud**" claim to ensure that the "**access_token**" was issued for your app.
  Its value should be the application ID of your app.

- Validate the "**iat**" and "**exp**" claims to ensure that the "**access_token**" has not expired.
  If expired, the user must re-authenticate to the Security Token Service (STS) and gain a new "**access_token**" that is valid.

There might be other validations of claims before allowing access:

- Ensure that the user has proper authorization / privileges.
- Ensuring that a certain strength of authentication has occurred, such as Multi-Factor Authentication
- E.g. if it is required that the user is member of an "AD Security Group", ensure and validate that this claim is received before allowing access. Presenting the application GUI and displaying no data is not allowed if the user is not authorized even if a valid "**access_token**" may exist. The user shall be prompted with "no access" page.

Ref: https://docs.microsoft.com/en-us/azure/active-directory/develop/active-directory-token-and-claims#validating-tokens and https://medium.com/the-new-control-plane/validating-an-adfs-jwt-token-1146bb529a2a

5.2 Requirement to “SAML 2.0 / WS-FED” Implementation

According to best practices, establishing a trust between the web application and Avinor is achieved by registering the web application as a “Relying Party Trust” in Avinor’s “Security Token Service (STS)” by exchanging “FederationMetaData.xml” over HTTPS URL. The vendor web application is using Avinor’s “Security Token Service” as the “Identity Provider (IdP)”, whereas Avinor issues the required Security Token that the application requires. Vendor web application will automatically redirect the web session to Avinor STS for authentication and validation of the user identity. This provides “Single Sign On” to the web application.

5.2.1 Usage of IdP-Initiated Sign On

Avinor do not support using “IdP-Initiated Sign On” as part of the federation integration since it is not according to best practice. Examples of usage scenarios is when the web application is not using any “Security Token Service” and can’t be registered as a “Relying Party Trust”, but only accept any possible claims that are bypassed by an STS service.

5.2.2 FederationMetadata.xml

According to best practice when performing federation and establishing trust between Avinor "Security Token Service (STS)" and the "Web Application" or 3rd party "Security Token Service", exchanging of "FederationMetadata.xml" must be carried out over fully qualified HTTPS URL (e.g. https://example.com/FederationMetaData/2007-06/FederationMetaData.xml).

Avinor's STS "FederationMetadata.xml" will contain information about:

- Avinor's DNS Address of "Security Token Service (STS)
- Endpoint of Avinor's STS Logon
- Claims offered by Avinor's STS
- Avinor's Token Signing Certificate

The Web Application "FederationMetadata.xml" must contain:

- Name of Relying Party Trust (DNS Address of the web application)
- Endpoint of Relying Party Trust Logon
- Claims that is required by the web application
- The certificate public key to encrypt Security Token (**Required**)

Since the "FederationMetadata.xml" is the vital key component to establish trust, the "FederationMetadata.xml" must be monitored automatically both by the "Security Token Service" and the "Web Server configuration" (representing the Web Application) to avoid service interrupt, whereas trust configuration must be maintained automatically. Recommended monitor interval is every 24hrs.

5.2.2.1 Avinor's Token Signing Certificate

Avinor "Security Token Service (STS)" will automatically renew the certificate used to sign "Security Token" issued without any notice up-front when the certificate expiries.

To avoid trust service interruption, the "Web Server configuration" hosting the "Web Application" must therefore automatically update its configuration to use the new valid "Token Signing Certificate" presented in Avinor's "Federationmetadata.xml". During the certificate renewal period of three weeks, the current and valid "Token Signing Certificate" is presented as "Primary Token Signing Certificate" and the new upcoming Token Signing certificate is presented as "Secondary" in "Federationmetadata.xml". This occurs without any notice to clients up-front, with the implication that the "Web Applicaltion" should have functionality in place to check up on the "Federationmetadata.xml" every day in order to pick up the certificate change in a timely and automatic fashion. Therefore, automatically or manually, it is crucial that the "Secondary" Token Certificate is registered as the future "Primary" Token Signing Certificate in the configuration of the "Web Application" as soon as possible after the new certificate is available in "Federationmetadata.xml". The "Secondary Token Signing Certificate" in "Federationmetadata.xml" will automatically be promoted to the "Primary Token Signing Certificate" 3 weeks after the "Secondary Token Signing Certificate" have been presented in the "Federationmetadata.xml", see example below

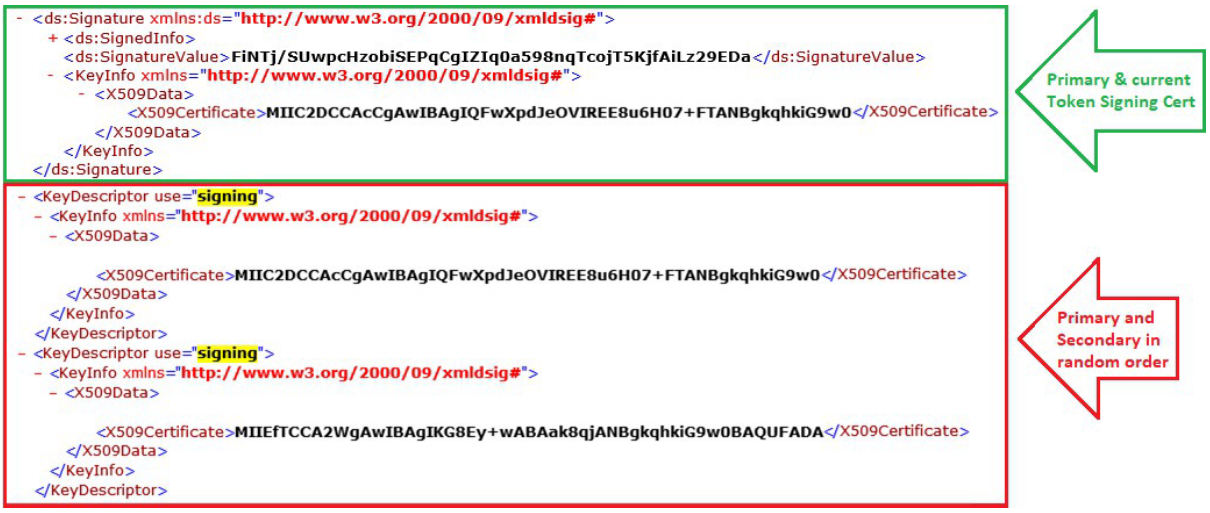


Figure 5.

As an example from a Microsoft environment; this is often referred to the “<trustedIssuers>” section in the “web.config” file where “Token Signing Certificate Thumbprint” is registered.

5.2.2.2 Application Token Encryption Certificate

According to best security practices, encrypting the “Security Token” cookie is required.
Avinor “Security Token Service (STS)” transports the “Security Token” as unencrypted cookie over TLS (by default), hence the cookie remains unencrypted in the web browser client. To avoid compromising Avinor’s identities, the “Security Token” cookie must be encrypted.

To mitigate this security vulnerability, the web server hosting the application, must use a dedicated encryption certificate for this purpose that is presented in the web application corresponding “FederationMetadata.xml”. Avinor “Security Token Service (STS)” will encrypt the “Security Token” cookie with corresponding Encryption Certificate (using the Public Key of the encryption certificate) presented in the web application “FederationMetadata.xml”. The private key of the encryption certificate shall only be installed and available for the web servers hosting this application.

The Encryption Certificate must fulfill following requirements:

- Must be registered in configuration file of the web server (web.config if IIS is used).
- Must be presented in “FederationMetadata.xml” as BASE64 format (see figure below)
- Must be automatically updated in the “FederationMetadata.xml” during renewal
- Must be dedicated to the application.
- Encryption Certificate should be named in following manner: “NameOfApplication-encryption.dnsdomain” example “application-encryption.avinor.no”
- It’s is not allowed to use the same TLS certificate used in the HTTPS transport-layer of the application as an Encryption Certificate
- The Encryption Certificate must use SHA256RSA signature algorithm or higher
- Self-signed Encryption Certificate is allowed (but not recommended)
- Encryption Certificate issued from Internal Issuing CA is allowed.
- Public issued Encryption Certificate is optional (but recommended)
  (This shall always be preferred choice if the application require high security)



Figure 6.

5.3 Single Logout Protocol (SLO)

The “Single Logout Protocol (SLO)” provides the possibility to sign-out all sessions (“Security Tokens”) issued by all “Security Token Services (STS)” involved in federation for the particular web application.

This means that when the end user decides to sign-out of the web application, the web application must perform “Single Logout Operation (SLO)” by performing sign-out of all “Security Tokens” from all authorities (“Security Token Services (STS)) issued during sign-on. The end user will be logged out of all web applications accessed during the federated session.

Further details and examples can be found here:

- **OpenIDConnect**
  https://docs.microsoft.com/en-us/windows-server/identity/ad-fs/development/ad-fs-logout-openid-connect
- **SAML 2.0 RFC**
  http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf
- **WS-FED**
  https://msdn.microsoft.com/en-us/library/azure/dn223670.aspx
- **Microsoft Azure SLO**
  https://msdn.microsoft.com/en-us/library/azure/dn195588.aspx

The “Single Logout Protocol (SLO)” is required and must be implemented for following protocols:

- OpenIDConnect
- SAML 2.0
- WS-Federation (WS-FED)

5.4 Token Signing Certificate

Avinor “Security Token Service (STS)” will automatically renew the certificate used to sign “Security Token” issued without any notice up-front when the certificate expiries.

To avoid trust service interruption, each application must validate and ensure that the issued token uses the correct Token Signing Certificate at all time.

Current and valid “Token Signing Certificate” is presented in Federationmetadata.xml for WS-FED and SAML protocol and in OpenIDConnect Metadata Endpoint. See above chapters for each various protocol requirements on how to validate the token signing certificate.

5.5 Authentication Libraries

Authentication to identity platforms, such as Azure Active Directory and ADFS may require many different sign-in methods, security, and compliance requirements for the application. Authentication Libraries enables developers to acquire security tokens, support multiple sign-in methods etc. from Azure Active Directory or ADFS without having to develop this from scratch.

5.5.1 Microsoft Authentication Library (MSAL)

Avinor highly recommends usage of MSAL authentication library.
MSAL enables developers to authenticate users and access secured web APIs by acquiring security tokens. The library is free to use from Microsoft. MSAL supports multiple languages and frameworks including various scenarios of application architecture and types.

Using MSAL provides following benefits:

- Developer do not need to maintain support of all authentication methods that Avinor defines.
- Developers do not need to maintain all compliance requirements of Azure AD.
- MSAL is maintained by Microsoft to support all required methods for authentication.
- There is no need to directly use the OAuth libraries or code against the protocol in your application etc.
- Handles Home Realm Discovery
- Can acquire tokens on behalf of a user or application (when applicable)
- Helps you specify which audience you want your application to sign in; Microsoft Accounts, Azure AD, Azure AD B2C, ADFS etc.
- Supports Cross Broker Assisted SignOn with Azure AD and Microsoft Authenticator App.

Read more about MSAL here:

Overview of the Microsoft Authentication Library (MSAL)
https://learn.microsoft.com/en-us/entra/identity-platform/msal-overview

Microsoft identity platform authentication libraries
https://learn.microsoft.com/en-us/entra/identity-platform/reference-v2-libraries

5.5.2 Azure Active Directory Authentication Library (ADAL)

The ADAL library is deprecated and is end of support. Applications using ADAL must be migrated to use the MSAL library.

Read more here:
https://learn.microsoft.com/en-us/entra/identity-platform/msal-migration

6 PKI SECURITY GUIDELINES

This chapter is written to ensure that Avinor, together with 3rd party organizations, utilizes PKI security according to best practices during federation. This chapter also describes Avinor's requirement to PKI and PKI configuration that is unsupported. PKI Security can also be referred to "Certificate Security". Each implementation intending to utilize federation must follow these guidelines.

6.1 Certificate Requirements

All certificates used during federation shall be based on the requirements described according to the PKI Security Guidelines chapter of this document. The table "Web Application Trust" describes certificate requirements between Avinor "Security Token Service (STS)" and the "Web Application". The table "Security Token trust" describes the certificate requirements between Avinor Security Token Service and 3rd party organization security token service (STS), example ADFS.

Web Application Trust:

Certificate Type	Self-Signed	Private Issued CA	Public Issued CA
Web Application HTTPS	Not Supported	Not Supported	Required
Security Token Signing Certificate	Supported	Recommended	Optional
Security Token Encryption Certificate	Not recommended	Recommended	Recommended
Security Token Service Certificate	Not Supported	Not Supported	Required

Table 13.

Security Token Trust

Certificate Type	Self-Signed	Private Issued CA	Public Issued CA
Security Token Signing Certificate	Supported	Supported	Recommended
Security Token Encryption Certificate	Supported	Supported	Recommended
Security Token Service Certificate	Not Supported	Not Supported	Required

Table 14.

6.2 PKI Secure Channel Protocols

There are many types of PKI Secure Channel Protocols (also known as Schannel) that can be utilized when transporting/encrypting/signing data with certificates, where the goal is to protect the data and the certificate. Some protocols are tagged as "End of Life", but very often enabled as default on servers by legacy reasons. Below table documents Avinor's requirements to these PKI Protocols.

PKI Schannel Protocol Name	Presence to client	Comment
TLS 1.3	Should be Enabled	Preferred Protocol
TLS 1.2	Should be Enabled	Should be enabled for backward compatibility
TLS 1.1	DISABLED	Protocol is End of Life
TLS 1.0	DISABLED	Protocol is End of Life
SSL 3.0	DISABLED	Protocol is End of Life
SSL 2.0	DISABLED	Protocol is End of Life
PCT 1.0	DISABLED	Protocol is End of Life
Multi-Protocol Unified Hello	DISABLED	Protocol is End of Life

Table 15.

6.3 PKI Security Test

Avinor requires that all web applications federated with Avinor “Security Token Service (STS)” must comply with certificate security test with rate Grade A at “Qualys SSL Labs” (https://www.ssllabs.com/ssltest/).

Security remediation must be carried out before federation can be establish if the Grade is less than Grade A.

By following the PKI Security Guidelines in this chapter, will result in Grade A at “Qualys SSL Labs”.

Please read more here
https://github.com/ssllabs/research/wiki/SSL-Server-Rating-Guide

6.4 PKI Cipher Suites

Cipher Suite is a set of cryptographic algorithms that "Secure Channel" protocols (e.g. TLS) uses to create keys and encrypt information. A cipher suite specifies one algorithm for each of the following tasks:

- Key Exchange
- Signature / Certificate type
- Bulk encryption
- Message authentication

A chiper suite is built upon this formatting:

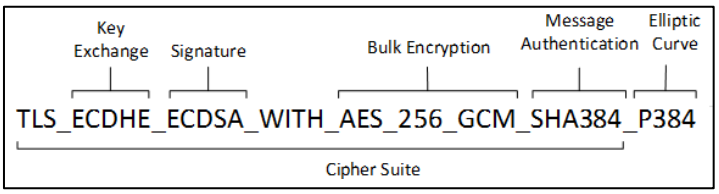


Figure 7.

Read and learn more here:
https://learn.microsoft.com/en-us/windows/win32/secauthn/cipher-suites-in-schannel

Key Exchange	Status
ECDHE	Preferred, Recommended and Supported
RSA	Not Supported by Avinor
DES	Not Supported by Avinor
NULL	Not Supported by Avinor

Table 16.

Signature / Certificate Type	Status
ECDHE	Preferred, Recommended and Supported
RSA	Preferred, Recommended and Supported

Table 17.

Message Authentication / HASH	Status
SHA 384	Preferred, Recommended and Supported
SHA 256	Preferred, Recommended and Supported
SHA 1	Not Supported by Avinor

Table 18.

Bulk Encryption	Status
GCM	Preferred, Recommended and Supported
CBC	Avoid usage – but supported for fallback.
Tripple DES	Not Supported by Avinor
DES	Not Supported by Avinor
RC4	Not Supported by Avinor
RC2	Not Supported by Avinor
Diffie-Hellman	Not Supported by Avinor
NULL	Not Supported by Avinor

Table 19.

6.4.1 TLS 1.3 Ciphers Suites

Below table documents Avinor preferred and supported Cipher Suites for TLS 1.3.

IANA	Cipher Suites	Protocol
0x1302	TLS_AES_256_GCM_SHA384	TLS 1.3
0x1301	TLS_AES_128_GCM_SHA256	TLS 1.3

Table 20.

6.4.2 TLS 1.2 Ciphers Suites

Below table documents Avinor preferred and supported Cipher Suites for TLS 1.2

IANA	Cipher Suites	Protocol	Cert Type	Support
0xc02c	TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384_P521*	TLS 1.2	ECC	Yes
0xc02c	TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384_P384	TLS 1.2	ECC	Yes
0xc024	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384_P521*	TLS 1.2	ECC	Avoid **
0xc024	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384_P384	TLS 1.2	ECC	Avoid **
0xc02b	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256_P521*	TLS 1.2	ECC	Yes
0xc02b	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256_P384	TLS 1.2	ECC	Yes
0xc02b	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256_P256	TLS 1.2	ECC	Yes
0xc023	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256_P521*	TLS 1.2	ECC	Avoid **
0xc023	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256_P384	TLS 1.2	ECC	Avoid **
0xc023	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256_P256	TLS 1.2	ECC	Avoid **
0xc028	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384_P521*	TLS 1.2	RSA	Avoid **
0xc028	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384_P384*	TLS 1.2	RSA	Avoid **
0xc028	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384_P256*	TLS 1.2	RSA	Avoid **
0xc027	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256_P521*	TLS 1.2	RSA	Avoid **
0xc027	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256_P384*	TLS 1.2	RSA	Avoid **
0xc027	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256_P256	TLS 1.2	RSA	Avoid **

Table 21.

ECC = Elliptic Curve Cryptography (ECC). ECC ciphers can only be used with ECC certificates.
RSA = Rivest-Shamir-Adleman Cryptography (RSA). RSA can only be used with RSA Certificates, this is the most used common cryptographic provider for public issued certificates.

* = The cipher is not enabled by default in Microsoft Operative System, but is supported, ref: https://msdn.microsoft.com/en-us/library/windows/desktop/aa374757(v=vs.85).aspx

** = These chippers is supported but considered as weak ciphers and should be avoided. https://learn.microsoft.com/en-us/dotnet/standard/security/vulnerabilities-cbc-mode

7 APPENDIX - ADFS FEDERATIONMETADATA

7.1 Sample Code - Update Web.Config ADFS FederationMetaData

Below Sample PowerShell Code, can be used to maintain the federation trust in an IIS web server "Web.Config" by monitoring the Token Signing Certificate presented ADFS FederationMetaData.xml. This script can / should be run daily as e.g. Scheduled Task. This sample code is created to system running IIS where the federation trust is configured in the web.config. The script enumerates all web sites and updates all web sites federated with ADFS.

--- PowerShell Sample Code Start ---

```
# ================================================================================================
#
# Microsoft PowerShell Source File -- Created with SAPIEN Technologies PrimalScript 2011
#
# NAME: ADFS - Update FederationMetaData with Certificate.ps1
#
# AUTHOR: Anders Horgen , Avinor
# DATE  : 2016.01.05
# Version: 1.0.3
# COMMENT:
#
#  ================================================================================================
#'=== CHANGE LOG                                                                                 ===
#'================================================================================================
#'=== DATE                        Version                 By                              Comment
                                                      ===
#'=== 2016.01.29        1.0.0           Anders Horgen       Orginal VBScript.                               ===
#'=== 2017.01.05        1.0.2           Anders Horgen       Fixed multiple entires in AppSettings           ===
#'=== 2017.05.11        1.0.3           Anders Horgen       Added Write-Log                                 ===
#'================================================================================================

CLS

# === Functions ================================================
Function Write-Log($Append = $true,$Text){
    $LogFile = "C:\ADFS_Federation_Monitoring\ADFS_Federation_Monitoring_IIS.log"
    $FileSize = ((Get-Item -Path $LogFile -ErrorAction SilentlyContinue ).Length / 1MB)
    $FileSizeMaximumMB = 2

    If ($FileSize -gt $FileSizeMaximumMB)
    {
        #OverWrite Logfile if it is over 2MB
        Remove-Item -Path $LogFile -Force -ErrorAction SilentlyContinue
        $Logtime = Get-Date -DisplayHint time -Format 'dd.MM.yyyy HH.mm.ss'
        if($Append){"$Logtime - $Text" | Out-File -FilePath $LogFile -Append -encoding default}
        else{"$Logtime - $Text" | Out-File -FilePath $LogFile -encoding default}

    }
    ELSE
    {
        $Logtime = Get-Date -DisplayHint time -Format 'dd.MM.yyyy HH.mm.ss'
        if($Append){"$Logtime - $Text" | Out-File -FilePath $LogFile -Append -encoding default}
        else{"$Logtime - $Text" | Out-File -FilePath $LogFile -encoding default}
    }

} #End Write-Log

Write-Log -Text "============================== Script Starting - IIS DinERP ================================="

# === Web Config Path ==========================================
#$Path = 'D:\InetPub\WebSites\Site01\wwwroot\ClaimApp\web.config'
#$Path = 'D:\InetPub\WebSites\Site03\wwwroot\web.config'

#$Path = 'C:\Temp\Web.Config'

Import-Module webAdministration
$IIS_Paths = Get-ChildItem IIS:\Sites\

foreach ($IIS_Path in $IIS_Paths)
{

    $WebSiteName = $IIS_Path.name
    $Path = $IIS_Path.physicalPath + '\web.config'


    Write-Host "Starting to Configure following IIS Site: " -NoNewLine; Write-Host "$WebSiteName" -ForegroundColor Yellow
    Write-Host  "=============================================================================================================="  -ForegroundColor  Red

    Write-Log -Text "Starting to Configure following IIS Site: $WebSiteName"
    Write-Log -Text  "--------------------------------------------------------------------------------------------------------------"

    $Web_Config_XML = [XML](Get-Content $Path)

    $dotNet3_5_Exist  =  $Web_Config_XML.configuration.'microsoft.identityModel'.service.issuerNameRegistry.trustedIssuers.add.count
    $dotNet4_0_Exist  =  $Web_Config_XML.configuration.'system.identityModel'.identityConfiguration.issuerNameRegistry.trustedIssuers.add.count




    If ($dotNet3_5_Exist -eq 0 -and $dotNet4_0_Exist -eq 0)
    {
        Write-Host "Web Site trust ADFS: " -NoNewLine; Write-Host "No" -ForegroundColor Yellow
        Write-Host ""

         Write-Log -Text "Web Site trust ADFS: No"
         Write-Log -Text "_"
    }
    ELSE
    {


    If ($dotNet3_5_Exist -ne 0)
       {
        Write-Host "IIS Web Site uses following .Net Framework version: " -NoNewLine; Write-Host "3.5" -ForegroundColor Yellow
        Write-Log -Text "IIS Web Site uses following .Net Framework version: 3.5"

        $Web_Config_Current_Trusted_ADFS_Signing_TrustedIssuer_Path  =  $Web_Config_XML.configuration.'microsoft.identityModel'.service.issuerNameRegistry.trustedIssuers
       }
       ELSE
       {
            Write-Host ""
            Write-Log -Text "_"
       }


    If ($dotNet4_0_Exist -ne 0)
       {
```

```
        Write-Host "IIS Web Site uses following .Net Framework version: " -NoNewLine; Write-Host "4.x" -ForegroundColor Yellow
        Write-Log -Text "IIS Web Site uses following .Net Framework version: 4.x"

        $Web_Config_Current_Trusted_ADFS_Signing_TrustedIssuer_Path  =  $Web_Config_XML.configuration.'system.identityModel'.identityConfiguration.issuerNameRegistry.trustedIssuers
    }
    ELSE
    {
        Write-Host ""
        Write-Log -Text "_"
    }

    Write-Host "Web Site trust ADFS: " -NoNewLine; Write-Host "Yes" -ForegroundColor Yellow
    Write-Log -Text "Web Site trust ADFS: Yes"


    $Web_Config_Current_Trusted_ADFS_Signing_Path  =  @($Web_Config_Current_Trusted_ADFS_Signing_TrustedIssuer_Path.add)



        # === Get ADFS FederationMetaData.xml ==============================
        $ADFS_FedXML_URL = ($Web_Config_XML.configuration.appSettings.add | Where-Object {$_.Value -like '*FederationMetaData.xml*'}).value
        [XML]$ADFS_FedXML = (New-Object System.Net.WebClient).DownloadString($ADFS_FedXML_URL)
        Write-Host "ADFS FederationMetaData Path: " -NoNewLine; Write-Host "$ADFS_FedXML_URL" -ForegroundColor Yellow
        Write-Log -Text "ADFS FederationMetaData Path: $ADFS_FedXML_URL"

        # === Get ADFS Signing Certificate and convert to Thumbprint ========
        $ADFS_FedXML_Cert_Signing_Path = @($ADFS_FedXML.EntityDescriptor.SPSSODescriptor.KeyDescriptor | where {$_.Use -like 'signing'})
        $ADFS_FedXML_Cert_Signing_Certificate_Count  =  $ADFS_FedXML_Cert_Signing_Path.Count

        If ($ADFS_FedXML_Cert_Signing_Certificate_Count -eq 2)
            {
                # === This section runs when ADFS has Primary and Secondary Signing Certificate presented in FederationMetaData.xml ===
                Write-Host "Contains 2 Signing Certificate"
                Write-Log -Text "Contains 2 Signing Certificate"

                $ADFS_FedXML_Cert_Signing_Certificate_Item1_Base64  =  $ADFS_FedXML_Cert_Signing_Path.Item(0).KeyInfo.X509Data.X509Certificate
                $ADFS_Signing_Certificate_Item1 =
[System.Security.Cryptography.X509Certificates.X509Certificate2]([System.Convert]::FromBase64String($ADFS_FedXML_Cert_Signing_Certificate_Item1_Base64))
                $ADFS_Signing_Certificate_Item1_Thumbprint  =  $ADFS_Signing_Certificate_Item1.Thumbprint

                $ADFS_FedXML_Cert_Signing_Certificate_Item2_Base64  =  $ADFS_FedXML_Cert_Signing_Path.Item(1).KeyInfo.X509Data.X509Certificate
                $ADFS_Signing_Certificate_Item2 =
[System.Security.Cryptography.X509Certificates.X509Certificate2]([System.Convert]::FromBase64String($ADFS_FedXML_Cert_Signing_Certificate_Item2_Base64))
                $ADFS_Signing_Certificate_Item2_Thumbprint  =  $ADFS_Signing_Certificate_Item2.Thumbprint

                $ADFS_Signing_Certificate_Item1
                $ADFS_Signing_Certificate_Item2
                Write-Host ""
                Write-Host "Ensure that web.config contains current ADFS Signing Certificate: " -NoNewline; Write-Host "$ADFS_Signing_Certificate_Item1_Thumbprint" -ForegroundColor Yellow
                Write-Host "Ensure that web.config contains current ADFS Signing Certificate: " -NoNewline; Write-Host "$ADFS_Signing_Certificate_Item2_Thumbprint" -ForegroundColor Yellow
                Write-Host ""

                Write-Log -Text "_"
                Write-Log -Text "Ensure that web.config contains current ADFS Signing Certificate: $ADFS_Signing_Certificate_Item1_Thumbprint"
                Write-Log -Text "Ensure that web.config contains current ADFS Signing Certificate: $ADFS_Signing_Certificate_Item2_Thumbprint"
                Write-Log -Text "_"


                $Web_Config_Current_Trusted_ADFS_Signing_Count  =  $Web_Config_Current_Trusted_ADFS_Signing_Path.Count

                # === This section check if Web.Config contains 1 or 2 ADFS signing cert. Ensure that only Primary and Secondary is registered.
                If ($Web_Config_Current_Trusted_ADFS_Signing_Count -eq 1)
                    {
                        $New_Web_Config_Trusted_ADFS_Signing = $Web_Config_Current_Trusted_ADFS_Signing_Path.Item(0).Clone()
                        $New_Web_Config_Trusted_ADFS_Signing.thumbprint  =  $ADFS_Signing_Certificate_Item2.Thumbprint.ToString()
                        $Web_Config_Current_Trusted_ADFS_Signing_TrustedIssuer_Path.PrependChild($New_Web_Config_Trusted_ADFS_Signing)
                        $Web_Config_XML.Save($Path)
                    }
                    ELSE
                    {

                        $Web_Config_Current_Trusted_ADFS_Signing_TrustedIssuer_Path.add.item(0).thumbprint  =  $ADFS_Signing_Certificate_Item1.Thumbprint.ToString()
                        $Web_Config_Current_Trusted_ADFS_Signing_TrustedIssuer_Path.add.item(1).thumbprint  =  $ADFS_Signing_Certificate_Item2.Thumbprint.ToString()
                        $Web_Config_XML.Save($Path)
                    }

            }
        ELSE
            {
                # === This section runs when ADFS has only Primary Signing Certificate presented in FederationMetaData.xml ===
                Write-Host "ADFS Contains 1 Signing Certificate, see below:"
                Write-Log -Text "ADFS Contains 1 Signing Certificate, see below:"

                $ADFS_FedXML_Cert_Signing_Certificate_Item1_Base64  =  $ADFS_FedXML_Cert_Signing_Path.Item(0).KeyInfo.X509Data.X509Certificate
                $ADFS_Signing_Certificate_Item1 =
[System.Security.Cryptography.X509Certificates.X509Certificate2]([System.Convert]::FromBase64String($ADFS_FedXML_Cert_Signing_Certificate_Item1_Base64))
                $ADFS_Signing_Certificate_Item1

                $Web_Config_Current_Trusted_ADFS_Signing_Count  =  $Web_Config_Current_Trusted_ADFS_Signing_Path.Count

                # === Check if Web.Config contains more than 1 Trusted Issuers ADFS Signing Certificate, remove Secondary when removed from ADFS ===
                If ($Web_Config_Current_Trusted_ADFS_Signing_Count -ne 1)
                    {
                        # ==== Loop and remove all entries of Trusted Issuers in web.config except for 1 entry in the array.
                        DO {

                            Write-host "Web.config contains old Trusted ADFS Signing Certificates - removing them..."
                            Write-host $Web_Config_Current_Trusted_ADFS_Signing_Count

                            Write-Log -Text "Web.config contains old Trusted ADFS Signing Certificates - removing them..."
                            Write-Log -Text "$Web_Config_Current_Trusted_ADFS_Signing_Count"


$Web_Config_Current_Trusted_ADFS_Signing_TrustedIssuer_Path.RemoveChild($Web_Config_Current_Trusted_ADFS_Signing_TrustedIssuer_Path.add.item($Web_Config_Current_Trusted_ADFS_Signing_Count
-1))
                            $Web_Config_Current_Trusted_ADFS_Signing_Count  =  $Web_Config_Current_Trusted_ADFS_Signing_Count  -1

                           } while ($Web_Config_Current_Trusted_ADFS_Signing_Count -ne 1)


                        $ADFS_FedXML_Cert_Signing_Certificate_Item1_Base64  =  $ADFS_FedXML_Cert_Signing_Path.Item(0).KeyInfo.X509Data.X509Certificate
                        $ADFS_Signing_Certificate_Item1 =
[System.Security.Cryptography.X509Certificates.X509Certificate2]([System.Convert]::FromBase64String($ADFS_FedXML_Cert_Signing_Certificate_Item1_Base64))
                        $ADFS_Signing_Certificate_Item1_Thumbprint  =  $ADFS_Signing_Certificate_Item1.Thumbprint.ToString()

                        $Web_Config_Current_Trusted_ADFS_Signing_TrustedIssuer_Path.add.thumbprint  =  $ADFS_Signing_Certificate_Item1_Thumbprint
                        $Web_Config_XML.Save($Path)



                    }
                ELSE
                    {

                        # === Ensure that web.config contains current ADFS Signing Certificate ===
                        $ADFS_FedXML_Cert_Signing_Certificate_Item1_Base64  =  $ADFS_FedXML_Cert_Signing_Path.Item(0).KeyInfo.X509Data.X509Certificate
                        $ADFS_Signing_Certificate_Item1 =
[System.Security.Cryptography.X509Certificates.X509Certificate2]([System.Convert]::FromBase64String($ADFS_FedXML_Cert_Signing_Certificate_Item1_Base64))
                        $ADFS_Signing_Certificate_Item1_Thumbprint = $ADFS_Signing_Certificate_Item1.Thumbprint.ToString()
                        Write-Host ""
                        Write-Host "Ensure that web.config contains current ADFS Signing Certificate: " -NoNewline; Write-Host "$ADFS_Signing_Certificate_Item1_Thumbprint" -
ForegroundColor Yellow
                        Write-Host ""
```

```
                        Write-Log -Text "_"
                        Write-Log -Text  "Ensure that web.config contains current ADFS Signing Certificate: $ADFS_Signing_Certificate_Item1_Thumbprint"
                        Write-Log -Text "_"

                        $Web_Config_Current_Trusted_ADFS_Signing_TrustedIssuer_Path.add.thumbprint  =  $ADFS_Signing_Certificate_Item1_Thumbprint
                        $Web_Config_XML.Save($Path)
                    }
                }
            }

    # Pause
        }
    Write-Log -Text "============================ Script Stopped ================================="
```

--- PowerShell Sample Code Stop ---

7.2 Sample Code - Update SharePoint ADFS FederationMetaData

Below Sample PowerShell Code, can be used to maintain the federation trust in "Microsoft SharePoint Server 2013" by monitoring the "Primary Token Signing Certificate" presented in ADFS FederationMetaData.xml.

This script can / should be run every 15minutes as e.g. Scheduled Task. Microsoft SharePoint 2013 server do not currently support registration of ADFS Secondary Token Signing Certificate, hence that is why it is crucial that this script must run frequent to avoid service interruption when ADFS renew its Token Signing Certificate by promoting the Secondary Token Signing Certificate to be used as Primary. Ref previous chapter ("Avinor's Token Signing Certificate") in this document that explains this process. The service downtime trust interruption here will 15minutes, when the Token Signing Certificate is changed.

--- PowerShell Sample Code Start ---

```
# ================================================================================================
#
# Microsoft PowerShell Source File -- Created with Windows PowerShell ISE
#
# NAME: ADFS - Update SharePoint from ADFS FederationMetaData - v1.0.4.ps1
#
# AUTHOR: Anders Horgen , Avinor
# DATE  : 2017.05.11
# Version: 1.0.4
# COMMENT:
#
#  ================================================================================================
#'=== CHANGE LOG                                                                                 ===
#'================================================================================================
#'=== DATE                    Version                 By                          Comment
#                                                    ==
#'=== 2016.03.16       1.0.0          Anders Horgen      Orginal VBScript.                        ===
#'=== 2016.03.17    1.0.1       Anders Horgen  Added Trusted Root Authority                  ===
#'=== 2016.06.09    1.0.2       Anders Horgen   Ensured that Download of XML don't use proxy  ===
#'=== 2016.06.10    1.0.3      Anders Horgen   Fixed detection of correct Primary Token Signing Cert ===
#'=== 2017.05.11   1.0.4     Anders Horgen  Added Logging and IISRESET                        ===
#'================================================================================================

cls
Add-PSSnapin "Microsoft.SharePoint.PowerShell"
# === Functions ================================
Function Write-Log($Append = $true,$Text){
    $LogFile = "D:\ADFS_Federation_Monitoring\ADFS_Federation_Monitoring_SharePoint.log"
    $FileSize = ((Get-Item -Path $LogFile -ErrorAction SilentlyContinue ).Length / 1MB)
    $FileSizeMaximumMB = 2

    If ($FileSize -gt $FileSizeMaximumMB)
    {
        #OverWrite Logfile if it is over 2MB
        Remove-Item -Path $LogFile -Force -ErrorAction SilentlyContinue
        $Logtime = Get-Date -DisplayHint time -Format 'dd.MM.yyyy HH.mm.ss'
        if($Append){"$Logtime - $Text" | Out-File -FilePath $LogFile -Append -encoding default}
        else{"$Logtime - $Text" | Out-File -FilePath $LogFile -encoding default}

    }
    ELSE
    {
        $Logtime = Get-Date -DisplayHint time -Format 'dd.MM.yyyy HH.mm.ss'
        if($Append){"$Logtime - $Text" | Out-File -FilePath $LogFile -Append -encoding default}
        else{"$Logtime - $Text" | Out-File -FilePath $LogFile -encoding default}
    }

} #End Write-Log

Write-Log -Text "============================== Script Starting - SharePoint DinERP ================================="

# === Declare Variables ================================

$SP_Servers = @(Get-SPServer | Select-Object Address, Role | Where-Object {$_.Role -like 'Application'})
$SP_ServersCount = $SP_Servers.count

Write-Log -Text "This SharePoint Farm consist of following Servers with the Application Role"
Foreach ($SP_Server in $SP_Servers)
{
    $SP_Server_String = $SP_Server.Address
    Write-Log -Text "$SP_Server_String"

}
Write-Log -Text "_"

$SP_Trusted_Identity_Token_Issuer = Get-SPTrustedIdentityTokenIssuer
$SP_Trusted_Identity_Token_Issuer_ADFS_URL = "https://" + ($SP_Trusted_Identity_Token_Issuer.ProviderUri.Authority) + "/FederationMetadata/2007-06/FederationMetadata.xml"

#'================================================================================================
#Below line is for debugging only
#$SP_Trusted_Identity_Token_Issuer_ADFS_URL = "https://adfs.avinor.no/FederationMetadata/2007-06/FederationMetaData.xml"
#'================================================================================================

$SP_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint = $SP_Trusted_Identity_Token_Issuer.SigningCertificate.Thumbprint
$SP_Trusted_Root_Authority = Get-SPTrustedRootAuthority
$SP_Trusted_Root_Authority_Object = $SP_Trusted_Root_Authority | Where-Object {$_.Certificate.Thumbprint -like $SP_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint}


        # === Get ADFS FederationMetaData.xml ================================
        $ADFS_FedXML_URL = $SP_Trusted_Identity_Token_Issuer_ADFS_URL

        $WebClient = New-Object System.Net.WebClient
        $WebClient.Proxy = [System.Net.GlobalProxySelection]::GetEmptyWebProxy()
        [XML]$ADFS_FedXML = $WebClient.DownloadString($ADFS_FedXML_URL)
        Write-Host "ADFS FederationMetaData Path: " -NoNewLine; Write-Host "$ADFS_FedXML_URL" -ForegroundColor Yellow
        Write-Log -Text "ADFS FederationMetaData Path: $ADFS_FedXML_URL"

        # === Get ADFS Signing Certificate and convert to Thumbprint ========
        $ADFS_FedXML_Primary_Token_Signing_Cert =
[System.Security.Cryptography.X509Certificates.X509Certificate2]([System.Convert]::FromBase64String($ADFS_FedXML.EntityDescriptor.Signature.KeyInfo.X509Data.X509Certificate))
        $ADFS_FedXML_Primary_Token_Signing_Cert_Thumbprint = $ADFS_FedXML_Primary_Token_Signing_Cert.Thumbprint

        $ADFS_FedXML_Cert_Signing_Path = @($ADFS_FedXML.EntityDescriptor.SPSSODescriptor.KeyDescriptor | where {$_.Use -like 'signing'})
        $ADFS_FedXML_Cert_Signing_Certificate_Count = $ADFS_FedXML_Cert_Signing_Path.Count



        If ($ADFS_FedXML_Cert_Signing_Certificate_Count -eq 2)
```

```
            {
                # === This section runs when ADFS has Primary and Secondary Signing Certificate presented in FederationMetaData.xml ===
                Write-Host "Contains 2 Signing Certificate"
                Write-Host ""

                Write-Log -Text "Contains 2 Signing Certificate"
                Write-Log -Text "_"

                # === Enumerate alle Token Signing Certificate and Identify which certificate is Primary and Secondary Token Signing Certificate ===
                ForEach ($ADFS_FedXML_Token_Signing_Certificate_Base64 in $ADFS_FedXML_Cert_Signing_Path)
                {
                    $ADFS_FedXML_Token_Signing_Certificate =
[System.Security.Cryptography.X509Certificates.X509Certificate2]([System.Convert]::FromBase64String($ADFS_FedXML_Token_Signing_Certificate_Base64.KeyInfo.X509Data.X509Certificate))
                    $ADFS_FedXML_Token_Signing_Certificate_Thumbprint = $ADFS_FedXML_Token_Signing_Certificate.Thumbprint

                    IF ($ADFS_FedXML_Token_Signing_Certificate_Thumbprint -like $ADFS_FedXML_Primary_Token_Signing_Cert_Thumbprint)
                    {

                        $ADFS_Primary_Token_Signing_Certificate = $ADFS_FedXML_Token_Signing_Certificate
                        $ADFS_Primary_Token_Signing_Certificate_IssueDate = ($ADFS_Primary_Token_Signing_Certificate.NotBefore).ToString()
                        $ADFS_Primary_Token_Signing_Certificate_Subject = $ADFS_Primary_Token_Signing_Certificate.Subject
                        $ADFS_Primary_Token_Signing_Certificate_Thumbprint = $ADFS_Primary_Token_Signing_Certificate.Thumbprint
                        $ADFS_Primary_Token_Signing_Certificate_is_SelfSigned = If ($ADFS_Primary_Token_Signing_Certificate.Issuer -like
$ADFS_Primary_Token_Signing_Certificate.Subject){$True}ELSE{$False}

                        Write-Host "ADFS Contains Primary Token Signing Certificate: " -NoNewline; Write-Host "$ADFS_Primary_Token_Signing_Certificate_Thumbprint -
$ADFS_Primary_Token_Signing_Certificate_Subject - Issued: $ADFS_Primary_Token_Signing_Certificate_IssueDate" -ForegroundColor Yellow
                        Write-Log -Text "ADFS Contains Primary Token Signing Certificate: $ADFS_Primary_Token_Signing_Certificate_Thumbprint -
$ADFS_Primary_Token_Signing_Certificate_Subject - Issued: $ADFS_Primary_Token_Signing_Certificate_IssueDate"
                    }
                    ELSE
                    {

                        $ADFS_Secondary_Token_Signing_Certificate = $ADFS_FedXML_Token_Signing_Certificate
                        $ADFS_Secondary_Token_Signing_Certificate_IssueDate = ($ADFS_Secondary_Token_Signing_Certificate.NotBefore).ToString()
                        $ADFS_Secondary_Token_Signing_Certificate_Subject = $ADFS_Secondary_Token_Signing_Certificate.Subject
                        $ADFS_Secondary_Token_Signing_Certificate_Thumbprint = $ADFS_Secondary_Token_Signing_Certificate.Thumbprint
                        $ADFS_Secondary_Token_Signing_Certificate_is_SelfSigned = If ($ADFS_Secondary_Token_Signing_Certificate.Issuer -like
$ADFS_Secondary_Token_Signing_Certificate.Subject){$True}ELSE{$False}
                        Write-Host "ADFS Contains Secondary Token Signing Certificate: " -NoNewline; Write-Host "$ADFS_Secondary_Token_Signing_Certificate_Thumbprint -
$ADFS_Secondary_Token_Signing_Certificate_Subject - Issued: $ADFS_Secondary_Token_Signing_Certificate_IssueDate" -ForegroundColor Yellow
                        Write-Log -Text "ADFS Contains Secondary Token Signing Certificate: $ADFS_Secondary_Token_Signing_Certificate_Thumbprint -
$ADFS_Secondary_Token_Signing_Certificate_Subject - Issued: $ADFS_Secondary_Token_Signing_Certificate_IssueDate"

                    }

                }

                Write-Host ""
                Write-Log -Text "_"


                # === This section check if Sharepoint contains 1 or 2 ADFS signing cert. Ensure that only Primary Token Certificate is registered.
                If ($ADFS_Primary_Token_Signing_Certificate_Thumbprint -notlike $SP_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint)
                    {
                        $SP_Trusted_Identity_Token_Issuer | Set-SPTrustedIdentityTokenIssuer -ImportTrustCertificate $ADFS_Primary_Token_Signing_Certificate
                        $SP_Trusted_Root_Authority_Object | Set-SPTrustedRootAuthority -Certificate $ADFS_Primary_Token_Signing_Certificate

                        $New_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint = (Get-SPTrustedIdentityTokenIssuer).SigningCertificate.thumbprint
                        $New_Trusted_Root_Authority_Certificate = (Get-SPTrustedRootAuthority | Where-Object {$_.Name -like
($SP_Trusted_Root_Authority_Object.Name)}).certificate.thumbprint


                            Write-Host ""
                            Write-Host "SharePoint contains incorrect Token Signing Certificate, update SharePoint..."
                            Write-Host "SharePoint Primary Token Signing Certificate:       " -NoNewline; Write-Host
"$SP_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint" -ForegroundColor Yellow
                            Write-Host "SharePoint Root Signing Certificate:                " -NoNewline; Write-Host
"$SP_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint" -ForegroundColor Yellow
                            Write-Host "ADFS Primary Token Signing Certificate:             " -NoNewline; Write-Host "$ADFS_Primary_Token_Signing_Certificate_Thumbprint" -
ForegroundColor Yellow
                            Write-Host ""
                            Write-Host "SharePoint is updated to use Issuer Certificate:    "  -NoNewline; Write-Host
"$New_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint"  -ForegroundColor  Yellow
                            Write-Host "SharePoint is updated to use Issuer Root Certificate: "  -NoNewline; Write-Host "$New_Trusted_Root_Authority_Certificate" -ForegroundColor Yellow


                            Write-Log -Text "_"
                            Write-Log -Text "SharePoint contains incorrect Token Signing Certificate, update SharePoint..."
                            Write-Log -Text "SharePoint Primary Token Signing Certificate:                  $SP_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint"
                            Write-Log -Text "SharePoint Root Signing Certificate:                           $SP_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint"
                            Write-Log -Text "ADFS Primary Token Signing Certificate:                        $ADFS_Primary_Token_Signing_Certificate_Thumbprint"
                            Write-Log -Text ""
                            Write-Log -Text "SharePoint is updated to use Issuer Certificate:               $New_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint"
                            Write-Log -Text "SharePoint is updated to use Issuer Root Certificate:         $New_Trusted_Root_Authority_Certificate"
                            Write-Log -Text "_"

                        Write-Host "Performing IISRESET on all $SP_ServersCount servers"
                        Write-Host "-------------------------------------------------------------------------------------------"

                        Write-Log -Text "Performing IISRESET on $SP_ServersCount servers"
                        Write-Log -Text "-----------------------------------------------------------"
                        Foreach ($SP_Server in $SP_Servers)
                        {

                                $SPServerString = $SP_Server.address + "." + $env:USERDNSDOMAIN

                                Write-Host "Performing IISRESET on $SPServerString"
                                Write-Log -Text "Performing IISRESET on $SPServerString"

                                $Invoke_Status = Invoke-Expression "IISRESET.exe $SPServerString"
                                $Invoke_StatusExitCode = $LASTEXITCODE

                                Write-Host "IISRESET Exit Status: $Invoke_StatusExitCode"
                                Write-Host ""
                                Write-Log -Text "IISRESET Exit Status: $Invoke_StatusExitCode"
                                Write-Log -Text "_"

                        }


                    }
                    ELSE
                    {

                        Write-Host ""
                        Write-Host "No need to update SharePoint Token Signing Certificate, as correct certificate is registered."
                        Write-Host "SharePoint Primary Token Signing Certificate: " -NoNewline; Write-Host "$SP_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint" -
ForegroundColor Yellow
                        Write-Host "ADFS Primary Token Signing Certificate:       " -NoNewline; Write-Host "$ADFS_Primary_Token_Signing_Certificate_Thumbprint" -ForegroundColor Yellow
                        Write-Host ""

                        Write-Log -Text "_"
                        Write-Log -Text "No need to update SharePoint Token Signing Certificate, as correct certificate is registered."
                        Write-Log -Text "SharePoint Primary Token Signing Certificate: $SP_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint"
                        Write-Log -Text "ADFS Primary Token Signing Certificate:       $ADFS_Primary_Token_Signing_Certificate_Thumbprint"
                    }


            }
        ELSE
            {
                # === This section runs when ADFS has only Primary Signing Certificate presented in FederationMetaData.xml ===
                Write-Host "ADFS Contains 1 Signing Certificate, see below:"
```

```
                Write-Log -Text "ADFS Contains 1 Signing Certificate, see below:"

                $ADFS_Primary_Token_Signing_Certificate = $ADFS_FedXML_Primary_Token_Signing_Cert
                $ADFS_Primary_Token_Signing_Certificate_IssueDate = ($ADFS_Primary_Token_Signing_Certificate.NotBefore).ToString()
                $ADFS_Primary_Token_Signing_Certificate_Subject = $ADFS_Primary_Token_Signing_Certificate.Subject
                $ADFS_Primary_Token_Signing_Certificate_Thumbprint = $ADFS_Primary_Token_Signing_Certificate.Thumbprint
                $ADFS_Primary_Token_Signing_Certificate_is_SelfSigned = If ($ADFS_Primary_Token_Signing_Certificate.Issuer -like
$ADFS_Primary_Token_Signing_Certificate.Subject){$True}ELSE{$False}
                Write-Host "ADFS Contains Primary Token Signing Certificate: " -NoNewline; Write-Host "$ADFS_Primary_Token_Signing_Certificate_Thumbprint -
$ADFS_Primary_Token_Signing_Certificate_Subject - Issued: $ADFS_Primary_Token_Signing_Certificate_IssueDate" -ForegroundColor Yellow
                Write-Log -Text "ADFS Contains Primary Token Signing Certificate: $ADFS_Primary_Token_Signing_Certificate_Thumbprint - $ADFS_Primary_Token_Signing_Certificate_Subject -
Issued: $ADFS_Primary_Token_Signing_Certificate_IssueDate"

                # === Check if SharePoint contains more than 1 Trusted Issuers ADFS Signing Certificate, remove Secondary when removed from ADFS ===
                If ($ADFS_Primary_Token_Signing_Certificate_Thumbprint -notlike $SP_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint)
                    {

                        $SP_Trusted_Identity_Token_Issuer | Set-SPTrustedIdentityTokenIssuer -ImportTrustCertificate $ADFS_Primary_Token_Signing_Certificate
                        $SP_Trusted_Root_Authority_Object | Set-SPTrustedRootAuthority -Certificate $ADFS_Primary_Token_Signing_Certificate

                        $New_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint = (Get-SPTrustedIdentityTokenIssuer).SigningCertificate.thumbprint
                        $New_Trusted_Root_Authority_Certificate = (Get-SPTrustedRootAuthority | Where-Object {$_.Name -like
($SP_Trusted_Root_Authority_Object.Name)}).certificate.thumbprint


                        Write-Host ""
                        Write-Host "SharePoint contains incorrect Token Signing Certificate, update SharePoint..."
                        Write-Host "SharePoint Primary Token Signing Certificate:          " -NoNewline; Write-Host
"$SP_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint" -ForegroundColor Yellow
                        Write-Host "SharePoint Root Signing Certificate:                   " -NoNewline; Write-Host
"$SP_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint" -ForegroundColor Yellow
                        Write-Host "ADFS Primary Token Signing Certificate:                " -NoNewline; Write-Host "$ADFS_Primary_Token_Signing_Certificate_Thumbprint" -ForegroundColor
Yellow
                        Write-Host ""
                        Write-Host "SharePoint is updated to use Issuer Certificate:       "  -NoNewline; Write-Host
"$New_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint" -ForegroundColor Yellow
                        Write-Host "SharePoint is updated to use Issuer Root Certificate: "  -NoNewline; Write-Host "$New_Trusted_Root_Authority_Certificate" -ForegroundColor Yellow

                        Write-Log -Text "_"
                        Write-Log -Text "SharePoint contains incorrect Token Signing Certificate, update SharePoint..."
                        Write-Log -Text "SharePoint Primary Token Signing Certificate:                  $SP_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint"
                        Write-Log -Text "SharePoint Root Signing Certificate:                           $SP_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint"
                        Write-Log -Text "ADFS Primary Token Signing Certificate:                        $ADFS_Primary_Token_Signing_Certificate_Thumbprint"
                        Write-Log -Text "_"
                        Write-Log -Text "SharePoint is updated to use Issuer Certificate:               $New_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint"
                        Write-Log -Text "SharePoint is updated to use Issuer Root Certificate:         $New_Trusted_Root_Authority_Certificate"
                        Write-Log -Text "_"


                        Write-Host "Performing IISRESET on all $SP_ServersCount servers"
                        Write-Host "-------------------------------------------------------------------------------------------"

                        Write-Log -Text "Performing IISRESET on $SP_ServersCount servers"
                        Write-Log -Text "-------------------------------------------------------------"
                        Foreach ($SP_Server in $SP_Servers)
                        {

                                $SPServerString = $SP_Server.address + "." + $env:USERDNSDOMAIN

                                Write-Host "Performing IISRESET on $SPServerString"
                                Write-Log -Text "Performing IISRESET on $SPServerString"

                                $Invoke_Status = Invoke-Expression "IISRESET.exe $SPServerString"
                                $Invoke_StatusExitCode = $LASTEXITCODE

                                Write-Host "IISRESET Exit Status: $Invoke_StatusExitCode"
                                Write-Host ""
                                Write-Log -Text "IISRESET Exit Status: $Invoke_StatusExitCode"
                                Write-Log -Text "_"

                        }


                    }
                ELSE
                    {

                        Write-Host ""
                        Write-Host "No need to update SharePoint Token Signing Certificate, as correct certificate is registered."
                        Write-Host "SharePoint Primary Token Signing Certificate: " -NoNewline; Write-Host "$SP_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint" -
ForegroundColor Yellow
                        Write-Host "ADFS Primary Token Signing Certificate:       " -NoNewline; Write-Host "$ADFS_Primary_Token_Signing_Certificate_Thumbprint" -ForegroundColor Yellow
                        Write-Host ""

                        Write-Log -Text "_"
                        Write-Log -Text "No need to update SharePoint Token Signing Certificate, as correct certificate is registered."
                        Write-Log -Text "SharePoint Primary Token Signing Certificate: $SP_Trusted_Identity_Token_Issuer_Token_Primary_Signing_Cert_Thumbprint"
                        Write-Log -Text "ADFS Primary Token Signing Certificate:       $ADFS_Primary_Token_Signing_Certificate_Thumbprint"
                        Write-Log -Text "_"

                    }
            }
Write-Log -Text "============================== Script Stopped ================================"
Write-Log -Text "_"
```

--- PowerShell Sample Code Stop ---

7.3 Sample Code – Update .NET App ADFS FederationMetaData

Below Sample .Net Framework 4 Code, can be used to maintain the federation trust in ".Net Web Application" running on "Internet Information Services (IIS) by monitoring the "Primary Token Signing Certificate" presented in ADFS FederationMetaData.xml. This is a more elegant automated routine build in .Net rather than the example explained in "6.1 Sample Code - Update Web.Config ADFS FederationMetaData" chapter in this document. The web application inside IIS will read the ADFS FederationMetDataLocation during startup of the application.

PS! The full .Net Code is also posted in GitHub at https://gist.github.com/hallatore/85532ecc62caa8023843ef6dcc72d284 (please see full details here)

--- Global.asax.cs Code Start ---

```
1  protected void Application_Start()
2  {
3  TrustedIssuers.ScheduleUpdate(ConfigurationManager.AppSettings["FederationMetadataLocation"]);
4  }
```

--- Global.asax.cs Sample Code Stop ---

--- web.config Code Start ---

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings>
    <add key="FederationMetadataLocation" value="https://sts.avinor.no/FederationMetadata/2007-06/FederationMetadata.xml" />
  </appSettings>
  </system.serviceModel>
  <system.identityModel>
    <identityConfiguration>
      <audienceUris>
        <add value="https://eks.avinor.no/" />
      </audienceUris>
      <certificateValidation certificateValidationMode="None" />
      <securityTokenHandlers>
        <remove type="System.IdentityModel.Tokens.SessionSecurityTokenHandler, System.IdentityModel, Version=4.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089" />
        <add type="System.IdentityModel.Services.Tokens.MachineKeySessionSecurityTokenHandler, System.IdentityModel.Services,
Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" />
      </securityTokenHandlers>
    </identityConfiguration>
  </system.identityModel>
  <system.identityModel.services>
    <federationConfiguration>
      <cookieHandler requireSsl="true" />
      <wsFederation passiveRedirectEnabled="true" issuer="https://sts.avinor.no/adfs/ls/" realm="https://YourAppURL/" requireHttps="true" />
      <serviceCertificate>
        <certificateReference findValue="<SigningThumbprint>" storeLocation="LocalMachine" storeName="My"
x509FindType="FindByThumbprint" />
      </serviceCertificate>
    </federationConfiguration>
  </system.identityModel.services>
</configuration>
```

--- web.config Sample Code Stop ---

8 APPENDIX – PKI

8.1 Sample Code – Configure Cipher Suites for Microsoft Windows

Below sample code can be used to configure usage of correct Cipher Suites, according to Avinor “PKI Security Guideline”

--- PowerShell Sample Code Start ---

```
# ================================================================================
#
# Microsoft PowerShell Source File -- Created with SAPIEN Technologies PrimalScript 2011
#
# NAME: Avinor_ChiperSuite.ps1
#
# AUTHOR: Anders Horgen , Avinor
# DATE  : 06.04.2016
# Version: 1.0.0
# COMMENT: This scrip shall only be run on Windows Servers and not at Windows Clients.
#          Running this script on Windows Client, will result in that clients can't access public services
#          that uses chiper suites that Avinor can't decide to use or not.
#
# Referance: https://technet.microsoft.com/en-us/library/dn786418.aspx
#            https://support.microsoft.com/en-us/kb/245030
#
#  ============================================================================================
#'=== CHANGE LOG                                                                                                                  ===
#'============================================================================================
#'=== DATE                          Version                 By                                      Comment
                                                        ===
#'=== 2016.04.06        1.0.0           Anders Horgen       Orginal VBScript.                                                       ===
#'============================================================================================

# ================================================================================
# === Configure SCHANNEL Protocols ===============================================
#  ================================================================================

    # === Ensure SCHANNEL RegPath Exist ==========================================
    $Reg_KeyExch_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL"
    New-Item -Path ($Reg_KeyExch_Path) -Force -ErrorAction SilentlyContinue

    # === Ensure SCHANNEL CipherSuites RegPath Exist =============================================
    $Reg_KeyExch_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\CipherSuites"
    New-Item -Path ($Reg_KeyExch_Path) -Force -ErrorAction SilentlyContinue

        # === Disable Multi-Protocol Unified Hello ===============================================
        $Reg_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\Multi-Protocol Unified Hello"
        New-Item -Path $Reg_Path -Force -ErrorAction SilentlyContinue
        New-Item -Path ($Reg_Path + "\Server") -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Server") -Name 'DisabledByDefault' -Value 1 -PropertyType DWORD -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Server") -Name 'Enabled' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue

        # === Disable PCT 1.0 ====================================================================
        $Reg_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\PCT 1.0"
        New-Item -Path $Reg_Path -Force -ErrorAction SilentlyContinue
        New-Item -Path ($Reg_Path + "\Server") -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Server") -Name 'DisabledByDefault' -Value 1 -PropertyType DWORD -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Server") -Name 'Enabled' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue

        # === Disable SSL 2.0 ====================================================================
        $Reg_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SSL 2.0"
        New-Item -Path $Reg_Path -Force -ErrorAction SilentlyContinue
        New-Item -Path ($Reg_Path + "\Client") -Force -ErrorAction SilentlyContinue
        New-Item -Path ($Reg_Path + "\Server") -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Client") -Name 'Enabled' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Server") -Name 'Enabled' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Client") -Name 'DisabledByDefault' -Value 1 -PropertyType DWORD -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Server") -Name 'DisabledByDefault' -Value 1 -PropertyType DWORD -Force -ErrorAction SilentlyContinue

        # === Disable SSL 3.0 ====================================================================
        $Reg_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SSL 3.0"
        New-Item -Path $Reg_Path -Force -ErrorAction SilentlyContinue
        New-Item -Path ($Reg_Path + "\Client") -Force -ErrorAction SilentlyContinue
        New-Item -Path ($Reg_Path + "\Server") -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Client") -Name 'Enabled' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Server") -Name 'Enabled' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Client") -Name 'DisabledByDefault' -Value 1 -PropertyType DWORD -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Server") -Name 'DisabledByDefault' -Value 1 -PropertyType DWORD -Force -ErrorAction SilentlyContinue


        # === Enable TLS 1.0 =====================================================================
        $Reg_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.0"
        New-Item -Path $Reg_Path -Force -ErrorAction SilentlyContinue
        New-Item -Path ($Reg_Path + "\Client") -Force -ErrorAction SilentlyContinue
        New-Item -Path ($Reg_Path + "\Server") -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Client") -Name 'Enabled' -Value 1 -PropertyType DWORD -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Server") -Name 'Enabled' -Value 1 -PropertyType DWORD -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Client") -Name 'DisabledByDefault' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Server") -Name 'DisabledByDefault' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue

        # === Enable TLS 1.1 =====================================================================
        $Reg_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.1"
        New-Item -Path $Reg_Path -Force -ErrorAction SilentlyContinue
        New-Item -Path ($Reg_Path + "\Client") -Force -ErrorAction SilentlyContinue
        New-Item -Path ($Reg_Path + "\Server") -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Client") -Name 'Enabled' -Value 1 -PropertyType DWORD -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Server") -Name 'Enabled' -Value 1 -PropertyType DWORD -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Client") -Name 'DisabledByDefault' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Server") -Name 'DisabledByDefault' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue

        # === Enable TLS 1.2 =====================================================================
        $Reg_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.2"
        New-Item -Path $Reg_Path -Force -ErrorAction SilentlyContinue
        New-Item -Path ($Reg_Path + "\Client") -Force -ErrorAction SilentlyContinue
        New-Item -Path ($Reg_Path + "\Server") -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Client") -Name 'Enabled' -Value 1 -PropertyType DWORD -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Server") -Name 'Enabled' -Value 1 -PropertyType DWORD -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Client") -Name 'DisabledByDefault' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_Path + "\Server") -Name 'DisabledByDefault' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue

# ================================================================================
# === Configure SCHANNEL Cipher Suites ===========================================
#  ================================================================================

        # === Ensure Cipher RegPath Exist ========================================
        $Reg_KeyExch_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers"
        New-Item -Path ($Reg_KeyExch_Path) -Force -ErrorAction SilentlyContinue

        # === Disable NULL Cipher Suite ==========================================
        $Reg_KeyExch_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\NULL"
        New-Item -Path ($Reg_KeyExch_Path) -Force -ErrorAction SilentlyContinue
```

```
        New-ItemProperty -Path ($Reg_KeyExch_Path) -Name 'Enabled' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue

        # === Disable DES 56/56 Cipher Suite ======================================================
        $Reg_Chipers_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers"

        ([Microsoft.Win32.RegistryKey]::OpenRemoteBaseKey([Microsoft.Win32.RegistryHive]::LocalMachine,
$env:COMPUTERNAME)).CreateSubKey('SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\DES   56/56')
        New-ItemProperty -Path ($Reg_Chipers_Path + "\" + "DES 56$([char]0x2F)56") -Name 'Enabled' -Value 0 -PropertyType DWORD -Force


        # === Disable RC2 40/128 Cipher Suite ======================================================
        $Reg_Chipers_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers"

        ([Microsoft.Win32.RegistryKey]::OpenRemoteBaseKey([Microsoft.Win32.RegistryHive]::LocalMachine,
$env:COMPUTERNAME)).CreateSubKey('SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC2   40/128')
        New-ItemProperty -Path ($Reg_Chipers_Path + "\" + "RC2 40$([char]0x2F)128") -Name 'Enabled' -Value 0 -PropertyType DWORD -Force


        # === Disable RC2 56/128 Cipher Suite ======================================================
        $Reg_Chipers_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers"

        ([Microsoft.Win32.RegistryKey]::OpenRemoteBaseKey([Microsoft.Win32.RegistryHive]::LocalMachine,
$env:COMPUTERNAME)).CreateSubKey('SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC2   56/128')
        New-ItemProperty -Path ($Reg_Chipers_Path + "\" + "RC2 56$([char]0x2F)128") -Name 'Enabled' -Value 0 -PropertyType DWORD -Force


        # === Disable RC2 128/128 Cipher Suite ======================================================
        $Reg_Chipers_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers"

        ([Microsoft.Win32.RegistryKey]::OpenRemoteBaseKey([Microsoft.Win32.RegistryHive]::LocalMachine,
$env:COMPUTERNAME)).CreateSubKey('SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC2   128/128')
        New-ItemProperty -Path ($Reg_Chipers_Path + "\" + "RC2 128$([char]0x2F)128") -Name 'Enabled' -Value 0 -PropertyType DWORD -Force


        # === Disable RC4 40/128 Ciphers Suite ======================================================
        $Reg_Chipers_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers"

        ([Microsoft.Win32.RegistryKey]::OpenRemoteBaseKey([Microsoft.Win32.RegistryHive]::LocalMachine,
$env:COMPUTERNAME)).CreateSubKey('SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4   40/128')
        New-ItemProperty -Path ($Reg_Chipers_Path + "\" + "RC4 40$([char]0x2F)128") -Name 'Enabled' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue


        # === Disable RC4 56/128 Ciphers Suite ======================================================
        $Reg_Chipers_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers"

        ([Microsoft.Win32.RegistryKey]::OpenRemoteBaseKey([Microsoft.Win32.RegistryHive]::LocalMachine,
$env:COMPUTERNAME)).CreateSubKey('SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4   56/128')
        New-ItemProperty -Path ($Reg_Chipers_Path + "\" + "RC4 56$([char]0x2F)128") -Name 'Enabled' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue


        # === Disable RC4 64/128 Ciphers Suite ======================================================
        $Reg_Chipers_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers"

        ([Microsoft.Win32.RegistryKey]::OpenRemoteBaseKey([Microsoft.Win32.RegistryHive]::LocalMachine,
$env:COMPUTERNAME)).CreateSubKey('SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4   64/128')
        New-ItemProperty -Path ($Reg_Chipers_Path + "\" + "RC4 64$([char]0x2F)128") -Name 'Enabled' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue


        # === Disable RC4 128/128 Ciphers Suite ======================================================
        $Reg_Chipers_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers"

        ([Microsoft.Win32.RegistryKey]::OpenRemoteBaseKey([Microsoft.Win32.RegistryHive]::LocalMachine,
$env:COMPUTERNAME)).CreateSubKey('SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\RC4   128/128')
        New-ItemProperty -Path ($Reg_Chipers_Path + "\" + "RC4 128$([char]0x2F)128") -Name 'Enabled' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue

        # === Disable Triple DES 168 Cipher Suite ======================================================
        # Please note that Nartec (IISCrypto40.exe) and Microsoft KB 245030 refer to that Triple DES 168 regkey shall be "Triple DES 168/168" this is wrong.
        # Ref: http://www.admin-enclave.com/en/articles/windows/169-howto-disable-triple-des-168-on-windows.html
        $Reg_KeyExch_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\Triple DES 168"
        New-Item -Path ($Reg_KeyExch_Path) -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_KeyExch_Path) -Name 'Enabled' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue

        # === Enable AES 128/128 Ciphers Suite ======================================================
        $Reg_Chipers_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers"

        ([Microsoft.Win32.RegistryKey]::OpenRemoteBaseKey([Microsoft.Win32.RegistryHive]::LocalMachine,
$env:COMPUTERNAME)).CreateSubKey('SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\AES   128/128')
        New-ItemProperty -Path ($Reg_Chipers_Path + "\" + "AES 128$([char]0x2F)128") -Name 'Enabled' -Value 0xffffffff -PropertyType DWORD -Force -ErrorAction SilentlyContinue


        # === Enabled AES 256/256 Ciphers Suite ======================================================
        $Reg_Chipers_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers"

        ([Microsoft.Win32.RegistryKey]::OpenRemoteBaseKey([Microsoft.Win32.RegistryHive]::LocalMachine,
$env:COMPUTERNAME)).CreateSubKey('SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Ciphers\AES   256/256')
        New-ItemProperty -Path ($Reg_Chipers_Path + "\" + "AES 256$([char]0x2F)256") -Name 'Enabled' -Value 0xffffffff -PropertyType DWORD -Force -ErrorAction SilentlyContinue


# ============================================================================================================
# === Configure SCHANNEL Cipher Hashes =======================================================================
#  ============================================================================================================

        # === Ensure Cipher Hashes RegPath Exist ======================================================
        $Reg_KeyExch_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Hashes"
        New-Item -Path ($Reg_KeyExch_Path) -Force -ErrorAction SilentlyContinue

        # === Disable MD5 Hash Cipher Suite ======================================================
        $Reg_KeyExch_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Hashes\MD5"
        New-Item -Path ($Reg_KeyExch_Path) -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_KeyExch_Path) -Name 'Enabled' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue

        # === Enable SHA Hash Cipher Suite ======================================================
        $Reg_KeyExch_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Hashes\SHA"
        New-Item -Path ($Reg_KeyExch_Path) -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_KeyExch_Path) -Name 'Enabled' -Value 0xffffffff -PropertyType DWORD -Force -ErrorAction SilentlyContinue

        # === Enable SHA256 Hash Cipher Suite ======================================================
        $Reg_KeyExch_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Hashes\SHA256"
        New-Item -Path ($Reg_KeyExch_Path) -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_KeyExch_Path) -Name 'Enabled' -Value 0xffffffff -PropertyType DWORD -Force -ErrorAction SilentlyContinue


        # === Enable SHA384 Hash Cipher Suite ======================================================
        $Reg_KeyExch_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Hashes\SHA384"
        New-Item -Path ($Reg_KeyExch_Path) -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_KeyExch_Path) -Name 'Enabled' -Value 0xffffffff -PropertyType DWORD -Force -ErrorAction SilentlyContinue


        # === Enable SHA512 Hash Cipher Suite ======================================================
        $Reg_KeyExch_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Hashes\SHA512"
        New-Item -Path ($Reg_KeyExch_Path) -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_KeyExch_Path) -Name 'Enabled' -Value 0xffffffff -PropertyType DWORD -Force -ErrorAction SilentlyContinue


# ============================================================================================================
# === Configure SCHANNEL Key Exchange Ciphers ================================================================
#  ============================================================================================================

        # === Ensure Cipher Hashes RegPath Exist ======================================================
        $Reg_KeyExch_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\KeyExchangeAlgorithms"
        New-Item -Path ($Reg_KeyExch_Path) -Force -ErrorAction SilentlyContinue

        # === Disable Diffie-Helleman Key Exchange Ciphers ======================================================
        $Reg_KeyExch_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\KeyExchangeAlgorithms\Diffie-Hellman"
        New-Item -Path ($Reg_KeyExch_Path)  -Force -ErrorAction SilentlyContinue
```

```
        New-ItemProperty -Path ($Reg_KeyExch_Path) -Name 'Enabled' -Value 0 -PropertyType DWORD -Force -ErrorAction SilentlyContinue


        # === Enable ECDH Key Exchange Ciphers ===================================================
        $Reg_KeyExch_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\KeyExchangeAlgorithms\ECDH"
        New-Item -Path ($Reg_KeyExch_Path)  -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_KeyExch_Path) -Name 'Enabled' -Value 0xffffffff -PropertyType DWORD -Force -ErrorAction SilentlyContinue


        # === Enable PKCS Key Exchange Ciphers ===================================================
        $Reg_KeyExch_Path = "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\KeyExchangeAlgorithms\PKCS"
        New-Item -Path ($Reg_KeyExch_Path)  -Force -ErrorAction SilentlyContinue
        New-ItemProperty -Path ($Reg_KeyExch_Path) -Name 'Enabled' -Value 0xffffffff -PropertyType DWORD -Force -ErrorAction SilentlyContinue


# ============================================================================================================
# === Configure Cipher Suites Order ==========================================================================
# ============================================================================================================


#  This section is not enabled by default, as above code is normally sufficient.
#  If there is an need to tune the Ciper Suites Order, below code is an example how this can be carried out.
#  If "Functions" don't exist, the operating system will decide the cipher Order Suite itself. Default behaviour.
#
#    # === Ensure Cipher Suite Order Exist ===================================================
#    $Reg_KeyExch_Path = "HKLM:\SOFTWARE\Policies\Microsoft\Cryptography\Configuration\SSL\00010002"
#    New-Item -Path ($Reg_KeyExch_Path) -Force -ErrorAction SilentlyContinue
#
#    # === Configure Suite Order =============================================================
#    $Cipher_Suite_Order = "TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384_P521," + `
#                          "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256_P521," + `
#                          "TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA_P521"  + `
#                          "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA_P521"
#
#    $Reg_KeyExch_Path = "HKLM:\SOFTWARE\Policies\Microsoft\Cryptography\Configuration\SSL\00010002"
#    New-ItemProperty -Path ($Reg_KeyExch_Path) -Name 'Functions' -PropertyType String -Value $Cipher_Suite_Order -Force -ErrorAction SilentlyContinue
```

--- PowerShell Sample Code Stop ---

9 APPENDIX – IDENTITIES PROVISIONING

9.1 Appendix – SCIM API

9.1.1 Get All Schemas

This method can used to retrieve all schemas objects. “attributes” field must contain all attributes used be the specific schema although not listed in this sample. For more details see RFC

Request Sample:

--- Request Sample Code Start ---

```
GET /v2/Schemas
Host: democompany.com
Accept: application/json
Authorization: Bearer dj8u9324dh3
```

--- Request Sample Code Stop ---

Table 22.

Response Sample:

--- Response Sample Code Start ---

```
{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:ListResponse"
  ],
  "totalResults": 3,
  "itemsPerPage": 3,
  "startIndex": 1,
  "Resources": [
    {
      "id": "urn:ietf:params:scim:schemas:core:2.0:User",
      "name": "User",
      "description": "User Schema",
      "attributes": [
        "..."
      ],
      "meta": {
        "resourceType": "Schema",
        "location": "https://democompany.com/scim/v2/Schemas/urn:ietf:params:scim:schemas:core:2.0:User"
      }
    },
    {
      "id": "urn:ietf:params:scim:schemas:extension:enterprise:2.0:User",
      "name": "Enterprise User",
      "description": "Enterprise User Schema",
      "attributes": [
        "..."
      ],
      "meta": {
        "resourceType": "Schema",
        "location":
"https://democompany.com/scim/v2/Schemas/urn:ietf:params:scim:schemas:extension:enterprise:2.0:User"
      }
    },
    {
      "id": "urn:ietf:params:scim:schemas:core:2.0:Group",
```

```
      "name": "Group",
      "description": "Group Schema",
      "attributes": [
        "..."
      ],
      "meta": {
        "resourceType": "Schema",
        "location":
"https://democompany.com/scim/v2/Schemas/urn:ietf:params:scim:schemas:core:2.0:Group"
      }
    }
  ]
}
```

--- Response Sample Code Stop ---

Table 23.

9.1.2 Get Schema Detail

This method can used to retrieve all schemas objects. “attributes” field must contain all attributes used be the specific schema although not listed in this sample. For more details see RFC

Request Sample:

--- Request Sample Code Start ---

```
GET /v2/Schemas/User
Host: democompany.com
Accept: application/json
Authorization: Bearer dj8u9324dh3
```

Table 24.

This method can used to retrieve all schemas objects. “attributes” field must contain all attributes used be the specific schema although not listed in this sample. For more details see RFC

Response Sample:

--- Response Sample Code Start ---


```
{
  "id": "urn:ietf:params:scim:schemas:core:2.0:User",
  "name": "User",
  "description": "User Schema",
  "attributes": [
    {
      "name": "userName",
      "type": "string",
      "multiValued": false,
      "required": true,
      "caseExact": false,
      "mutability": "readWrite",
      "returned": "default",
      "uniqueness": "server",
      "description": "Unique identifier for the User, typically used by the user to directly authenticate to the service provider. Each User MUST include a non-empty userName value. This identifier MUST be unique across the service provider's entire set of Users."
    },
    {
      "name": "name",
      "type": "complex",
      "multiValued": false,
      "required": false,
      "mutability": "readWrite",
      "returned": "default",
      "uniqueness": "none",
      "description": "The components of the user's real name. Providers MAY return just the full name as a single string in the formatted sub-attribute, or they MAY return just the individual component attributes using the other sub-attributes, or they MAY return both.  If both variants are returned, they SHOULD be describing the same name, with the formatted name indicating how the component attributes should be combined.",
      "subAttributes": [
        {
          "name": "formatted",
          "type": "string",
          "multiValued": false,
          "required": false,
          "caseExact": false,
```

```
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "The full name, including all middle names, titles, and suffixes as appropriate,
formatted for display (e.g., 'Ms. Barbara J Jensen, III')."
      },
      {
        "name": "familyName",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "The family name of the User, or last name in most Western languages (e.g., 'Jensen'
given the full name 'Ms. Barbara J Jensen, III')."
      },
      {
        "name": "givenName",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "The given name of the User, or first name in most Western languages (e.g., 'Barbara'
given the full name 'Ms. Barbara J Jensen, III')."
      },
      {
        "name": "middleName",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "The middle name(s) of the User (e.g., 'Jane' given the full name 'Ms. Barbara J
Jensen, III')."
      },
      {
        "name": "honorificPrefix",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "The honorific prefix(es) of the User, or title in most Western languages (e.g., 'Ms.'
given the full name 'Ms. Barbara J Jensen, III')."
      },
      {
        "name": "honorificSuffix",
        "type": "string",
        "multiValued": false,
```

```
          "required": false,
          "caseExact": false,
          "mutability": "readWrite",
          "returned": "default",
          "uniqueness": "none",
          "description": "The honorific suffix(es) of the User, or suffix in most Western languages (e.g., 'III' given the full name 'Ms. Barbara J Jensen, III')."
        }
      ]
    },
    {
      "name": "displayName",
      "type": "string",
      "multiValued": false,
      "required": false,
      "caseExact": false,
      "mutability": "readWrite",
      "returned": "default",
      "uniqueness": "none",
      "description": "The name of the User, suitable for display to end-users.  The name SHOULD be the full name of the User being described, if known."
    },
    {
      "name": "nickName",
      "type": "string",
      "multiValued": false,
      "required": false,
      "caseExact": false,
      "mutability": "readWrite",
      "returned": "default",
      "uniqueness": "none",
      "description": "The casual way to address the user in real life, e.g., 'Bob' or 'Bobby' instead of 'Robert'. This attribute SHOULD NOT be used to represent a User's username (e.g., 'bjensen' or 'mpepperidge')."
    },
    {
      "name": "profileUrl",
      "type": "reference",
      "multiValued": false,
      "required": false,
      "caseExact": false,
      "mutability": "readWrite",
      "returned": "default",
      "uniqueness": "none",
      "referenceTypes": [
      "external"
      ],
      "description": "A fully qualified URL pointing to a page representing the User's online profile."
    },
    {
      "name": "title",
      "type": "string",
      "multiValued": false,
      "required": false,
      "caseExact": false,
      "mutability": "readWrite",
      "returned": "default",
      "uniqueness": "none",
      "description": "The user's title, such as 'Vice President.'"
```

```
    },
    {
      "name": "userType",
      "type": "string",
      "multiValued": false,
      "required": false,
      "caseExact": false,
      "mutability": "readWrite",
      "returned": "default",
      "uniqueness": "none",
      "description": "Used to identify the relationship between the organization and the user. Typical values used might be 'Contractor', 'Employee', 'Intern', 'Temp', 'External', and 'Unknown', but any value may be used."
    },
    {
      "name": "preferredLanguage",
      "type": "string",
      "multiValued": false,
      "required": false,
      "caseExact": false,
      "mutability": "readWrite",
      "returned": "default",
      "uniqueness": "none",
      "description": "Indicates the User's preferred written or spoken language.  Generally used for selecting a localized user interface; e.g., 'en_US' specifies the language English and country US."
    },
    {
      "name": "locale",
      "type": "string",
      "multiValued": false,
      "required": false,
      "caseExact": false,
      "mutability": "readWrite",
      "returned": "default",
      "uniqueness": "none",
      "description": "Used to indicate the User's default location for purposes of localizing items such as currency, date time format, or numerical representations."
    },
    {
      "name": "timezone",
      "type": "string",
      "multiValued": false,
      "required": false,
      "caseExact": false,
      "mutability": "readWrite",
      "returned": "default",
      "uniqueness": "none",
      "description": "The User's time zone in the 'Olson' time zone database format, e.g., 'America/Los_Angeles'."
    },
    {
      "name": "active",
      "type": "boolean",
      "multiValued": false,
      "required": false,
      "mutability": "readWrite",
      "returned": "default",
      "uniqueness": "none",
```

```
      "description": "A Boolean value indicating the User's administrative status."
    },
    {
      "name": "password",
      "type": "string",
      "multiValued": false,
      "required": false,
      "caseExact": false,
      "mutability": "writeOnly",
      "returned": "never",
      "uniqueness": "none",
      "description": "The User's cleartext password. This attribute is intended to be used as a means to
specify an initial password when creating a new User or to reset an existing User's password."
    },
    {
      "name": "emails",
      "type": "complex",
      "multiValued": true,
      "required": false,
      "mutability": "readWrite",
      "returned": "default",
      "uniqueness": "none",
      "description": "Email addresses for the user. The value SHOULD be canonicalized by the service
provider, e.g., 'bjensen@example.com' instead of 'bjensen@EXAMPLE.COM'.",
      "subAttributes": [
        {
          "name": "value",
          "type": "string",
          "multiValued": false,
          "required": false,
          "caseExact": false,
          "mutability": "readWrite",
          "returned": "default",
          "uniqueness": "none",
          "description": "Email addresses for the user. The value SHOULD be canonicalized by the service
provider, e.g., 'bjensen@example.com' instead of 'bjensen@EXAMPLE.COM'."
        },
        {
          "name": "display",
          "type": "string",
          "multiValued": false,
          "required": false,
          "caseExact": false,
          "mutability": "readWrite",
          "returned": "default",
          "uniqueness": "none",
          "description": "A human-readable name, primarily used for display purposes."
        },
        {
          "name": "type",
          "type": "string",
          "multiValued": false,
          "required": false,
          "caseExact": false,
          "mutability": "readWrite",
          "returned": "default",
          "uniqueness": "none",
          "canonicalValues": [
```

```
          "work",
          "home",
          "other"
         ],
         "description": "A label indicating the attribute's function, e.g., 'work' or 'home'."
        },
        {
         "name": "primary",
         "type": "boolean",
         "multiValued": false,
         "required": false,
         "mutability": "readWrite",
         "returned": "default",
         "uniqueness": "none",
         "description": "A Boolean value indicating the 'primary' or preferred attribute value for this attribute,
e.g., the preferred mailing address or primary email address.  The primary attribute value 'true' MUST
appear no more than once."
        }
      ]
    },
    {
     "name": "phoneNumbers",
     "type": "complex",
     "multiValued": true,
     "required": false,
     "mutability": "readWrite",
     "returned": "default",
     "uniqueness": "none",
     "description": "Phone numbers for the User. The value SHOULD be canonicalized by the service
provider according to the format specified in RFC 3966, e.g., 'tel:+1-201-555-0123'.",
     "subAttributes": [
      {
       "name": "value",
       "type": "string",
       "multiValued": false,
       "required": false,
       "caseExact": false,
       "mutability": "readWrite",
       "returned": "default",
       "uniqueness": "none",
       "description": "Phone number of the User."
      },
      {
       "name": "display",
       "type": "string",
       "multiValued": false,
       "required": false,
       "caseExact": false,
       "mutability": "readWrite",
       "returned": "default",
       "uniqueness": "none",
       "description": "A human-readable name, primarily used for display purposes."
      },
      {
       "name": "type",
       "type": "string",
       "multiValued": false,
       "required": false,
```

```
          "caseExact": false,
          "mutability": "readWrite",
          "returned": "default",
          "uniqueness": "none",
          "canonicalValues": [
          "work",
            "home",
            "mobile",
            "fax",
            "pager",
            "other"
          ],
          "description": "A label indicating the attribute's function, e.g., 'work', 'home', 'mobile'."
        },
        {
          "name": "primary",
          "type": "boolean",
          "multiValued": false,
          "required": false,
          "mutability": "readWrite",
          "returned": "default",
          "uniqueness": "none",
          "description": "A Boolean value indicating the 'primary' or preferred attribute value for this attribute,
e.g., the preferred mailing address or primary email address. The primary attribute value 'true' MUST
appear no more than once."
        }
      ]
    },
    {
      "name": "ims",
      "type": "complex",
      "multiValued": true,
      "required": false,
      "mutability": "readWrite",
      "returned": "default",
      "uniqueness": "none",
      "description": "Instant messaging addresses for the User.",
      "subAttributes": [
        {
          "name": "value",
          "type": "string",
          "multiValued": false,
          "required": false,
          "caseExact": false,
          "mutability": "readWrite",
          "returned": "default",
          "uniqueness": "none",
          "description": "Instant messaging address for the User."
        },
        {
          "name": "display",
          "type": "string",
          "multiValued": false,
          "required": false,
          "caseExact": false,
          "mutability": "readWrite",
          "returned": "default",
          "uniqueness": "none",
```

```
        "description": "A human-readable name, primarily used for display purposes."
      },
      {
        "name": "type",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "canonicalValues": [
        "aim",
          "gtalk",
          "icq",
          "xmpp",
          "msn",
          "skype",
          "qq",
          "yahoo"
        ],
        "description": "A label indicating the attribute's function, e.g., 'aim', 'gtalk', 'xmpp'."
      },
      {
        "name": "primary",
        "type": "boolean",
        "multiValued": false,
        "required": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "A Boolean value indicating the 'primary' or preferred attribute value for this attribute,
e.g., the preferred mailing address or primary email address. The primary attribute value 'true' MUST
appear no more than once."
      }
    ]
  },
  {
    "name": "photos",
    "type": "complex",
    "multiValued": true,
    "required": false,
    "mutability": "readWrite",
    "returned": "default",
    "uniqueness": "none",
    "description": "URLs of photos of the User.",
    "subAttributes": [
      {
        "name": "value",
        "type": "reference",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "referenceTypes": [
          "external"
```

```
        ],
        "description": "URLs of a photo of the User."
      },
      {
        "name": "display",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "A human-readable name, primarily used for display purposes."
      },
      {
        "name": "type",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "canonicalValues": [
        "photo",
          "thumbnail"
        ],
        "description": "A label indicating the attribute's function, i.e., 'photo' or 'thumbnail'."
      },
      {
        "name": "primary",
        "type": "boolean",
        "multiValued": false,
        "required": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "A Boolean value indicating the 'primary' or preferred attribute value for this attribute,
e.g., the preferred mailing address or primary email address. The primary attribute value 'true' MUST
appear no more than once."
      }
    ]
  },
  {
    "name": "addresses",
    "type": "complex",
    "multiValued": true,
    "required": false,
    "mutability": "readWrite",
    "returned": "default",
    "uniqueness": "none",
    "description": "A physical mailing address for this User.",
    "subAttributes": [
      {
        "name": "formatted",
        "type": "string",
        "multiValued": false,
        "required": false,
```

```
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "The full mailing address, formatted for display or use with a mailing label. This attribute MAY contain newlines."
      },
      {
        "name": "streetAddress",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "The full street address component, which may include house number, street name, P.O. box, and multi-line extended street address information. This attribute MAY contain newlines."
      },
      {
        "name": "locality",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "The city or locality component."
      },
      {
        "name": "region",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "The state or region component."
      },
      {
        "name": "postalCode",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "The zip code or postal code component."
      },
      {
        "name": "country",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
```

```
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "The country name component."
      },
      {
        "name": "type",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "canonicalValues": [
        "work",
          "home",
          "other"
        ],
        "description": "A label indicating the attribute's function, e.g., 'work' or 'home'."
      }
    ]
  },
  {
    "name": "groups",
    "type": "complex",
    "multiValued": true,
    "required": false,
    "mutability": "readOnly",
    "returned": "default",
    "uniqueness": "none",
    "description": "A list of groups to which the user belongs, either through direct membership, through
nested groups, or dynamically calculated.",
    "subAttributes": [
      {
        "name": "value",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readOnly",
        "returned": "default",
        "uniqueness": "none",
        "description": "The identifier of the User's group."
      },
      {
        "name": "$ref",
        "type": "reference",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readOnly",
        "returned": "default",
        "uniqueness": "none",
        "referenceTypes": [
        "Group"
        ],
        "description": "The URI of the corresponding 'Group' resource to which the user belongs."
```

```
      },
      {
        "name": "display",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readOnly",
        "returned": "default",
        "uniqueness": "none",
        "description": "A human-readable name, primarily used for display purposes."
      },
      {
        "name": "type",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "canonicalValues": [
        "direct",
          "indirect"
        ],
        "description": "A label indicating the attribute's function, e.g., 'direct' or 'indirect'."
      }
    ]
  },
  {
    "name": "entitlements",
    "type": "complex",
    "multiValued": true,
    "required": false,
    "mutability": "readWrite",
    "returned": "default",
    "uniqueness": "none",
    "description": "A list of entitlements for the User that represent a thing the User has.",
    "subAttributes": [
      {
        "name": "value",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "The value of an entitlement."
      },
      {
        "name": "display",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
```

```
        "uniqueness": "none",
        "description": "A human-readable name, primarily used for display purposes."
      },
      {
        "name": "type",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "A label indicating the attribute's function."
      },
      {
        "name": "primary",
        "type": "boolean",
        "multiValued": false,
        "required": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "A Boolean value indicating the 'primary' or preferred attribute value for this attribute,
e.g., the preferred mailing address or primary email address. The primary attribute value 'true' MUST
appear no more than once."
      }
    ]
  },
  {
    "name": "roles",
    "type": "complex",
    "multiValued": true,
    "required": false,
    "mutability": "readWrite",
    "returned": "default",
    "uniqueness": "none",
    "description": "A list of roles for the User that collectively represent who the User is, e.g., 'Student',
'Faculty'.",
    "subAttributes": [
      {
        "name": "value",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "The value of a role."
      },
      {
        "name": "display",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
```

```
        "uniqueness": "none",
        "description": "A human-readable name, primarily used for display purposes."
      },
      {
        "name": "type",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "A label indicating the attribute's function."
      },
      {
        "name": "primary",
        "type": "boolean",
        "multiValued": false,
        "required": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "A Boolean value indicating the 'primary' or preferred attribute value for this attribute,
e.g., the preferred mailing address or primary email address. The primary attribute value 'true' MUST
appear no more than once."
      }
    ]
  },
  {
    "name": "x509Certificates",
    "type": "complex",
    "multiValued": true,
    "required": false,
    "mutability": "readWrite",
    "returned": "default",
    "uniqueness": "none",
    "description": "A list of certificates issued to the User.",
    "subAttributes": [
      {
        "name": "value",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "The value of an X.509 certificate."
      },
      {
        "name": "display",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
```

```
        "description": "A human-readable name, primarily used for display purposes."
      },
      {
        "name": "type",
        "type": "string",
        "multiValued": false,
        "required": false,
        "caseExact": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "A label indicating the attribute's function."
      },
      {
        "name": "primary",
        "type": "boolean",
        "multiValued": false,
        "required": false,
        "mutability": "readWrite",
        "returned": "default",
        "uniqueness": "none",
        "description": "A Boolean value indicating the 'primary' or preferred attribute value for this attribute,
e.g., the preferred mailing address or primary email address. The primary attribute value 'true' MUST
appear no more than once."
      }
    ]
  }
 ],
 "meta": {
  "resourceType": "Schema",
  "location": "https://democompany.com/scim/v2/Schemas/urn:ietf:params:scim:schemas:core:2.0:User"
 }
}
```


Table 25.

9.1.3 Create User

Use HTTP POST to create a user

Request Sample:

--- Request Sample Code Start ---

```
POST /v2/Users  HTTP/1.1
Accept: application/json
Authorization: Bearer 9ea8b38f80038981fe38f60419672646
Host: democompany.com
Content-Length: ...
Content-Type: application/json

{
  "schemas":["urn:ietf:params:scim:schemas:core:2.0:User"],
  "externalId":"GMOLA",
  "userName":"GMOLA",
  "name":{
  "familyName":"Nordmann",
  "givenName":"Ola"
  }
}
```

--- Request Sample Code Stop ---

Table 26.

Response Sample:

--- Response Sample Code Start ---

```
HTTP/1.1 201 Created
Content-Type: application/json
Location:  https://democompany.com/v2/Users/2819c223-7f76-453a-919d-413861904646

{
  "schemas":["urn:ietf:params:scim:schemas:core:2.0:User"],
  "id":"2819c223-7f76-453a-919d-413861904646",
  "externalId":"GMOLA",
  "meta":{
  "resourceType":"User",
    "created":"2018-09-01T12:21:11.714Z",
    "lastModified":"2018-09-01T12:21:11.714Z",
    "location": "https://democompany.com/v1/Users/23123"
  },
  "name":{
    "familyName":"Nordmann",
    "givenName":"Ola"
  },
  "userName":"GMOLA",
  "active": true
}
```

--- Response Sample Code Stop ---

Table 27.

9.1.4 Read User

Use HTTP GET to fetch user data. To fetch a single user record the following GET method is applied

Request Sample:

--- Request Sample Code Start ---

```
GET  /v2/Users/2819c223-7f76-453a-919d-413861904646
Host: democompany.com
Accept: application/json
Authorization: Bearer dj8u9324dh3
```

--- Request Sample Code Stop ---

Table 28.

Response Sample:

--- Response Sample Code Start ---

```
{
  "schemas": ["urn:ietf:params:scim:schemas:core:2.0:User"],
  "id":"2819c223-7f76-453a-919d-413861904646",
  "externalId":"GMOLA",
  "meta":{
    "resourceType": "User",
    "created":"2018-09-01T12:21:11.714Z",
    "lastModified":"2018-09-01T12:21:11.714Z",
    "location":"https://democompany.com/v2/Users/23123"
  },
  "name":{
    "familyName": "Nordmann",
    "givenName": "Ola"
  },
  "userName": "GMOLA",
  "active": true,
  "phoneNumbers":[
    {
      "value":"+4712345678",
      "type":"work"
    }
  ],
  "emails":[
    {
      "value":"ola.nordmann@avinor.no",
      "type":"work",
      "primary": true
    }
  ],
  "groups":
    [ {
      "value":  "e9e30dba-f08f-4109-8486-d5c6a331660a",
      "$ref": "https://democompany.com/v2/Groups/e9e30dba-f08f-4109-8486-d5c6a331660a",
      "display": "Supervisor"
    }, {
      "value":  "c3a26dd3-27a0-4dec-a2ac-ce211e105f97",
      "$ref": "https://democompany.com/v2/Groups/c3a26dd3-27a0-4dec-a2ac-ce211e105f97",
      "display": "Sales"
    } ]
}
```

--- Response Sample Code Stop ---

Table 29.

9.1.5 Update User Attribute

The HTTP Patch method is applied for updating specific user attributes

Request Sample:

--- Request Sample Code Start ---

```
PATCH /v2/Users/2819c223-7f76-453a-919d-413861904646 HTTP/1.1
Accept: application/json
Authorization: Bearer 9ea8b38f80038981fe38f60419672646
Host: democompany.com
Content-Length: ...
Content-Type: application/json

{
  "active": true
}
```

Table 30.

Response Sample:

--- Response Sample Code Start ---

```
HTTP/1.1 200 OK
Content-Type: application/json
Location: https://democompany.com/v2/Users/2819c223-7f76-453a-919d-413861904646
Content-Length: ...
Content-Type: application/json

{
  "schemas": ["urn:ietf:params:scim:schemas:core:2.0:User"],
  "id":"2819c223-7f76-453a-919d-413861904646",
  "externalId":"GMOLA",
  "meta":{
    "resourceType": "User",
    "created":"2018-09-01T12:21:11.714Z",
    "lastModified":"2018-09-01T12:21:11.714Z",
    "location":"https://democompany.com/v2/Users/23123"
  },
  "name":{
    "familyName": "Nordmann",
    "givenName": "Ola"
  },
  "userName": "GMOLA",
  "active": true,
  "phoneNumbers":[
    {
      "value":"+4712345678",
      "type":"work"
    },
    {
      "value":"+4755443322",
      "type":"workmobile"
    }
  ],
  "emails":[
    {
      "value":"ola.nordmann@avinor.no",
```

```
      "type":"work",
      "primary": true
    }
  ]
}
```

--- Response Sample Code Stop ---

Table 31.

9.1.6 Update User

HTTP Patch can also be applied to update multiple attributes for an existing user

Request Sample:

--- Request Sample Code Start ---

```
PATCH /v2/Users/2819c223-7f76-453a-919d-413861904646 HTTP/1.1
Accept: application/json
Authorization: Bearer 9ea8b38f80038981fe38f60419672646
Host: democompany.com
Content-Length: ...
Content-Type: application/json

{
  "phoneNumbers":[
    {
      "value":"+4712345678",
      "type":"work"
    },
    {
      "value":"+4755443322",
      "type":"workmobile"
    }
  ],
  "emails":[
    {
      "value":"ola.nordmann@avinor.no",
      "type":"work",
      "primary": true
    }
  ]
}
```

--- Request Sample Code Stop ---

Table 32.

Response Sample:

--- Response Sample Code Start ---

```
HTTP/1.1 200 OK
Content-Type: application/json
Location: https://democompany.com/v2/Users/2819c223-7f76-453a-919d-413861904646
Content-Length: ...
Content-Type: application/json

{
  "schemas": ["urn:ietf:params:scim:schemas:core:2.0:User"],
  "id":"2819c223-7f76-453a-919d-413861904646",
  "externalId":"GMOLA",
  "meta":{
    "resourceType": "User",
    "created":"2018-09-01T12:21:11.714Z",
    "lastModified":"2018-09-01T12:21:11.714Z",
    "location":"https://democompany.com/v2/Users/23123"
  },
  "name":{
    "familyName": "Nordmann",
    "givenName": "Ola"
  },
  "userName": "GMOLA",
  "active": true,
  "phoneNumbers":[
    {
      "value":"+4712345678",
      "type":"work"
    },
    {
      "value":"+4755443322",
      "type":"workmobile"
    }
  ],
  "emails":[
    {
      "value":"ola.nordmann@avinor.no",
      "type":"work",
      "primary": true
    }
  ]
}
```

--- Response Sample Code Stop ---

Table 33.

9.1.7 Bulk Operation

The Bulk endpoint is an optional feature that allows to perform multiple create, update, and delete operations using one request.

Example Request

--- Request Sample Code Start ---

```
POST /v2/Bulk HTTP/1.1
Accept: application/json
Authorization: Bearer 9ea8b38f80038981fe38f60419672646
Host: democompany.com
Content-Length: ...
Content-Type: application/json

{
  "schemas": ["urn:scim:api:messages:2.0:BulkRequest"],
   "failOnErrors": 1,
    "Operations": [
    {
      "method": "POST",
      "path": "/Users",
      "bulkId": "clientBulkId1",
      "data": {
          "schemas": ["urn:scim:schemas:core:2.0:User"],
          "externalId":" GMOLA",
          "userName":"GMOLA ",
          "name":{
          "familyName":"Nordmann",
          "givenName":"Ola"
          }
       }
    }

    {
      "method":"PATCH",
      "path":"/Users/33124",
      "bulkId":"clientBulkId2",
      "data":{
          "schemas": ["urn:scim:schemas:core:2.0:User"],
          "active":false
       }
  }

  {
    "method":"DELETE",
    "path":"/Users/43125",
    "bulkId":"clientBulkId3"
  }
}
```

--- Request Sample Code Stop ---

Table 34.

Parameters:

- failOnErrors: Specifies the number of errors that the service provider will accept before the operation is terminated with an error response.

Example Response:

--- Response Sample Code Start ---

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "schemas": ["urn:ietf:params:scim:schemas:core:2.0:User"],
   "Operations": [
     {
       "location": "https://democompany.com/v2/Users/23123",
       "method": "POST",
       "bulkId": "clientBulkId1",
       "status": {"code": "201"}
     },
     {
       "location": "https://democompany.com/v2/Users/33124",
       "method": "PATCH",
       "bulkId": "clientBulkId2",
       "status": {"code": "200"}
      },
      {
        "location": "https://democompany.com/v2/Users/43124",
        "method": "DELETE",
        "bulkId": "clientBulkId1",
        "status": {"code": "204"}
      }
   ]
}
```

--- Response Sample Code Stop ---

Table 35.

9.1.8 Search User(s)

Use filtered search to get one or more users based on a specific search criteria.

Request Sample:

```
--- Request Sample Code Start ---
GET /v2/Users?filter=lastUpdate gt "2012-01-01"&sortBy=lastUpdate&sortOrder=ascending
Host: democompany.com
Accept: application/json
Authorization: Bearer dj8u9324dh3
---  Request Sample Code Stop ---
```

Table 36.

Response Sample:

```
--- Response Sample Code Start ---
[
{
  "schemas": ["urn:ietf:params:scim:schemas:core:2.0:User"],
  "id":"2819c223-7f76-453a-919d-413861904646",
  "externalId":"GMOLA",
  "meta":{
    "resourceType": "User",
    "created":"2018-09-01T12:21:11.714Z",
    "lastModified":"2018-09-01T12:21:11.714Z",
    "location":"https://democompany.com/v2/Users/23123"
  },
  "name":{
    "familyName": "Nordmann",
    "givenName": "Ola"
  },
  "userName": "GMOLA",
  "status": "Active",
  "phoneNumbers":[
    {
      "value":"+4712345678",
      "type":"work"
    }
  ],
  "emails":[
    {
      "value":"ola.nordmann@avinor.no",
      "type":"work",
      "primary": true
    }
  ],
  "groups":
    [ {
      "value":  "e9e30dba-f08f-4109-8486-d5c6a331660a",
      "$ref": "https://democompany.com/v2/Groups/e9e30dba-f08f-4109-8486-d5c6a331660a",
      "display": "Supervisor"
    }, {
      "value":  "c3a26dd3-27a0-4dec-a2ac-ce211e105f97",
      "$ref": "https://democompany.com/v2/Groups/c3a26dd3-27a0-4dec-a2ac-ce211e105f97",
      "display": "Sales"
    } ]
},
{
```

```
  "schemas": ["urn:ietf:params:scim:schemas:core:2.0:User"],
  "id":"24561",
  "externalId":"CAKHA",
  "meta":{
    "resourceType": "User",
    "created":"2018-05-01T19:37:12.214Z",
    "lastModified":"2018-05-01T19:37:12.214Z",
    "location":"https://democompany.com/v1/Users/24561"
  },
  "name":{
    "familyName": "Hansen",
    "givenName": "Kari"
  },
  "userName": "CAKHA",
  "active": true,
  "phoneNumbers":[
    {
      "value":"+4787654321",
      "type":"work"
    }
  ],
  "emails":[
    {
      "value":"kari.hansen@avinor.no",
      "type":"work",
      "primary": true
    }
  ],
  "groups":
    [{
      "value":  "c3a26dd3-27a0-4dec-a2ac-ce211e105f97",
      "$ref": "https://democompany.com/v2/Groups/c3a26dd3-27a0-4dec-a2ac-ce211e105f97",
      "display": "Sales"
    } ]
}
]
```


--- Response Sample Code Stop ---

Table 37.

9.1.9 Get All Groups

The following example shows how to list all groups.

Request Sample:

```
--- Request Sample Code Start ---
GET /v2/Groups/ HTTP/1.1
Accept: application/json
Authorization: Bearer 9ea8b38f80038981fe38f60419672646
Host: democompany.com
Content-Length: ...
Content-Type: application/json
---  Request Sample Code Stop ---
```

Table 38.

Response Sample:

```
--- Response Sample Code Start ---
HTTP/1.1 200 OK
Content-Type: application/json
Location: https://democompany.com/v2/Groups/
Content-Length: ...
Content-Type: application/json

{
  "schemas": [
    "urn:ietf:params:scim:api:messages:2.0:ListResponse"
  ],
  "totalResults": 2,
  "itemsPerPage": 25,
  "startIndex": 1,
  "Resources": [
    {
      "schemas": [
        "urn:ietf:params:scim:schemas:core:2.0:Group"
      ],
      "id": "1",
      "externalId": "G1",
      "displayName": "Sales",
      "groupType": "Organization",
      "meta": {
        "resourceType": "Group",
        "created": "2018-04-17T11:05:29-05:00",
        "lastModified": "2018-04-17T11:05:29-05:00",
        "location": "https://democompany.com/scim/v2/Groups/1"
      }
    },
    {
      "schemas": [
        "urn:ietf:params:scim:schemas:core:2.0:Group"
      ],
      "id": "2",
      "externalId": "G2",
      "displayName": "Supervisors",
      "groupType": "Organization",
      "meta": {
```

```
        "resourceType": "Group",
        "created": "2018-04-17T11:05:29-05:00",
        "lastModified": "2018-04-17T11:05:29-05:00",
        "location": "https://democompany.com/scim/v2/Groups/2"
      }
    }
  ]
}
```

--- Response Sample Code Stop ---

Table 39.

9.1.10 Get Group Detail

The following example shows how to get details for a specific group.

Request Sample:

```
--- Request Sample Code Start ---
GET /v2/Groups/e9e30dba-f08f-4109-8486-d5c6a331660a HTTP/1.1
Accept: application/json
Authorization: Bearer 9ea8b38f80038981fe38f60419672646
Host: democompany.com
Content-Length: ...
Content-Type: application/json
--- Request Sample Code Stop ---
```

Table 40.

```
--- Response Sample Code Start ---
HTTP/1.1 200 OK
Content-Type: application/json
Location: https://democompany.com/v2/Groups/e9e30dba-f08f-4109-8486-d5c6a331660a
Content-Length: ...
Content-Type: application/json

{
  "schemas": [
    "urn:ietf:params:scim:schemas:core:2.0:Group"
  ],
  "id": "1",
  "externalId": "G1",
  "displayName": "Sales",
  "groupType": "Organization",
  "members": [
    {
      "display": "Ola Nordmann",
      "value":  "2819c223-7f76-453a-919d-413861904646",
      "$ref":   "https://democompany.com/scim/v2/Users/2819c223-7f76-453a-919d-413861904646"
    },
    {
      "display": "Kari Nordmann",
      "value": "24561",
      "$ref": "https://democompany.com/scim/v2/Users/24561"
    }
  ],
```

```
  "meta": {
    "resourceType": "Group",
    "created": "2018-04-17T11:05:29-05:00",
    "lastModified": "2018-04-17T11:05:29-05:00",
    "location": "https://democompany.com/scim/v2/Groups/1"
  }
}
```

--- Response Sample Code Stop ---

Table 41.

9.1.11 Add User to Group

The following example shows how to add a member to a group.

Request Sample:

--- Request Sample Code Start ---

```
PATCH /v2/Groups/e9e30dba-f08f-4109-8486-d5c6a331660a HTTP/1.1
Accept: application/json
Authorization: Bearer 9ea8b38f80038981fe38f60419672646
Host: democompany.com
Content-Length: ...
Content-Type: application/json

{ "schemas":
    ["urn:ietf:params:scim:api:messages:2.0:PatchOp"],
    "Operations":[
      {
       "op":"add",
       "path":"members",
       "value":[
        {
          "display": "Ola Nordmann",
          "$ref": "https://dempcompany.com/v2/Users/2819c223-7f76-453a-919d-413861904646",
          "value": "2819c223-7f76-453a-919d-413861904646"
        }
       ]
      }
    ]
  }
```

--- Request Sample Code Stop ---

Table 42.

Response Sample:

--- Response Sample Code Start ---

```
HTTP/1.1 201 No Content
Content-Type: application/json
Location: https://democompany.com/v2/Groups/ e9e30dba-f08f-4109-8486-d5c6a331660a
Content-Length: ...
Content-Type: application/json
```

--- Response Sample Code Stop ---

Table 43.

If the user was already a member of this group, no changes should be made to the resource, and a success response should be returned. The server responds with either the entire updated Group or no response body.

9.1.12 Remove User from Group

The following example shows how to remove a member to a group.

Request Sample:

--- Request Sample Code Start ---

```
PATCH /v2/Groups/e9e30dba-f08f-4109-8486-d5c6a331660a HTTP/1.1
Accept: application/json
Authorization: Bearer 9ea8b38f80038981fe38f60419672646
Host: democompany.com
Content-Length: ...
Content-Type: application/json

{ "schemas":
    ["urn:ietf:params:scim:api:messages:2.0:PatchOp"],
    "Operations":[
     {
      "op":"remove",
      "path":"members",
      "value":[
       {
         "value": "2819c223-7f76-453a-919d-413861904646"
       }
      ]
     }
   ]
  }
```

--- Request Sample Code Stop ---

Table 44.

Response Sample:

--- Response Sample Code Start ---

```
HTTP/1.1 201 No Content
Content-Type: application/json
Location: https://democompany.com/v2/Groups/ e9e30dba-f08f-4109-8486-d5c6a331660a
Content-Length: ...
Content-Type: application/json
```

--- Response Sample Code Stop ---

Table 45.

--- o0o ---