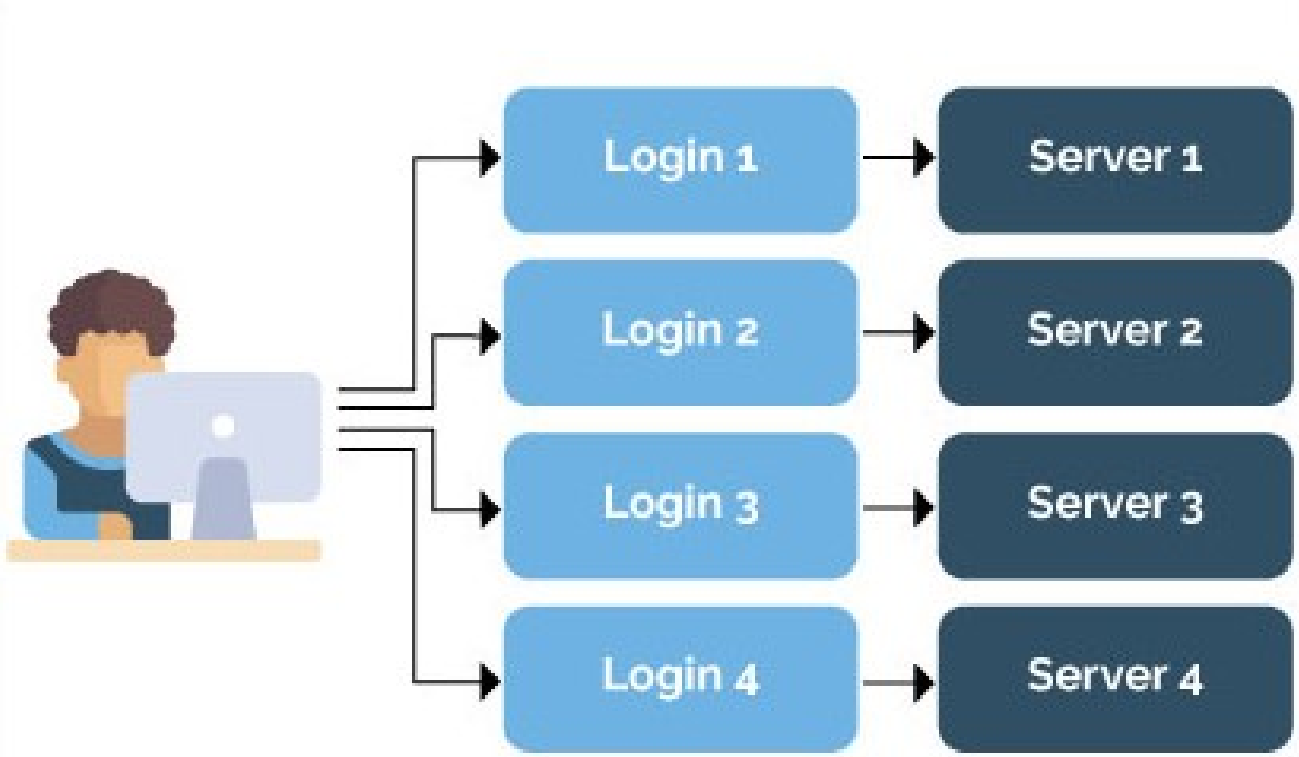

Login 1
Server 1
Login 2
Server 2
Login 3
Server 3
Login 4
Server 4

Quelle: [1]

Single Sign On bei Webanwendungen

Daniel Ebert 65926

Hochschule Aalen

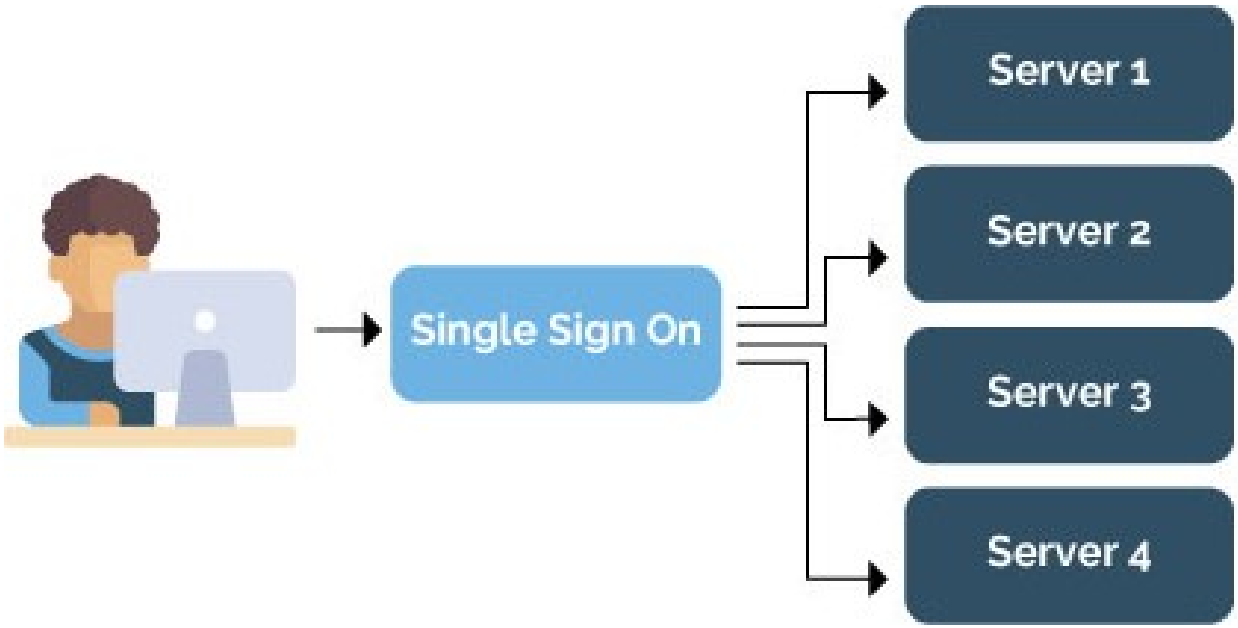
daniel.ebert01@studmail.htw-aalen.de

2. Januar 2021

Overview

SSO

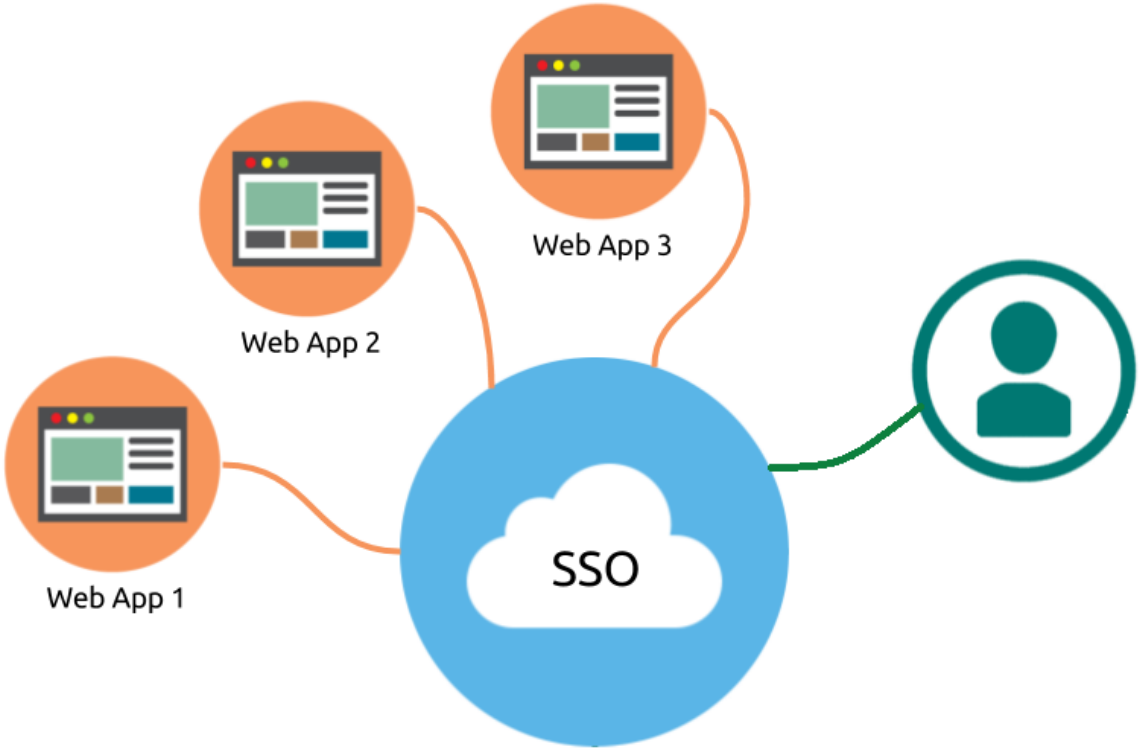
- Benutzer muss sich nur einmal mit Authentifizierungsverfahrens authentifizieren
- Danach übernimmt SSO-Mechanismus das Authentifizieren



Quelle: [1]

Beispiel SSO Use Case

- Eine Gruppe von Webseiten nimmt am SSO teil
- Benutzer loggt sich auf einer dieser Website mit Passwort ein
- Danach wird Benutzer auf anderen Websiten automatisch eingeloggt



Quelle: [2]

Vorteile

- Psychological Acceptability
  - Keine mehrfachen Anmeldungen
  - Ein Passwort für mehrere Webanwendungen
- Einfachere Entwicklung
  - Ein System zur Authentifizierung
- Einfachere Administration
  - Alle Benutzerdaten an einem Ort

Nachteile

- Single Point of Failure
  - Redundanz duch mehrere Keycloak Instanzen
- Bereits existierende Systeme an SSO anbinden kann schwierig sein
  - Keycloak Adapter vereinfachen Anbindung

Alternativen

- Verschiedene Arten von SSO
  - SSO für Webanwendungen
  - Alternative: SSO für Intranet/Enterprise
- Verschiedene Protokolle für SSO
  - OpenID Connect (OIDC)
  - Alternativen: SAML 2.0, Kerberos
- Verschiedene Implementierungen von SSO-Systemen
  - Keine zwei SSO Implementierungen sind gleich
  - Keycloak
  - Alternativen: Okta, Auth0, Apereo CAS

OIDC Teilnehmer/Rollen

- Benutzer
- Clients
- OpenID Provider

Benutzer

- Menschliche Teilnehmer
- Kann sich einloggen
- Benutzer hat Claims zugeordnet

Quelle: [3]

Claims

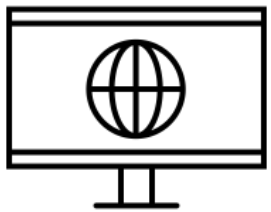
- Key-Value Paare
- Information über Benutzer, z.B. Name, Adresse, email
- Information über Authentifizierung, z.B. wann und bei wem hat Authentifizierung des Benutzers stattgefunden
- Beispiel Claims:
  - 'sub' (Subject) Claim: Benutzer ID, erstellt von Issuer
  - 'iss' (Issuer) Claim: Bei welcher Instanz authentifiziert

Quelle: [7]

Clients

Drei Arten von Clients:

1. Ruft andere Clients/Services im Namen des authentifizierten Benutzers auf
   - z.B. Frontend Anwendungen
   - Rufen geschützte Ressourcen von Backend Services ab
2. Backend Services die Ressourcen bereitstellen
   - z.B. Ressourcenserver
   - Ressourcen beschränkt für authentifizierte und authorisierte Benutzer
3. Native Anwendungen
   - Laufen auf Gerät des Benutzers
   - Für Web SSO nicht interessant



Quelle: [4, 5]

OpenID Provider

- Authentifiziert Benutzer
- Speichert z.B. Claims der Benutzer und Konfiguration der Clients
- Stellt Endpunkte bereit für:
  - Authentifizierung eines Benutzers
  - UserInfo Endpunkt (Informationen über Benutzer)
  - Token Endpunkt (Tokens mit späterer Verfallszeit und/oder weniger Rechten)

Quelle: [6]

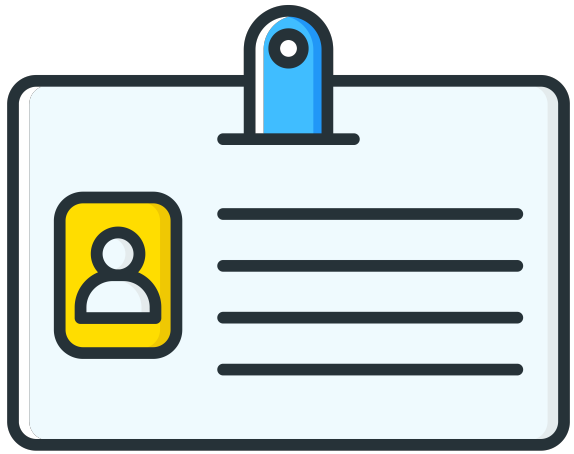
Tokens

- Tokens im JWT-Format
- Erstellt und Signiert vom OpenID Provider
- Verschiedene Arten von Tokens:
  - ID Token
  - Access Token
  - Refresh Token
- 3 teilige Struktur:

Quelle: [8]

ID Token

- Enthält Claims über die Authentifizierung des Benutzers
  - z.B. 'auth_time' Claim: Wann hat Authentifizierung stattgefunden
- Optional weitere Claims mit Informationen über den Benutzer
- Ist nur für den an der Authentifizierung des Benutzers beteiligten Client gedacht
- Client validiert mit ID Token die Authentifizierung des Benutzers



Quelle: [9]

Access Token

- Schlüssel für Ressourcen, welche für die authentifizierten und authorisierten Benutzer beschränkt ist
- Kann z.B. als Bearer Token im HTTP Authorization Header an Backend Services enthalten sein
- Sollte außer 'sub' Claim keine Informationen über Benutzer enthalten
- Informationen über Benutzer kann mit Access Token bei UserInfo Endpunkt angefordert werden
- Enthält 'scope' Claim
  - Spezifiziert Ressourcen und Benutzer Claims
  - Mit Access Token Zugriff hat man auf Spezifiziertes Zugriff
  - Ressourcen bei Backend Services
  - Benutzer Claims bei UserInfo Endpunkt

Access Token

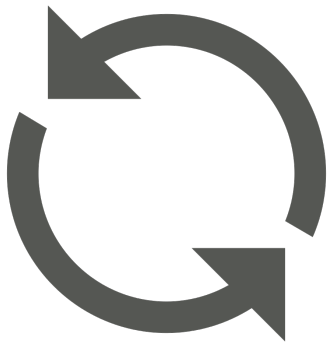
Access Token

Benutzer Claims
und Ressourcen
im scope Claim

Quelle: [10, 11]

Refresh Token

- OIDC Tokens haben Verfallszeit ('exp' Claim)
- Abgelaufene Tokens sind nicht mehr gültig
- Mit Refresh Tokens können über den Token Endpunkt neue Tokens angefordert werden
  - Optional neue Tokens mit weniger Rechten

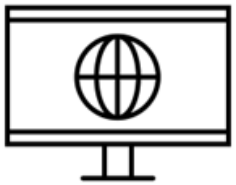


Quelle: [12]

OIDC Protokoll Flows

- Authorization Code Flow
- Zugriff auf geschützte Ressourcen
- Validieren des Access Tokens

Authorization Code Flow Schritt 1



Client

1. Authorization Request

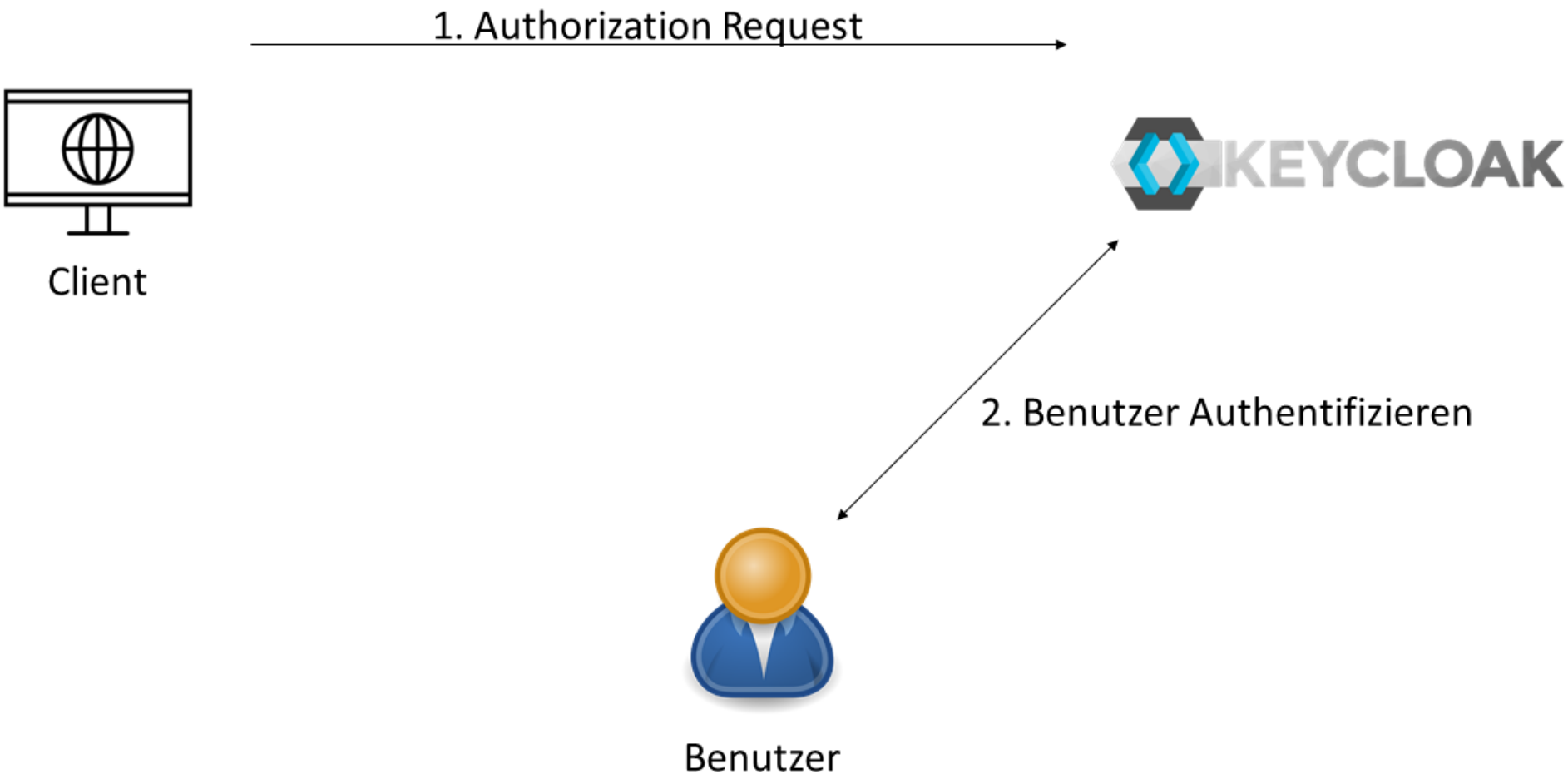
Benutzer

Authorization Code Flow Schritt 1 Beispiel

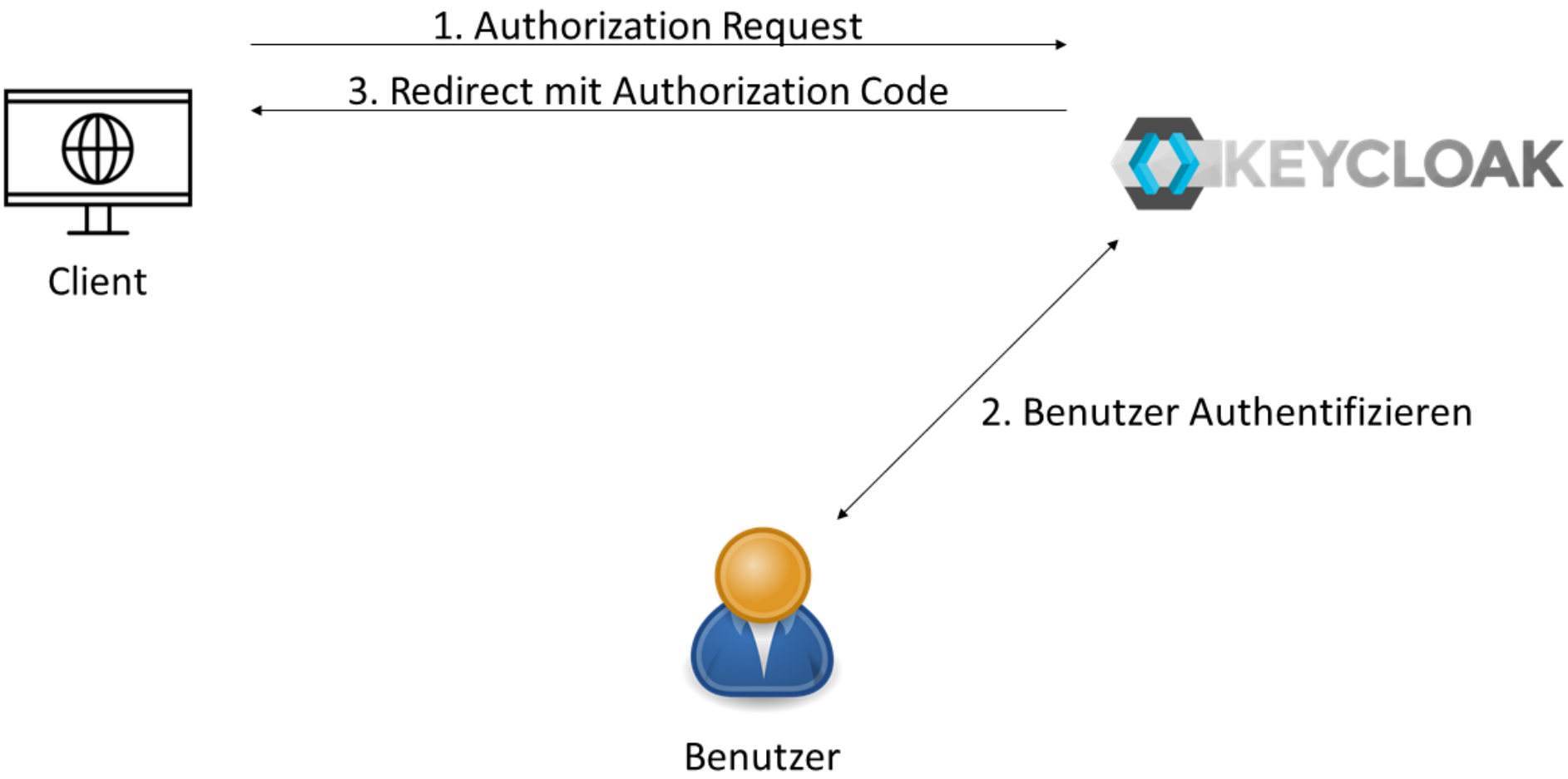
```
GET https://keycloak/auth/realms/.../auth?
        client_id=frontend1
        &redirect_uri=https%3A%2F%2Ffrontend.com%2F
        &state=0909ff6a-53b2-4253-8690-aff72d2cfff1
        &response_mode=fragment
        &response_type=code
        &scope=openid%20profile
        &nonce=67bad316-d8c1-45d1-9559-2a1c4726ce91
```

Listing 1: Beispiel Authorization Request

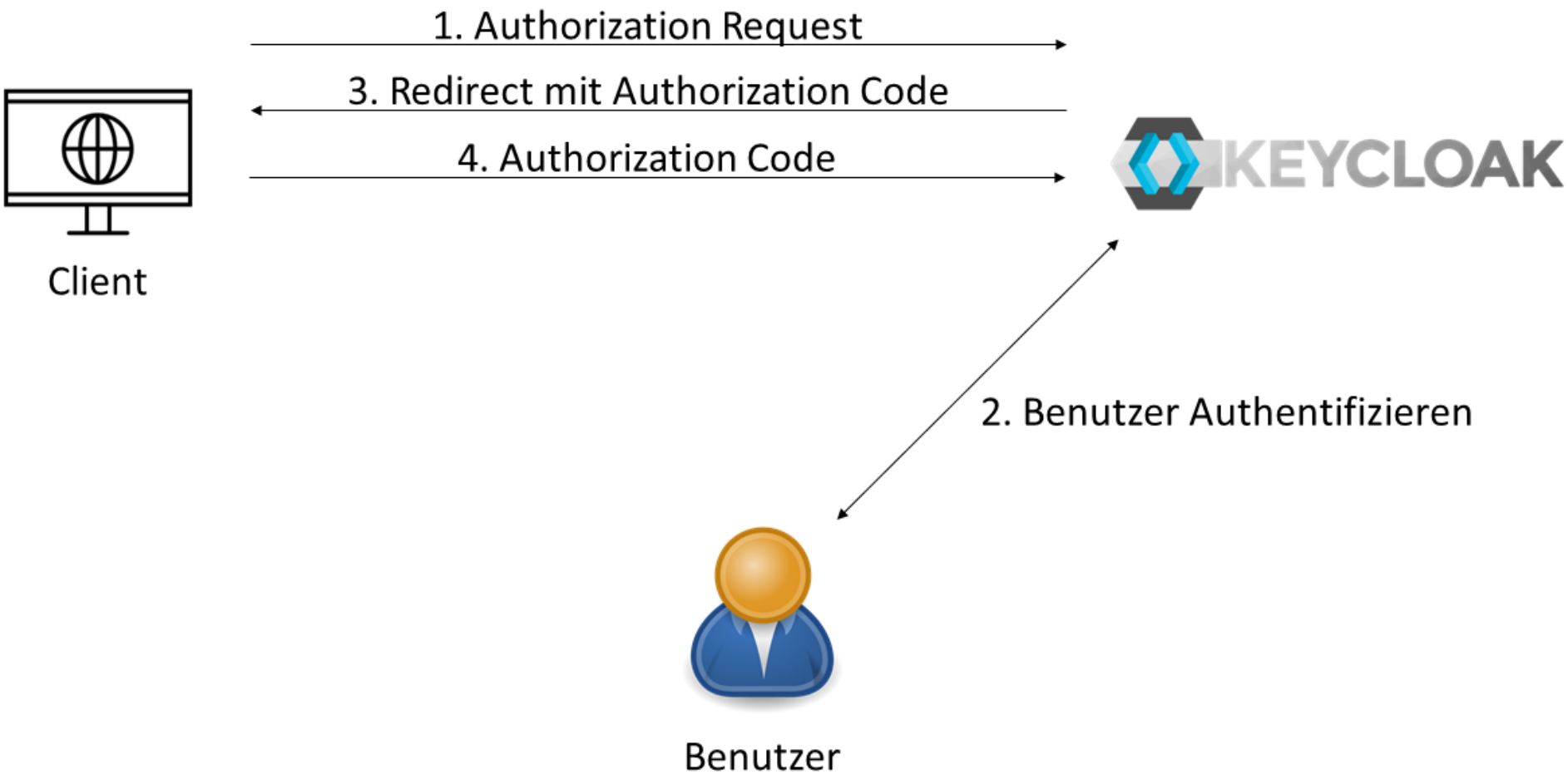
Authorization Code Flow Schritt 2



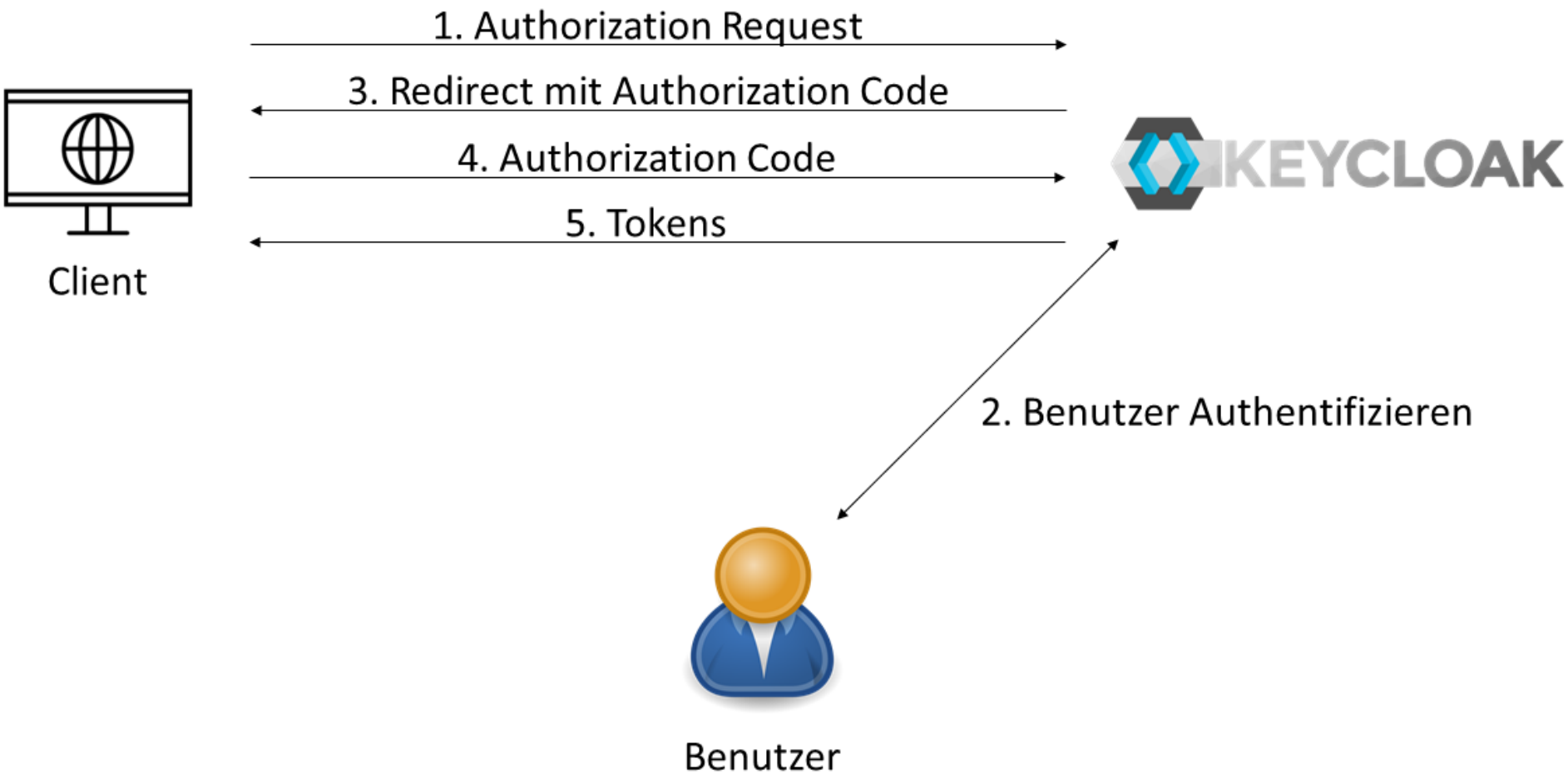
Authorization Code Flow Schritt 3



Authorization Code Flow Schritt 4

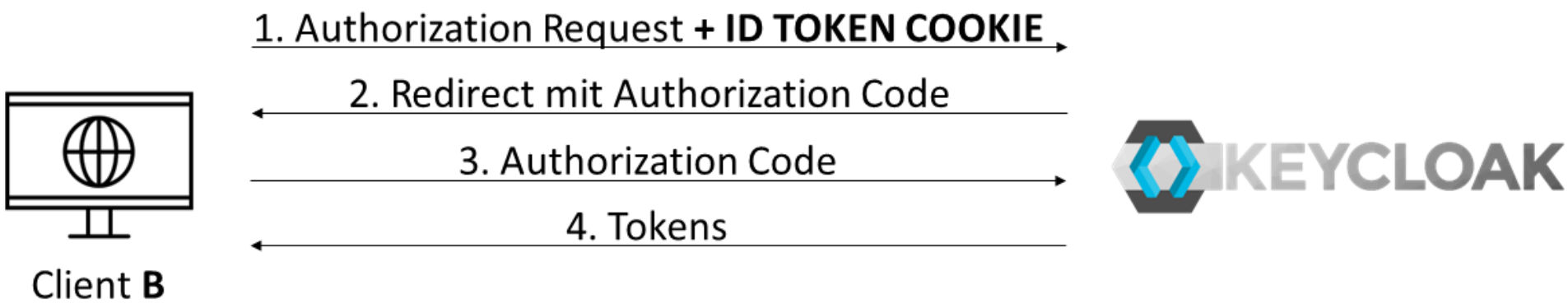


Authorization Code Flow Schritt 5



Zweite Authentifizierung

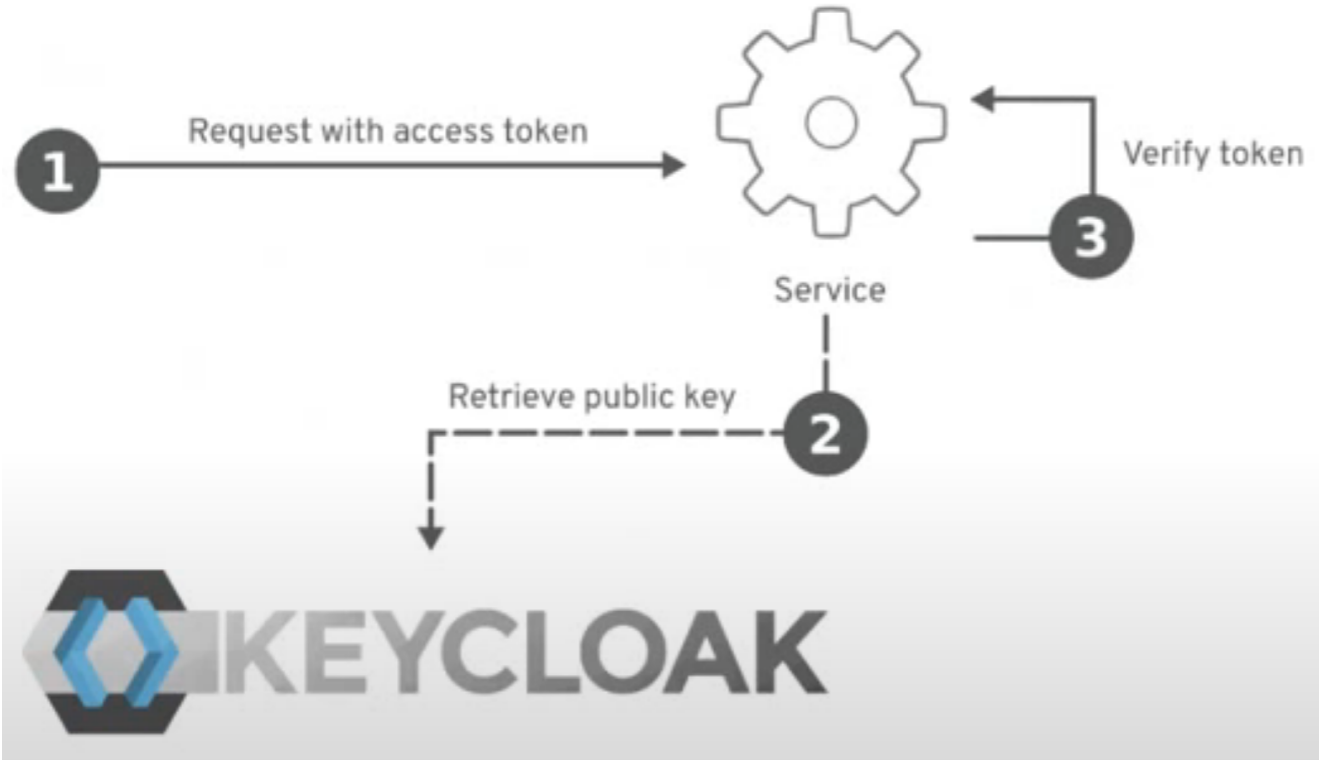
- Nach erster erfolgreicher Authentifizierung wird Cookie mit ID Token für Keycloak's Authorization Endpunkt gesetzt



Zugriff auf geschützte Ressourcen

- Access Token im Authorization Header der HTTP Anfrage
- Jeder kann Access Token einsetzen
- Einschränken der abrufbaren Ressourcen und Benutzer Claims durch 'scope' und Audience Claims des Access Tokens
- Empfänger muss 'scope' und Audience Claims überprüfen

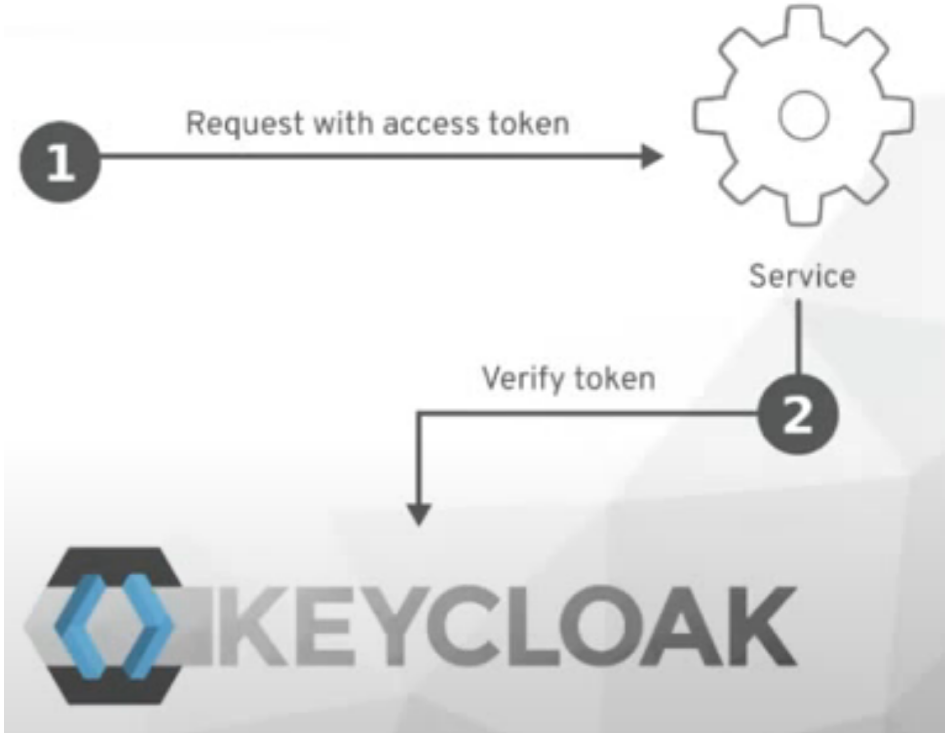
Validieren des Access Tokens - Offline Variante



- Public Key gecached
- Verringerte Latenz
- Verringerte Auslastung des Keycloak Servers
- Access Token kann nicht wiederrufen und invalidiert werden

Quelle: [13]

Validieren des Access Tokens - Online Variante



- Höhere Latenz
- Höhere Auslastung des Keycloak Servers
- Access Token kann wiederrufen und invalidiert werden

Quelle: [13]

Threat Model

- Client Schwachstellen
- Authorization Server Schwachstellen
- Token Schwachstellen

Client Schwachstellen

- Cross-Site-Request-Forgery (CSRF)
  - Z.B. Angreifer sendet gefälschte Authentifizierungsantwort an Client's redirect URI
  - State Token (Anti-CSRF Token)
- Registrierung der redirect_uri
  - Angreifer gibt sich als Client aus
  - Möglich wenn redirect_uri zu unspezifisch
- Diebstahl von Tokens
  - Z.B. durch XSS
- OIDC frei von Kryptographie
  - Verwendung von TLS voraussetzen, z.B. mit HSTS

Authorization Server Schwachstellen

- Redirection URI manipulieren
  - Authorization Code zu URI des Angreifers umleiten
- Session Hijacking
  - Z.B. Authorization Code aus Browser-Verlauf auslesen
  - Deshalb darf Authorization Code nur einmal eingelöst werden

Token Schwachstellen

- Token Modification
  - Empfänger von Access Token muss digitale Signatur überprüfen
- Token Replay
  - Gute Verschlüsselung, geringe Token Verfallszeit
- Token Redirect
  - Empfänger ID im Token
- Token Disclosure
  - Token selbst könnte sensible Informationen enthalten

Notiz an Herr Professor Karg

- Notiz an Herr Professor Karg: Abschließend folgt noch eine Demo der implementierten Beispielwebanwendung in der VM. Vom Inhalt wird diese Demo ungefähr so sein wie das Kapitel 1.4 "Umsetzung von SSO innerhalb einer Beispielwebanwendung" unserer Ausarbeitung.

Quellen Teil 1

1. https://www.renovodata.com/blog/2019/01/17/single-sign-on
2. https://insready.com/en/blog/single-sign-using-oauth2-and-jwt-distributed-architecture
3. https://commons.wikimedia.org/wiki/File:User_icon_2.svg
4. https://thenounproject.com/term/web-application/3085910/
5. https://icon-library.com/icon/web-service-icon-9.html
6. https://design.jboss.org/keycloak/logo/images/keycloak_logo_600px.png

Quellen Teil 2

7 https://upload.wikimedia.org/wikipedia/commons/5/59/OneDrive_Folder_Icon.svg; user data icon by Mohamad Arif Prasetyo from the Noun Project

8 https://www.toptal.com/web/cookie-free-authentication-with-json-web-tokens-an-example-in-laravel-and-angularjs

9 https://iconarchive.com/show/job-seeker-icons-by-inipagi/id-card-icon.html

10 https://de.cleanpng.com/png-jry6v5/

11 https://thenounproject.com/search/?q=resource&i=2979093 resource by Hrbon from the Noun Project

12 https://commons.wikimedia.org/wiki/File:Refresh_icon.svg

13 https://www.youtube.com/watch?v=mdZauKsMDiI

Danke & Ihre Fragen