Identity Access Management mit IdentityServer

DAS OPENID CONNECT UND OAUTH 2 FRAMEWORK FÜR .NET

Agenda

- Security
- Identity Access Management
- Protokolle
- Security Token
- Grant Types
- IdentityServer Beispiel

About Me: Konstantin KLETZANDER

- Software Entwickler & Trainer
  - ~ 2 Jahre IT-Trainer
  - ~ 6 Jahre Software Entwickler / IT Consultant
  - MCSD: Web Applications
  - FH-Studium in Hagenberg & St. Pölten
- Web: https://kkonstantin.com/

Security

FÜR WEB ANWENDUNGEN

Bekannte Gefahren für Software Systeme

- **Cross-Site Scripting**
  - Clientseitige Scripts, die Cookies und Session Tokens stehlen oder den DOM manipulieren können
- **SQL Injection**
  - Schadhafte SQL Anweisungen, die eine Datenbank zerstören können
- **Cross-Site Request Forgery (CSRF bzw. XSRF)**
  - User wird dazu veranlasst, schadhafte Requests abzuschicken
- **Open Redirects**
  - URL Weiterleitungen auf gefährliche externe Websites z.B. nach Login

HTTPS

- HTTPS = Hyper Text Transfer Protocol Secure
- TLS = Transport Layer Security, SSL = Secure Sockets Layer
- SSL/TLS verschlüsselt Informationen, die zwischen Client und Server übertragen werden.
- HTTPS schützt Client und Server mittels „Handshake“ auf Vertrauensbasis
- *Webstandard - HTTPS sollte für alle Ressourcen verwendet werden!*

HTTPS

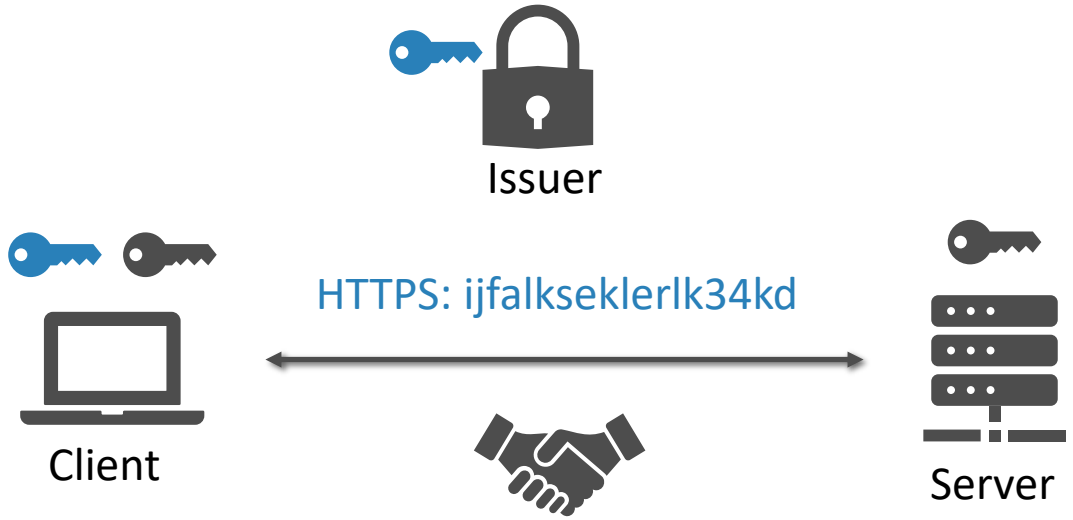
SSL / TLS

HTTPS

GEFÄHRDTE RESSOURCEN:

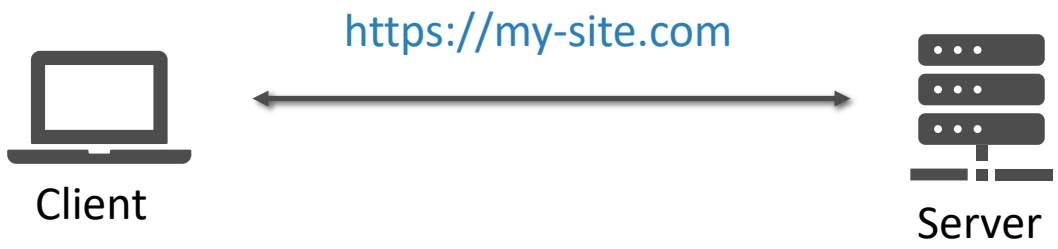
- **Client:**
  - Benutzernamen und Passwörter
  - Cookies
  - Persönliche Informationen
    - Benutzernamen, Passwörter, Bankinformationen, …

- **Server:**
  - Login / Session Token
  - Cookies
  - Persönliche Informationen



HSTS

- HSTS = HTTP Strict Transport Security
- Verpflichtet den Client, für eine bestimmte Zeit, alle Requests über HTTPS zu senden.
- **Parameter:**
  - Max-age
  - IncludeSubDomains
  - Preload

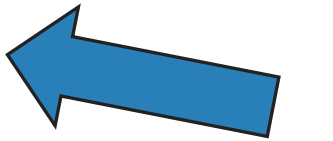


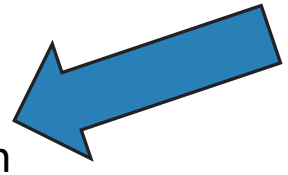

```
services.AddHsts();

app.UseHsts()
```

CORS

- Cross-Origin Resource Sharing

Origin: https://my-site.com





Access-Control-Allow-Origin: https://my-site.com

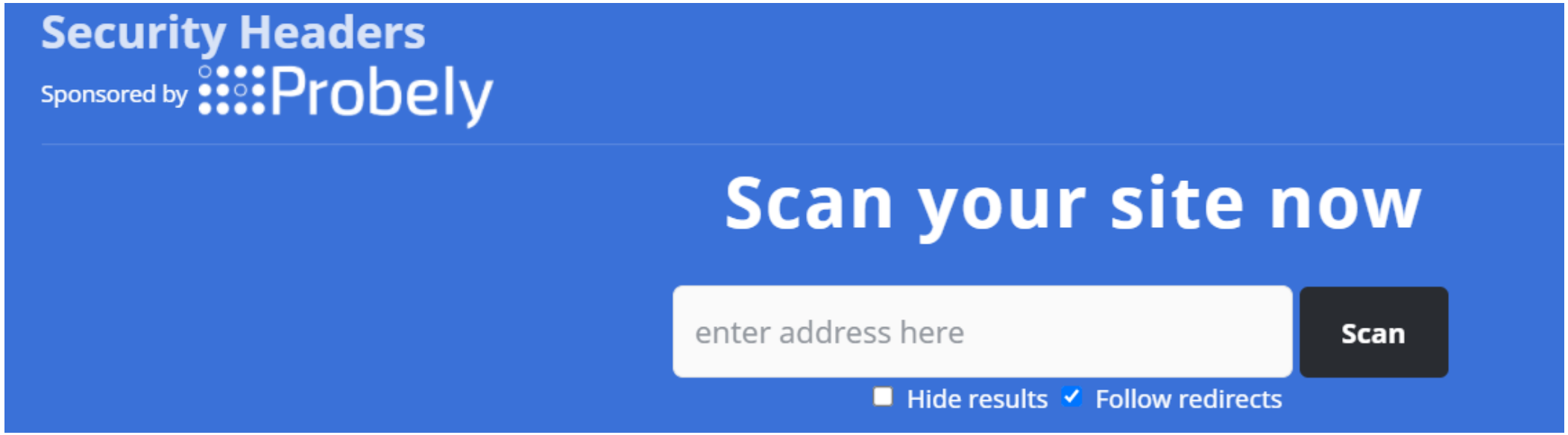
CORS

- Browser erlaubt keine Domän-übergreifenden AJAX Datenübertragungen
- Cross Origin Ressource Sharing muss serverseitig aktiviert werden!

```
services.AddCors();

app.UseCors(options => options.AllowAnyOrigin().AllowAnyHeader().AllowAnyMethod());
```

Security Headers

- SecurityHeaders.com
- NuGet: NetEscapades Security Headers



Authentifizierung und Autorisierung

AUTHENTIFIZIERUNG

- Identity
  - Wer ist der Benutzer?
- Registrierter Benutzer
  - Lokale Membership-Datenbank
  - Third-Party Anbieter

AUTORISIERUNG

- Permission
  - Was darf der Benutzer?
  - Welche Ressourcen darf der Benutzer verwenden?
- Authentifizierter Benutzer

Claims, Roles, Policies

- **Möglichkeiten zur Autorisierung:**
  - Claims
  - Rollenbasierte Zugriffskontrolle
  - Claims und Code
  - Policybasierte Zugriffskontrolle

*“Darf ein **Subjekt** eine bestimmte **Aktion** ausführen?”*

Claims

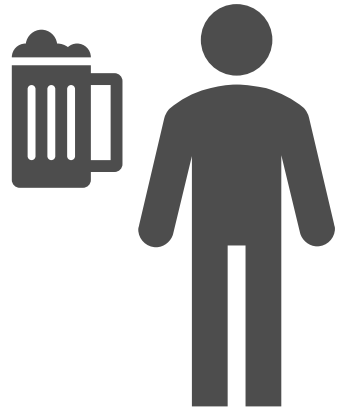
- Claim = Berechtigung eines Benutzers oder einer Organisation
- Die Authentifizierung liefert Identity Claims
- **Beispiel:**
  - Das Identity-System wird um Berechtigungs-Claims erweitert
  - **Vorteil:**
    - Detailierte Kontrolle über Berechtigungen
  - **Nachteil:**
    - Auswuchs von Berechtigungen
    - Manuelles Löschen ungültiger Berechtigungen
    - Identity ist nicht Autorisierung!

```
{
   “sub”: “12341234”,
   “name”: “Erika Mustermann”,
   “role”: “manager”,
   “department”: “sales”,
   “canViewReports”: “true”,
   “canPlaceOrders”: “true”,
   “orderLimit”: “10000”
}
```

Berechtigungen im wirklichen Leben

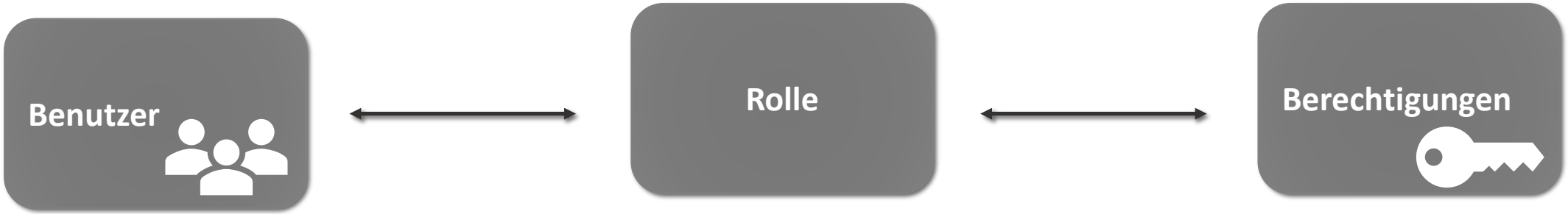
Name: Max Mustermann
Geburtsdatum: 12. Mai 1990

~~CAN ORDER ALCOHOL: YES~~



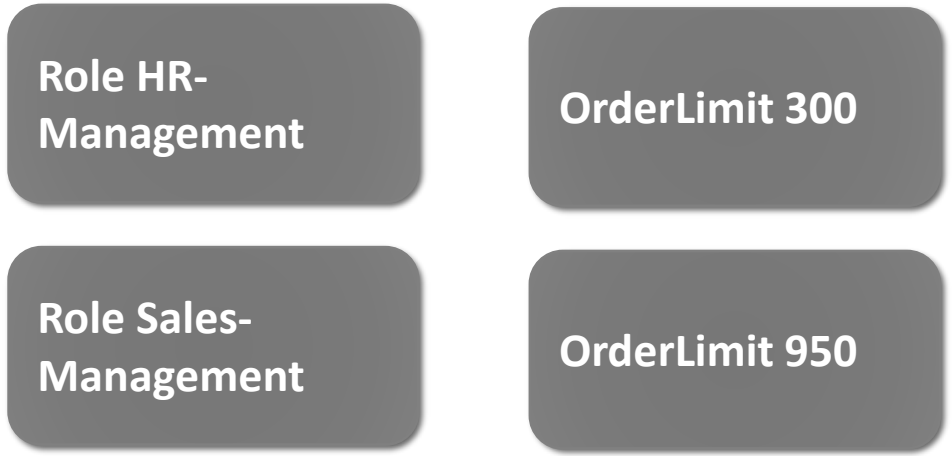
Alkohol Policy

Rollenbasierte Zugriffskontrolle



Rollen

- **Beispiel:**
  - Die Management-Rolle hat die Berechtigung Bestellungen bis zu einem Limit aufzugeben.
  - Manager können jedoch nur bis zu einem Abteilungslimit Bestellungen aufgeben.



Rollen-Problem: Berechtigungen sind auf eine Rolle limitiert!

Claims und Code

```
{
 “sub”: “12341234”,
 “name”: “Erika Mustermann”,
 “role”: “manager”,
 “department”: “sales”
}
```

→

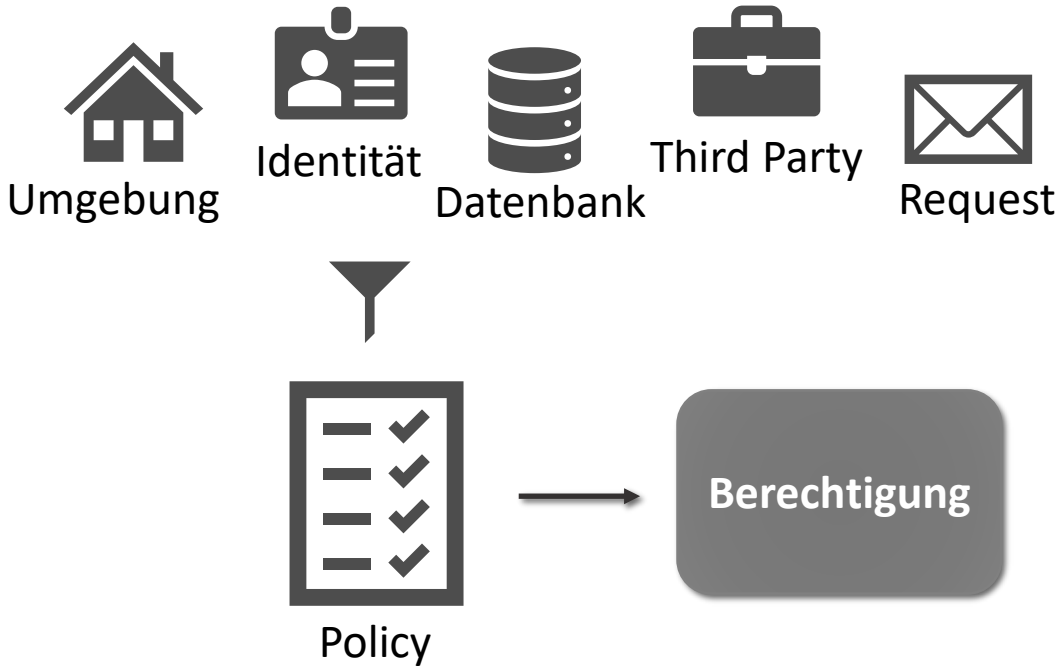
```
decimal orderLimit =
GetOrderLimit(department);

If(role==“sales” && orderValue < orderLimit)
```

→ **Berchtigung** **oder** **Veweigerung**

- **Vorteile:**
  - Besser Skalierbarer
  - Weniger manuelle Verwaltung notwendig
- **Nachteile:**
  - Anwendungsentwickler implementieren Security Regeln

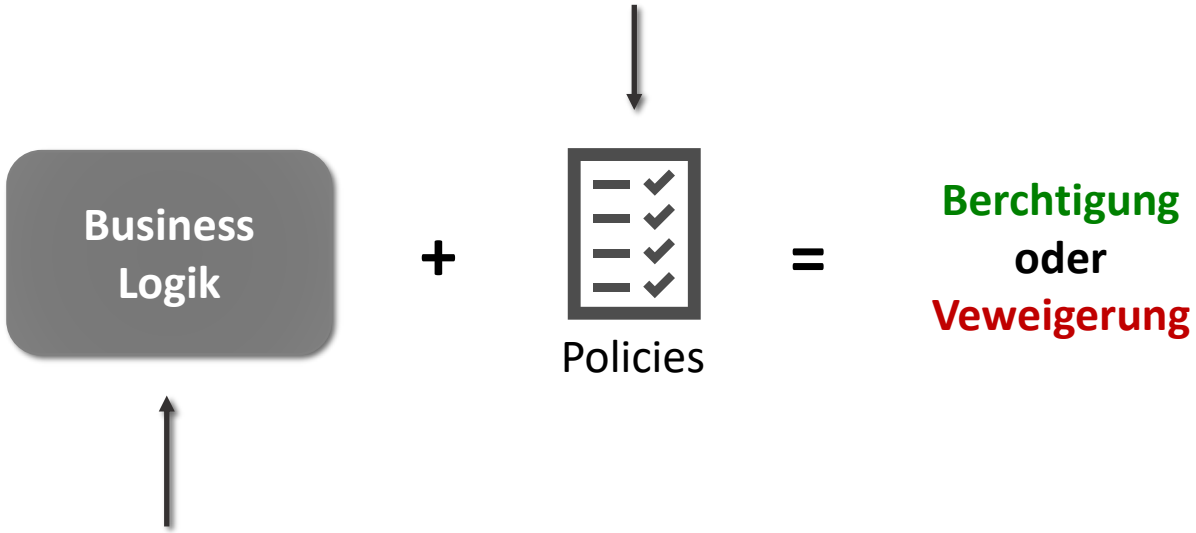
Policybasierte Zugriffskontrolle

- Synonym: Attribute Based Access Control (ABAC)
- Policy = Regelwerk, Richtlinie
- Ermöglicht eine detailierte Kontrolle der Autorisierung
- Policies treffen Autorisierungsentscheidungen
- Autorisierung auf Basis verschiedener Quellen:
  - Claims
  - Umgebung
  - Requests inklusive
  - Datenbanken, zusätzlicher Datenspeicher, REST APIs, etc.
- Autorisierung von Applikation getrennt
  - Sollte einfach zu lessen sein
  - Sollte leicht zu verwalten sein
  - Unabhängiges Deployment



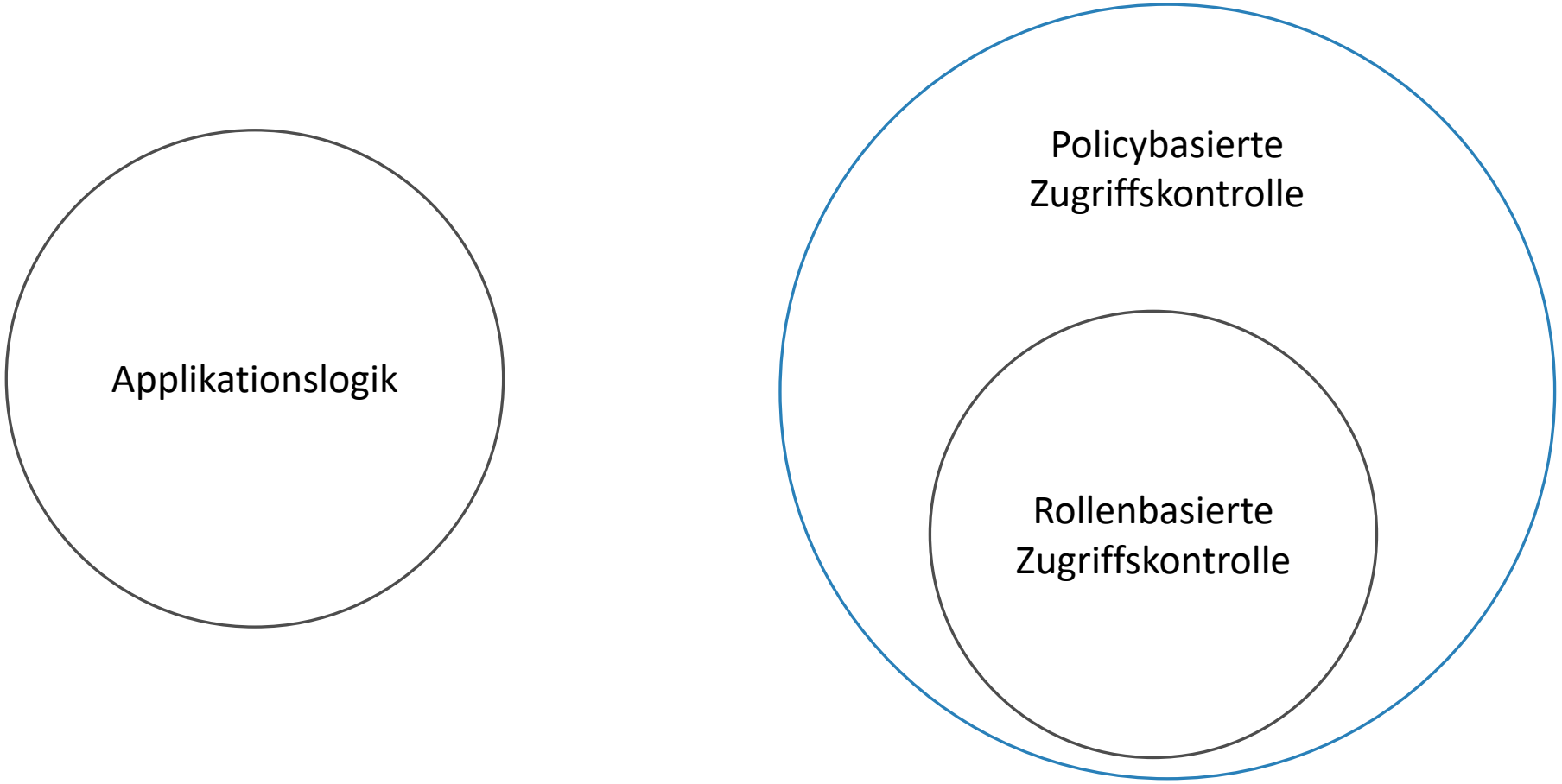
Policybasierte Zugriffskontrolle

Unabhängig von den Applikationen deployed!



Von bestehenden Applikationen verwaltet

Wie soll Autorisierung implementiert werden?

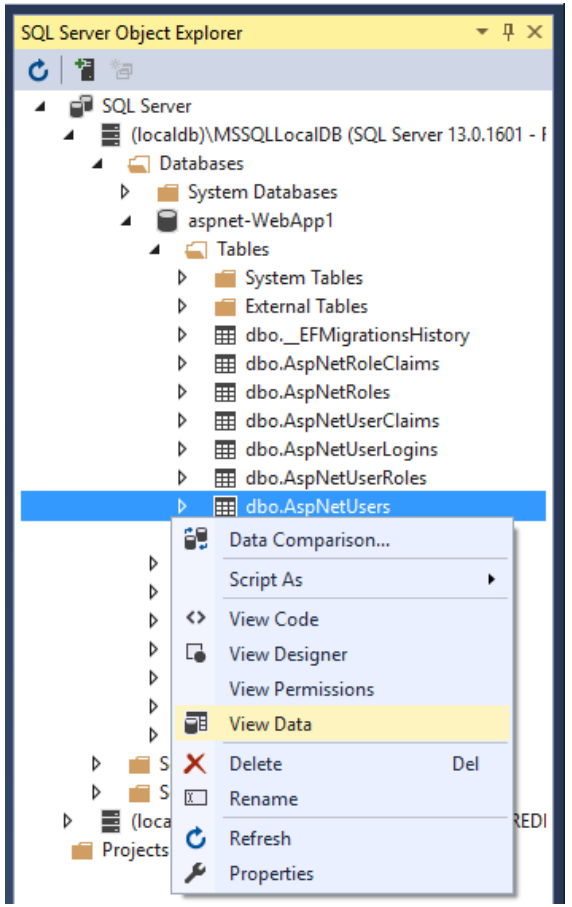


Wie soll Autorisierung implementiert werden?

- Claims sollten nicht als Permissions verwendet werden
- Rollenbasierte Zugriffskontrolle eignet sich gut für weniger umfangreiche Berechtigungen
- Claims und Code bietet eine flexible Lösung
  - Schwer zu verwalten
  - Änderungen in den Securityregeln verlangen ein Deployment der Applikation
- Policybasierte Zugriffskontrolle
  - Securityregeln sind unabhängig von der Anwendung
  - Einfach zu verwalten
  - Ensteht aus natürlichen Informationen

Security mit ASP.NET Identity

- **Implementierung in einer Anwendung:**
  - Membership System mit Login Funktionalität
  - Ermöglicht einfache Benutzerverwaltung
  - Unterstützt externe Anbieter wie Google, Facebook, Microsoft, etc.
  - Unterstützt SQL Server, Azure und andere Datenspeicher



Mehrere Apps, mehrere User??

Identity Access Management

AUTHENTIFIZIERUNG, AUTHORISIERUNG AUF ORGANISATIONSEBENE

Identity Access Management

Identity Access Management (IAM) beschreibt

- Prozesse zur Verwaltung von Benutzerkonten und Ressourcen in einem Organisationsnetzwerk,
- sowie der Berechtigungsverwaltung für Benutzer auf Applikationen und Systeme.

- **Vorteile:**
  - Flexibilität, Effizienz, User Experience und niedrigere Kosten
  - Zentralisierte Zugriffssteuerung
  - Reduktion von Datenschutzverletzungen
  - Weniger Helpdesk Anfragen…

Identity Access Management

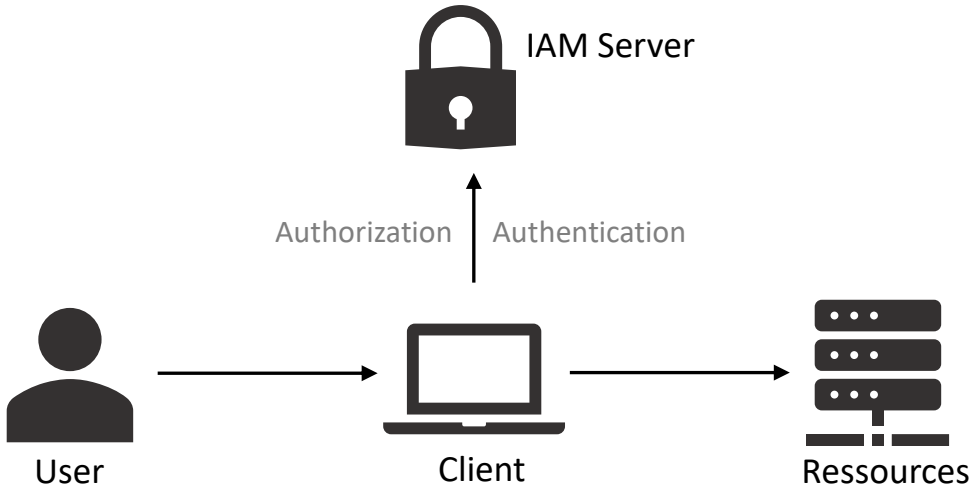
- **Identity Access Management (IAM)**
  - Beispiel: Anmeldung bei Google Anwendung (z.B. Gmail, YouTube)
  - Redirect bei Benutzeranmeldung am Client zu IAM Server: *accounts.google.com*

  - **IAM Server** kümmert sich um die Benutzer-Authentifizierung
  - **Unterstützte Protokolle:** OAuth 2.0 und OpenID Connect, SAML, WS-Federation
  - **Kommunikation:** verschlüsselt mit TLS/SSL => **HTTPS ist erforderlich**

  - **Ziel:** Zentralisiertes Access Management und Lifecycle Management von Applikationen

Identity Access Management

- **IAM Server:** Zentraler Identity Access Management Server
- **User:** Ein Benutzer, mit Benutzername und Passwort am IAM registriert
- **Clients:** Eine Softwareanwendung, die am IAM mit einer ClientId registriert ist.
- **Ressources:** Ressourcen werden vom IAM geschützt, das können APIs oder Benutzerdaten sein.
- **Scope:** Gültigkeitsbereich für eine Ressource



Benutzerdaten-Verwaltung

- **Mechanismen für Benutzerlogin und –Logout in Webanwendungen:**
  - **Cookies**

    ```
    Cookie: USER_ID=1234
    ```

  - **Sessions (+ Cookies)**

    ```
    Cookie: MY_SESSION_ID=WZ731IGdvdCABtZS4gWRE0gbWUgbY234gdHdpdHNvb2tpZ
    ```

  - **Security Token (+ OpenId Connect, + Oauth 2, + Cookies, + Web storage)**

Protokolle OpenID Connect & Oauth 2

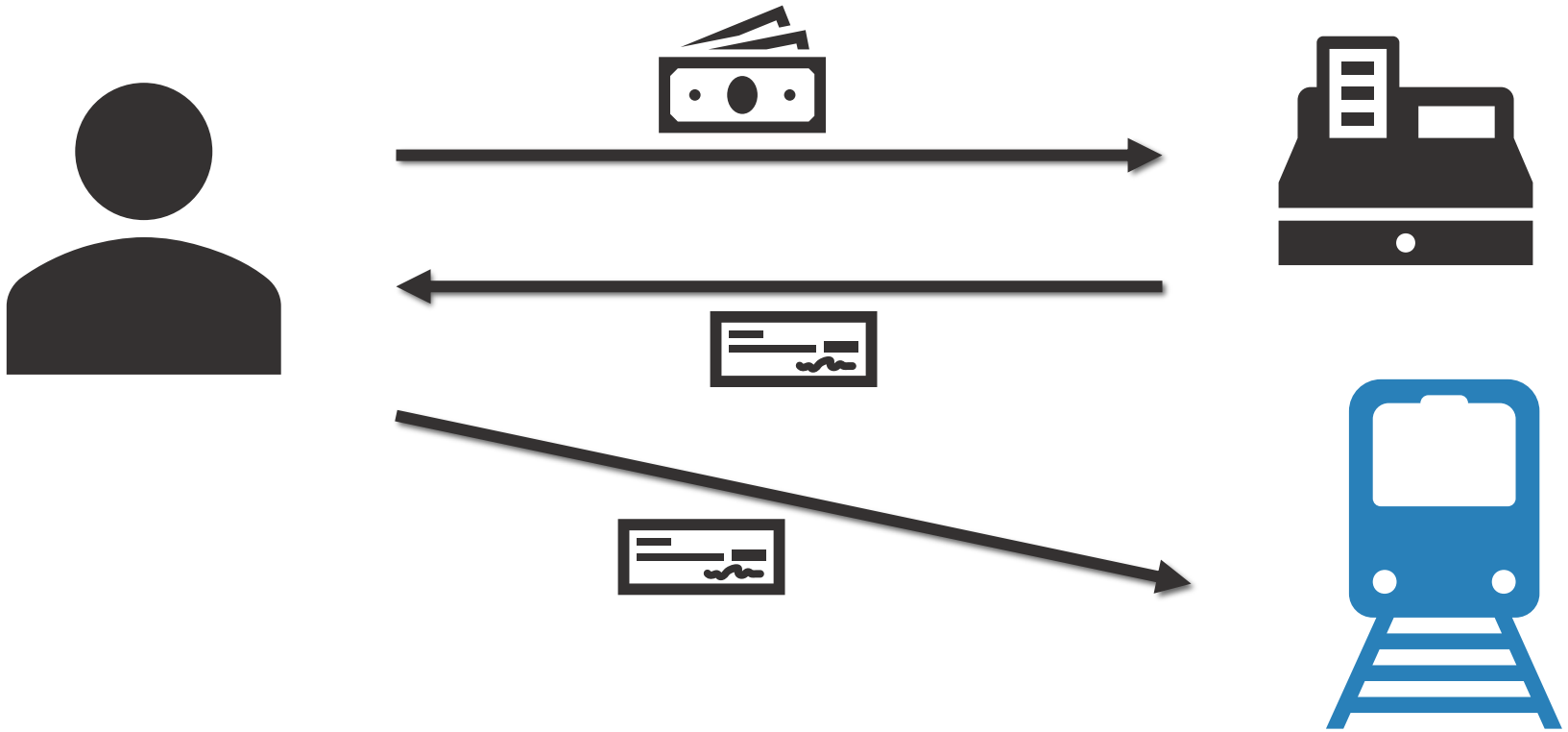
- Aktuelle IAM Server basieren auf den Protokollen OpenID Connect und OAuth 2
- Die Protokolle regeln Sicherheitsbedenken:
  - Authentifizierung *(=> Identity Resources)*
  - API Zugriff *(=> API Resources)*
- Die Protokolle ermöglichen die Absicherung der Requests mittels Security Token

"OpenID Connect and OAuth 2.0 are very similar – in fact OpenID Connect is an extension on top of OAuth 2.0."

Quelle: IdentityServer4 Docs

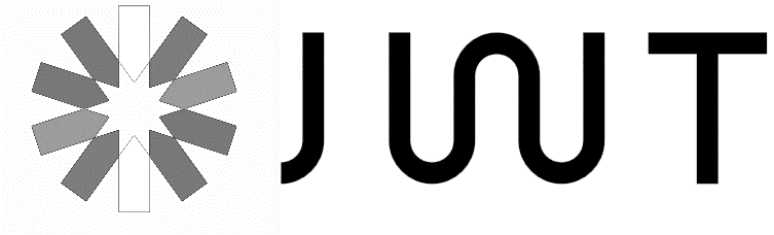
Security Token

Beispiel: Fahrschein Authentifizierung & Autorisierung



Security Token

- JWT = JSON Web Token, RFC 7519
- Informationen in JWT:
  - Credentials
  - Claims
  - Weitere Informationen



jwt.ms

- **JWT ≠ Bearer Token!!**
  - Bearer Token: Einfacher String, in DB gespeichert (alter Ansatz)
  - JWT: Standard um Claims zu encoden und zu verfizieren (moderner Ansatz)

jwt.ms

Enter token below (it never leaves your browser):

eyJhbGciOiJSUzI1NiIsImtpZCI6IkEyNzMwRjFGMkQ1RDI3NUU1QTlDMDlBRkYyNUI3QUE3IiwidHlwIjoiYXQrand0In0.eyJuYmYiOjE2MTg2NTU2MzQsImV4cCI6MTYxODY1OTIzNCwiaXNzIjoiaHR0cHM6Ly9sb2NhbGhvc3Q6NTAwMSIsImF1ZCI6ImFwaTEiLCJjbGllbnRfaWQiOiJvYXV0aENsaWVudCIsImp0aSI6IjUwN0M1QjQzRkU2MjU1NDJGQzUzOEY4Nzg5MTY4RTAxIiwiaWF0IjoxNjE4NjU1NjM0LCJzY29wZSI6WyJhcGkxLnJlYWQiXX0.sXlDe2JqlzQ5AhHuwettdnj6sF0XqcK3HMxHTBOPHfT8cedmxGuB3OEJgCPtiLpnAkSqAtou6WZis400sZK9ZNfyCn5EVnh6Otzw4i3DAn4XPqonqlaovHm_Zcen2RYhVLo6sigt3tsSmR0UqQufnxPrBJL_kCwJn5B1LtS85f2Grv2mXYxF8-YfUiOnSzA66i3PPRr5NpZ0k-BDcKW6AxFQ6X6Jn8krI9WLXQjI3Y24YIwQf6fKzu-mebxlyg-EQx6V5MHt_UDb82XajK3mxS7BLCfDTw-Z1lDo_mJnmjnhY-Le6NABvGfC3GLg-uhCI_ra1-C9DYb_HO1Aa0fTGQ

Decoded Token | Claims


```
{
  "alg": "RS256",
  "kid": "A2730F1F2D5D275E5A9C09AFF25B7AA7",
  "typ": "at+jwt"
}.{
  "nbf": 1618655634,
  "exp": 1618659234,
  "iss": "https://localhost:5001",
  "aud": "api1",
  "client_id": "oauthClient",
  "jti": "507C5B43FE625542FC538F8789168E01",
  "iat": 1618655634,
  "scope": [
    "api1.read"
  ]
}.[Signature]
```


Header

Payload

Signature

Security Token

- **Identity Token:**
  - Ergebnis eines Authentifizierungsprozesses
  - Beinhaltet mindestens Sub Claim (= Subject Claim)
    - = Wie und wann hat sich der Benutzer authentifiziert

- **Access Token:**
  - Ermöglicht den Zugriff auf API-Ressourcen
  - Client fordert Access Token an und leitet diesen an API weiter
  - Beinhaltet Informationen über Client und ggf. Benutzer
  - API verwendet diese Informationen, um einen Datenzugriff zu autorisieren

Grant Types

- Abläufe bzw. Flows zur Tokenanforderung vom Client am IAM Server sind **Grant Types**
- Diese Abläufe sind in den Protokollen OpenID Connect & OAuth 2.0 spezifiziert.
- **Grant Types** in der Praxis:
  - **Client Credentials** (RFC 6749, M2M Applikationen)
  - **Resource Owner and Password**
  - **Authorization Code + PKCE** (RFC 7636, Interaktive Clients)
  - **Refresh Token**
  - Implicit
  - Hybrid
  - Device Flow
  - Extension Grants

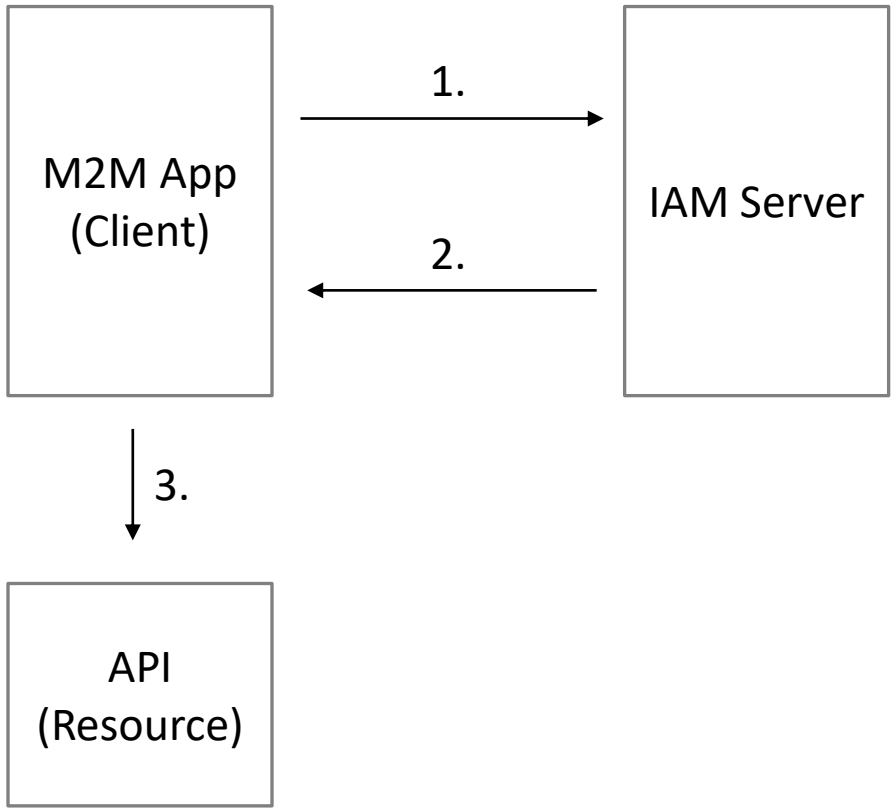
Client Credentials Flow

- Tokenanforderung mittels Grant Type: Client Credentials, RFC 6749
- **Empfohlen für Maschine-zu-Maschine Kommunikation** (z.B. API zu API)
- Der Client authentifiziert sich am Tokenendpunkt des IAM Servers mit den folgenden Daten:
  - *Client Id*
  - *Client Secret*

Client Credentials Flow

Prozessbeschreibung

1. M2M Applikation (Client) authentifiziert sich am IAM Server
2. Der IAM Server Antwortet mit einem Access Token
3. Der Access Token kann von der Anwendung verwendet werden, um eine API aufzurufen und Daten abfragen zu können



Resource Owner and Password Flow

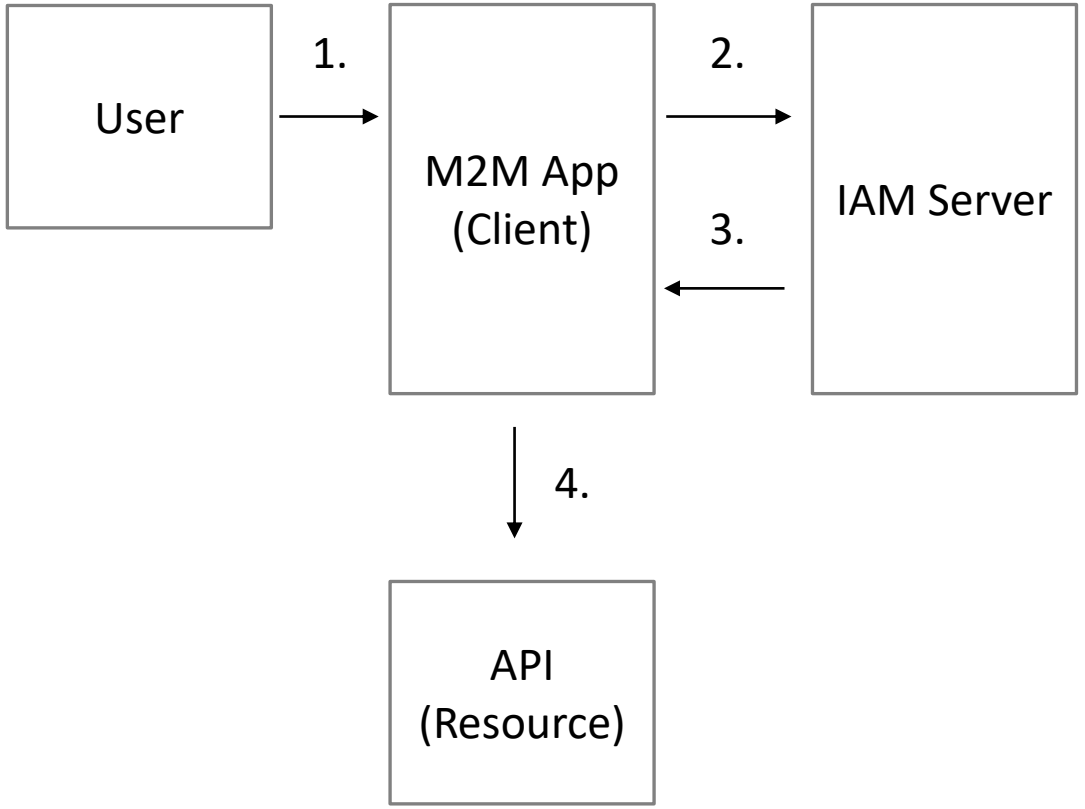
- Tokenanforderung mittels Grant Type: Resource Owner and Password, RFC 6749
- Maschine-zu-Maschine Kommunikation (z.B. API zu API)
  - **Nur für VERTRAUENSWÜRDIGE Applikationen**
  - und wenn kein “Interaktiver Client” möglich ist (Authorization Code + PKCE)
  - **Nachteil: Benutzerinformationen können im Backend gespeichert werden**

- Der Client authentifiziert sich am Tokenendpunkt des IAM Servers mit den folgenden Daten:
  - *Client Id*
  - *Client Secret*
  - *Benutzername*
  - *Passwort*

Resource Owner and Password Flow

Prozessbeschreibung

1. Benutzerlogin am Client
2. M2M Applikation (Client) authentifiziert sich am IAM Server
3. Der IAM Server Antwortet mit einem Access Token
4. Der Access Token kann von der Anwendung verwendet werden, um eine API aufzurufen und Daten abfragen zu können



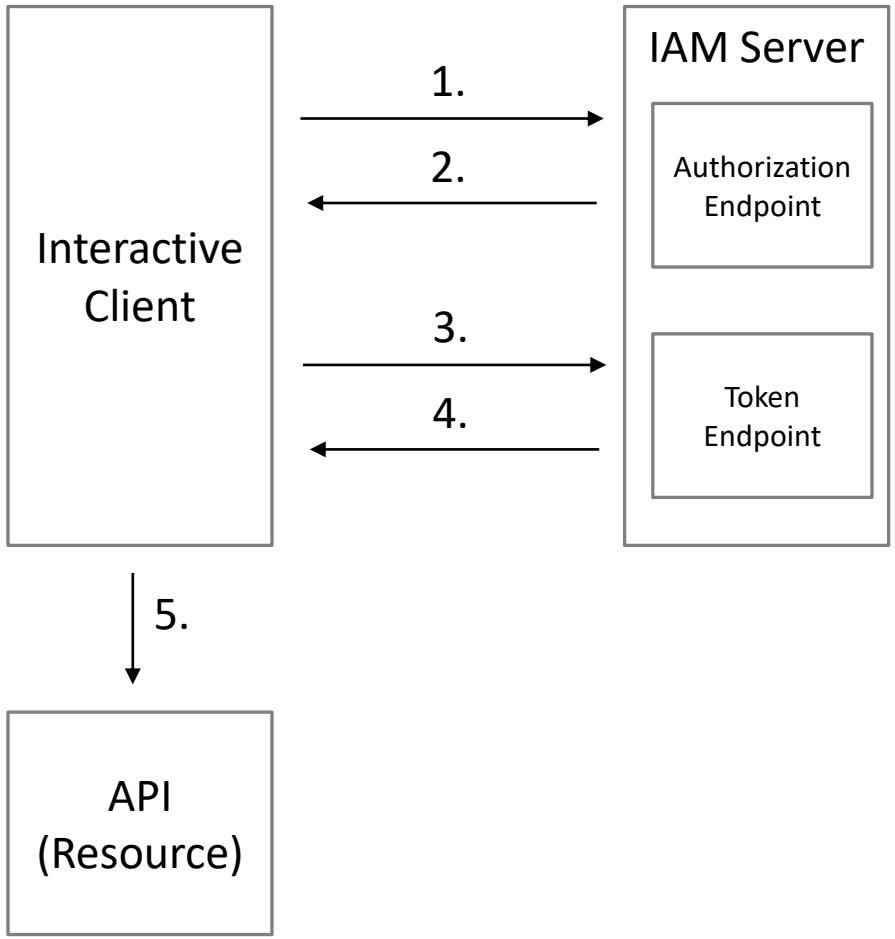
Authorization Code + PKCE Flow

- Tokenanforderung mittels Grant Type: Authorization Code + PCKE, RFC 7636
- PKCE = Proof Key for Code Exchange
- **Empfohlen für Interaktive Clients** (Web Anwendungen, Postman, …)
- Der Client authentifiziert sich am Tokenendpunkt des IAM Servers mit den folgenden Daten:
  - *Client Id*
  - *Client Secret*
  - *Benutzername*
  - *Passwort*

Authorization Code + PKCE Flow

Prozessbeschreibung

1. Client authentifiziert sich am IAM Server (Authentifizierungsendpunkt)
2. Der IAM Server antwortet mit einem Authorization Code
3. Client schickt einen Access Token Request + Authorization Code an den IAM Server (Tokenendpunkt)
4. Der IAM Server antworter mit einem Access Token
5. Der Access Token kann vom Client verwendet werden, um eine API aufzurufen und Daten abfragen zu können.



IdentityServer

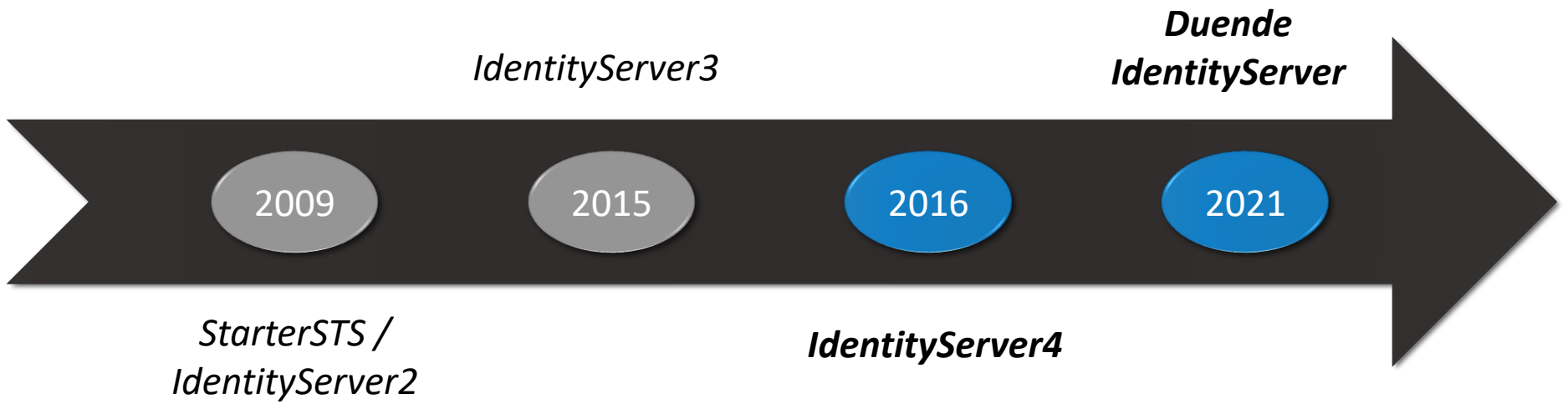
IDENTITY ACCESS MANAGEMENT SERVER FÜR .NET

IdentityServer

- **IdentityServer** ist ein Identity Access Management Server – *unterstützt ASP.NET Identity!*
- Alternative Bezeichnungen: Security Token Service, Identity Provider, Authorization Server, …
- IdentityServer ein “OpenID Connect Provider” - implementiert OpenID Connect und OAuth 2
- **Features:**
  - Ermöglicht eine zentralisierte Authentifizierungslogik
  - Unterstützt lokalen Account Store oder externe Identity Provider (Google, Facebook, 2FA, …)
  - Schützen von Ressourcen (API oder Benutzerdaten)
  - Session Management und Single Sign-On
  - Clients verwalten und authentifizieren
  - Ausstellen von Identity und Access Token für die Clients
  - Tokenvalidierung
  - Robust, von Microsoft empfohlen

Technische Entwicklung

- Die Entwicklung von IdentityServer
- Microsoft Support für IdentityServer4 gleich wie .NET 5 (bis ~ Dezember 2022)



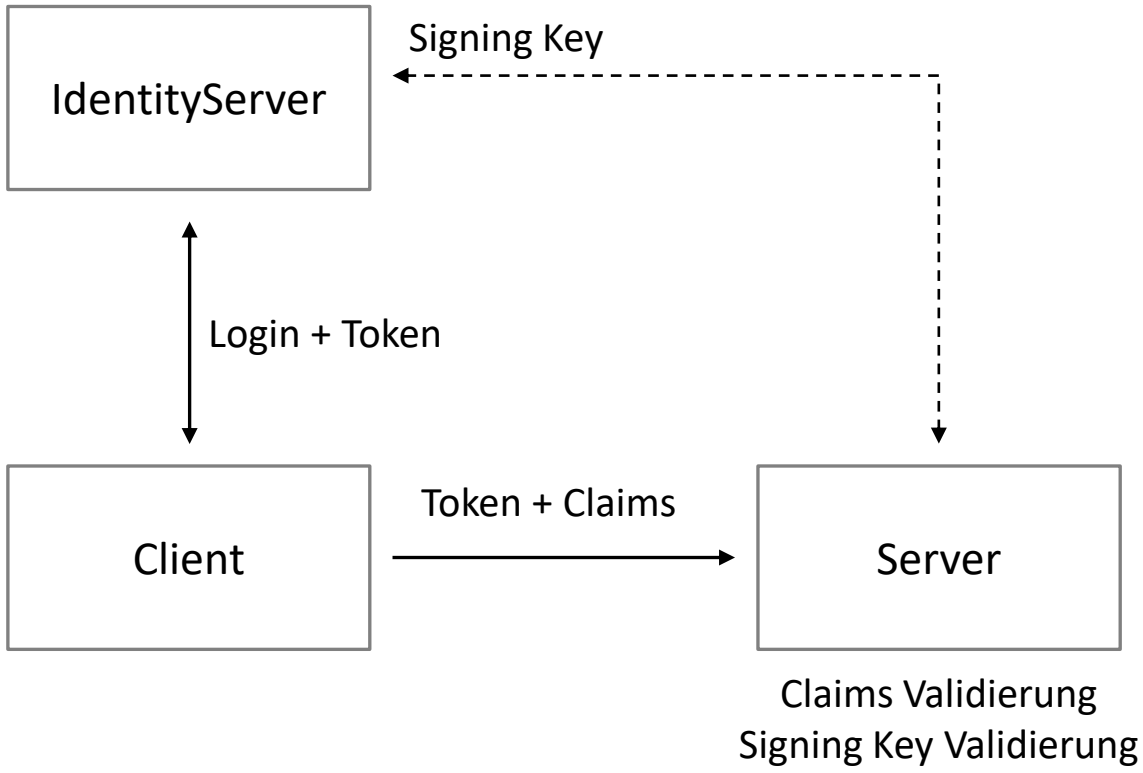
Lizenz

- Open-Source und gratis in der Verwendung (bis Version4)
- Darf für kommerzielle Projekte verwendet werden
- Das Projekt ist Apache 2.0 lizensiert
- Voraussetzung für die Verwendung ist die Einhaltung der .NET Foundation Verhaltensregeln

- **IdentityServer4 ist die letzte Open Source Version von IdentityServer!**
- Die Entwickler Dominick Baier und Brock Allen führen das Projekt komerziell weiter: Duende

- *Kommerzielle Alternativen?* Duende, Auth0, Azure AD, …

Workflow


IdentityServer
Signing Key
Login + Token
Client
Token + Claims
Server
Claims Validierung
Signing Key Validierung

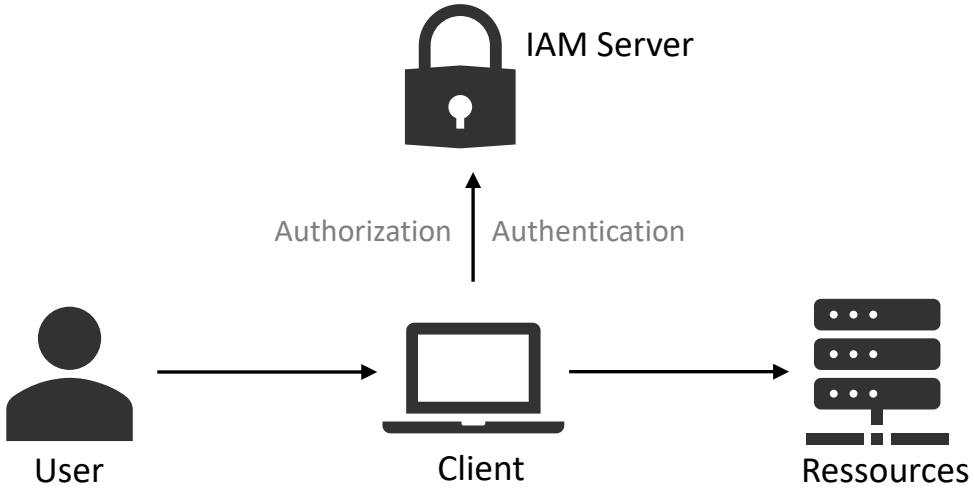
OpenID Connect Discovery Document

Endpoint: /.well-known/openid-configuration


```
{
  "issuer": "https://localhost:5001",
  "jwks_uri": "https://localhost:5001/.well-known/openid-configuration/jwks",
  "authorization_endpoint": "https://localhost:5001/connect/authorize",
  "token_endpoint": "https://localhost:5001/connect/token",
  "userinfo_endpoint": "https://localhost:5001/connect/userinfo",
  "end_session_endpoint": "https://localhost:5001/connect/endsession",
  "check_session_iframe": "https://localhost:5001/connect/checksession",
  "revocation_endpoint": "https://localhost:5001/connect/revocation",
  "introspection_endpoint": "https://localhost:5001/connect/introspect",
  "device_authorization_endpoint": "https://localhost:5001/connect/deviceauthorization",
  "frontchannel_logout_supported": true,
  "frontchannel_logout_session_supported": true,
  "backchannel_logout_supported": true,
  "backchannel_logout_session_supported": true,
  "scopes_supported": [
    "openid",
    "profile",
    "email",
    "role",
    "api1.read",
    "api1.write",
    "offline_access"
  ],
```

Clients, Resources und User

- **Clients**: Welche Anwendungen benötigen Absicherung durch IdentityServer?
- **Resources** & **Scopes**: Was darf ein Client verwenden? (1 Resource hat N Scopes)
- **Identity** Resource: Standard OpenID Connect Scopes email, profile, Custom Scope: role
- **Api** Resources & **Api** Scopes: Geschützte Api Resource und Berechtigung z.B.: customerapi.read



User/Client/Resource Management

- **IdentityServer Admin-Tool:** AdminUI (Kostenpflichtig !!!), User, Client, Resources -Management

- **Open-Source Alternativen:**
  - Clients und Resources in AppSettings verwalten
  - User Management mit ASP.NET Identity
    - Inspiration 1: Custom User Management
    - Inspiration 2: IdentityManager, Tutorial IdentityManager
    - Inspiration 3: Skoruba Admin

Profile Service

- Profile Service Docs
- Aufruf wenn IdentityServer User-Claims an Client-App schicken muss
- 2x Aufruf bei Request von Identity und Access Token => Verschiedene Claims in den Token möglich

Services / Profile Service

```
public class ProfileService : IProfileService
{
    public Task GetProfileDataAsync(ProfileDataRequestContext context)
    {
        context.IssuedClaims.Add(new Claim("test-claim", "test-value"));
        //...
    }

    public Task IsActiveAsync(IsActiveContext context)
    {
        context.IsActive = true;
        // ...
    }
}
```

External Authentication

- External Authentication Docs
- ASP .NET Identity unterstützt “Out Of The Box” OIDC, Google, Facebook, Microsoft, Twitter

Beispiel Code Google:

```
services.AddAuthentication()
   .AddGoogle("Google", options =>
   {
      options.SignInScheme = IdentityServerConstants.ExternalCookieAuthenticationScheme;

      options.ClientId = "xxx.apps.googleusercontent.com";
      options.ClientSecret = "secret";
   });
```

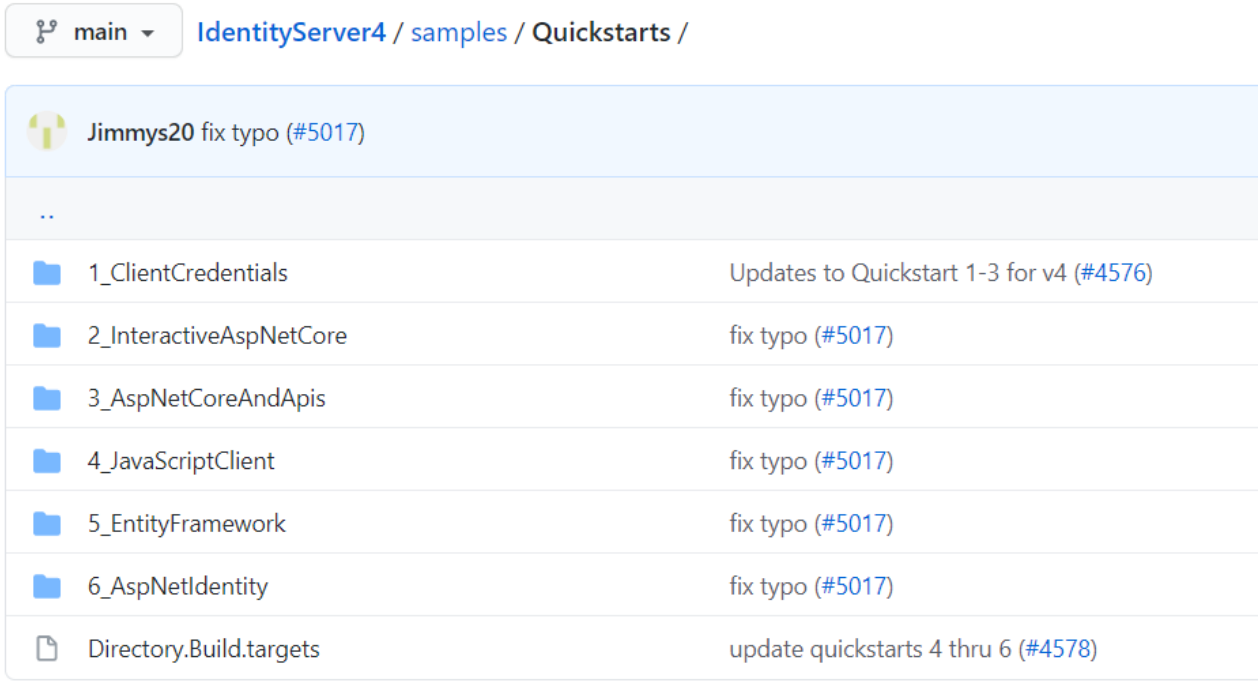
Federation Gateway/Active Directory

- IdentityServer kann als Federation Gateway verwendet werden
- Unterstützung für externe Logins mit WS-Federation und SAML
- Single Sign-On mit IdentityServer als Federation Gateway ist möglich!

- **Voraussetzungen:**
  - Kommunikation über TLS/SSL (HTTPS)
  - Federation Gateway (WS-Federation, SAML) Server Instanz

Identity Server Beispiele

- Code von den Entwicklern:
  - IdentityServer Docs
  - Quickstarts GitHub Repository
  - Quickstart UI
    - dotnet new -i identityserver4.templates
    - dotnet new is4ui



Beispiel

MIT IDENTITYSERVER4

Beispiel

- Implementierung von IdentityServer4 in .NET 5.0
- OAuth und OpenId Connect
- Absichern einer API
- Verwendung von Clients

- GitHub Repository: https://github.com/kkonstantin01/IdentityServerDemo

Nützliche Links & Quellen

- IdentityServer4 Docs, IdentityServer4 Docs (PDF)
- IdentityServer4 GitHub
- IdentityServer4 Microsoft
- IdentityServer.com
- LeastPrivilege (IdentityServer Entwickler Blog)
- Getting Started with IdentityServer4
- Create Certificates for IdentityServer4 signing using .NET Core
- ASP.NET Core Swagger UI Authorization using IdentityServer4
- IdentityServer4 Adding custom properties to User
- Microsoft Token Decoder
- Microsoft Oauth 2.0 and OpenID Connect Docs