Autorisierung im Web und in Microservices

Bernd Ua | probucon Business Consulting GmbH&Co KG

embarcadero®

Vorstellung

- Bernd Ua
  - Geschäftsführer von probucon
  - Trainer, Consultant und nicht zuletzt Entwickler
  - Mehr zwei Jahrzehnte Erfahrung im Delphi Umfeld
  - Program Chair der Delphi Entwicklerkonferenz Ekon
  - Embarcadero MVP

Bernd Ua

Themen

- Überblick
- Authentifizierung und Autorisierung mit Services
- JWT
- Einfache Authentifizierungsverfahren
- Beispiele und Umsetzung

AUTHORISIERUNG/ AUTHENTIFIZIERUNG

Autorisierung versus Authentifizierung

- Authentifizierung
  - Entspricht der Identifizierung von Personen, Nutzern
  - Nachweis der Identität
- Autorisierung
  - Den technischen Zugang/Zugriff für authentifizierte Nutzer gestatten

- Oder kurz
  - Authentifizierung -> Wer bin ich ?
  - Autorisierung -> Was darf ich ?

embarcadero®

Microservice Architekturen

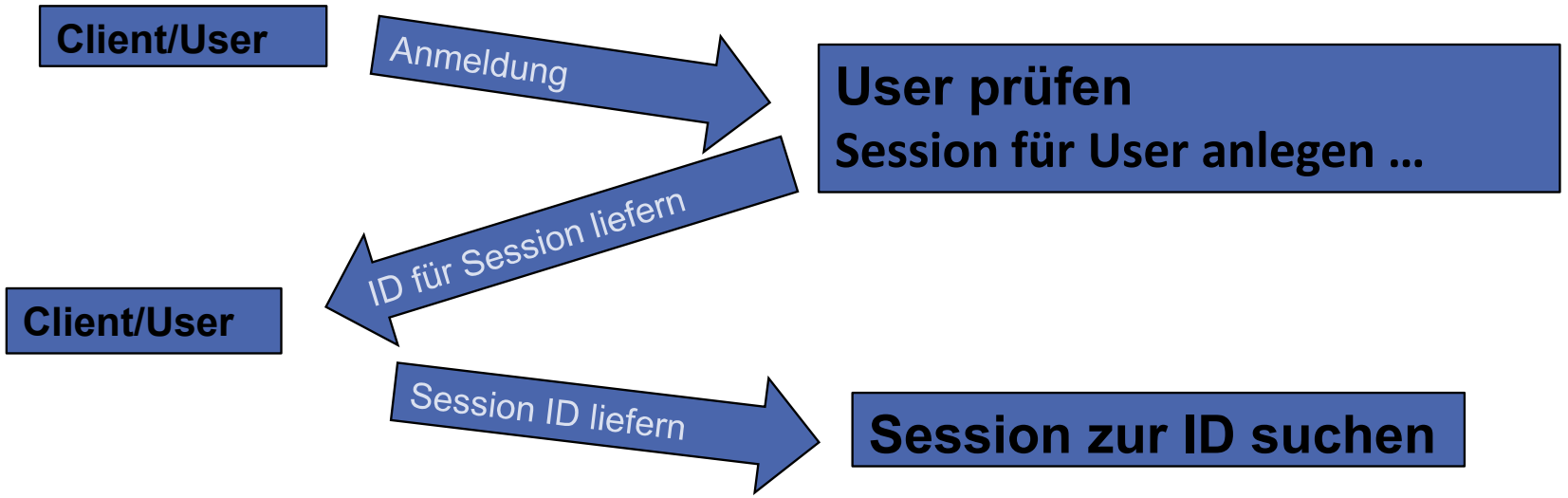
- Die klassische Client-Server-Anmeldung taugt hier wenig
- Sollen User-Passwörter durch alle Services geschleift werden ?
- Sollen Passwörter in zig Services hinterlegt und deployed werden ?

REST soll stateless sein

- Ist eine „klassische“ Anmeldung stateless ?
- Nein! Der Server „merkt“ sich pro Session/User die Autorisierungs-Informationen

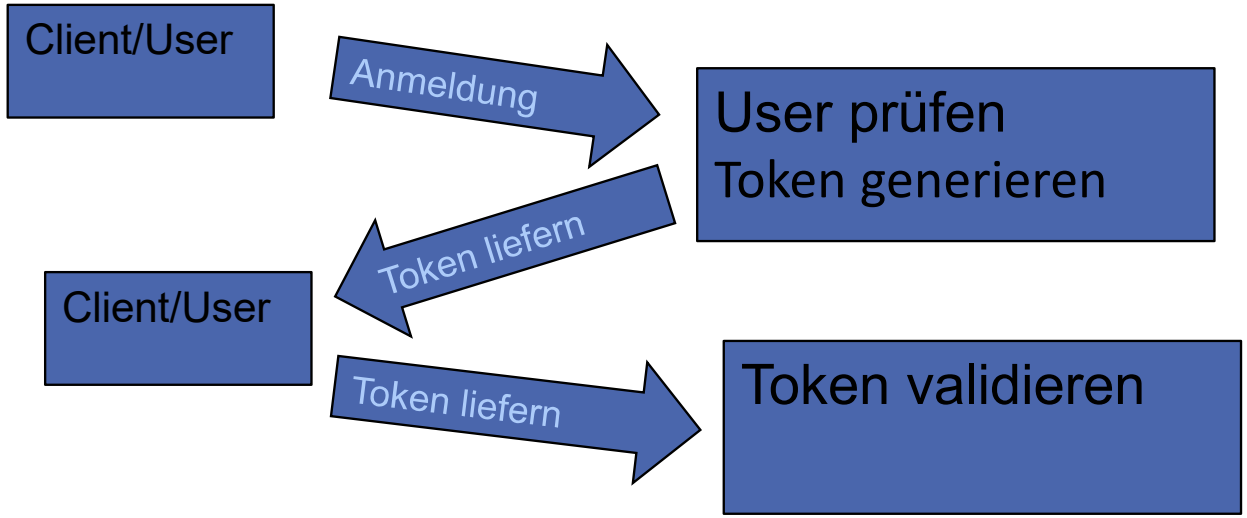
Probleme bei Sessions

- Sessions-Ids werden gerne als Cookie zum Client gesendet
- Anfällig für Cross-Site-Forgery-Attacken
- Skaliert schlecht



Anmeldungen ohne „state"

- Einmalige Userprüfung
- Danach liefern des Tokens mit jedem Request



Vorteile

- State am Client
  - Bessere Skalierung
  - Anmeldung geht bei Serverrestart nicht verloren
- Uservalidierung wird verkürzt
- Uservalidierung kann ausgelagert werden

Standards

- für Gestaltung und Übertragung von Tokens existieren Standards
- SAML V2.0 (Security Assertion Markup Language) – OASIS Standard von 2005 für SOAP
- Simple Web Token (war als Standard von MS eingereicht)

JWT

- Aktueller De facto Standard für das Ablegen von Daten und Login Informationen auf dem Server
- Ein nach RFC 7519 genormtes Access-Token = „JavaScript Web Token“
- Im Gegensatz zu klassischen Session-IDs mit Serverseitigen Daten skalierbar
- https://jwt.io/

Aufbau eines JWT Tokens

- Besteht aus drei Teilen
  - Header
  - Payload
  - Signatur

Inhalte

- Header
  - Im JSON wird der Typ der Content(optional) und der Signatur-Algorithmus festgehalten
    { "alg": "HS256", "typ": "JWT" }

Payload

- Enthält die Inhalte als JSON
- Einige Schlüssel sind reserviert (iss,sub,aud etc)
- Die Werte im payload werden auch als Claims bezeichnet

Signatur

- der Header und der Payload werden Base64 kodiert und durch einen Punkt getrennt
- Das Ergebnis wird mit der angegebenen Methode gehasht
- Anschließend werden alle Elemente Base64 codiert und mit . (Punkt) aneinander gehängt

Übertragung

- Erfolgt im HTTP-Header wahlweise
- im Authorization-Feld als Bearer-Token
  Authorization: Bearer bGcieyJhbGiJIU...
- Oder im im Cookie-Feld:
  Cookie: token=bGcieyJhbGiJIU...

Verfahren

- User sendet Username/Passwort
- Server validiert den User und sendet ein Token zurück
- Das Token ist kryptografisch signiert (und nicht verschlüsselt !)
- Mit den folgenden Aufrufen sendet der user das Token mit
- Der Server validiert es bei jedem Aufruf

Implementierungen in Delphi

- Jose JWT Library
- Unter anderem von MARS und TMS XDATA verwendet
- Die wichtigsten Klassen
  - TJWT kapselt einen Token
  - TJWK kapselt einen Schlüssel
  - TJose stellt Hilfsfunktionen wie Verify etc bereit

embarcadero®

Jose JWT Library

- Github Repository : https://github.com/paolo-rossi/delphi-jose-jwt
- Installation :
- Download von Github
- Entpacken
- Pfad zu Source/Common und Source/JOSE zum Suchpfad des Projekts oder sogar der Bibliothek zuordnen

Verwendung

- Gut geeignet für User-Security in einem Service
- Auch gut geeignet für separate Anmeldeserver
- Nicht vergessen:
- die Daten sind lediglich gegen Veränderungen geschützt und nicht verschlüsselt !
- Keine „geheimen“ Daten dort ablegen

AUTHENTIFIZIERUNGS-VERFAHREN

Verschiedene Verfahren

- Einfache Authentifizierung
- Webserver Authentifizierung
- OAuth1
- OAuth2

Einfache Authentifizierung

- Übertragung von konfigurierbaren Schlüssel/Wert Paaren für User und Passwort
- Bei Get in URL
- Bei Post im Body

HTTPAuthentifizierung

- Username und Passwort werden Base64 Codiert in den Header eingebettet
- „Basic Authentication“
- Authorization Header nach RFC2617

OAuth1 und OAuth2

- Beide Verfahren sind dafür gedacht, Autorisierungen in verteilten System zu ermöglichen
- Ein Dienst A soll auf Dienst B zugreifen können, ohne das ein User seine Zugangsdaten direkt hinterlegt

Links

- Jose JWT Library
- https://github.com/paolo-rossi/delphi-jose-jwt

- Artikel und Source
- https://probucon.de/blog/2020/07/coderage-germany-2020/

- Entwicklerkonferenz Ekon
- https://entwickler-konferenz.de/de/

- Danke für Ihre Aufmerksamkeit
- Bernd Ua

embarcadero®