Agenda

1. Intro:
   - OAuth 2.0
   - Intro-Lab: OAuth2 in Action
   - OpenID Connect 1.0
2. Hands-On Part
   - Resource Server
   - Client (Auth Code Flow)
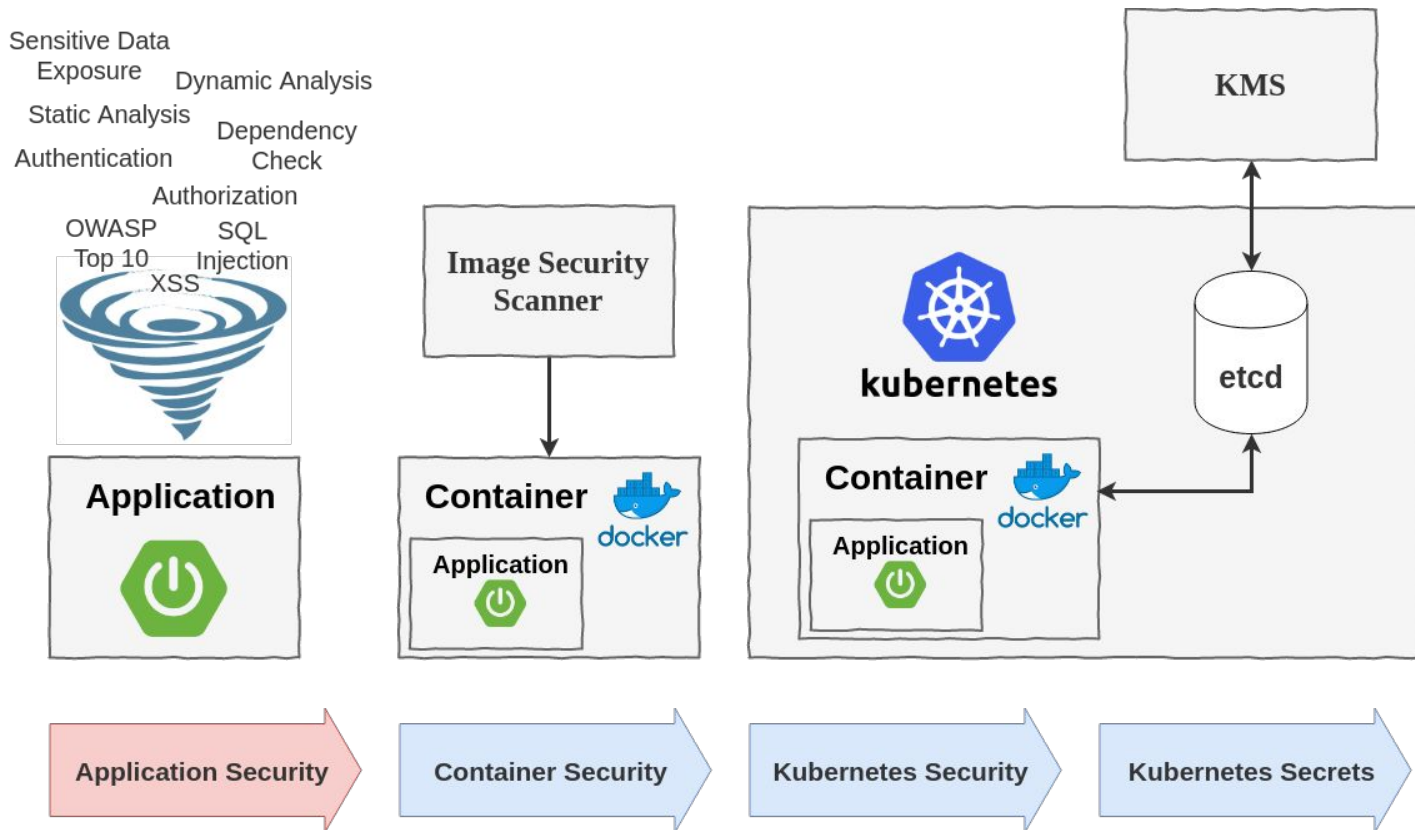3. More Demos (as time allows)
   - Testing Support
   - ...

NOVATEC

Workshop Code

Workshop-Tutorial:

https://andifalk.gitbook.io/openid-connect-workshop

Workshop Source-Code:

https://github.com/andifalk/secure-oauth2-oidc-workshop

NOVATEC

Authentication & Authorization: Only ONE part of Security!

Sensitive Data Exposure
Dynamic Analysis
Static Analysis
Dependency Check
Authentication
Authorization
OWASP Top 10
SQL Injection
XSS

Application

Image Security Scanner

Container docker

Application

KMS

kubernetes

etcd

Container docker

Application

Application Security

Container Security

Kubernetes Security

Kubernetes Secrets

4

NOVATEC

Authentication

Knowledge Factor (something the user knows):
Password, PIN, security question,...

Ownership Factor (something the user has):
ID card, security token, cell phone holding a software token,...

Inherence Factor (something the user is):
Fingerprint, retinal pattern,...

NOVATEC

Authentication

- **Single-Factor Authentication**

- **Multi-Factor Authentication**

NOVATEC

Common Authentication Mechanisms

- Basic Authentication / Digest Access Authentication
- Form-based Authentication (i.e. using Session Cookies)
- Client-Certificates (Mutual TLS)
- Kerberos Tickets
- Proprietary mechanisms like API-Tokens, Siteminder etc.
- SAML Assertion Tokens
- JSON Web Tokens
- OAuth 2.0 & OpenID Connect 1.0
- WebAuthn / FIDO2

NOVATEC

OAuth 2.0 & OpenID Connect

Introduction

NOVATEC

OAuth 2.0

RFC 6749: OAuth 2.0 Authorization Framework

NOVATEC

OAuth 2.0

"The OAuth 2.0 authorization framework enables a **third-party application** to obtain **limited access to an HTTP service**, either **on behalf of a resource owner** by orchestrating an approval interaction between the resource owner and the HTTP service, or by allowing the third-party application to obtain access on **its own behalf**"

(RFC 6749)

NOVATEC

OAuth 2.0 - A bunch of RFC's (Request for Comments)

OAuth 2.0

NEW

RFC 8628: Device Grant	RFC 7591: Client Registration	RFC 8693: Token Exchange
RFC 6750: Bearer Token	RFC 7636: PKCE	RFC 8252: Native Apps
RFC 6819: Threat Model	RFC 7662: Token Introspection	RFC 7009: Token Revocation
RFC 6749: OAuth 2.0 Authorization Framework		

NOVATEC

"OAuth 2.1" - Even more Draft Specifications...

"OAuth 2.1"

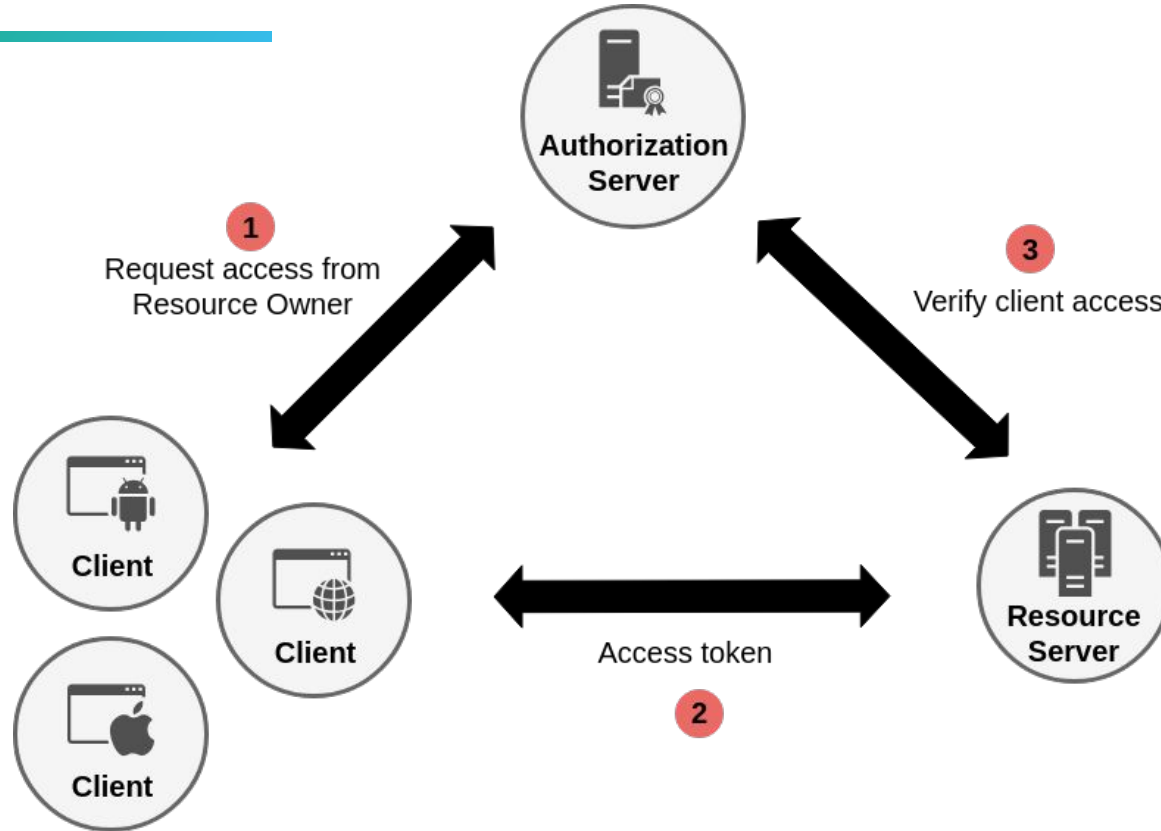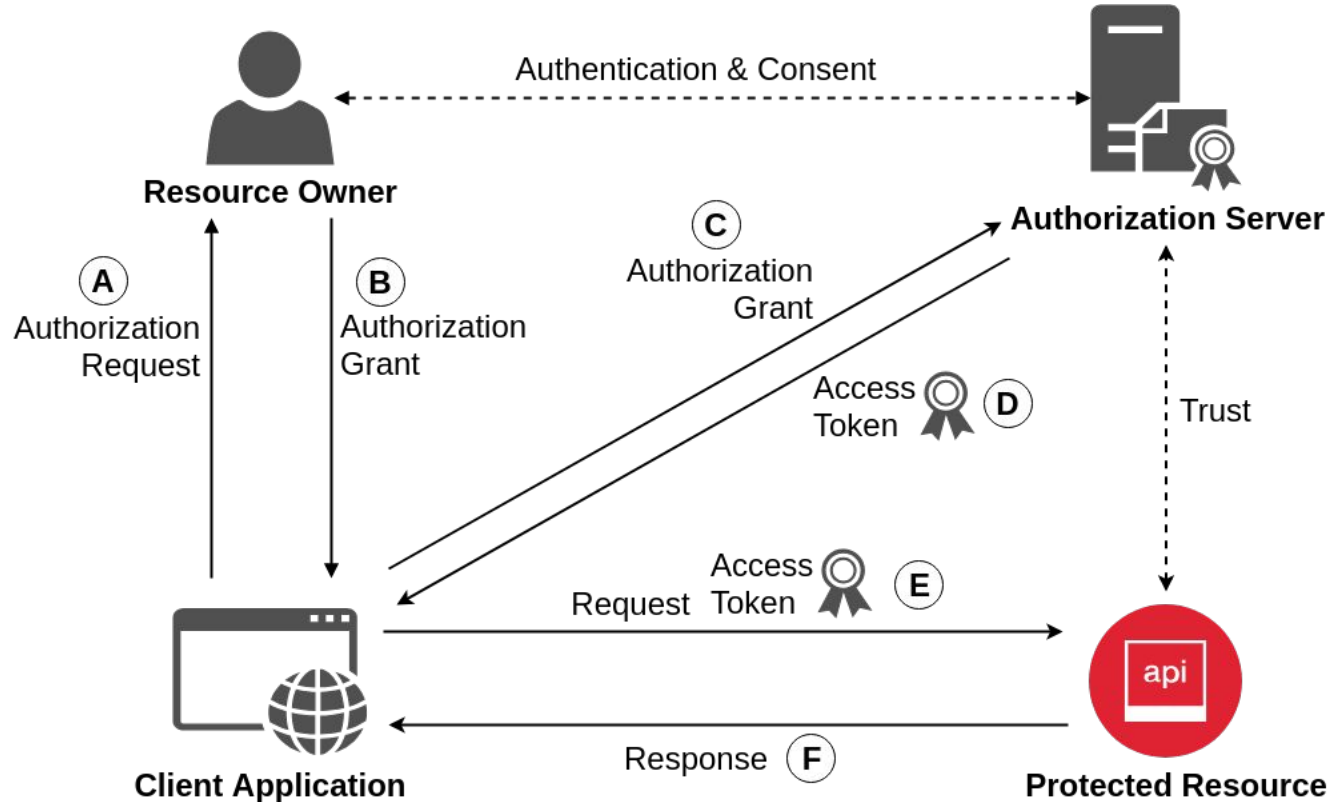|		
JWT Response for OAuth Token Introspection	JSON Web Token Best Current Practices	OAuth 2.0 Rich Authorization Requests
JWT Secured Authorization Request	Resource Indicators for OAuth 2.0	OAuth 2.0 Security Best Current Practice
OAuth 2.0 for Browser-Based Apps	JWT Profile for OAuth 2.0 Access Tokens	OAuth 2.0 Mutual-TLS Client Authentication

NOVATEC

OAuth 2.0 is an authorization delegation framework

NOVATEC

OAuth 2.0 Roles

Authorization Server

1 Request access from Resource Owner

3 Verify client access

Client

Client

Client

Resource Server

2 Access token

NOVATEC

OAuth 2.0 Protocol Flow

OAuth 2.0 Grant Flows

Client Type	Flow	Refresh Tokens
Confidential	Authorization Code	X
Public (Native)	Authorization Code (PKCE)	X
Public (SPA)	Implicit	--
Trusted	RO Password Creds	X
No Resource Owner	Client Credentials	--

NOVATEC

OAuth 2.0 Authorization Code Grant Flow

NOVATEC
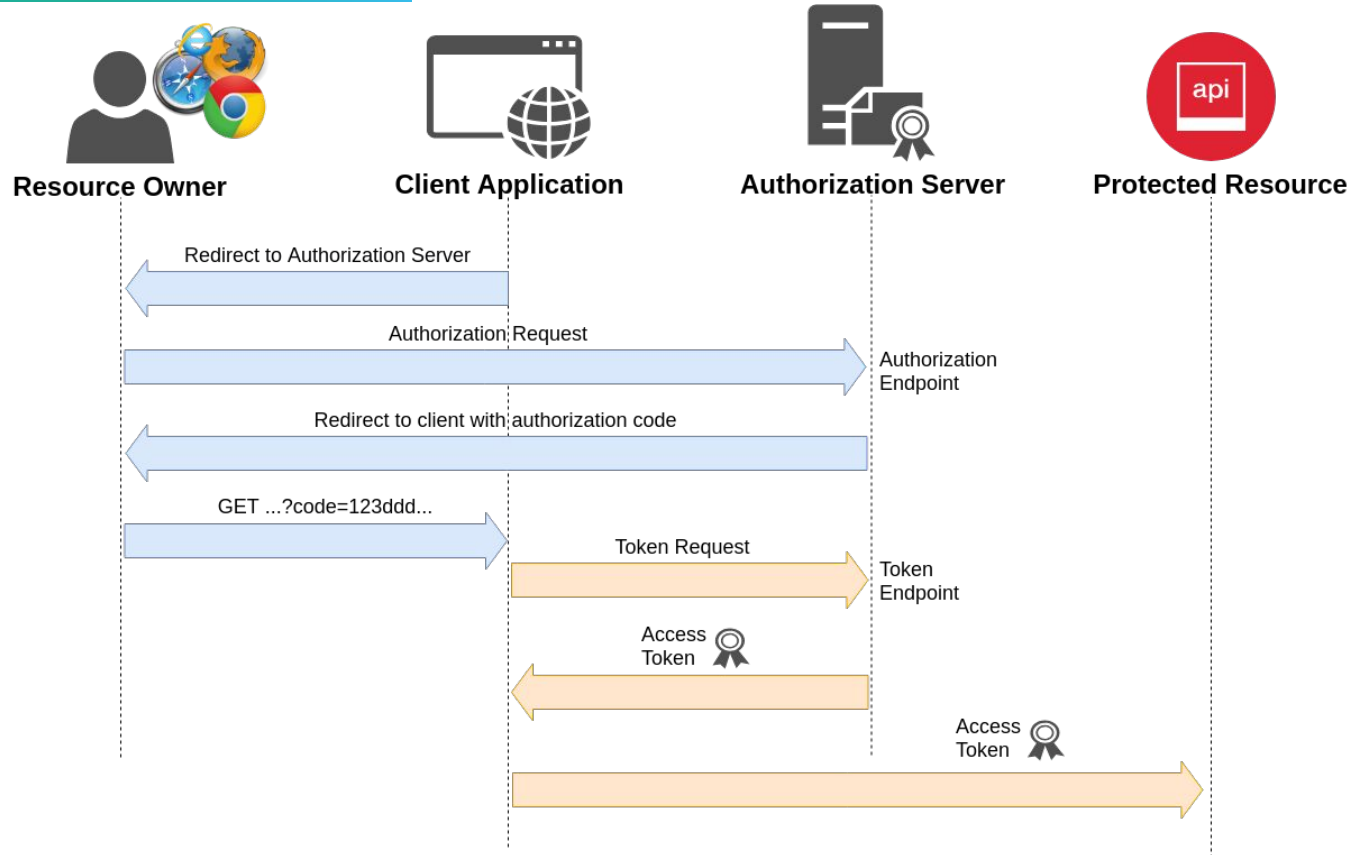
Practice Time

Intro-Lab: Authorization Code Grant Flow in Action

NOVATEC

OpenID Connect 1.0

Identity Layer on top of the OAuth 2.0 Protocol

NOVATEC

OAuth 2.0 is NOT an Authentication Protocol!

> **Jim Manico**
> @manicode
>
> Follow
>
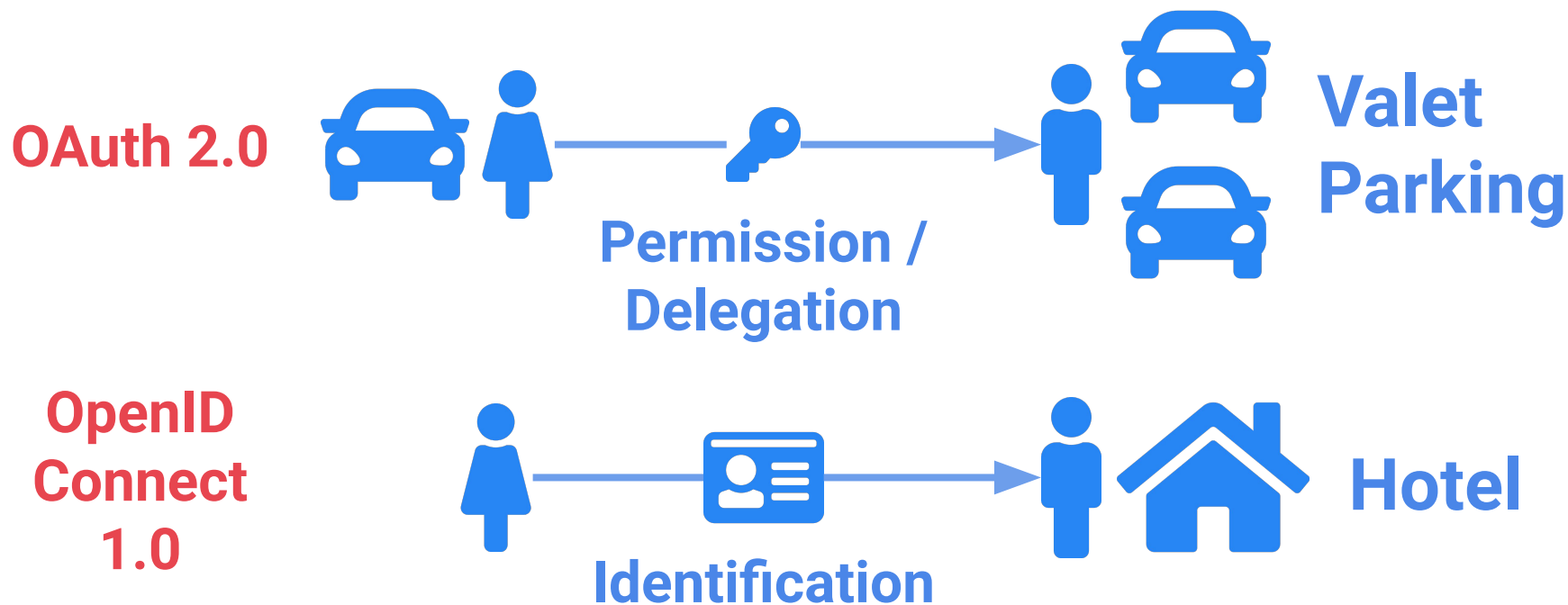> Repeat after me: OAuth 2.0 is NOT AN AUTHENTICATION PROTOCOL.
> oauth.net/articles/authe ...
>
> 12:22 PM - 2 Feb 2017 from Kapaa, HI

[OAuth 2.0 is not an authentication protocol](#)

NOVATEC

OAuth 2.0 vs. OpenID Connect 1.0

OAuth 2.0

Permission / Delegation

Valet Parking

OpenID Connect 1.0

Identification

Hotel

NOVATEC

OpenID Connect 1.0 Standards Layer

OpenID Connect 1.0

OAuth 2.0 Authorization Framework (RFC 6749)

Javascript Object Signing and Encryption (JOSE)

JSON Web Token (JWT)

JSON Web Signature (JWS)   JSON Web Encryption (JWE)   JSON Web Key (JWK)

JSON Web Algorithms (JWA)

NOVATEC
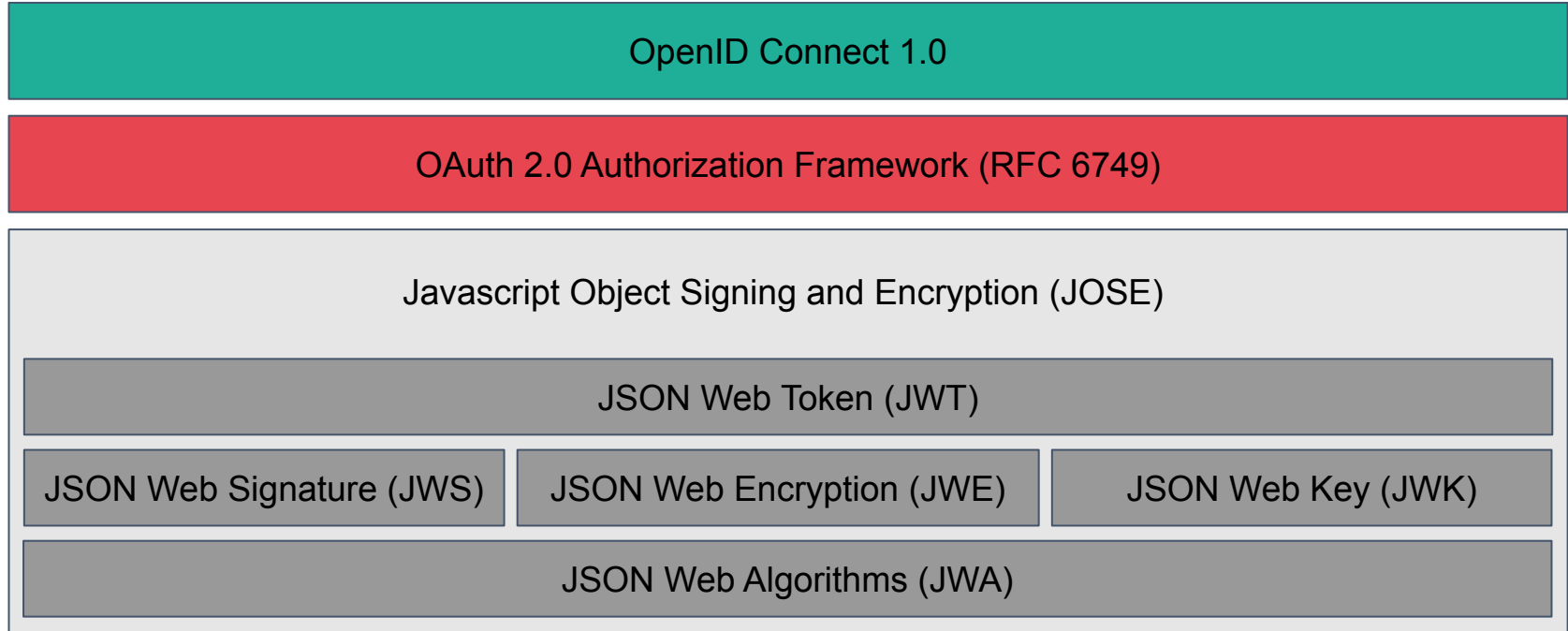
JSON Web Algorithms (JWA)

- Cryptographic algorithms and identifiers for JWS, JWE, and JWK specifications
- Digital Signatures and MACs
- Algorithms for Key Management
- Algorithms for Content Encryption
- Algorithms for Keys

https://tools.ietf.org/html/rfc7518

NOVATEC

JSON Web Key (JWK)

{"keys": [
      {"kty":"EC",
       "crv":"P-256",
       "x":"MKBCTNIcKUSDii11ySs3526iDZ8AiTo7Tu6KPAqv7D4",
       "y":"4Etl6SRW2YiLUrN5vfvVHuhp7x8PxltmWWlbbM4IFyM",
       "use":"enc",
       "kid":"1"},
      {"kty":"RSA",
       "n": "0vx7agoebGcQSuuPiLJXZptN9nn...",
       "e":"AQAB",
       "alg":"RS256",
       "kid":"2011-04-29"}]}

https://tools.ietf.org/html/rfc7517

NOVATEC

JSON Web Signature (JWS)

- JSON Web Signature (JWS) represents content secured with digital signatures or Message Authentication Codes (MACs) using JSON-based data structures
- A document using JWS can answer two questions about the JSON payload:
  - Has the JSON object been altered after creation?
  - Who created this JSON object?

https://tools.ietf.org/html/rfc7515

NOVATEC

JSON Web Encryption (JWE)

- Data structure representing an encrypted and integrity-protected message
- As of July 2019 only identity server of PingIdentity supports JWE
- NOT supported by Spring Security 5.x (See github issue 4435) !

https://tools.ietf.org/html/rfc7516
https://github.com/spring-projects/spring-security/issues/4435

NOVATEC

JSON Web Token (JWT)

- JSON Web Tokens consist of three parts separated by dots ("."), which are:
  - Header
  - Payload
  - Signature
- Each part is Base64Url encoded
- Signature supports symmetric or asymmetric algorithms (e.g. HMAC or RSA)
- Signature = HMACSHA256(
  base64UrlEncode(header) + "." + base64UrlEncode(payload), secret)

https://tools.ietf.org/html/rfc7519
https://tools.ietf.org/html/draft-ietf-oauth-jwt-bcp
https://tools.ietf.org/html/draft-ietf-oauth-proof-of-possession

NOVATEC

JSON Web Token (JWT) - Decoded Form

```
{
  "alg": "RS256",
  "typ": "JWT",
  "kid": "lp_FcMZ8D7U6EEUCiZyWAF21NcwjX_ddwJ5a3eCPMwQ"
}
```
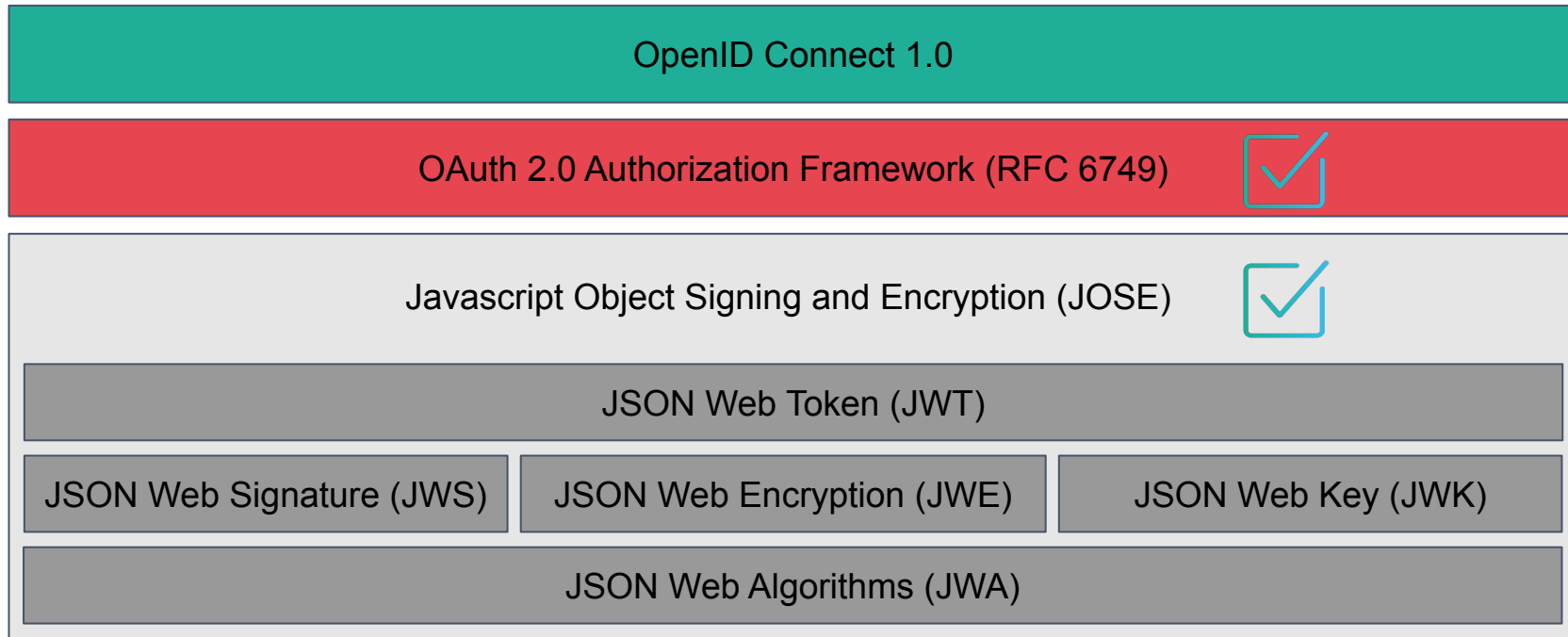**HEADER**

```
{
  "exp": 1571745342,
  "iat": 1571745042,
  "iss": "http://localhost:8080/auth/realms/workshop",
  "aud": ["library-service","account"],
  "sub": "08d3bcaa-5ffd-4b8d-909e-bb567881384b"
}
```
PAYLOAD

NOVATEC

https://jwt.io

ALGORITHM RS256

Encoded PASTE A TOKEN HERE

eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwi
a2lkIiA6ICJscF9GY01aOEQ3VTZFRVVDaVp5V0FG
MjFOY3dqWF9kZHdKNWEzZUNQTXdRIn0.eyJqdGki
OiI1YjkxYTNmYS03MWVhLTQ2YzktYjFkMS0xYzM0
ZGYwNTVkZWIiLCJleHAiOjE1NzE3NDUzNDIsIm5i
ZiI6MCwiaWF0IjoxNTcxNzQ1MDQyLCJpc3MiOiJo
dHRwOi8vbG9jYWxob3N0OjgwODAvYXV0aC9yZWFs
bXMvd29ya3Nob3AiLCJhdWQiOlsibGlicmFyeS1z
ZXJ2aWNlIiwiYWNjb3VudCJdLCJzdWIiOiIwOGQz
YmNhYS01ZmZkLTRiOGQtOTA5ZS1iYjU2Nzg4MTM4
NGIiLCJ0eXAiOiJCZWFyZXIiLCJhen AiOiJsaWJy
YXJ5LWNsaWVudCIsImF1dGhfdGltZSI6MCwic2Vz
c2lvbl9zdGF0ZSI6IjdmZTBkMTM1LWJjMzktNDYz
MC04ZmM5LTEyODA0ZTEwYTUyOCIsImFjciI6IjEi
LCJhbGxvd2VkLW9yaWdpbnMiOlsiaHR0cDovL2xv
Y2FsaG9zdDo5MDkwIl0sInJlYWxtX2FjY2VzcyI6
eyJyb2xlcyI6WyJsaWJyYXJ5X2FkbWluIiwib2Zm
bGluZV9hY2Nlc3MiLCJ1bWFfYXV0aG9yaXphdGlv
biJdfSwicmVzb3VyY2VfYWNjZXNzIjp7ImFjY291
bnQiOnsicm9sZXMiOlsibWFuYWdlLWFjY291bnQi
LCJtYW5hZ2UtYWNjb3VudC1saW5rcyIsInZpZXct

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "RS256",
  "typ": "JWT",
  "kid": "lp_FcMZ8D7U6EEUCiZyWAF21NcwjX_ddwJ5a3eCPMwQ"
}
```

PAYLOAD: DATA

```
{
  "jti": "5b91a3fa-71ea-46c9-b1d1-1c34df055deb",
  "exp": 1571745342,
  "nbf": 0,
  "iat": 1571745042,
  "iss": "http://localhost:8080/auth/realms/workshop",
  "aud": [
    "library-service",
    "account"
  ],
  "sub": "08d3bcaa-5ffd-4b8d-909e-bb567881384b",
  "typ": "Bearer",
  "azp": "library-client",
  "auth_time": 0,
  "session_state": "7fe0d135-bc39-4630-8fc9-12804e10a528",
  "acr": "1",
  "allowed-origins": [
    "http://localhost:9090"
  ],
```

29

NOVATEC

OpenID Connect 1.0 Standards Layer

OpenID Connect 1.0

OAuth 2.0 Authorization Framework (RFC 6749) ☑

Javascript Object Signing and Encryption (JOSE) ☑

JSON Web Token (JWT)

JSON Web Signature (JWS) | JSON Web Encryption (JWE) | JSON Web Key (JWK)

JSON Web Algorithms (JWA)

NOVATEC

OpenID Connect 1.0 (OIDC)

- Based on OAuth 2.0
- Additions:
  - ID Token (JWT format is mandatory)
  - User Info Endpoint (Mandatory)
  - Hybrid Grant Flow (Mandatory)
  - OpenID Provider Configuration Information (Discovery, Optional)

https://openid.net/specs/openid-connect-core-1_0.html
https://openid.net/specs/openid-connect-registration-1_0.html
https://openid.net/specs/openid-connect-discovery-1_0.html

NOVATEC

OpenID Connect 1.0 Claims

Claim	Required	Description
iss	☑	Issuer Identifier
sub	☑	Unique Subject Identifier
aud	☑	Target audience(s) of an ID Token
exp	☑	Expiration time
iat	☑	Time at which the JWT was issued
auth_time	☑	Time of End-User authentication
nonce		Used to associate a client with an ID Token

NOVATEC

JWT Best Practices

- Perform Algorithm Verification
- Validate Issuer and Subject
- Use and Validate Audience
- Do Not Trust Received Claims

https://tools.ietf.org/wg/oauth/draft-ietf-oauth-jwt-bcp/

NOVATEC

OpenID Connect 1.0 Roles
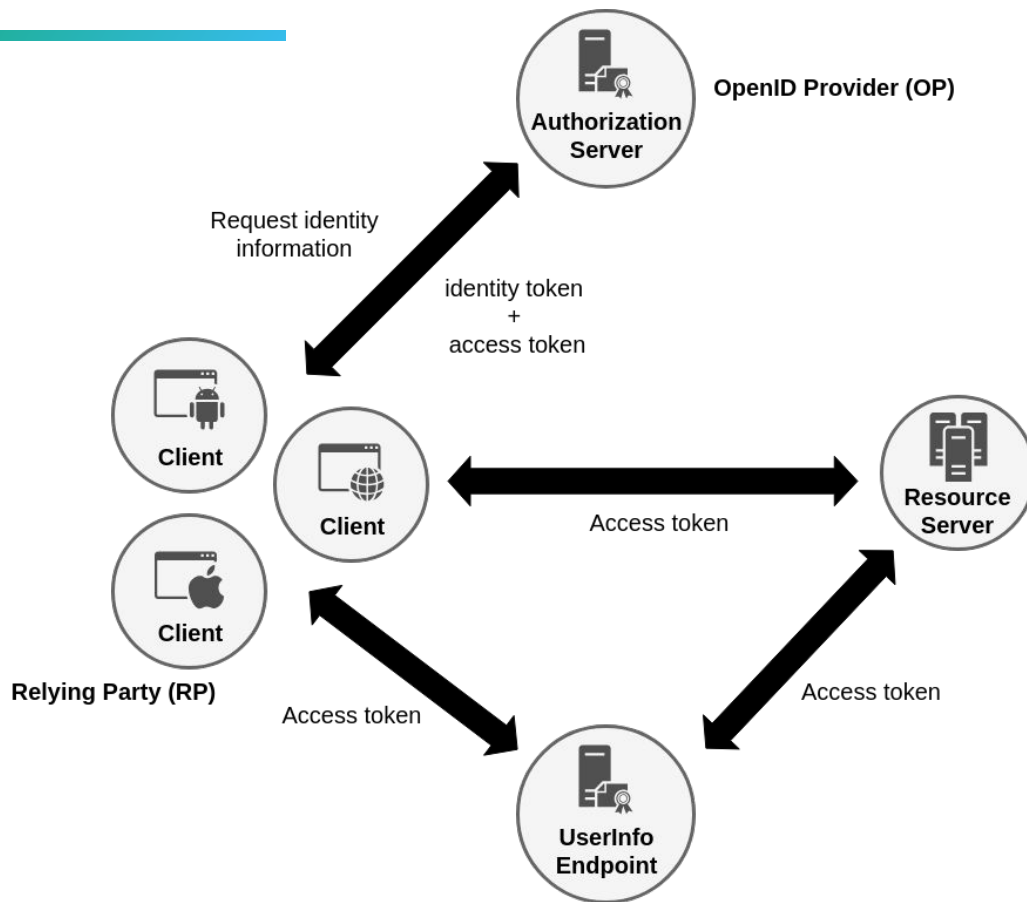
OpenID Connect 1.0 User Info Endpoint

```
GET /userinfo HTTP/1.1
Host: identityserver.example.com
Authorization: Bearer SlAV32hkKG
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
 "sub": "248289761001",
 "name": "Jane Doe",
 "given_name": "Jane",
 "family_name": "Doe",
 "preferred_username": "j.doe",
 "email": "janedoe@example.com",
 "picture": "http://example.com/janedoe/me.jpg"
}
```

NOVATEC

OpenID Connect 1.0 Discovery

```json
{
  "authorization_endpoint": "https://idp.example.com/auth",
  "grant_types_supported": [
      "authorization_code",
      "implicit",
      "refresh_token"
  ],
  "issuer": "https://idp.example.com",
  "jwks_uri": "https://idp.example.com/keys",
  "token_endpoint": "https://idp.example.com/token",
  "userinfo_endpoint": "https://idp.example.com/userinfo",
  ...
}
```

https://example.com/.well-known/openid-configuration

NOVATEC

OpenID Connect 1.0: Access Token Types

JWT Token (Self-contained)	Opaque Token (Reference)
✅ Offline-Validation (Signature/Expiration)	❗ Validation call to introspection-endpoint
✅ Contains all required information	❗ Additional call to get required information
✅ Protocol agnostic	❗ Bound to Http
❗ Cannot be revoked	✅ May be revoked

NOVATEC

Token Validation
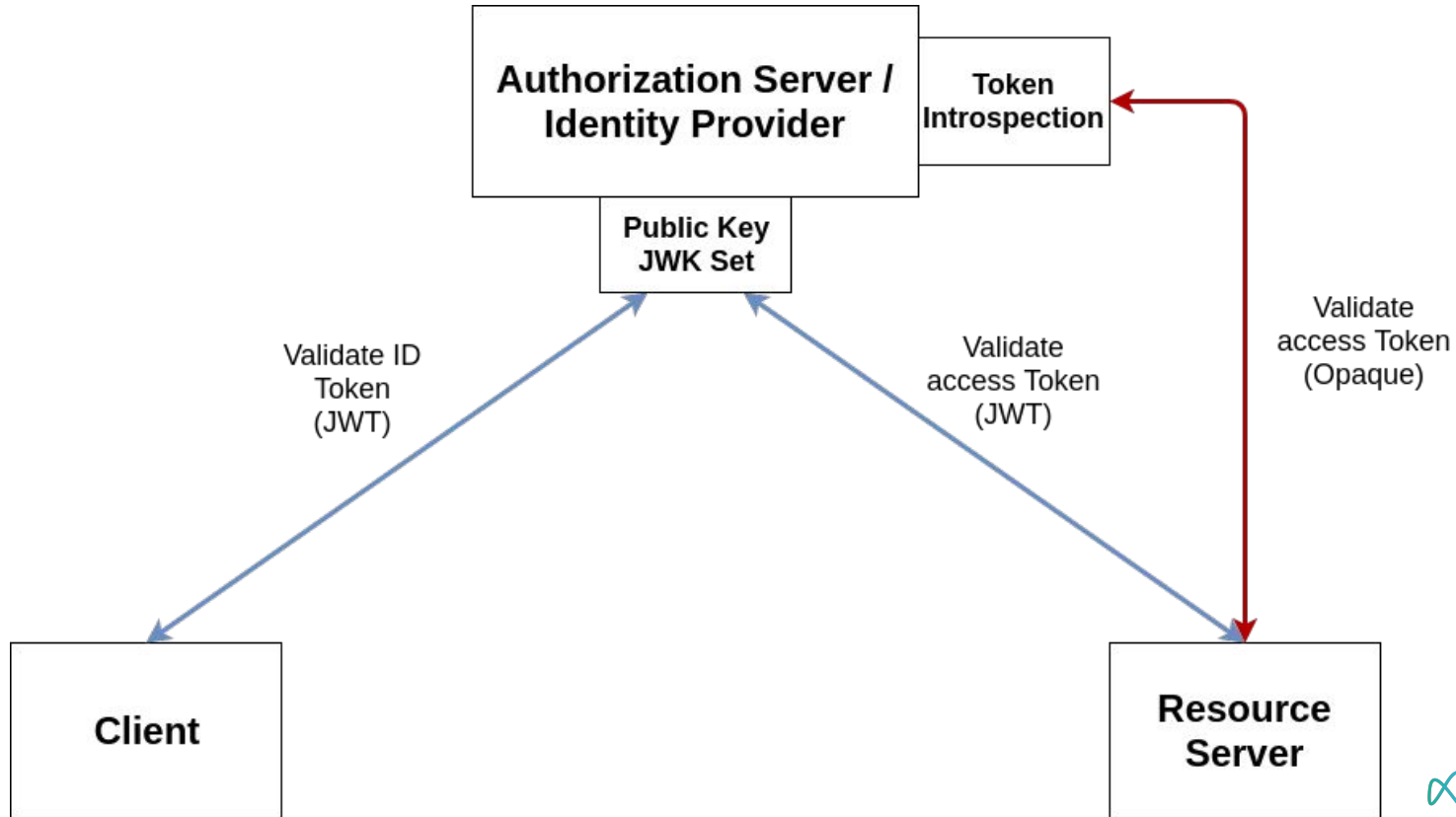
OAuth 2.0 Access Token JWT Profile (Draft)

- Required claims: iss, exp, aud, sub, client_id
- Consider privacy restrictions for identity claims
- Authorization claims according to SCIM Core (RFC7643):
  - Groups
  - Entitlements
  - Roles

System for Cross-domain Identity Management (SCIM)
JSON Web Token (JWT) Profile for OAuth 2.0 Access Tokens

NOVATEC

Access Token - Best Practices

- Bind Access Token to Client Id
- Limit Access Token Lifetime (5-60 minutes)
- Limit Access Token to Resources (Scope, Audience)
- Perform Required Validations
  - Opaque Tokens: Introspection
  - JWT: Signature, Expiration, Subject, Issuer, Audience

OAuth 2.0 Threat Model and Security Considerations (RFC 6819)
OAuth 2.0 Security Best Current Practice

NOVATEC

Refresh Token - Best Practices

- Bind Refresh Token to Client Id
- Rotate Refresh Token on Each Use
- Maximum Lifetime for each Refresh Token (e.g. 12h)
- Overall Maximum Time Limit for Refresh Tokens (e.g. 24h)
- No Refresh Token for Public Clients ("Silent Refresh")

OAuth 2.0 Threat Model and Security Considerations (RFC 6819)
OAuth 2.0 for Browser-Based Apps

NOVATEC

OpenID Connect Identity Providers

- RedHat/JBoss Keycloak (https://www.keycloak.org)
- Auth0 (https://auth0.com)
- Okta (https://www.okta.com)
- ForgeRock (https://www.forgerock.com/platform/identity-management)
- CloudFoundry UAA (https://github.com/cloudfoundry/uaa)
- PingFederate (https://www.pingidentity.com/en/platform/single-sign-on/sso-overview.html)
- Azure Active Directory (https://azure.microsoft.com/en-us/services/active-directory)
- ...

See: https://openid.net/developers/certified/#OPServices

NOVATEC

JBoss Keycloak as OIDC Identity Provider

- Open Source identity and access management product by RedHat/JBoss
- Currently based on JBoss Wildfly Application Server
- Implements OpenID Connect 1.0, OAuth 2.0 and SAML 2.0
- Provides a centralized user management
- Keycloak-X will move away from WildFly to Quarkus (https://quarkus.io)

https://www.keycloak.org/
https://www.keycloak.org/2019/10/keycloak-x.html

NOVATEC

Spring Security 5 Basics

NOVATEC

Spring Security High Level View

HTTP(S) Request →

HTTP(S) Response ←

HTTP Firewall

Security Filter Chain

✖ Strict-Transport-Security
✖ Content-Security-Policy
✖ X-Frame-Options
✖ X-XSS-Protection
✖ X-Content-Type-Options
✖ Referrer-Policy
✖ Feature-Policy

Security Response Headers

NOVATEC

Authentication in Spring Security

```
ExceptionTranslationFilter
        │
        ▼
AuthenticationEntryPoint
        │
        ▼
AuthenticationManager ──────▶ AuthenticationProvider ──────▶ UserDetailsService

                                                              UserDetails
```

ExceptionTranslationFilter

AuthenticationEntryPoint

AuthenticationManager

AuthenticationProvider

UserDetails

UserDetailsService

NOVATEC

Authentication in Spring Security

AuthenticationProvider

DaoAuthenticationProvider   JwtAuthenticationProvider   OpaqueTokenAuthenticationProvider

NOVATEC
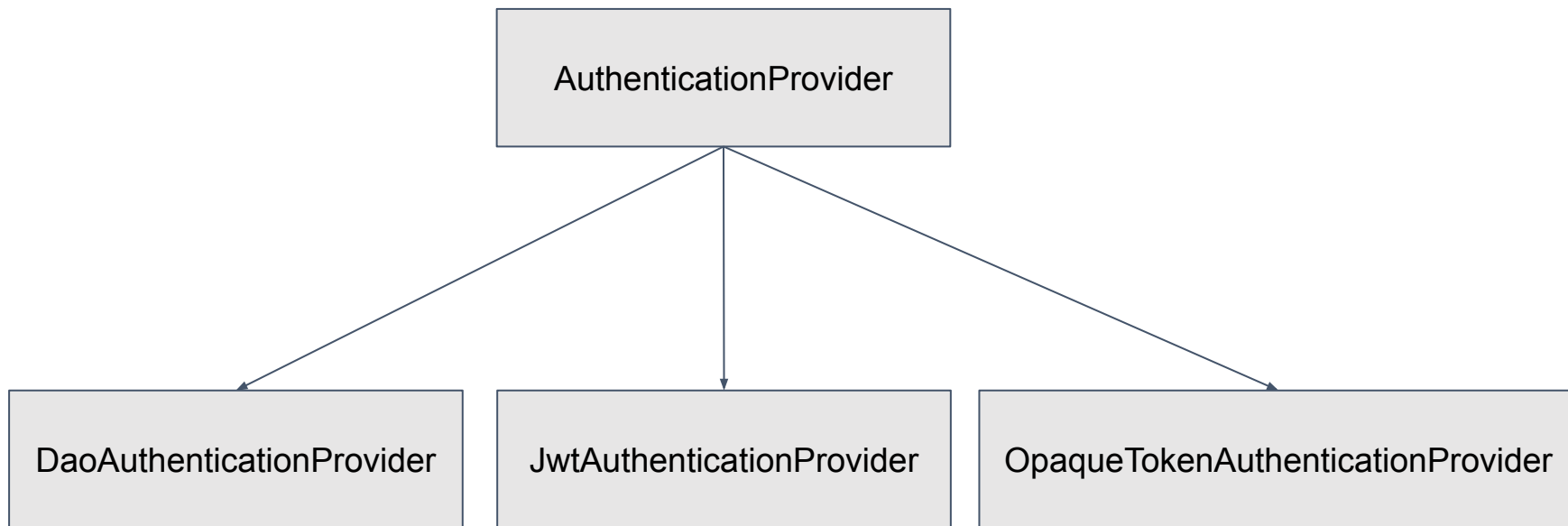
Authorization in Spring Security

@PreAuthorize
@Secured

SecurityContextHolder ← AbstractSecurityInterceptor

SecurityContext

AuthenticationManager → Authentication → GrantedAuthority

Principal

48

NOVATEC
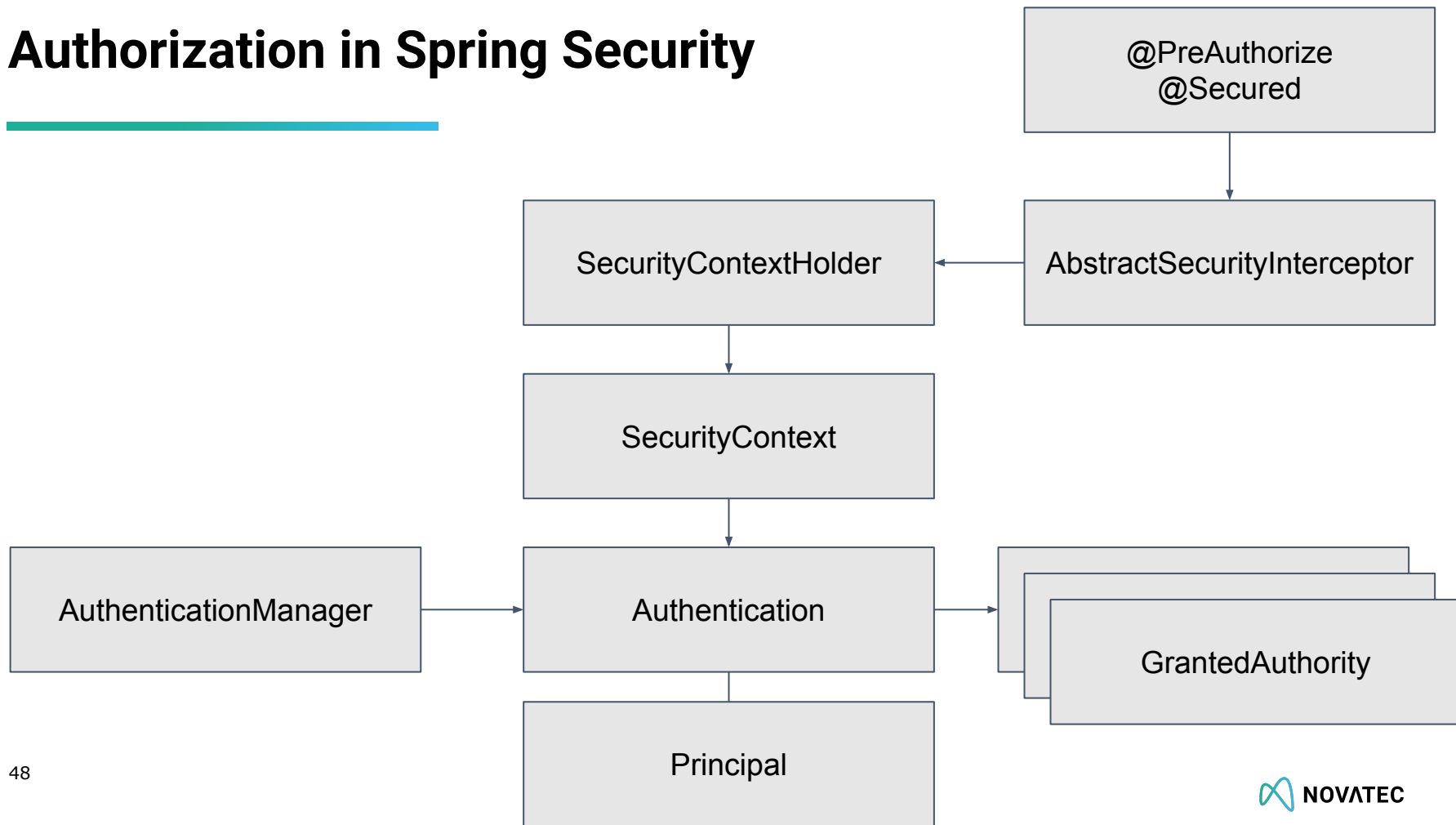
"Legacy" Spring Security 4.x OAuth2 Technology Stack

spring-security-oauth2-autoconfigure
spring-security-oauth2
spring-security-jwt
spring-boot-starter-security
spring-boot

NOVATEC

New Spring Security 5.x OIDC Technology Stack

spring-boot-starter-oauth2-client	spring-boot-starter-oauth2-resource-server
spring-security-oauth2-jose	
OpenID com.nimbusds:oauth2-oidc-sdk	
spring-boot	

https://connect2id.com/products/nimbus-oauth-openid-connect-sdk

NOVATEC
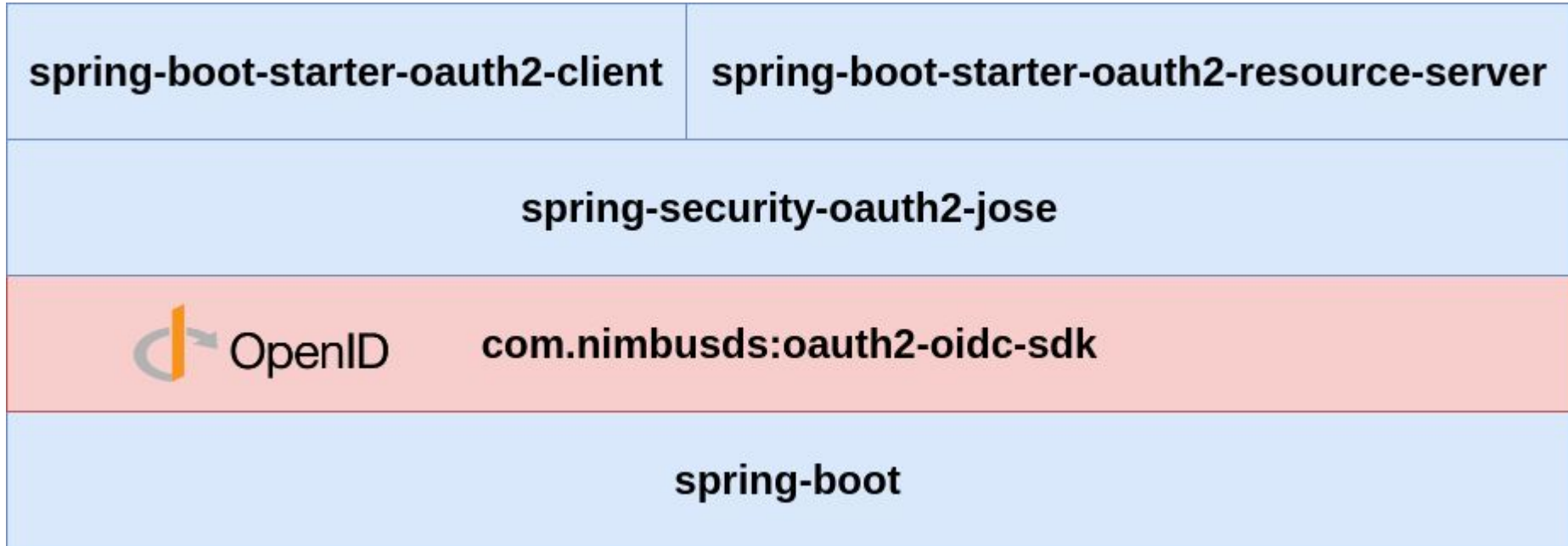
OpenID Connect on the Server side (The Resource Server)

NOVATEC

Authentication in a single Resource server

Authorization Server

JWK   Public Key

Validate JWT signature

Client   JSON Web Token (JWT)   Resource Server

GET / HTTP/1.1
Host: localhost:8080
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1N...

NOVATEC

JWT Authentication in Spring Security 5

NOVATEC
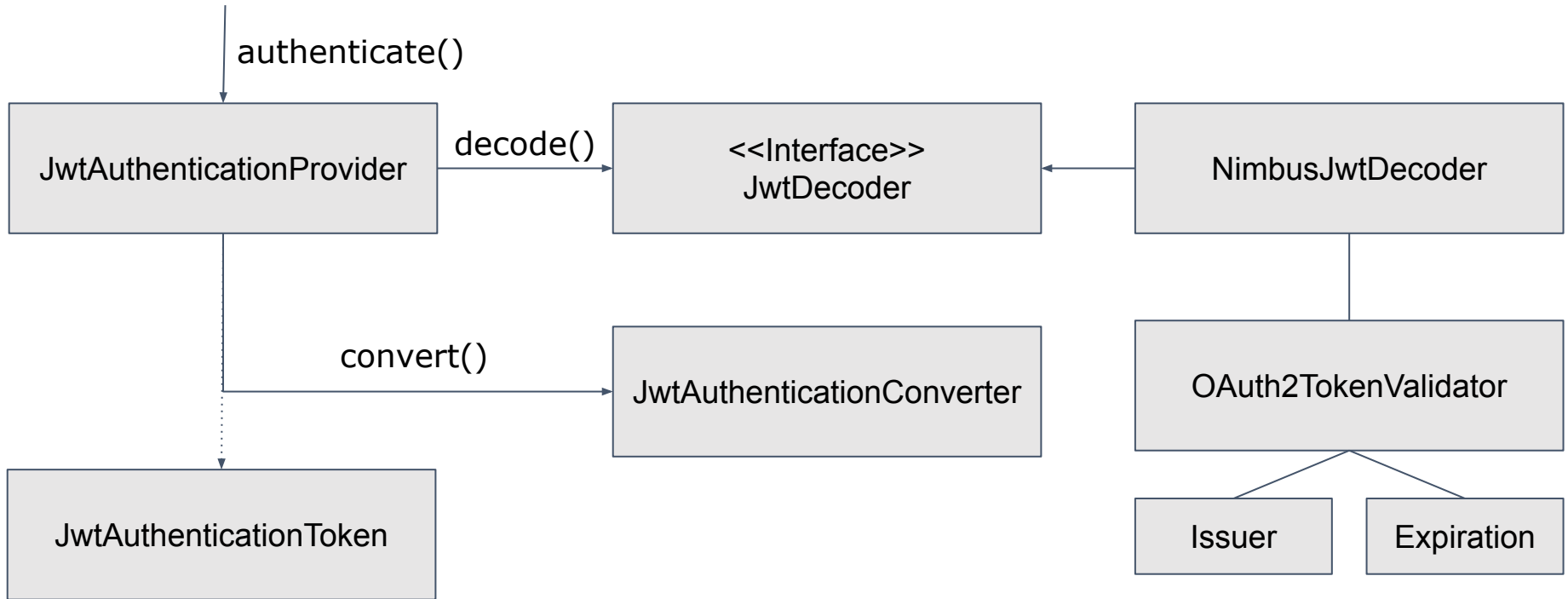
Practice Time

Lab 1: Implementing a Resource Server

NOVATEC

The Hands-On Application: Use Cases

- Online Book Library
  - Administer Library Users
  - Administer Books
  - List available Books
  - Borrow a Book
  - Return a previously borrowed Book

https://github.com/andifalk/secure-oauth2-oidc-workshop

NOVATEC

The Hands-On Application: Architecture
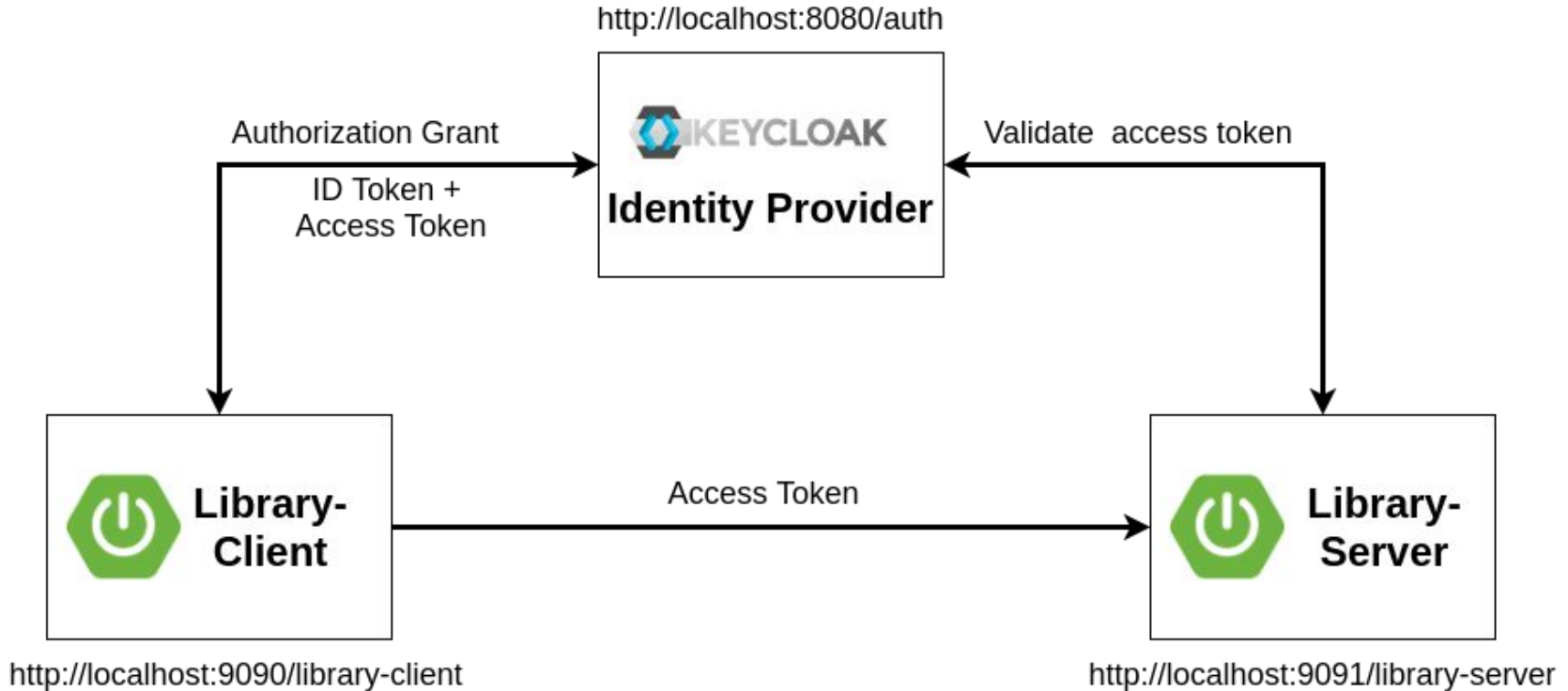
http://localhost:8080/auth

KEYCLOAK
Identity Provider

Authorization Grant
ID Token +
Access Token

Validate access token

Library-Client

Library-Server

Access Token

http://localhost:9090/library-client

http://localhost:9091/library-server

NOVATEC

OAuth 2.0 Resource Owner Password Credentials Flow (1)

```
+---------+
| Resource |
|  Owner  |
|         |
+---------+
     v
     |    Resource Owner
  (A) Password Credentials
     |
     v
+---------+                                         +---------------+
|         |>--(B)---- Resource Owner ------->|               |
|         |           Password Credentials   | Authorization |
| Client  |                                   |    Server     |
|         |<--(C)---- Access Token --------<|               |
|         |        (w/ Optional Refresh Token)|               |
+---------+                                         +---------------+
```

NOVATEC
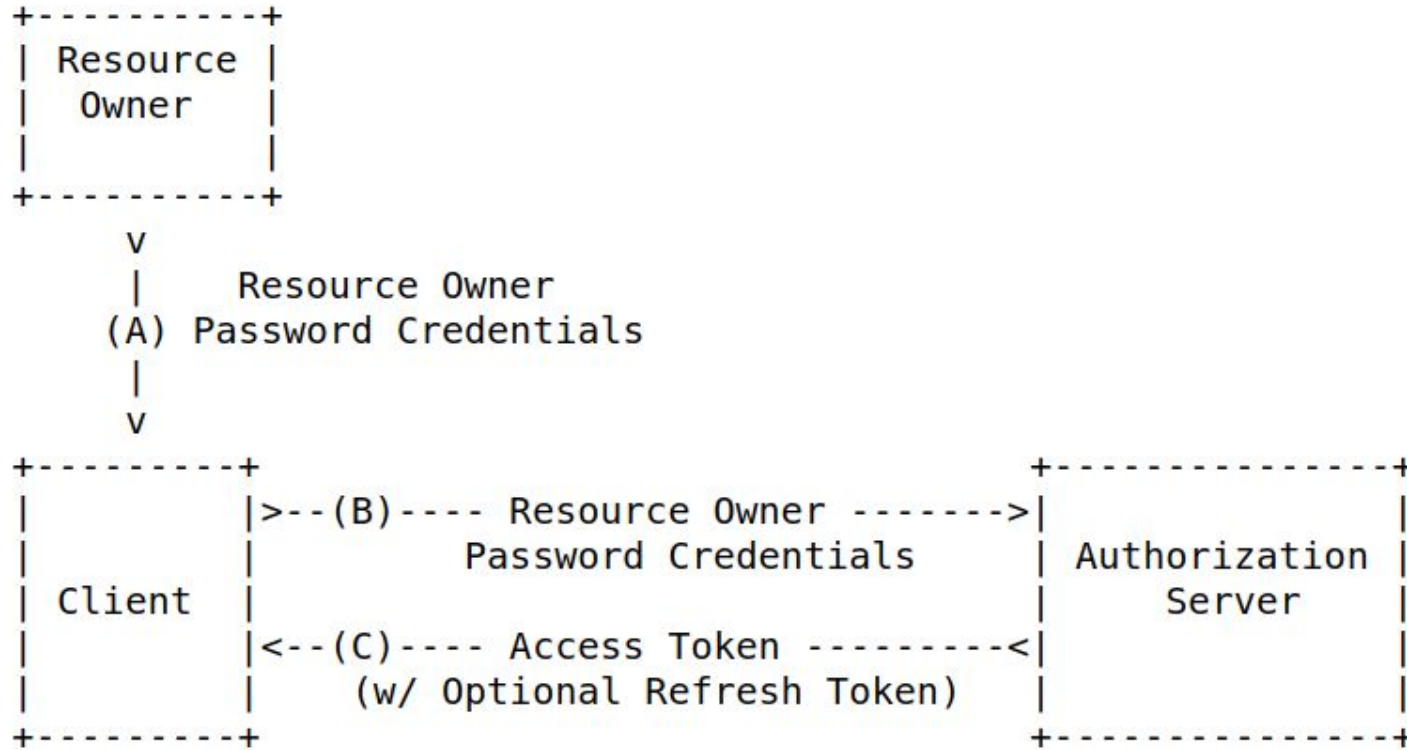
OAuth 2.0 Resource Owner Password Credentials Flow (2)

```
POST /token HTTP/1.1
     Host: server.example.com
     Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW
     Content-Type: application/x-www-form-urlencoded
     grant_type=password&username=johndoe&password=A3ddj3w
```

```
POST /token HTTP/1.1
     Host: server.example.com
     Content-Type: application/x-www-form-urlencoded
grant_type=password&username=johndoe&password=A3ddj3w
&client_id=123&client_secret=xyz
```

NOVATEC

Advanced OpenID Connect Scenarios

NOVATEC

Token-Relay with an API Gateway
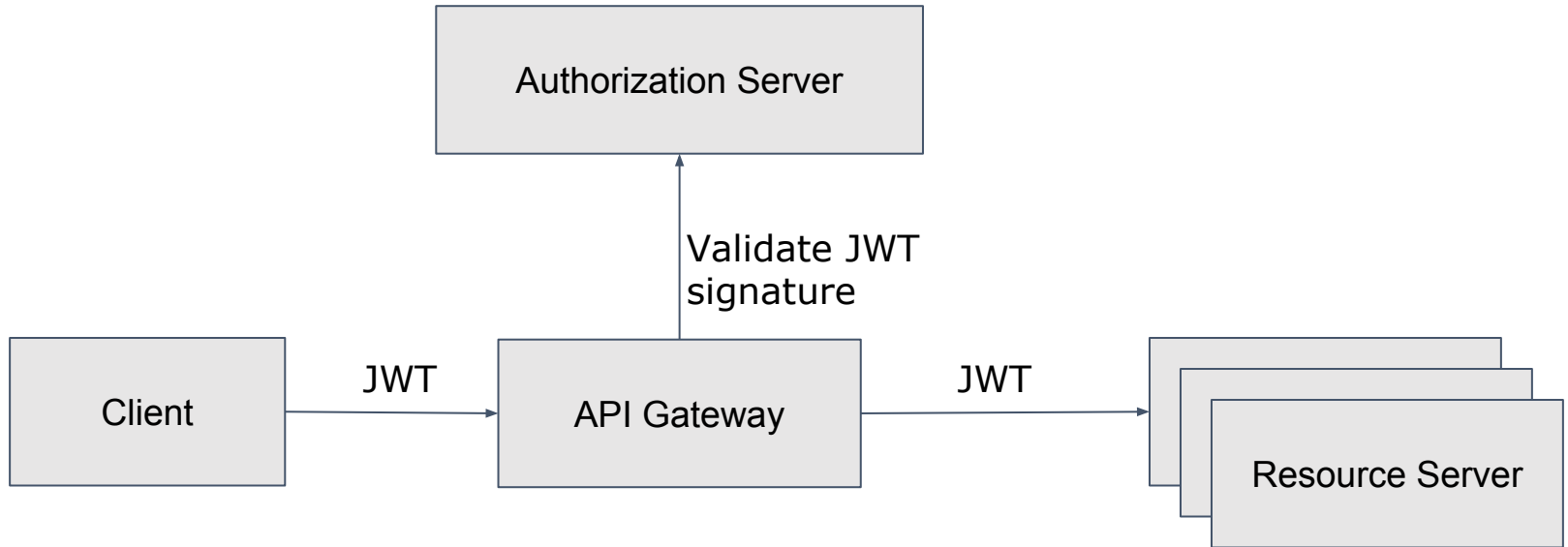
Token-Exchange with an API Gateway

Authorization Server

Validate JWT
signature & Token
Exchange

JWT

Audience
A

Resource Server A

Client

JWT

API Gateway

JWT

Audience B

Resource Server B

https://tools.ietf.org/html/rfc8693

NOVATEC
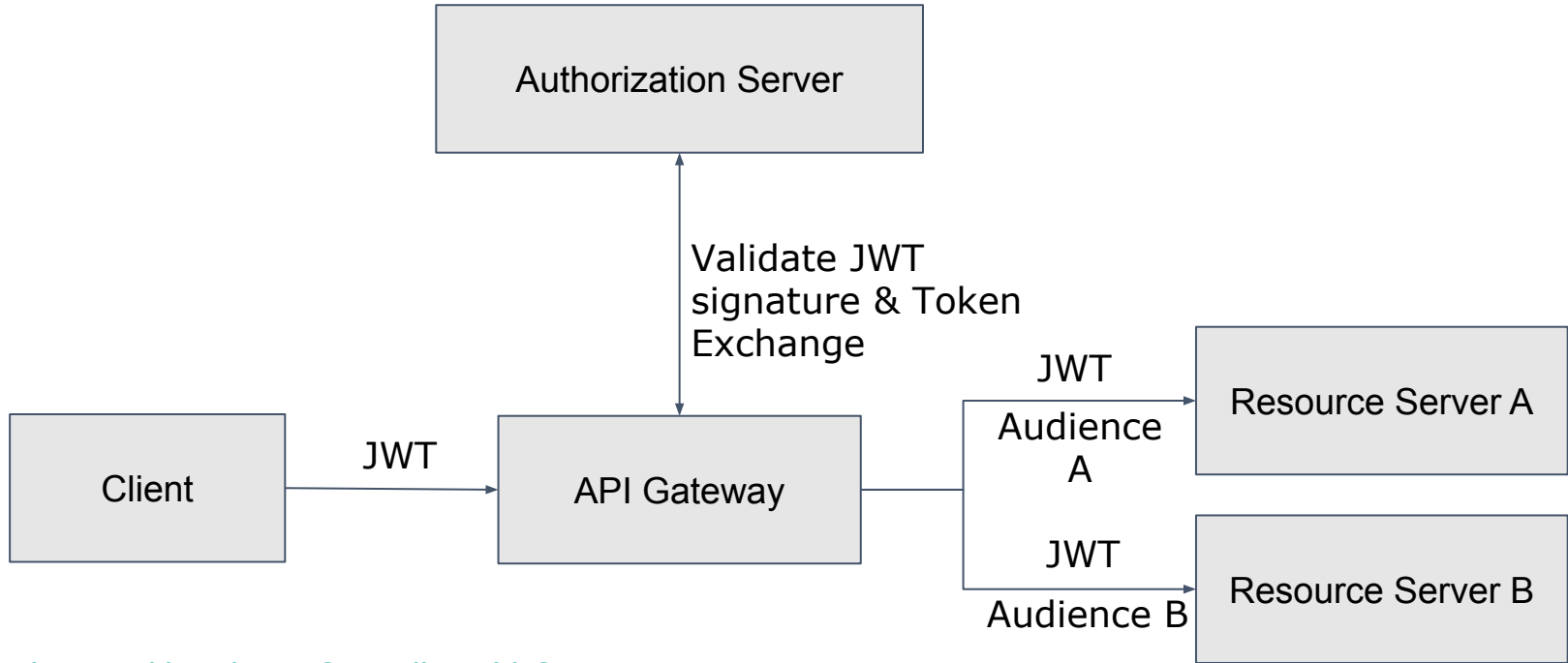
Microservice-to-Microservice calls

NOVATEC
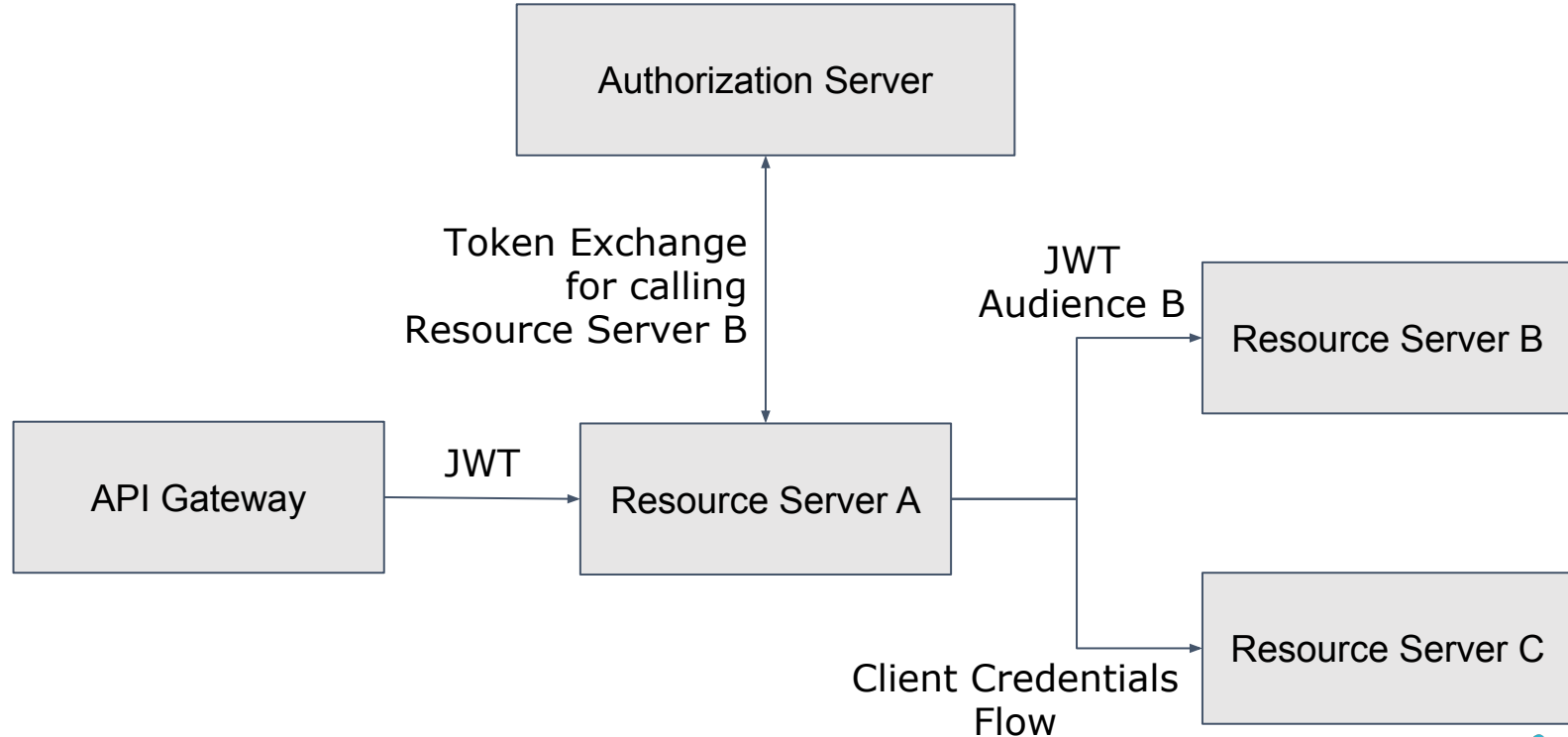
RFC 8693: OAuth 2.0 Token Exchange (Request)

POST authserver.com/oauth/token HTTP/1.1
    Host: serverA.com
    Authorization: Basic cnMwODpsb25nLXNlY3VyZS1yYW5k20tc2VjcmV0
    Content-Type: application/x-www-form-urlencoded

    grant_type=urn:ietf:params:oauth:grant-type:token-exchange
    &resource=http://serverB.com/api
    &subject_token=accVkjcJyb4BWCxGsndESCJQbdFMogUC5PbRDqceLTC
    &subject_token_type=urn:ietf:params:oauth:token-type:access_token

https://tools.ietf.org/html/rfc8693

NOVATEC

RFC 8693: OAuth 2.0 Token Exchange (Response)

HTTP/1.1 200 OK
Content-Type: application/json

```
{
    "access_token":"eyJhbGciOiJFUzI1NiIsImtpZCI6IjllciJ9.eyJhdWQiOiJo...",
    "issued_token_type":  "urn:ietf:params:oauth:token-type:access_token",
    "token_type":"Bearer",
    "expires_in":60
}
```

https://tools.ietf.org/html/rfc8693

NOVATEC

OpenID Connect on the Client side

NOVATEC

OAuth 2.0 Grant Flows

Client Type	Flow	Refresh Tokens
Confidential	Authorization Code	X
Public (Native)	Authorization Code (PKCE)	X
Public (SPA)	Implicit	--
Trusted	RO Password Creds	X
No Resource Owner	Client Credentials	--

NOVATEC

OpenID Connect Libraries

- oidc-client (Javascript) https://github.com/IdentityModel/oidc-client-js
- angular-oauth2-oidc (Typescript)
  https://github.com/manfredsteyer/angular-oauth2-oidc
- angular-auth-oidc-client (Typescript)
  https://github.com/damienbod/angular-auth-oidc-client
- IdentityModel.OidcClient (C#/.Net)
  https://github.com/IdentityModel/IdentityModel.OidcClient
- Nimbus OAuth 2.0 SDK (Java)
  https://connect2id.com/products/nimbus-oauth-openid-connect-sdk
- OIDC RP library (Python) https://github.com/openid/JWTConnect-Python-OidcRP
- ...

See: https://openid.net/developers/certified/#OPServices

NOVATEC

Spring Security: ClientRegistration

ClientRegistration
Repository

ClientRegistration

Metadata of OIDC
client registered at
the provider

NOVATEC
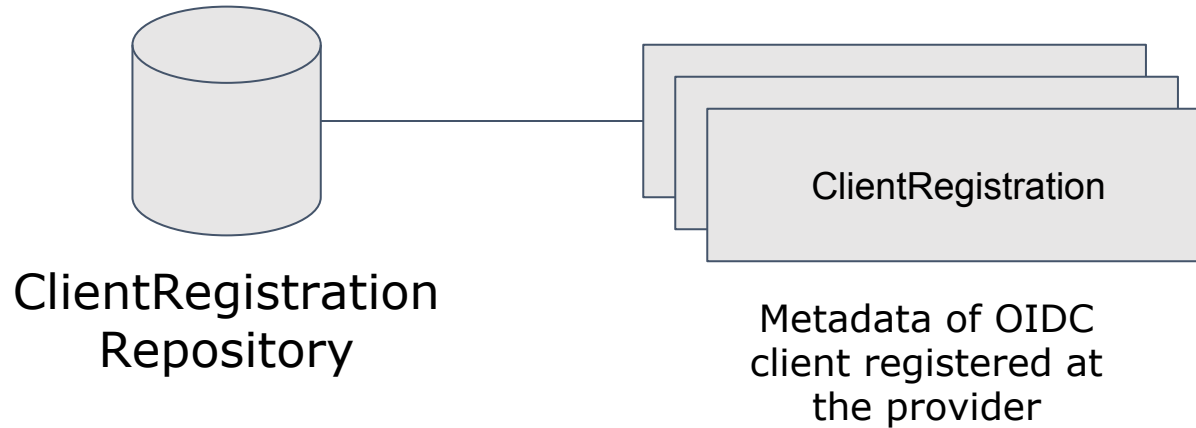
Spring Security: OAuth2AuthorizedClient

```
   _____              _____         _____
  /          /|            |                      |       |                         |
 /          / |            |  OAuth2Authorized    |------ |   ClientRegistration    |
|          |  |            |      Client          |       |_____|
|          |  |  ----------|                      |
|          |  |            |_____|       |_____|
|          | /                                            |  <<Resource Owner>>      |
|_____|/             State required to allow a        |     Principal           |
                          client to call a protected       |_____|
                          resource
OAuth2AuthorizedClient                                     |_____|
    Repository                                             |   OAuth2AccessToken     |
                                                           |_____|

                                                           |_____|
                                                           |   OAuth2RefreshToken    |
                                                           |_____|
```

NOVATEC
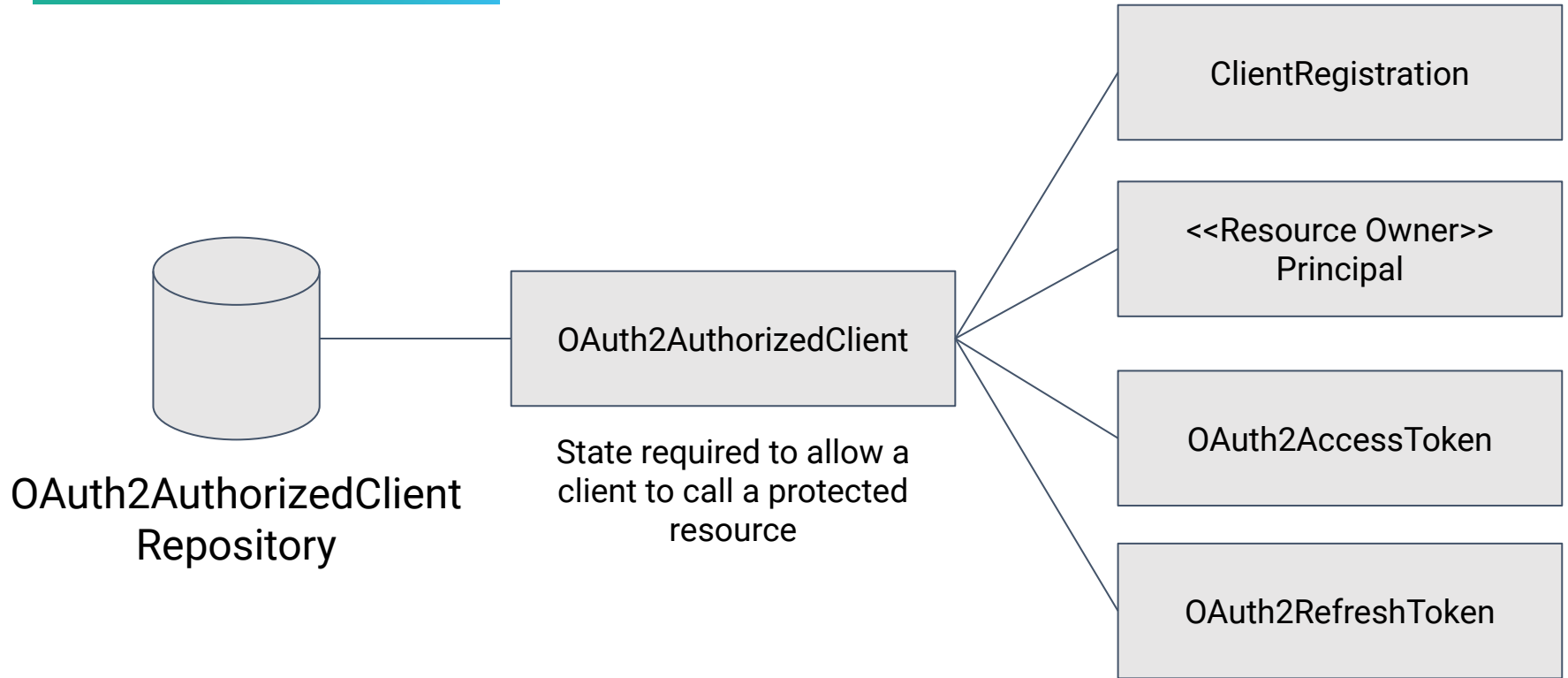
Spring Security: OidcUser

@AuthenticationPrincipal

OidcUser

OidcUserInfo

OidcIdToken

<<Claims>
Map<String, Object>

NOVATEC
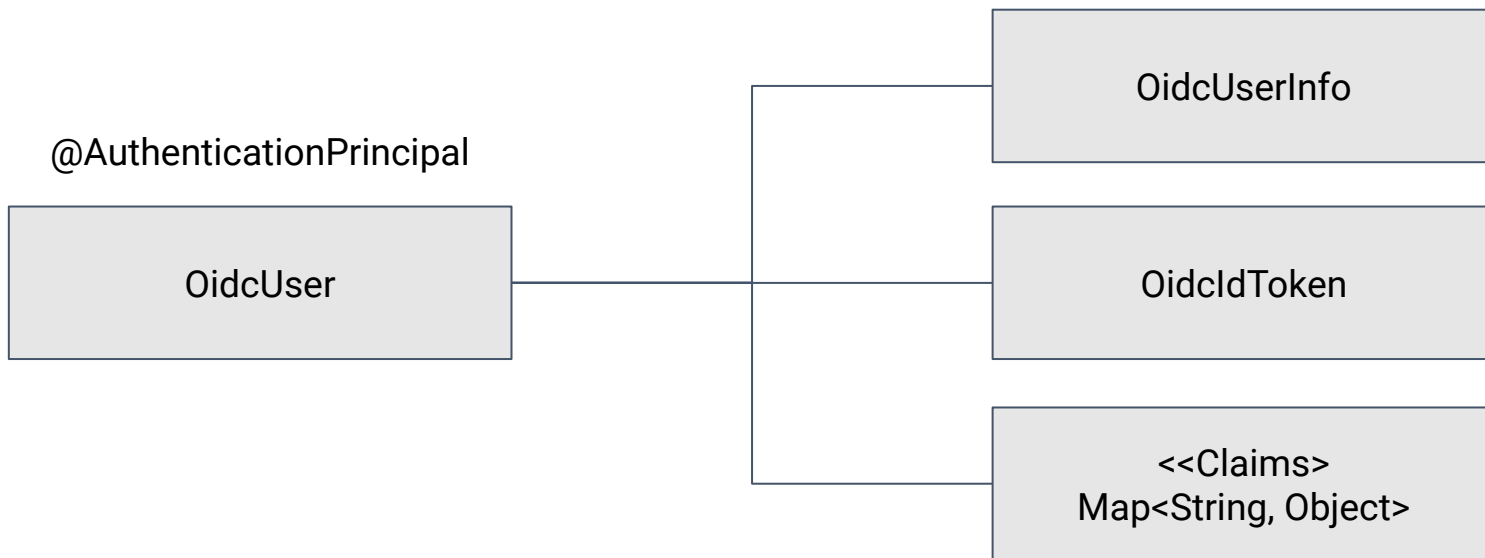
Practice Time

Lab 2: Implementing the client side (Authorization Code Flow)

NOVATEC

OAuth 2.0 / OIDC Client (Spring MVC / Thymeleaf)

Library Client

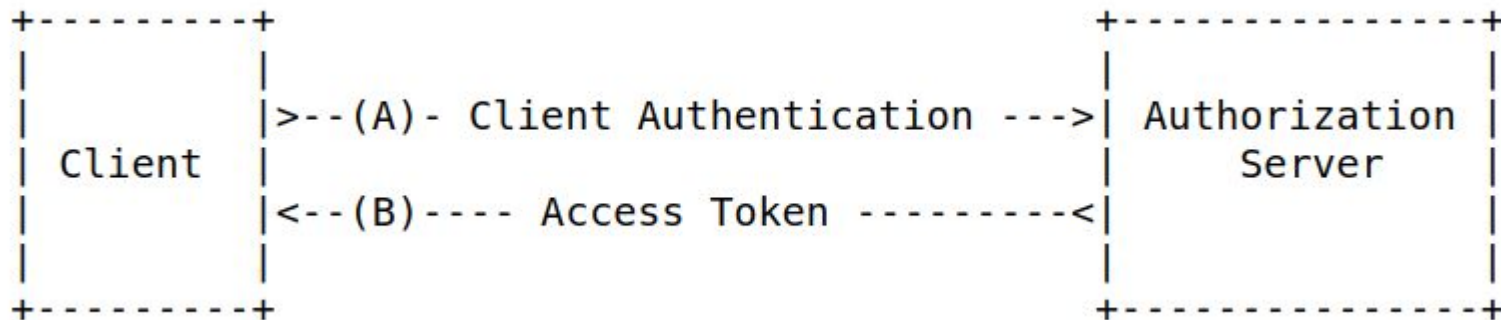A nice library to borrow books   Hello Bruce Wayne

Author(s)	Title	Description	ISBN	
[Bob C. Martin]	Clean Code	Even bad code can function. But if code isn't clean, it can bring a development organization to its knees. Every year, countless hours and significant resources are lost because of poorly written code. But it doesn't have to be that way. Noted software expert Robert C. Martin presents a revolutionary paradigm with Clean Code: A Handbook of Agile Software Craftsmanship . Martin has teamed up with his colleagues from Object Mentor to distill their best agile practice of cleaning code "on the fly" into a book that will instill within you the values of a software craftsman and make you a better programmer—but only if you work at it.	9780132350884	Return
[Josh Long, Kenny Bastiani]	Cloud Native Java	What separates the traditional enterprise from the likes of Amazon, Netflix, and Etsy? Those companies have refined the art of cloud native development to maintain their competitive edge and stay well ahead of the competition. This practical guide shows Java/JVM developers how to build better software, faster, using Spring Boot, Spring Cloud, and Cloud Foundry.	9781449374648	
[Craig Walls]	Spring in Action: Covers Spring 4	Spring in Action, Fourth Edition is a hands-on guide to the Spring Framework, updated for version 4. It covers the latest features, tools, and practices including Spring MVC, REST, Security, Web Flow, and more. You'll move between short snippets and an ongoing example as you learn to build simple and efficient J2EE applications. Author Craig Walls has a special knack for crisp and entertaining examples that zoom in on the features and techniques you really need.	9781617291203	Borrow
[Gene Kim, Jez Humble, Patrick Debois]	The DevOps Handbook	Wondering if The DevOps Handbook is for you? Authors, Gene Kim, Jez Humble, Patrick Debois and John Willis developed this book for anyone looking to transform their IT organization—especially those who want to make serious changes through the DevOps methodology to increase productivity, profitability and win the marketplace.	9781942788003	Borrow

NOVATEC

Practice Time

Lab 3: Implementing the client side (Client Credentials Flow)

NOVATEC

OAuth 2.0 Client Credentials Grant Flow (1)

```
+--------+                              +---------------+
|        |                              |               |
|        |>--(A)- Client Authentication --->| Authorization |
| Client |                              |    Server     |
|        |<--(B)---- Access Token --------<|               |
|        |                              |               |
+--------+                              +---------------+
```
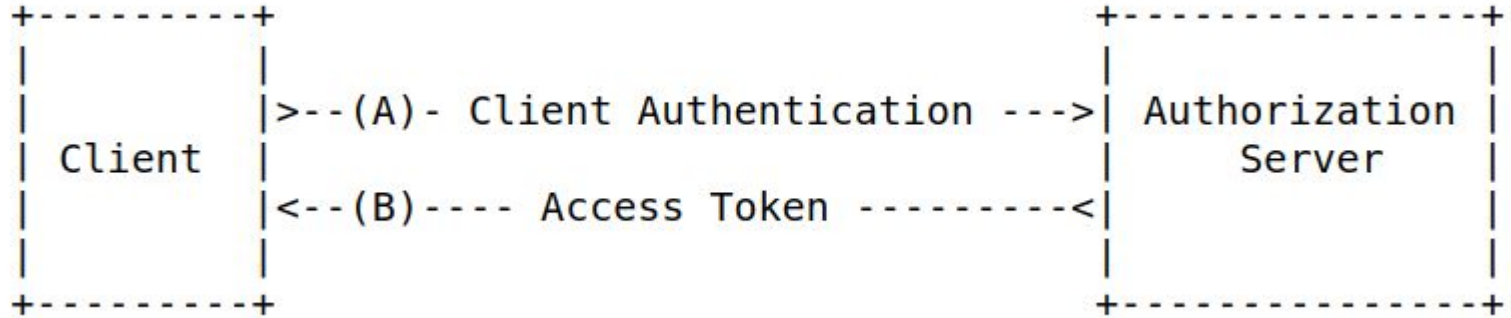
POST /token HTTP/1.1
   Host: server.example.com
   Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW
   Content-Type: application/x-www-form-urlencoded
   grant_type=client_credentials

NOVATEC

OAuth 2.0 Client Credentials Grant Flow (2)

```
+---------+                                              +----------------+
|         |                                              |                |
|         |>--(A)- Client Authentication --->|           | Authorization  |
| Client  |                                              |     Server     |
|         |<--(B)---- Access Token ---------<|           |                |
|         |                                              |                |
+---------+                                              +----------------+
```

POST /token HTTP/1.1
    Host: server.example.com
    Content-Type: application/x-www-form-urlencoded
    grant_type=client_credentials&client_id=123&client_secret=xyz

NOVATEC

Practice Time

Lab 4: Testing JWT Auth+Authz

NOVATEC

General Testing Strategies for JWT Auth+Authz

- Using Spring JWT test support (MockMvc / WebTestClient)
- Using self-signed JWT Tokens
- Using real identity provider (using Testcontainers)
- Only test the authorization layer

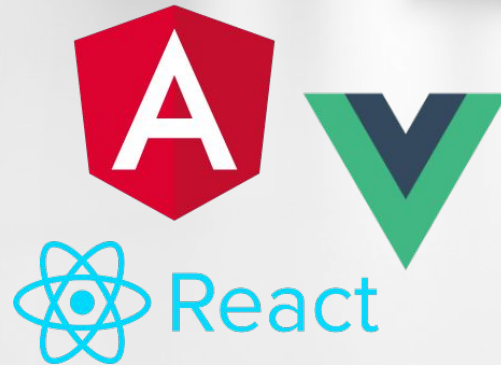https://docs.spring.io/spring-security/site/docs/current/reference/htmlsingle/#testing-bearer-authentication
https://docs.spring.io/spring-security/site/docs/current/reference/htmlsingle/#testing-bearer-authentication-2
 https://docs.spring.io/spring-security/site/docs/current/reference/htmlsingle/#oauth2resourceserver-jwt-decoder-public-key
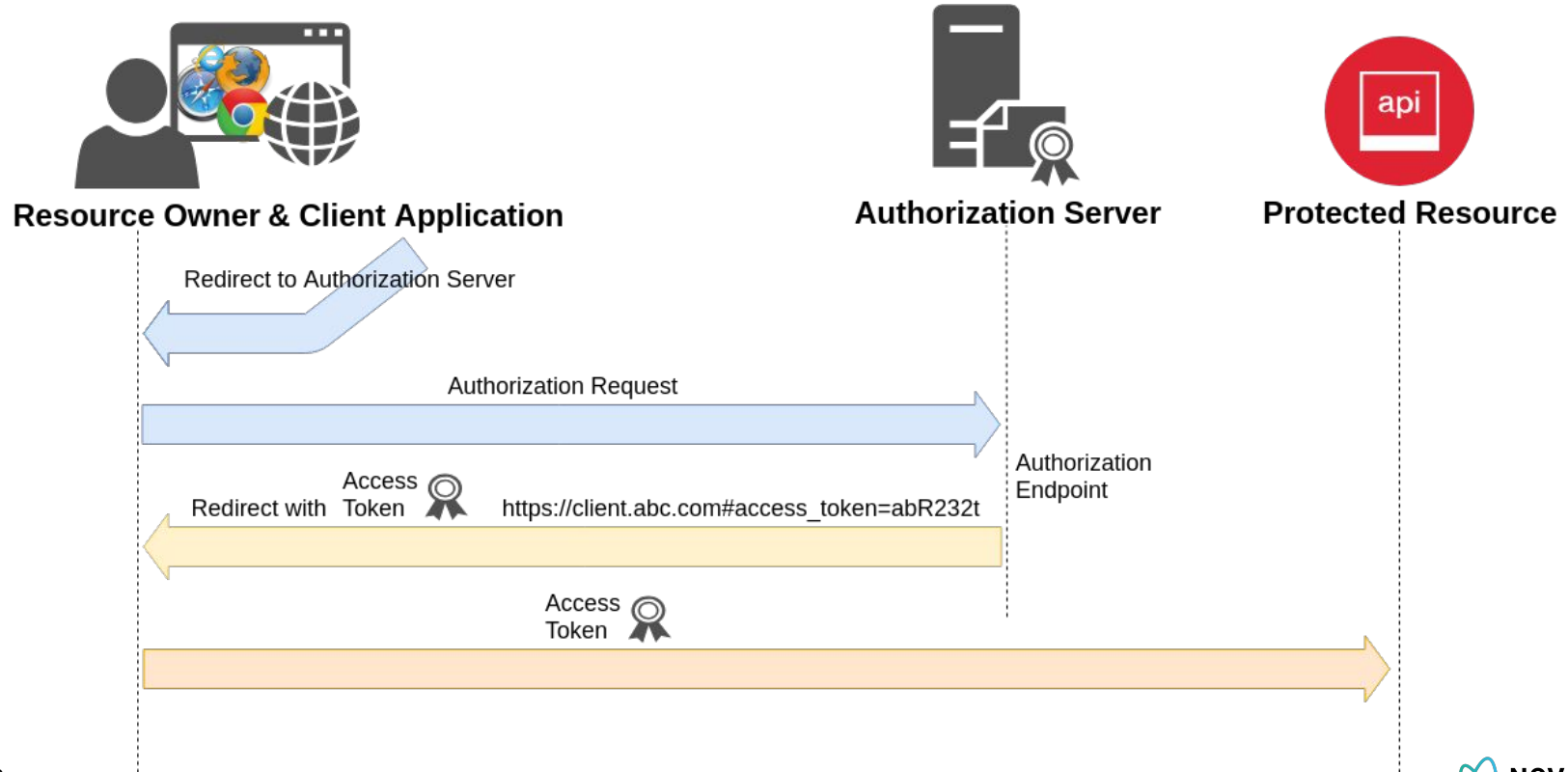https://www.testcontainers.org/
https://github.com/dasniko/testcontainers-keycloak

NOVATEC

What about Single Page Applications?

NOVATEC

OAuth 2.0 Grant Flows

Client Type	Flow	Refresh Tokens
Confidential	Authorization Code	X
Public (Native)	Authorization Code (PKCE)	X
Public (SPA)	Implicit	--
Trusted	RO Password Creds	X
No Resource Owner	Client Credentials	--

NOVATEC

OAuth 2.0 Implicit Flow

Resource Owner & Client Application   **Authorization Server**   **Protected Resource**

Redirect to Authorization Server

Authorization Request

Authorization Endpoint

Redirect with Access Token   https://client.abc.com#access_token=abR232t

Access Token

NOVATEC

OAuth 2.0 Implicit Grant Flow is at Risk !

Don't use the OAuth Implicit Grant any longer!

https://tools.ietf.org/html/draft-ietf-oauth-security-topics

NOVATEC
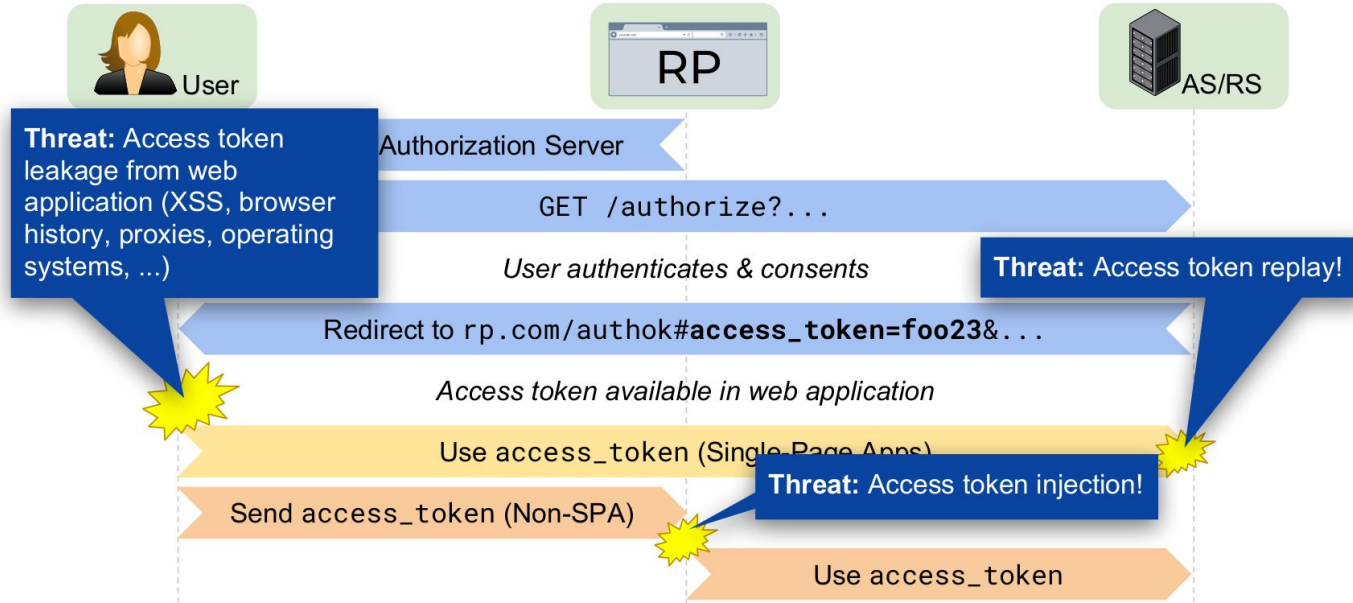
"OAuth 2.1" Grant Flows

Client Type	Flow	Refresh Tokens
Confidential	**Authorization Code (PKCE)**	X
Public (Native)	Authorization Code (PKCE)	X
Public (SPA)	**Authorization Code (PKCE)**	--
~~Trusted~~	~~RO Password Creds~~	~~X~~
No Resource Owner	Client Credentials	--

NOVATEC

Authorization Code + PKCE Grant Flow

- PKCE-Proof Key for Code Exchange ("pixy")
- Mitigates authorization code interception attack
- Public clients cannot keep secrets ("client_secret")
  - PKCE adds a dynamically created cryptographically random key: The "code verifier"

NOVATEC

OAuth 2.0 Auth Code + PKCE Grant Flow

```
                                    +--------------------+
                                    |   Authz Server     |
+----------+                        | +----------------+ |
|          |--(A)- Authorization Request --->|         | |
|          |       + t(code_verifier), t_m  | | Authorization | |
|          |                        | |   Endpoint     | |
|          |<-(B)---- Authorization Code -----|         | |
|          |                        | +----------------+ |
| Client   |                        |                    |
|          |                        | +----------------+ |
|          |--(C)-- Access Token Request ---->|         | |
|          |         + code_verifier | |    Token       | |
|          |                        | |   Endpoint     | |
|          |<-(D)------ Access Token ---------|         | |
+----------+                        | +----------------+ |
                                    +--------------------+
```

code_verifier: Random string (43-128 characters)

t(code_verifier): "code_challenge", Hashed code_verifier

t_m: Hashing algorithm to be used ("S256" or "None")

https://tools.ietf.org/html/rfc7636

84

NOVATEC

What's new in Spring Security 5.2&5.3

NOVATEC

Spring Security 5.2

- Client Support for PKCE
- OpenID Connect RP-Initiated Logout
- Support for OAuth 2.0 Token Introspection
- Resource Server Multi-tenancy (Servlet & Reactive)
- Use symmetric keys with JwtDecoder
- JWT Flow API in Test Support

https://spring.io/blog/2019/10/01/spring-security-5-2-goes-ga

NOVATEC

Support for Opaque Tokens

```java
class ResSrvConfig extends WebSecurityConfigurerAdapter {

    @Override
    protected void configure(HttpSecurity http)
            throws Exception {
        http.oauth2ResourceServer()
                .opaqueToken()
                    .introspectionUri(this.introspectionUri)
                    .introspectionClientCredentials(
                        this.clientId, this.clientSecret);
    }
}
```

NOVATEC

Resource Server Multi-Tenancy

```java
class ResSrvConfig extends WebSecurityConfigurerAdapter {

    @Override protected void configure(HttpSecurity http) {
        http.oauth2ResourceServer()
            .authenticationManagerResolver(
                multitenantAuthenticationManager());
    }

    @Bean AuthenticationManagerResolver<HttpServletRequest>
        multiTenantAuthMgr() {...}

    AuthenticationManager jwt() {...}
    AuthenticationManager opaque() {...}
}
```

NOVATEC

Support for Symmetric Keys

```java
class ResSrvConfig extends WebSecurityConfigurerAdapter {

    @Value("${spring.security.oauth2.resourceserver.
                  jwt.key-value}") RSAPublicKey key;

    @Override protected void configure(HttpSecurity http) {
        http.oauth2ResourceServer().jwt().decoder(jwtDecoder());
    }

    @Bean JwtDecoder jwtDecoder() throws Exception {
        return NimbusJwtDecoder.
                  withPublicKey(this.key).build();
    }
}
```

NOVATEC

JWT Flow API in Test Support

```java
public class OAuth2ResourceServerTest {

    @Test
    public void testRequestPostProcessor() {
        mockMvc.perform(get("/message")
          .with(mockAccessToken().scope("message:read")))
          .andExpect(status().isOk())

        mockMvc.perform(get("/")
          .with(jwt().claim(SUB, "the-subject")))
          .andExpect(status().isOk())
    }
}
```

NOVATEC

Lambda DSL

```
protected void configure(HttpSecurity http) throws Exception {
    http.authorizeRequests(authorizeRequests ->
        authorizeRequests
            .antMatchers("/blog/**").permitAll()
            .anyRequest().authenticated()
    ).formLogin(formLogin ->
        formLogin
            .loginPage("/login")
            .permitAll()
    );
}
```

https://spring.io/blog/2019/11/21/spring-security-lambda-dsl

Spring Security 5.3

- Support OAuth 2.0 / OIDC Authorization Server:
    - OpenID Connect 1.0 (Authorization Code Flow)
    - PKCE
    - OAuth 2.0 Client Credentials Grant
    - JWT Access Token format
    - JWK Set Endpoint
    - Opaque Access Token format

Spring Security team has decided to no longer provide support for authorization servers (...but still in discussion...)
https://spring.io/blog/2019/11/14/spring-security-oauth-2-0-roadmap-update

NOVATEC

Spring Security 5.3

- Resource server support for multiple trusted JWT access token issuers
- OidcIdToken.Builder & OidcUserInfo.Builder
- Opaque Token MockMvc & Reactive Test Support
- Oidc Login MockMvc & Reactive Test Support
- ...

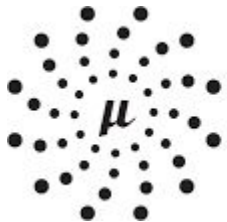https://spring.io/blog/2020/01/08/spring-security-5-3-0-m1-released

NOVATEC

What about other Microframeworks?

NOVATEC

OpenID Connect in other Micro-Frameworks

Already good support for JWT based authentication in:

- Micronaut
  https://micronaut-projects.github.io/micronaut-security/latest/guide/#jwt
- Quarkus https://quarkus.io/guides/jwt-guide

NOVATEC

OpenID Connect in other Micro-Frameworks

See the Bonus-Labs containing demos for Micronaut and Quarkus

NOVATEC

Books and Online References

NOVATEC

Books and Online References (1)

- Justin Richer et.al: OAuth2 in Action (Manning, 2017, ISBN 978-1617293276)
- Michael Schwartz et.al: Securing the Perimeter (Apress, 2018, ISBN 978-1484226001)
- RFC 6749: The OAuth 2.0 Authorization Framework
- RFC 6750: OAuth 2.0 Bearer Token Usage
- RFC 6819: OAuth 2.0 Threat Model and Security Considerations
- RFC 7636: Proof Key for Code Exchange ("Pixy")
- OpenID Connect Core 1.0 Specification
- OpenID Connect Dynamic Client Registration 1.0
- OpenID Connect Discovery 1.0
- RFC 7519: JSON Web Token (JWT)
- JSON Web Token Best Current Practices

NOVATEC

Books and Online References (2)

- [Why you should stop using the OAuth implicit grant](#)
- [OAuth 2.0 Security Best Current Practice](#)
- [OAuth 2.0 for Browser-Based Apps](#)
- [OAuth 2.0 Mutual TLS Client Authentication and Certificate-Bound Access Tokens](#)
- [JSON Web Token (JWT) Profile for OAuth 2.0 Access Tokens](#)
- [OAuth 2.0 Token Exchange](#)

NOVATEC

Books and Online References (3)

- [Resource Indicators for OAuth 2.0](#)
- [Spring Security 5.2 Reference Documentation](#)
- [Microservices Security Patterns & Protocols with Spring Security (Devoxx Video)](#)
- [Microservices Security Patterns & Protocols (SpringOne Platform 2019 Video)](#)
- [How to secure your Spring apps with Keycloak by Thomas Darimont (Video)](#)

NOVATEC

NOVATEC

Andreas Falk
Managing Consultant

Mobil: +49 151 46146778
E-Mail: andreas.falk@novatec-gmbh.de

Novatec Consulting GmbH
Dieselstraße 18/1
D-70771 Leinfelden-Echterdingen

T. +49 711 22040-700
info@novatec-gmbh.de
www.novatec-gmbh.de