Single Sign-on für Webanwendungen

Chemnitzer Linuxtage 2024
17. März 2024

Silke Meyer
silke.meyer@univention.de
@smeyer@univention.social / @freiefunken@mastodon.social

univention

Univention

- Debian-basierte Appliance für Identity und Access Management (IAM) rund um OpenLDAP
  - zwei Ausrichtungen: allgemeiner Univention Corporate Server und UCS@school
- neu: containerbasiertes IAM mit den Kernkomponenten von UCS
  - Univention Nubus
- Zusammenarbeit mit öffentlichen Verwaltungen, Bildungsträgern, Firmen

Agenda

1. Use Cases für Web-SSO
2. Einführung in die beteiligten Komponenten
3. Die Protokolle SAML 2.0 und OIDC 1.0
4. Freie Identity Provider-Software: Shibboleth IdP und Keycloak
5. Demo

Use Cases für Single Sign-on

- Nutzer*innen-Perspektive: nur 1x einloggen und die IdP-Session im Browser für den Zugriff auf diverse Anwendungen nutzen
- Administrator*innen-Perspektive:
  - keine dienstspezifischen Credentials, Passwörter bleiben immer im IdM
  - Absicherung *eines* zentralen Logins (z.B. mit MFA)
  - Kontrolle über Attributfreigaben (Welcher Dienst bekommt welche weiteren Attribute aus dem IdM?)
- native Apps und Geräteauthentifizierung (OAuth/OIDC)

Einführung in die beteiligten Komponenten

Identity Provider

- Authentisierung:
  - eine oder mehrerer dahinterliegender Datenquellen mit den Nutzeraccounts
  - Identity Management System (IdM), Verzeichnisdienst, Datenbank, Webservice
- nach erfolgreichem Login Attributfreigabe:
  - Abfrage konfigurierter Regeln
  - Herausgabe ausgewählter Attribute an den anfragenden Dienst (z.B. User Identifier, Berechtigungsinformationen)

Service Provider (a.k.a. Relying Party)

- Wording: oft ist der angebundene Dienst selbst gemeint
- Software, die eine Ressource schützt und den Zugriff auf sie reguliert
  - Delegation des Logins an einen IdP
  - Entgegennahme von Attributen und Weitergabe an die eigentliche Anwendung
  - implementierungsspezifisch: Autorisierung aufgrund dieser Informationen in SP-Software oder Anwendung

Ablauf im einfachsten Fall

- Nutzer*in klickt im Browser bei Dienst „Login“
- Umleitung zur Loginseite eines IdP
- Login → IdP authentisiert → "Wer ist das?" Authentication
- Umleitung zur geschützten Anwendung → IdP überträgt Informationen (z.B. Attribute) an SP
- SP prüft, ob aufgrund der Infos Zugriff gegeben wird → „Was darf die hier?" Authorization
- Nutzer*in ist eingeloggt und darf Dinge tun
- Nutzer*in klickt auf zweiten angeschlossenen Dienst und ist schon angemeldet

Autorisierung von Zugriffen auf Dienst-Seite

- Autorisierungsstrategien
  - alle authentisierten Nutzer*innen
  - auf Basis von Gruppenzugehörigkeiten im IdM (GBAC), Bsp. Team
  - auf Basis von Rollen (RBAC), Bsp. alle in der Rolle Azubi
  - auf Basis von Attributen (ABAC), Bsp. alle mit einer Berechtigung → feingranular und flexibel
- im IdP: Konfiguration der richtigen Informationsfilter für jeden angebundenen Dienst
- unterschiedliche Umsetzung in Protokollen und in Software-Lösungen

Vertrauensstellung

- Vorab-Austausch bestimmter Eckdaten für die Kommunikation: Metadaten
  - eindeutige Identifier der Systeme
  - Kommunikationsendpunkte
  - Zertifikate für Signierung / Verschlüsselung
- optional: signierte Metadaten (mit Signaturvalidierung ;))
- Wie heißt die Gegenstelle? Mit welchen Zertifikaten kann die signierte Kommunikation validiert werden? Welche Kommunikationsstandards unterstützt die Gegenstelle? An welche Endpunkte sollen die Requests geschickt werden?
- SAML: xml / OIDC: JSON

Die Protokolle SAML 2.0 und OIDC 1.0

Die Protokolle SAML 2.0 und OIDC 1.0

SAML 2.0 (2005/2008)	OIDC 1.0 (2012)
Browser-only (fast)	Browser, native Apps, Geräte (versch. Flows)
xml-basiert, HTTP- / SOAP-Requests	JSON-basiert, HTTP-/REST-API
Authentisierung + Autorisierung	Erweiterung von OAuth 2.0 um Konzept von Identität
sehr ausgereift, sehr verbos	geringere Payload
Vertrauen über Vorab-Austausch von (signierten) xml-Metadaten	Vertrauen über Vorab-Austausch von Client ID, Client Secret u. Redirect URI
Spezifikation durch OASIS	Spezifikation durch OpenID Foundation

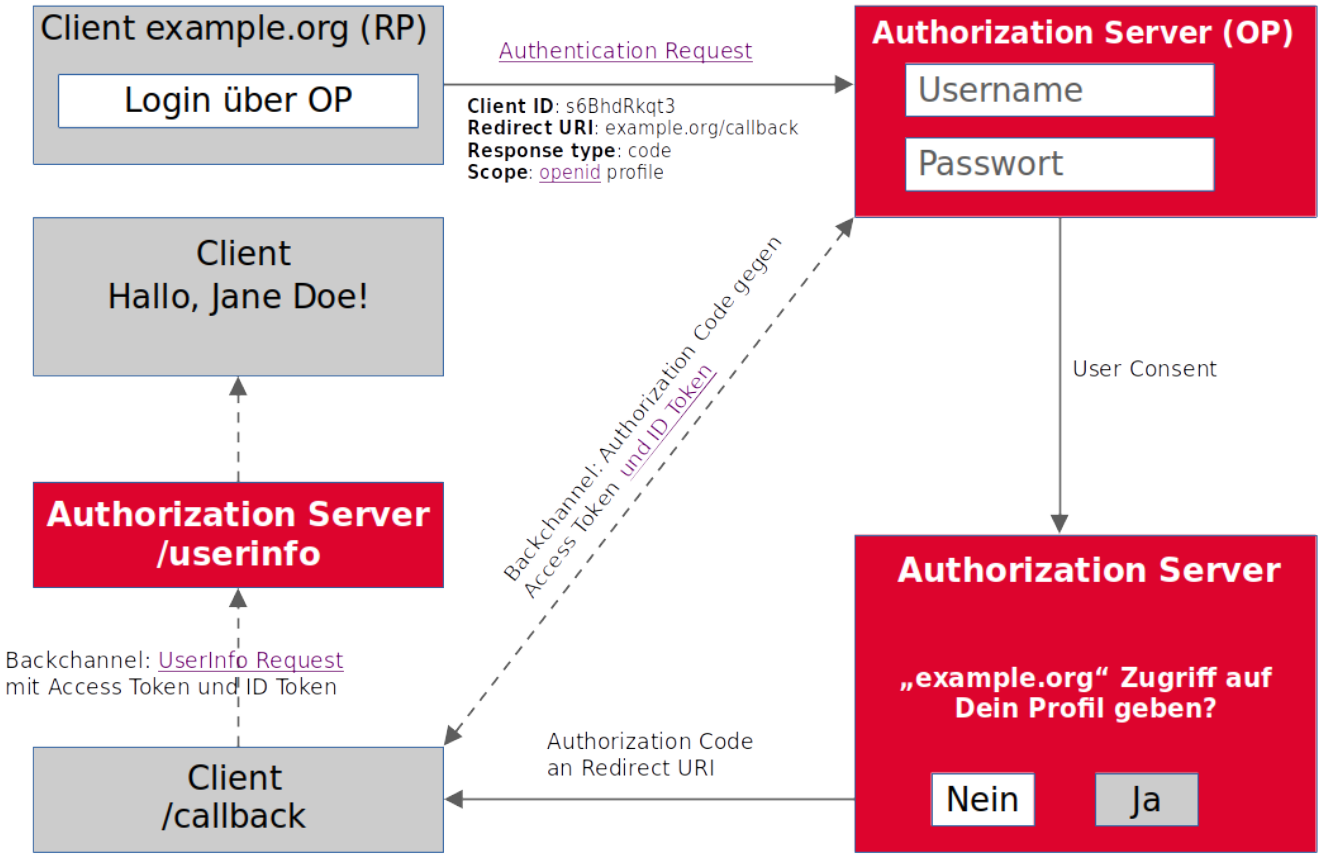
Terminologien (Auszug)

SAML	OIDC
Identity Provider / IdP	OpenID Connect Provider / OP
Service Provider / SP	Relying Party / RP
Entity ID (unique Identifier des IdP)	Client ID (unique Identifier der RP)
Attribut	Claim
-	Scope (thematisch gebündelte Claims)
SAML Assertion (xml)	Access Token, Refresh Token, ID Token (JWT)
Signieren/Verschlüsseln mit x509-Zert.	Signieren/Verschlüsseln mit JWK

Von OAuth 2.0 (2012) zu OIDC 1.0 (2014)

- OAuth 2.0
  - delegierte Autorisierung ohne Preisgabe des Passwortes
  - Informationen mit Dritten teilen (z.B. mit einer Website)
  - API-Autorisierung über Access Tokens (JSON Web Tokens)
- OIDC 1.0
  - Erweiterung von OAuth 2.0 um einen ID Token zur Abbildung einer Identität
  - i.d.R. generierte Identifier: subject claim (subject public, subject pairwise)
  - Angefragter zusätzlicher Scope „openid“ im Authentication request löst den Unterschied zu OAuth 2.0 aus

OIDC 1.0



Quelle: Nate Barbettini, Okta https://www.youtube.com/watch?v=996OiexHze0

OIDC Token Response (gekürzt)

- Übertragung im HTTP-Header
- HTTP/1.1 200 OK
  Content-Type: application/json
  Cache-Control: no-store
  Pragma: no-cache
  {
  "**access_token**": "SlAV32hkKG",
  "token_type": "Bearer",
  "**refresh_token**": "8xLOxBtZp8",
  "expires_in": 3600,
  "**id_token**": "eyJ0eXAiOiJKiJIUzI1NiJ9.EyJpc3MiOiJA4MTkzOD.dBjft4CVP-mhb1p1r_wWOEjXk"
  }

Freie Identity Provider-Software (Auswahl)

Shibboleth IdP und Keycloak

Shibboleth IdP	Keycloak IdP
Shibboleth Consortium, Apache 2.0-Lizenz	unter Dach von RedHat, Apache 2.0-Lizenz
Java + Datenbank	Java + Datenbank
SAML, CAS, m. Plugins: OIDC, OAuth 2.0	SAML, OIDC, OAuth 2.0
- Konfig über zig xml- und properties-Dateien	- Konfig über unintuitive Web-GUI
+ aussagekräftiges Logging	+ REST API und vorgesehener Betrieb in Cloud-Kontexten

SSO-Debugging

- An welcher Stelle im Login-Flow tritt der Fehler auf?
- Welche Seite meldet den Fehler? → beide Seiten betrachten, wenn möglich
- versch. Browser / Profile nutzen
- Logs
  - slapd-Log z.B. auf „stats“ setzen (syslog/Journal)
  - Shibboleth: 3 Einstellungen auf „DEBUG“:
    `idp.loglevel.idp, idp.loglevel.messages, idp.loglevel.encryption`
- Keycloak-Logging…
- Log der jeweiligen Client-Anwendung prüfen

SSO-Debugging

- Sind die in den JSON-Konfigurationen / xml-Metadaten publizierten Kommunikationsendpunkte erreichbar?
- Stimmt das tatsächlich verwendete Schlüsselmaterial überein mit dem in den JSON-Konfigurationen / xml-Metadaten angegebenen?
- Loadbalancer oder Proxy vor dem IdP?
- Welche Attribute / Claims erwartet die geschützte Anwendung? Welche werden vom IdP übertragen?

Demo

- xml-Metadaten, JSON-Konfigurationsangaben
- Konfiguration auf IdP-Seite:
  - Grundeinstellungen, IdM-Anbindung Shibboleth IdP / Keycloak IdP
  - SAML SP bzw. OIDC RP in Shib / KC
  - Freigaberegeln für angebundene Dienste in Shib / KC
- SAML-Login, OIDC-Login, Logdateien
- eine IdP-Session bietet Zugriff auf mehrere Dienste

Links

- Buch: Stian Thorgersen, Pedro Igor Silva: Keycloak – Identity and Access Management for Modern Applications, Packt Publishing, Birmingham/Mumbai 2023 (2. Aufl.) → Repo
- Shibboleth IdP: Doku, Download
- Keycloak
- Zum Spielen:
  - Apache-Modul OIDC RP: mod_auth_openidc
  - Apache-Modul Shibboleth SP mod_shib (SAML)
  - Blog-Artikel von Björn Schießle zur Nextcloud-Anbindung

Kontaktdaten

Silke Meyer
IT Consultant
<silke.meyer@univention.de>
+49 421 22232-106

Univention GmbH
Mary-Somerville-Str. 1
28359 Bremen

Univention GmbH Berlin
Mariannenstr. 9-10
10999 Berlin

Univention North America Inc.
7241 185th AVE NE #3206
Redmond, WA 98073-3206