Security Concepts

SAML

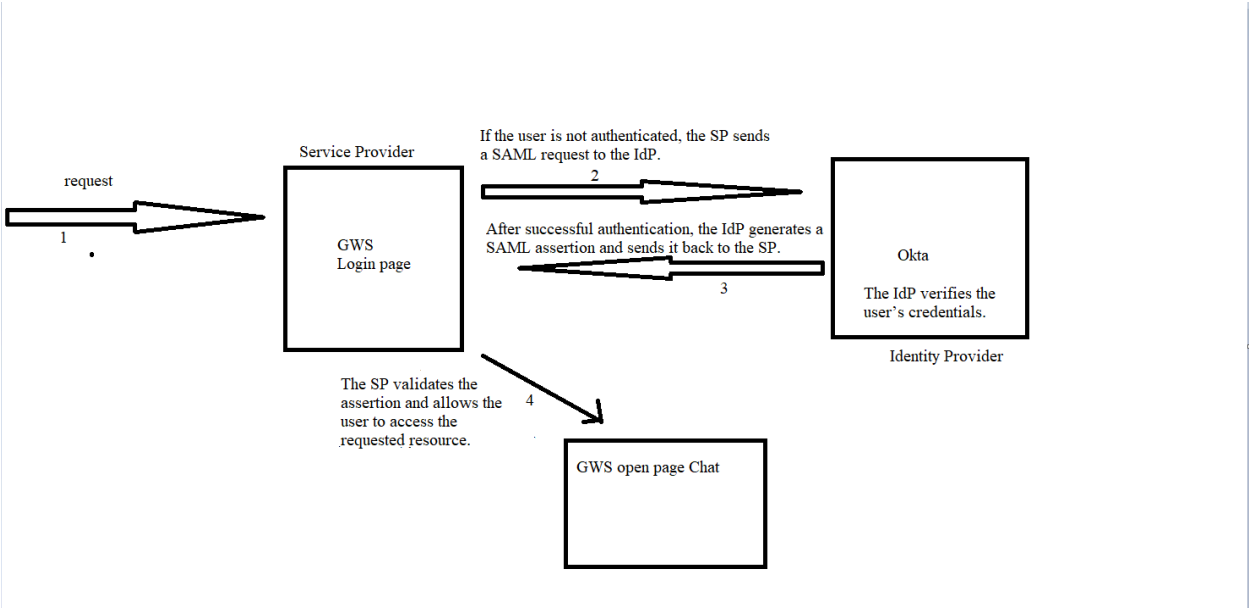
SAML (Security Assertion Markup Language) is an open standard for authentication and authorization that enables Single Sign-On (SSO) across multiple applications. It allows identity providers (IdPs) to securely pass authentication credentials to service providers (SPs), enabling seamless user authentication without requiring multiple logins.

Key Components of SAML:

1. **Identity Provider (IdP)** – The entity that authenticates users and issues SAML assertions (e.g., Okta, Azure AD, ADFS).
2. **Service Provider (SP)** – The application or service that relies/depends on the IdP for authentication (e.g., Salesforce, Google Workspace).
3. **SAML Assertion** – An XML document containing authentication and authorization information.
4. **SAML Protocol** – Defines how SAML requests and responses are exchanged.
5. **SAML Binding** – Specifies how SAML messages are transported (e.g., HTTP-Redirect, HTTP-POST).

How SAML SSO Works:

1. **User Requests Access** – The user attempts to access a protected resource at the SP.
2. **SP Redirects to IdP** – If the user is not authenticated, the SP sends a SAML request to the IdP.
3. **User Authenticates** – The IdP verifies the user's credentials.
4. **IdP Sends SAML Response** – After successful authentication, the IdP generates a SAML assertion and sends it back to the SP.
5. **SP Grants Access** – The SP validates the assertion and allows the user to access the requested resource.



SAML SSO Flow Diagram

```
+----------------+       +----------------+       +----------------+
|    User        |       |  Service       |       | Identity       |
| (Browser/App)  | ----> |  Provider (SP) | ----> | Provider (IdP) |
|                |       |                |       | (Auth System)  |
+----------------+       +----------------+       +----------------+
        |                        |                        |
(1) User Requests Access         |                        |
        |                        |                        |
        |  (2) Redirect to IdP   |                        |
        |----------------------> |                        |
        |                        |                        |
        |                        |  (3) User Authentication |
        |                        |----------------------> |
        |                        |                        |
        |                        |  (4) Generate SAML Token|
        |                        | <----------------------|
        |                        |                        |
        |  (5) SAML Response     |                        |
        | <--------------------- |                        |
        |                        |                        |
        |  (6) Validate Token    |                        |
        |----------------------> |                        |
        |                        |                        |
        |  (7) Grant Access      |                        |
        | <--------------------- |                        |
+----------------+       +----------------+       +----------------+
|    User        |       |  Service       |       | Identity       |
| (Browser/App)  |       |  Provider (SP) |       | Provider (IdP) |
+----------------+       +----------------+       +----------------+
```

Step-by-Step Explanation:

1. **User Requests Access** – The user tries to access an application (SP).
2. **SP Redirects to IdP** – The SP sends a SAML authentication request to the IdP.
3. **User Authenticates** – The IdP asks the user for credentials (if not already logged in).
4. **Generate SAML Assertion** – Once authenticated, the IdP generates a signed SAML response.
5. **SAML Response to SP** – The IdP sends the SAML assertion back to the SP.
6. **SP Validates Token** – The SP verifies the SAML assertion's validity.
7. **User Granted Access** – If valid, the user is authenticated and granted access to the application.

Below are examples of a **SAML Authentication Request (SP → IdP)** and **SAML Response (IdP → SP)** to help you debug authentication issues in Volt MX Foundry.

1. SAML Authentication Request (SP → IdP)

When a user tries to log in, Volt MX (Service Provider) sends this **AuthnRequest** to the Identity Provider (IdP):

```
<samlp:AuthnRequest xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
    ID="_12345abc"
    Version="2.0"
    IssueInstant="2025-01-30T12:00:00Z"
    Destination="https://idp.example.com/sso"
    AssertionConsumerServiceURL="https://sp.example.com/auth/saml/acs"
    ProtocolBinding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
    ForceAuthn="false">

    <saml:Issuer xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
        https://sp.example.com
    </saml:Issuer>

    <samlp:NameIDPolicy AllowCreate="true"
        Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress"/>
</samlp:AuthnRequest>
```

Key Fields:

- **ID**: Unique identifier for the request.
- **Destination**: The IdP's SSO endpoint.
- **AssertionConsumerServiceURL**: Where the IdP should send the response.
- **Issuer**: Identifies the SP (Volt MX).

---

2. SAML Response (IdP → SP)

If authentication is successful, the IdP sends this **SAML Response** to Volt MX:

```
<samlp:Response xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
    ID="_response123"
    Version="2.0"
```

```
IssueInstant="2025-01-30T12:01:00Z"
Destination="https://sp.example.com/auth/saml/acs"
InResponseTo="_12345abc">

<saml:Issuer xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
   https://idp.example.com
</saml:Issuer>

<samlp:Status>
   <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
</samlp:Status>

<saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
   ID="_assertion456"
   IssueInstant="2025-01-30T12:01:00Z"
   Version="2.0">

   <saml:Issuer>https://idp.example.com</saml:Issuer>

   <saml:Subject>
      <saml:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress">
         user@example.com
      </saml:NameID>
      <saml:SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
         <saml:SubjectConfirmationData NotOnOrAfter="2025-01-30T12:10:00Z"
            Recipient="https://sp.example.com/auth/saml/acs"
            InResponseTo="_12345abc"/>
      </saml:SubjectConfirmation>
   </saml:Subject>

   <saml:Conditions NotBefore="2025-01-30T12:00:00Z"
                    NotOnOrAfter="2025-01-30T12:10:00Z">
      <saml:AudienceRestriction>
         <saml:Audience>https://sp.example.com</saml:Audience>
      </saml:AudienceRestriction>
   </saml:Conditions>

   <saml:AuthnStatement AuthnInstant="2025-01-30T12:01:00Z">
      <saml:AuthnContext>
         <saml:AuthnContextClassRef>
```

```
            urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
          </saml:AuthnContextClassRef>
        </saml:AuthnContext>
      </saml:AuthnStatement>

      <saml:AttributeStatement>
        <saml:Attribute Name="FirstName">
          <saml:AttributeValue>John</saml:AttributeValue>
        </saml:Attribute>
        <saml:Attribute Name="LastName">
          <saml:AttributeValue>Doe</saml:AttributeValue>
        </saml:Attribute>
        <saml:Attribute Name="Role">
          <saml:AttributeValue>Admin</saml:AttributeValue>
        </saml:Attribute>
      </saml:AttributeStatement>

    </saml:Assertion>
</samlp:Response>
```

Key Fields:

- **StatusCode**: "Success" means authentication was successful.
- **NameID**: The authenticated user's email.
- **Conditions**: Defines the valid time range for the assertion.
- **AudienceRestriction**: Ensures the response is intended for Volt MX.
- **AuthnStatement**: Authentication method used (PasswordProtectedTransport).
- **AttributeStatement**: Contains additional user details like name and role.