MikroServisler ve Güvenlik

Temel Güvenlik Kavramları

- Kimliklendirme (Authentication)
  - Bir kullanıcı veya uygulamanın **kimliğinin doğrulanması işlemi**dir
  - Kullanıcı için :username/password, Servis veya uygulama için : apikey/secret
  - Kimliği doğrulanan nesneye (**subject**), **principal** adı verilir

Temel Güvenlik Kavramları

- Yetkilendirme (Authorization)
  - Principal'ın **secure resource** üzerinde yapmak istediği işlemlerin denetlenmesi işlemidir
  - Bu denetlemeye göre işlemin yapılmasına izin verilir veya işlem reddedilir
  - Secure resource bir URL, servis metodu veya domain nesnesi olabilir

Temel Güvenlik Kavramları

JAVA Eğitimleri

- Yetkilendirme Türleri
  - Role Based Access Control - **RBAC**
    - User ve grupların rollerine göre sistem üzerinde belirli işlemlerin yapılması sağlanır
    - user/group ve role
  - Access Control List - **ACL**
    - User ve rollere belirli bir secure resource üzerinde belirli işlemleri(CRUD) yapmalarını denetler (grant/deny permission)

Monolith vs Microservice Güvenliği

- Monolith uygumaların security mimarisinde kullanılan yapıların microservice mimarisinde kullanılması çok uygun olmamaktadır
  - In memory security context yönetimi
  - Centralized session yönetimi
- Microservis mimarisinde security context bilgisinin **request header**'ları üzerinde taşınması söz konusudur

API Gateway ve Güvenlik

- Microservice'lere istemci tarafından doğrudan erişimin bir takım **dezavantajları** söz konusudur
  - Doğrudan sunucu ile etkileşime girilmesi arka kısımdaki servis mimarisinin istemci tarafına sızması demektir
  - Yetkilendirilmemiş isteklerin dahili network'e ulaşması söz konusu olabilir
  - Farklı istemciler için farklı auth metotlarının bütün servislerde ayrı ayrı ele alınması gerekebilir

API Gateway ve Güvenlik

JAVA Eğitimleri

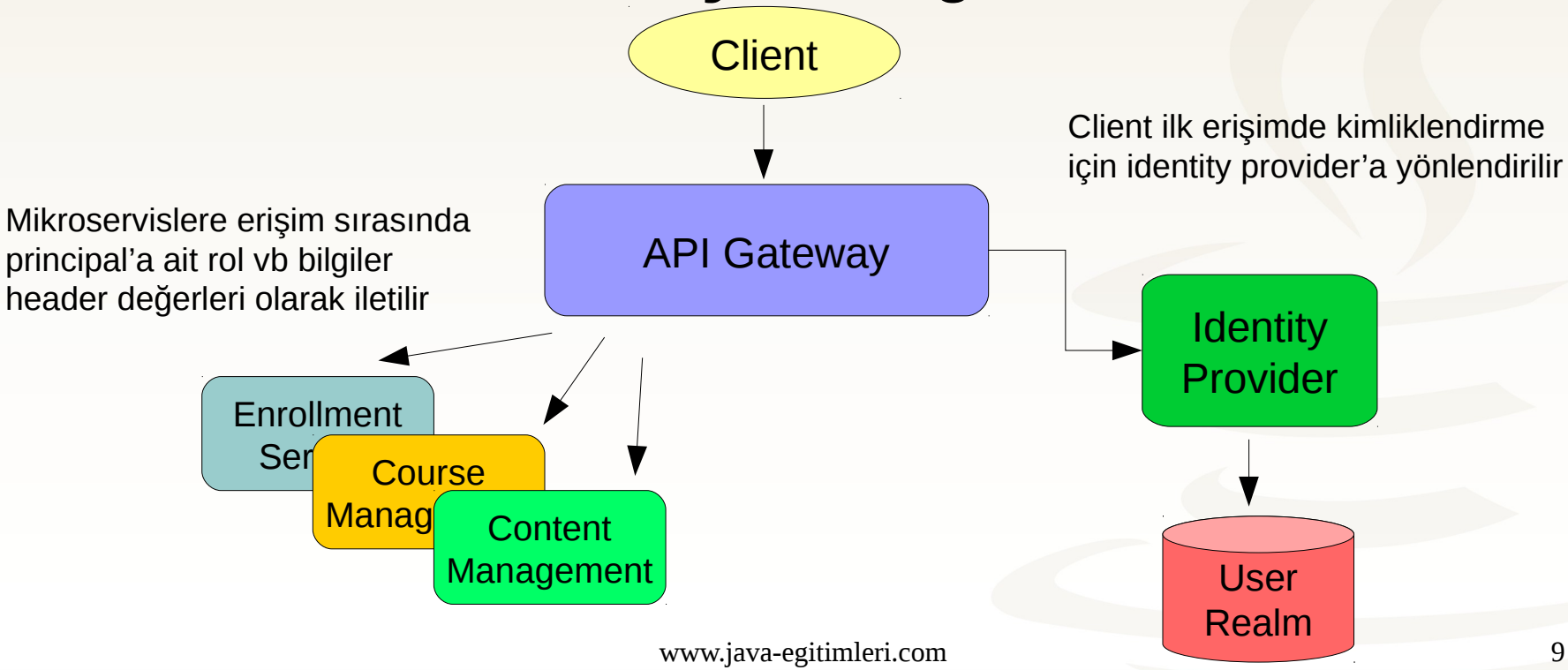
- Servis instance'ları genellikle **API Gateway'in arkasında** yer alır ve istemciler API Gateway üzerinden ilgili servislere erişirler
- Erişim sırasında servisler isteği gönderen kullanıcıyı **kimliklendirme işlemine tabi tutmak** ve servisi **invoke etmeye yetkisi olup olmadığını kontrol etmek** isterler
- API Gateway, mikroservisler ile dış dünya arasında bir **SSO proxy vazifesi** görür

API Gateway ve Güvenlik

- Gelen isteği API Gateway kimliklendirme işlemine tabi tutar ve ardından ilgili servise yönlendirirken request içerisine **“access token” (Json Web Token)** ilave eder
- Servisler bu **token üzerinden yetki kontrolü** yapabilirler
- Servisler **diğer servislere erişirken** de bu token’ı kullanırlar

API Gateway ve Güvenlik

- Her bir mikroservisin ayrı ayrı identity provider ile **tekrar tekrar konuşmasının** önüne geçer
- Bir tür **SSO kabiliyeti** sağlamaktadır



Mikroservisler Arasında Güvenlik

- Mikroservisler arasında **iletişim sırasında** kimliklendirme ve yetkilendirme ihtiyaçları için **farklı yöntemler** kullanılabilir
  - Basic Auth
  - OAuth with JWT
  - Signed JWT
  - API Keys

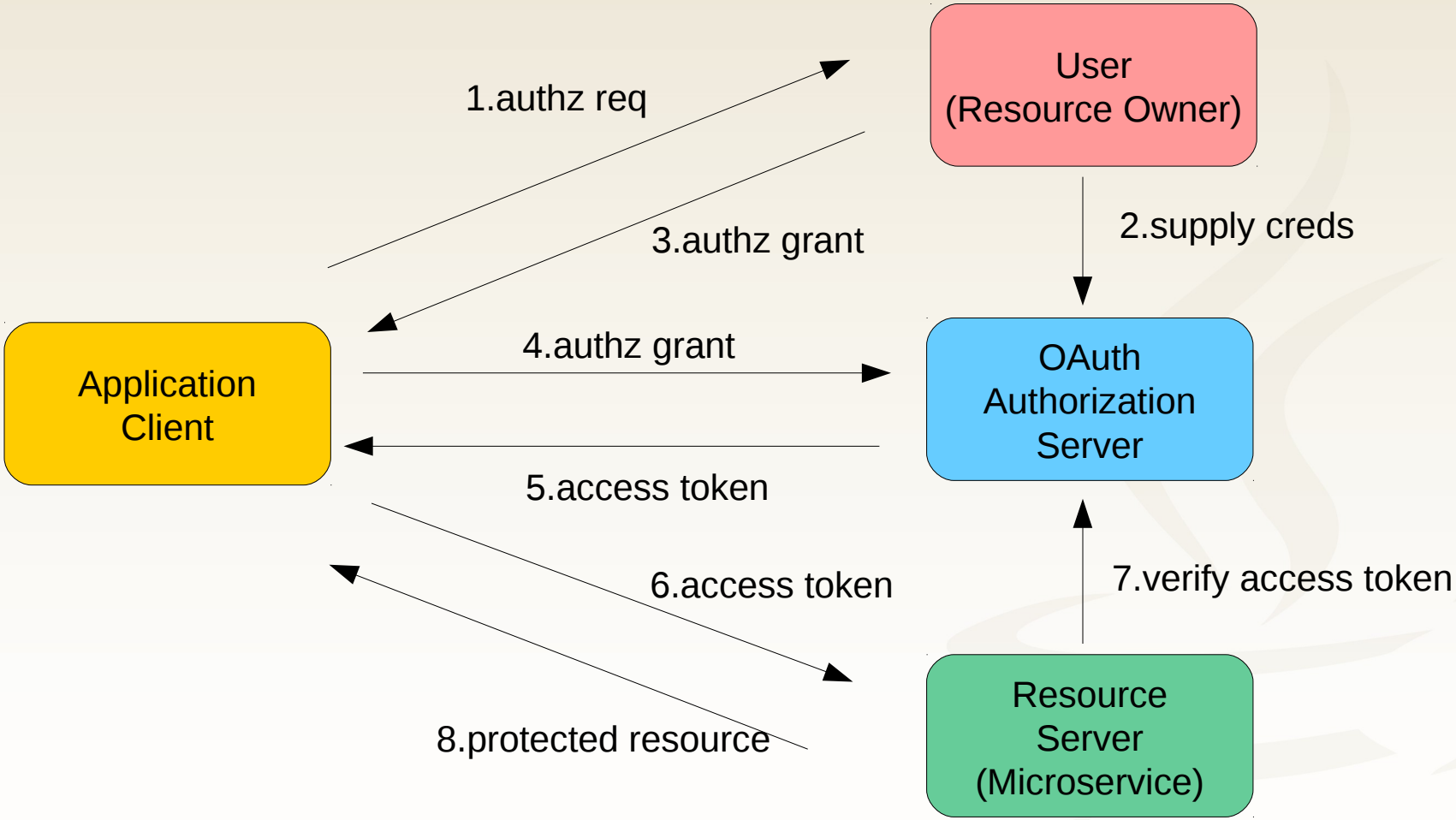
Basic Authentication

- Kullanıcı adı ve şifresinin **base64 encoding**'e tabi tutulmuş hali request içerisinde authorization header şeklinde taşınarak sunucu tarafında kimliklendirme yapılır
- HTTP protokolünün bir standardıdır

OAuth with JWT

- OAuth bir **authorization standardı**dır
- İstemci uygulamaların veya servislerin, sunucudaki **protected resource**'a erişimlerini **yetkilendirilmiş biçimde** gerçekleştirmelerini sağlar
- Yetkilendirme, istemci ve sunucu arasında **credential bilgilerini paylaşmadan** gerçekleşir
- Yerine yetkilendirme sırasında JWT **access token** kullanılır

OAuth'un İşleyişi



Access & Refresh Tokens

- Access token valid ise **yetkilendirme gerçekleşmiş** demektir
- Access **token'ın sızması** durumunda ilgisiz bir kullanıcının istekleride yetkilendirilmiş olacaktır
- **Revoke** edilmesi **mümkün değil**dir
- Dolayısı ile access token'ın **yaşam süresi kısa tutulmalı**dır

Access & Refresh Tokens

JAVA Eğitimleri

- Bu durumda da **kullanıcının sürekli kimliklendirilmesi** ihtiyacı ortaya çıkar
- Bunun önüne geçmek için kullanıcı kimliklendirmesi aşamasında önce **refresh token** üretilir
- Refresh token'ların **ömrü daha uzun** sürelidir ve **revoke edilebilir**
- Refresh token kullanılarak **access token'lar üretilip** yetkilendirmede kullanılabilir

JSON Web Token

- İstemci ve sunucu arasında güvenli bir biçimde **kullanıcıya ait kimlik ve yetki bilgilerini transfer etmek** için geliştirilmiş bir standarttır
- Kullanıcı sisteme **login olduğunda** JSON Web Token oluşturularak dönülür
- Bu token içerisinde kullanıcı ile ilgili sunucu tarafında **yetkilendirmede kullanılacak bilgiler** mevcuttur

JWT

- **Müteakip isteklerde** istemci sunucuya bu token bilgisi ile erişir
- Token bilgisi request içerisinde **Authorization** header ile gönderilir
- İstemci ve sunucu arasında transfer edilen bilgi **sayısal imza**ya tabi tutulduğu için **güvenli** kabul edilir
- Sayısal imza **secret bir key** veya **RSA public/private key pair**'leri ile yapılabilir

JWT'nin Yapısı

- JWT token **üç bölümden** oluşur
  - Header
  - Payload
  - Signature
- Header bölümünde **token tipi ve hashing** algoritması yer alır
- Payload bölümünde ise **kullanıcı/istemci hakkında claims** şeklinde tabir edilen bilgi ve ilave metadata yer alır

JWT’nin Yapısı

- Signature ise **base64 encoding**’e tabi tutulmuş **header ve payload içeriğinin imzalanmış hali**dir
- Signature sayesinde **istemcinin kimliği ve token’ın başka birisi tarafından değiştirilmediği** garanti edilir

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZS
I6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5
MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf
36POk6yJV_adQssw5c
```

API Keys

- Servis çağrısını oluşturan **parametreler**, **servisin ID'si** ve **timestamp** bilgisi birleştirilerek **API key verisi** elde edilir ve sunucuya servis çağrısı sırasında gönderilir
- Bu sayede servis çağrısını yapan **istemci servis tespit edilebilir** ve kimliklendirmesi yapılmış olunur
- Ayrıca **inputun** transfer sırasında **değişmediğ**inden de emin olunur

API Keys ve HMAC

- Bunun için API key'in **hash**'i elde edilerek, ardından shared secret key ile encrypt edilir
- Encrypt edilen bu bilgiye **Hash-based Message Authentication Code** (HMAC) adı verilir
  - Hash (service id + input params + timestamp)
  - HMAC = Encrypt (hash value)
- HMAC verisi **base64 ile encode edilerek** network üzerinde taşınabilir hale getirilir

HMAC Süreci

İletişim

- Harezmi Bilişim Çözümleri
- Kurumsal Java Eğitimleri
- http://www.java-egitimleri.com
- info@java-egitimleri.com