EDUSERV IT · IAM PLAYBOOK

SSO UNLOCKED

Federation & SSO Playbook

Federation & SSO in IAM — 8 Sections for Beginners & Professionals

WHAT'S INSIDE — AT A GLANCE

This cheat sheet covers 8 essential sections on Federation & SSO in IAM — from first principles to real systems, common mistakes, and interview readiness.

01	**Definition & Why It Exists**	What SSO & Federation are and the problem they solve
02	**Key Components**	Every building block involved in SSO & Federation
03	**Real-World SSO Flow**	Step-by-step: how SAML SSO works in a live system
04	**Real-World Federation Flow**	Step-by-step: how cross-org Federation works
05	**Real-World Scenario**	Enterprise login with Alex, Okta & Salesforce
06	**Common Mistakes**	High-value lessons engineers get wrong
07	**Architecture View**	Where Federation & SSO sit in your system
08	**Interview Power Points**	Key facts expected in IAM interviews

- **SSO = one login, many apps**
- **IdP = the system that vouches for you**
- **SP = the app you want to access**
- **Federation = trust between different organisations**

01 DEFINITION & WHY IT EXISTS

What they are and the real problem they solve

DEFINITION

SSO — Log in once — access all your applications without re-entering credentials.

Federation — A trust agreement letting separate organisations accept each other's verified identities.

KEY CONCEPTS

- **The Problem** — Every app requiring a separate login creates password fatigue, weak reused passwords, and slow offboarding. One breach can compromise everything.
- **SSO solves this** — One login at the IdP unlocks all connected apps. Offboard once—access is cut off at the source. No more managing hundreds of separate credentials.
- **Federation extends it** — SSO works within one organisation. Federation lets a user from Org A access Org B's systems using their own Org A credentials. No new account. No shared password.
- **Why it matters** — Enterprises use 100+ SaaS apps. Compliance frameworks (SOC2, HIPAA, ISO27001) require fast deprovisioning. B2B access requires Federation — not shared accounts.

REAL-WORLD EXAMPLE

— You arrive at work and log into your laptop once.

— You open Slack, Salesforce, GitHub, and AWS.

— None of them ask for your password again.

— Your IdP (Okta) vouches for you to each app — that's SSO.

— A partner company's engineer accesses your Jira using their own company login — that's Federation.

WHY IT MATTERS IN INDUSTRY

— Password sprawl is a top breach vector — SSO eliminates most of it.

— One disable in the IdP cuts all connected apps instantly.

— Zero Trust demands verified identity on every access — SSO enables this at scale.

— B2B supply chain access requires Federation — not shared internal accounts.

02 KEY COMPONENTS INVOLVED

The building blocks of every SSO & Federation system

DEFINITION

SSO and Federation involve multiple components working together — not just a redirect to a login page.

KEY CONCEPTS

- **Identity Provider (IdP)** — The trusted authority that authenticates users and issues signed assertions or tokens — Okta, Azure AD, Google Workspace, Ping Identity.
- **Service Provider (SP)** — The application the user wants to access — Salesforce, GitHub, Slack, AWS Console. It relies on the IdP to confirm identity. Never handles passwords.
- **Browser** — The messenger. In SAML flows it carries the authentication request from SP to IdP and delivers the signed assertion back to the SP.
- **Session** — After SSO login, a session exists at the IdP. As long as that IdP session is active, the user can access all connected apps without logging in again.
- **Trust Relationship** — The pre-established agreement between IdP and SP — certificates exchanged, metadata shared, endpoints configured. No trust setup = no SSO.
- **SAML 2.0** — Enterprise SSO protocol using signed XML assertions delivered via the browser. Common in corporate apps — Salesforce, Workday, ServiceNow.
- **OIDC / OAuth 2.0** — Modern SSO standard. OAuth 2.0 handles authorisation (what can be accessed). OIDC adds authentication (who the user is) using signed JWTs. Used in web, mobile, and APIs.
- **Single Logout (SLO)** — Coordinated logout that terminates sessions at the IdP AND all SPs simultaneously. Must be explicitly configured — not automatic by default.

REAL-WORLD EXAMPLE

— "Sign in with Google" uses OIDC — Google is the IdP, the app is the SP.

— Google issues a signed JWT confirming your identity.

— The app validates the token — your password never reaches the app.

— Enterprise apps like Salesforce use SAML 2.0 instead of OIDC.

WHY IT MATTERS IN INDUSTRY

— SAML = enterprise legacy apps. OIDC = modern web, mobile, APIs.

— OAuth 2.0 alone does NOT prove identity — OIDC adds the identity layer on top.

— The IdP session is the key to SSO — it is what allows silent re-authentication.

— No metadata exchange = no Federation trust = assertion rejected on sight.

03 REAL-WORLD SSO FLOW

Step-by-step — How SAML SSO actually works in a live system

DEFINITION

SSO follows a structured flow: user requests app → redirected to IdP → authenticated → signed assertion issued → SP validates → access granted.

SAML FLOW

1	**User Requests App** *User → SP*	User clicks a protected app. No active session — the SP redirects them to the IdP login page.
2	**SP Redirects to IdP** *SP → IdP*	SP sends a signed SAML AuthnRequest to the IdP via the browser. The app never sees the user's password.
3	**User Authenticates** *IdP verifies*	User logs in at the IdP with credentials + MFA. The IdP verifies WHO the user is and creates an IdP session.
4	**IdP Issues Assertion** *IdP → Browser*	IdP creates a signed XML assertion: "I verified this user. Their name is X, role is Y." Signed with the IdP's private key.
5	**Browser POSTs to SP** *Browser → SP*	Browser delivers the XML assertion directly to the SP's ACS endpoint via HTTP POST — front-channel delivery.
6	**SP Validates & Grants** *SP → User*	SP validates the IdP's signature using the pre-exchanged certificate. Valid = access granted. No password needed.

OIDC — HOW IT DIFFERS

OIDC (modern web & mobile) follows the same concept but with one critical difference: instead of delivering the token through the browser, the IdP returns a short-lived **authorisation code** to the browser. The app then exchanges that code for the actual JWT token via a secure **server-to-server back-channel call** — the token never travels through the browser. This is what makes OIDC more secure for modern apps and APIs. For mobile apps and SPAs, **PKCE** (Proof Key for Code Exchange) is also required to prevent code interception.

KEY INSIGHT	The SP never handles the user's password — ever. It only receives a signed assertion from the IdP and validates the digital signature. The trust is entirely in the IdP's cryptographic signature — not in the user's credentials.

04 REAL-WORLD FEDERATION FLOW

Step-by-step — How cross-organisation Federation actually works

DEFINITION

Federation lets a user from Org A access Org B's systems using their own Org A credentials — no new accounts needed at Org B.

FEDERATION FLOW

1	**Trust Setup (One-Time)** *IdP-A ↔ IdP-B*	Org A and Org B exchange SAML metadata — certificates, entity IDs, and endpoint URLs. This establishes the trust. Without this, Federation cannot work.
2	**User from Org A requests access** *User → SP (Org B)*	An Org A employee clicks a resource at Org B. Org B's SP detects no active session and triggers the flow.
3	**SP redirects to Org A's IdP** *SP → IdP-A*	Org B's SP redirects the user to Org A's IdP. Org B trusts Org A to verify their own users.
4	**User authenticates at Org A** *IdP-A verifies*	The user logs in with their Org A credentials. Org B never sees these credentials.
5	**Org A IdP issues assertion** *IdP-A → Browser*	Org A's IdP issues a signed assertion: "I vouch for this user." Signed with Org A's private certificate.
6	**Org B SP validates** *Browser → SP (Org B)*	Org B validates the assertion against Org A's certificate — exchanged in Step 1. The trust relationship makes this possible.
7	**Access granted at Org B** *SP → User*	Validation passes. The user accesses Org B's system using their Org A identity. No new password. No new account.

KEY INSIGHT	SSO = one login across one organisation's apps. Federation = one organisation trusting another organisation's IdP. The metadata exchange in Step 1 is what makes Federation possible — it is the foundation of the trust relationship between two separate organisations.

05 REAL-WORLD SCENARIO

Enterprise Login — Alex, Okta & Salesforce behind the scenes

SCENARIO

1	**Alex opens Salesforce** *Browser / SP*	Alex clicks the Salesforce bookmark. No active session. Salesforce detects this and triggers the SSO flow.
2	**Salesforce redirects to Okta** *SP → IdP*	Salesforce sends a SAML AuthnRequest to Okta, redirecting Alex's browser to the Okta login page.
3	**Alex logs into Okta** *IdP Authentication*	Alex enters corporate credentials and approves an MFA push. Okta verifies WHO Alex is and creates an IdP session.
4	**Okta issues a SAML assertion** *IdP → Browser*	Okta creates a signed XML assertion: "Alex is authenticated. Role = Sales-Rep. Email = alex@company.com."
5	**Browser delivers to Salesforce** *Browser → SP*	Browser POSTs the assertion to Salesforce's ACS URL. Salesforce validates Okta's signature against the certificate in its metadata.
6	**Alex accesses Salesforce** *Access Granted*	Signature valid. Alex is logged in. No Salesforce password was ever needed.
7	**Alex opens Slack next** *SSO in Action*	Slack detects no session and sends its own AuthnRequest to Okta. Because Alex's Okta session cookie is still active, Okta silently issues a new assertion for Slack — no second login required.
8	**Alex's account deprovisioned** *Offboarding*	Alex leaves. One disable in Okta terminates the IdP session. All apps — Salesforce, Slack, GitHub, AWS — lose access simultaneously.

06 COMMON MISTAKES

High-value lessons — what beginners and engineers get wrong

×	**SSO and Federation are the same**	SSO = one login within one organisation. Federation = trust between separate organisations. Federation enables cross-org SSO — they are related but not identical.
×	**OAuth 2.0 authenticates users**	OAuth 2.0 is an authorisation framework — it grants access to resources. It does NOT prove identity. OIDC is built on top of OAuth 2.0 to add authentication. Never use OAuth 2.0 alone for login.
×	**SSO automatically improves security**	SSO centralises risk at the IdP. A compromised IdP exposes every connected app. MFA at the IdP is non-negotiable — SSO without MFA can make things worse, not better.
×	**Trust relationship is automatic**	Federation requires explicit setup — metadata exchange, certificate sharing, endpoint configuration between both parties. No setup = assertion rejected, even if it's correctly signed.
×	**Logout from one app logs out everywhere**	Single Logout (SLO) is a separate, explicitly configured flow — not automatic. Without SLO, logging out of Salesforce leaves active sessions in Slack, GitHub, and AWS.

07 ARCHITECTURE VIEW

Where Federation & SSO sit in your system

◆ **Centralised Auth Layer**	The IdP sits between all apps and the user. Every app delegates authentication to it — no app handles passwords independently.
◆ **SP-Initiated vs IdP-Initiated**	SP-initiated: user tries to access app → redirected to IdP. IdP-initiated: user logs into IdP portal → launches app. IdP-initiated is less secure — no AuthnRequest to validate.
◆ **Front-channel vs Back-channel**	SAML: token travels via browser (front-channel). OIDC: token exchanged server-to-server (back-channel). Back-channel is more secure — token never travels through the browser.
◆ **Federation Trust Model**	IdP and SP exchange metadata in advance. This pre-established trust is what makes assertions trustworthy across organisations.
◆ **Single Logout (SLO)**	Terminates sessions at the IdP AND all SPs simultaneously. Must be explicitly configured — not automatic in most deployments.
◆ **Audit Trail**	Every login, assertion, and logout is logged at the IdP. Essential for compliance and detecting anomalous access.

08 INTERVIEW POWER POINTS

Key facts — expect these in IAM interviews

Q1	**SSO = one login grants access to multiple apps within one organisation. Federation = trust between separate organisations — related but not the same concept.**
Q2	**IdP = authenticates the user and issues signed assertions or tokens. SP = the app that relies on the IdP. The SP never handles the user's password directly.**
Q3	**SAML 2.0 = XML-based, browser front-channel delivery. OIDC = JWT-based, token exchanged via secure back-channel. OAuth 2.0 = authorisation only — does NOT authenticate users.**
Q4	**Federation requires a pre-established trust relationship — metadata exchange (certificates, entity IDs, endpoints) between IdP and SP. No metadata = assertion rejected.**
Q5	**SP-initiated is more secure than IdP-initiated. Without an AuthnRequest to validate, IdP-initiated flows are vulnerable to CSRF attacks.**
Q6	**Single Logout (SLO) must be explicitly configured. Without it, logging out of one app leaves active sessions at all other SPs.**

MEMORY HOOK

Federation & SSO = One office ID card that opens every door in the building. You prove WHO you are at the front desk (IdP) once. Every door (SP) trusts the front desk's stamp — no re-checking needed. Federation = the front desk also accepts ID cards from partner buildings. SLO = handing back the ID card so ALL doors lock at once.