



# Identity Management with KEYCLOAK

Thomas Darimont  
*Java User Group Saarland*



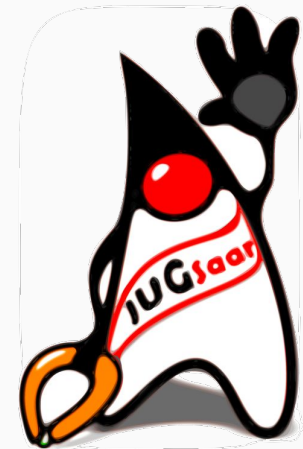
25.10.2016

# Thomas Darimont

t.darimont@eurodata.de



- Dipl. Wirtschaftsinformatiker BA
- Software Architect @ eurodata AG
- Open Source Enthusiast
- Former Spring Team Member
- Java User Group Saarland
- 14+ Years in the Industry



- Core Identity Management concepts
- Single Sign-On Overview
- Web SSO Foundations
- Keycloak Project Overview
- Keycloak Integration Demos
- Keycloak Production Setup

# Identity Management

## Identity Management - in a nutshell...



- Identity Management (IdM)
  - User Account Management
  - User Groups and Roles
  - User Authentication
  - User Authorization
  - Single Sign-On
- Federated Identity Management (FIdM)
  - Management of User Accounts across Domains / Companies
  - e.g. Windows AD + legacy User Store
- Identity & Access Management (IAM)
  - Includes Identity Management
  - Includes management of fine-grained Permissions
  - Provisioning of IT-Resources

- Authentication (AuthN)
  - Who is the User?
  - Identity
- Authorization (AuthZ)
  - What is the user allowed to do?
  - Role based Access Control (RBAC)
  - Permissions
- Single Sign-On
  - use of multiple apps / services with a single login
- Provisioning
  - automatic allocation and assignment of IT Resources
  - Deprovisioning - Tear down of provided IT Resources

- **Reduced Help-Desk**
  - Self-Service Account Management
  - Password Management and User Profile updates
- **Improved Service**
  - Quick provisioning of Services
  - Centralized User Account
  - Single Sign-On across multiple Applications
- **Improved Security**
  - Uniform Password Policies
  - Sophisticated Audit capabilities
  - Multi-Factor Authentication



# Single Sign-On

- Single Sign-On (SSO)
  - Uniform Login
  - For multiple applications
- Variants
  - Ticket based
  - Certificate based
  - Portal based
- Single Logout (SLO)
  - Global - log-out from all Applications
  - Local - “log-out” of a single Application
- Different Scenarios
  - Cloud (Google, Facebook, Amazon)
  - In-house (Active Directory)

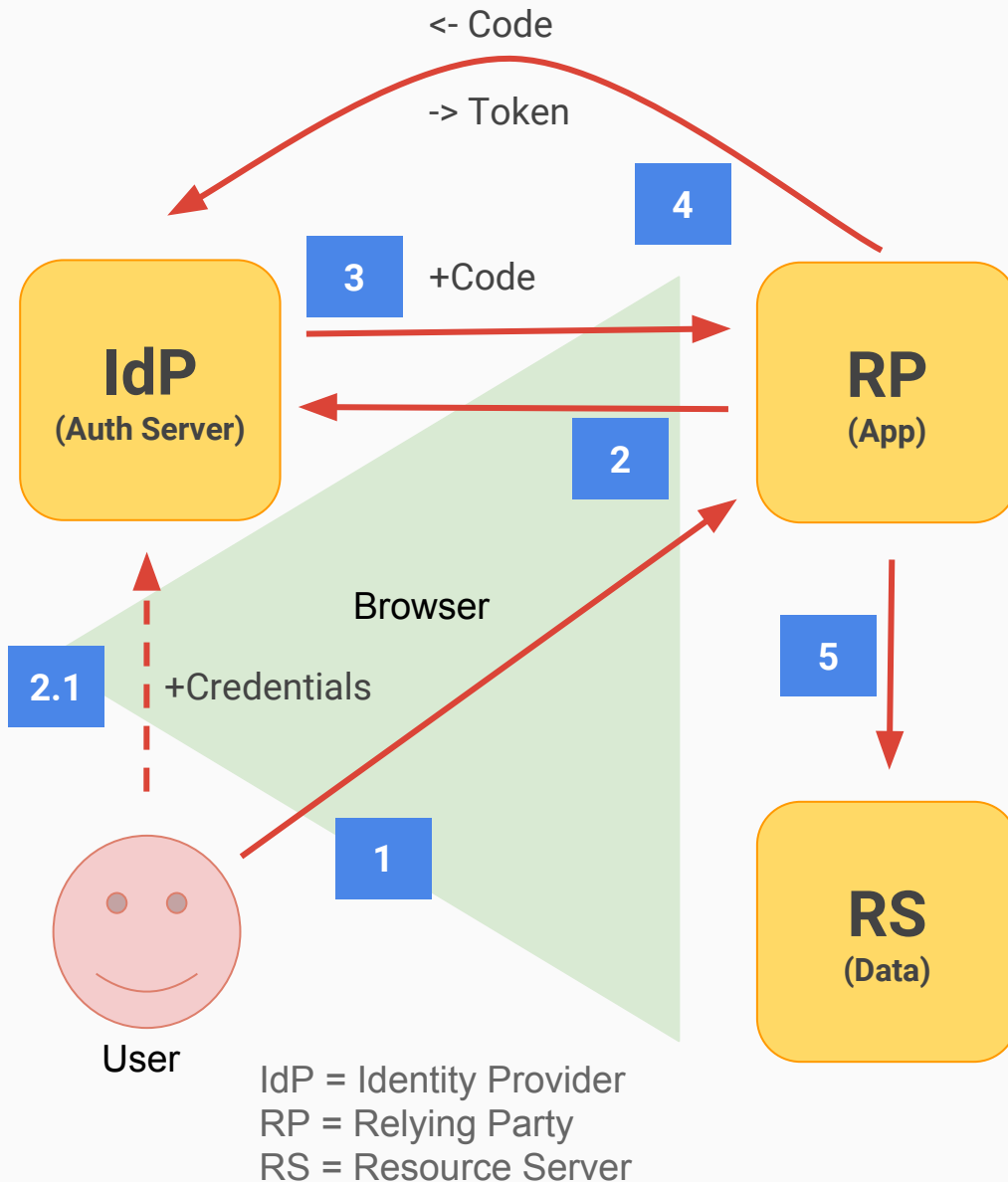
| Protocol   | Usage examples  |
|--|---|
| Proprietary                                      | Atlassian Crowd, “Facebook Login”                     |
| Kerberos   | Active Directory                                      |
| CAS  | JASIG   |
| SAML   | Shibboleth, <b>Keycloak</b>                           |
| <sup>old</sup> OAuth / OpenID 1.0 / 2.0          | Google, Yahoo, Facebook, ...                          |
| <sup>new</sup> <b>OAuth 2.0 / OpenID Connect</b> | Google, Microsoft, Ping Identity, ... <b>Keycloak</b> |

# OAuth 2.0

- Authorization Protocol “Framework” for Access Delegation
- Specified by IETF OAuth WG [oauth.net/2/](https://oauth.net/2/)
- Complex, but very well documented
- Multiple Protocol Flows
- Exchange of Tokens
  - AccessToken: “Ticket” to access a Resource - short-lived (Minutes)
  - RefreshToken: For requesting new Tokens - long-lived (Days)
- Claim
  - Piece of Data within the Token
  - e.g. Email Address or Username
- Scopes
  - Named Collections of Claims or Operations
  - Full Name, Email or Write access to your Facebook Wall

- Identity Provider (IdP)
  - Also known as “Authorization Server”
- Relying Party (RP)
  - Client / Application / Service
- Resource Server (RS)
  - Provides Resources to other Clients / Users
  - Often same as RP

# OAuth 2.0 Flow Example - Authorization Code Flow



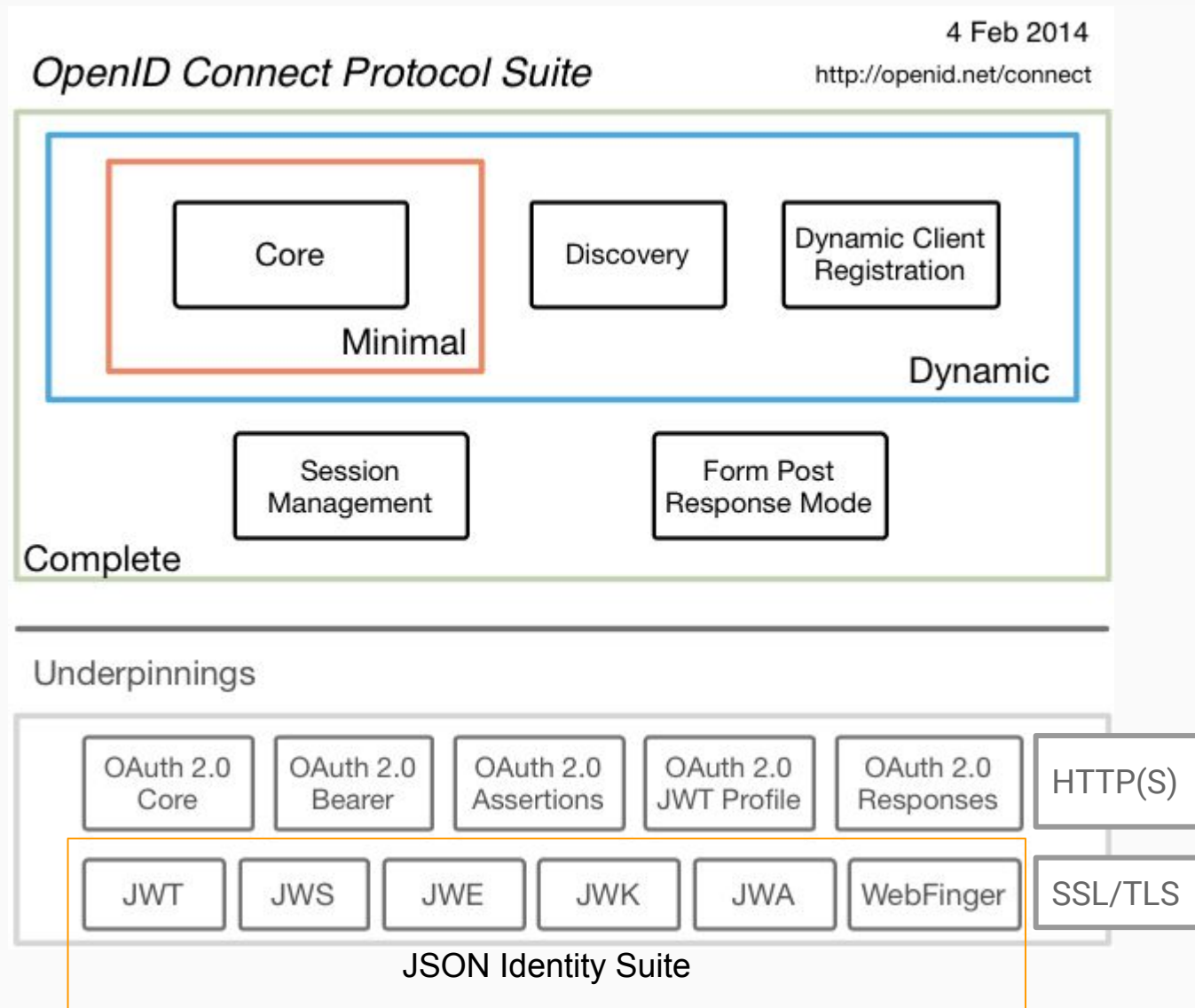
- 1 Unauthenticated User access App
- 2 App redirects User to IdP
- 2.1 User submits Credentials to IdP
- 3 IdP redirects User with Code to RP
- 4 RP exchanges Code to Tokens w. IdP
- 5 RP uses Token to access RS on users behalf

# OpenID Connect



- Authentication Protocol based on OAuth 2.0
- “Identity Layer” on top of OAuth 2.0
- Simple, widespread, flexible & well documented
- Recommended for Mobile- / and Web-Applications
- Features
  - Additional Token: **ID Token** with User Information
  - Session Management for SSO + SLO
  - User Info Endpoint / centralized User Account
  - Discovery - Self-Describing Endpoints
  - Client Self-Registration
  - Front- / Back-Channel Logout

# OpenID Connect Profiles



# What is OpenID Connect? How does it work?



Base Protocol ↗

- [OpenID Connect Basic Client Implementer's Guide 1.0 - draft 37](#)
- OpenID Provider (OP)
  - OAuth 2.0 Auth Server that is capable of Authenticating the User
  - Manages User Accounts and Credentials
  - Manages Tokens (Create, Validate, Invalidate)
- Relying Party (RP)
  - aka “Service Provider” | “Resource Server” | “OAuth Client”
  - Provides Service / Data / Application

- JWT - pronounced like “jot”
- Part of the JSON Identity Suite
- Contains information about an Entity (User) relevant for a Client (App)
- Compact representation
- Information is digitally signed
  - Secret (HMAC)
  - Public/Private Key pair (RSA)
- Usage
  - HTTP POST Body
  - HTTP Header
  - URL Query String (insecure)
  - **recommended always** via HTTPS
- Good overview <https://jwt.io/introduction/>

## JSON Web Token - Example

**<header-base64>.<payload-base64>.<signature-base64>**

Encoded

[illegible]Decoded EDIT THE PAYLOAD AND SECRET (ONLY HS256 SUPPORTED)

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "RS256",
  "typ": "JWT",
  "kid": "NR2QXkvaVCNn18bmTqmjKIyn2Tx2o40A3N3ggS-EYJQ"
}
```

PAYLOAD: DATA

```
{
  "jti": "84b9a06a-6eea-4deb-b3d6-342363119a38",
  "exp": 1472802390,
  "nbf": 0,
  "iat": 1472802090,
  "iss": "https://test-
sso.eurodata.de/u/auth/realms/eurodata-dev",
  "aud": "app-edrewe",
  "sub": "d12bdd95-cfa1-41fc-a91c-8b005dcfd68a",
  "typ": "Bearer",
  "azp": "app-edrewe",
  "auth_time": 0,
  "session_state": "1fd79003-49d5-4a74-81d9-d2d1a0a1bc00",
  "acr": "1",
  "client_session": "dbde966d-180a-469d-9213-36a1af888e59",
  "allowed-origins": [],
  "resource_access": {
    "realm-management": {
      "roles": [
        "view-users"
      ]
    },
    "app-edrewe": {
      "roles": [
        "user"
      ]
    },
    "account": {
      "roles": [
        "manage-account",

```

# Keycloak

- Identity Management Solution
- Open Source by JBoss (Redhat)
- Current Version 2.3.0.CR1 (Oct 2016)
- Support for User / Group / Role Management
- Self-Service Account-Management
- Single Sign-on (SSO) and Single Logout (SLO) Support
  - OAuth 2.0 / OpenID Connect
  - SAML 2.0
- Identity Brokering, User Federation, Social Login
- Multi-Factor Authentication (MFA) with TOTP/HOTP
- Multi-Tenancy
- Themes and L10N
- Multiple Persistence Technologies supported (RDBMS, MongoDB, File)
- Support for High Availability
- Web based Admin Console
- Many extension points
- Commercial Product Available (Redhat SSO - based on Keycloak 1.9.8.Final)



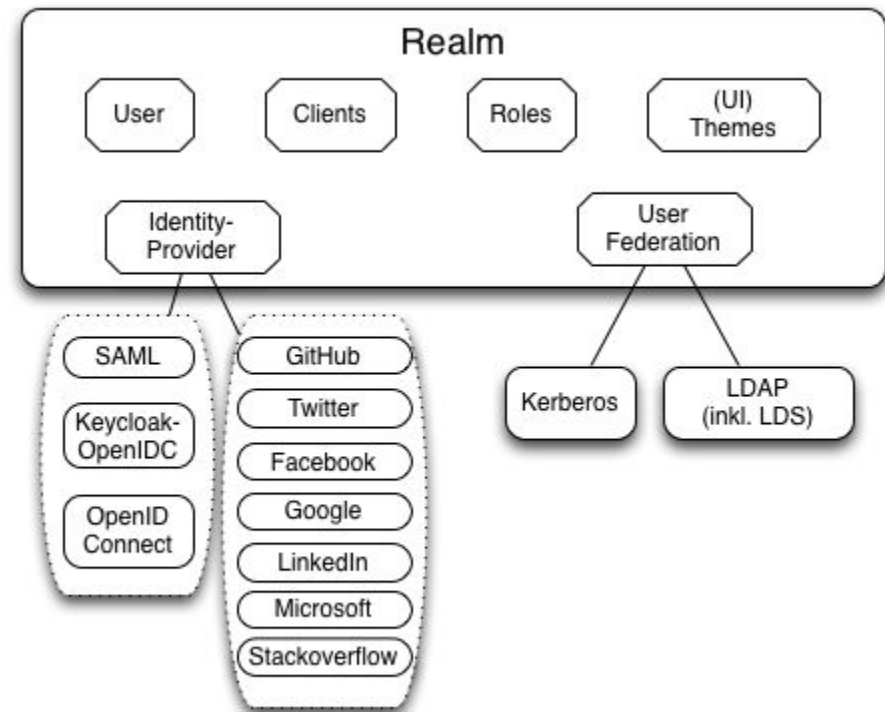
<http://www.keycloak.org>



- [OAuth 2.0 Threat Model and Security Considerations](#)
- [Keycloak Threat Model](#)
- Security Best Practices
- Secure communication over HTTPS/SSL
- Host based Access Control
- Hierarchical Role based Access Control - (H-RBAC)
- Encrypted Credentials
  - Passwords: PBKDF2 hash function with Salt 20.000 Iterations by default
  - OTP: HmacSHA1
- Security Defenses
  - Browser Security
  - Brute-force Protection (reCaptcha)
- User / Admin action Audit Log

# Keycloak - Terminology

- **Realm**
  - Domain for User Accounts and Applications
  - Authentication, Policies, User Federation
  - Sessions, Events
  - Theme, Email, Timeouts
  - Global Roles
- **Client**
  - OAuth 2.0 Client
  - Represents an Application
  - Client-specific Roles
  - Protocol Mappers
  - Authentication / Authorization
- **Group**
  - User Groups
  - Group specific Roles
- **User**
  - User Account
  - User specific Roles
  - Group Membership
  - User Attributes
  - Password / MFA



# Keycloak Technology Stack

## Frontend

- Angular JS (1.4.4)
- PatternFly
- Bootstrap



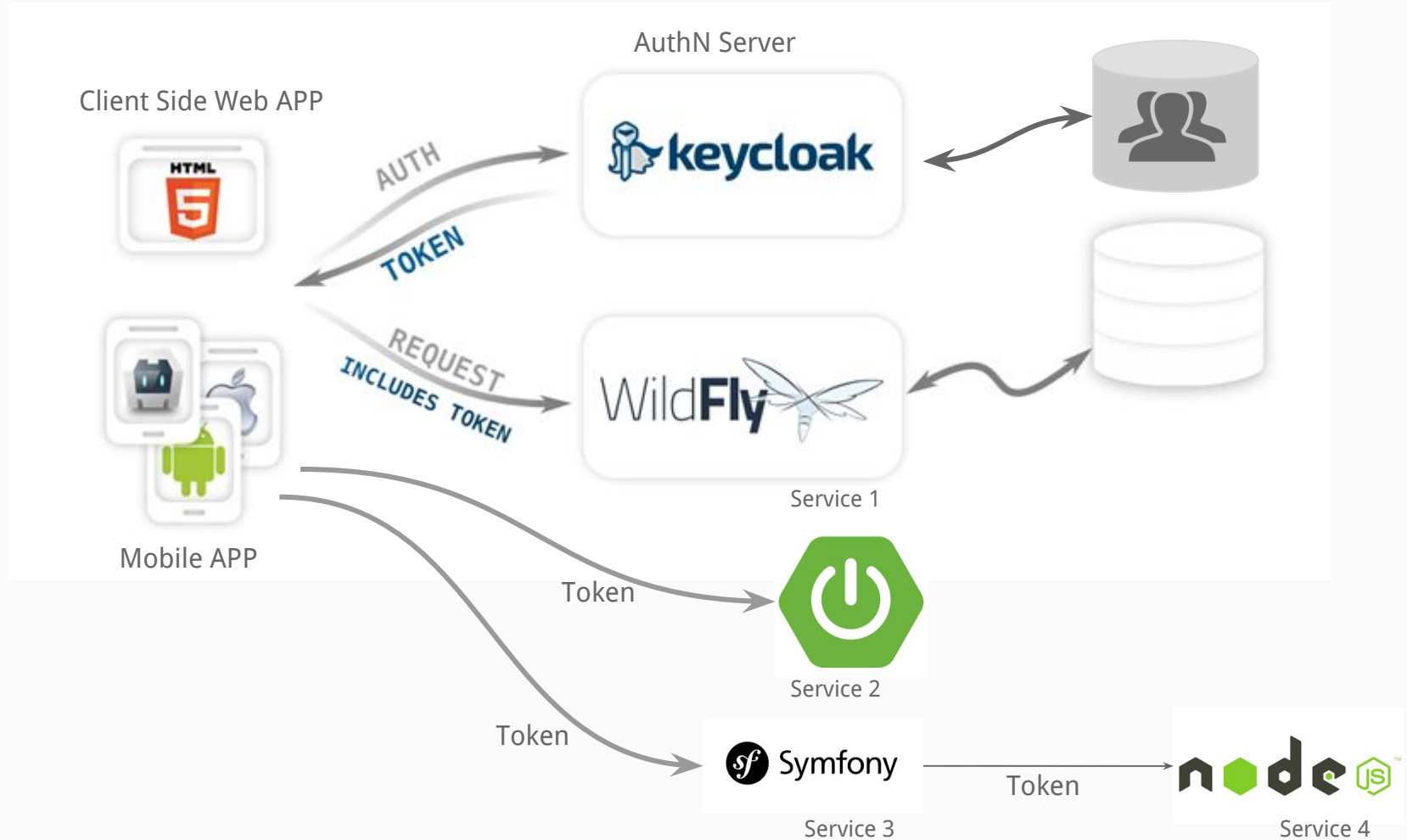
↕ HTTP(S) / REST

## Backend

- JAX-RS (Resteasy)
- Commons HTTP Client
- Freemarker
- Jackson 2.0
- JPA (Hibernate)
- Mongo DB (Optional)
- Infinispan (JGroups)
- JBoss Logging
- Apache Directory API
- Wildfly 10.1.x.Final



# Keycloak - Web SSO Infrastructure



# Keycloak in Action



# Keycloak - Integration Demo

- Java EE 7 Petclinic - Spring Pet Clinic Clone
- Java EE 7 Web Application based on JSF, JAX-RS, JPA, Wildfly 10.0
- <https://github.com/thomasdarimont/javaee7-petclinic>
- Integration via Keycloak-Servlet-Filter-Adapter

The screenshot shows the GitHub repository page for `phasenraum2010 / javaee7-petclinic`. At the top, there are buttons for `Watch` (4), `Unstar` (13), and `Fork` (23). Below this is a navigation bar with `Code`, `Issues` (6), `Pull requests` (0), `Wiki`, `Pulse`, and `Graphs`. The repository description is "Java EE 7 Petclinic" with a link to `http://javaee7petclinic-port80guru.rhcloud.com`. A summary bar shows `83 commits`, `1 branch`, `3 releases`, and `2 contributors`. Below this is a bar with `Branch: master`, a `New pull request` button, and buttons for `New file`, `Upload files`, `Find file`, `HTTPS`, a link to `https://github.com/phasenraum2010/javaee7-petclinic`, and a `Download ZIP` button. At the bottom, a commit message from `phasenraum2010` says "updated mvn site" with the latest commit `d212517` made 18 days ago.

# Keycloak - Before

## Java EE 7 Petclinic

|  |   |   |   |   |  |
|--|---|---|---|---|--|
|  Home |  Find Owners |  Veterinarians |  Specialties |  Pet Types |  Help |
|--|---|---|---|---|--|

## Welcome

### First Steps:

- add some Pet Types like dog,cat,mouse,...
- add some Specialties for Veterinarians like dentist, anesthetist, radiology,...
- add a Veterinarian
- add an Owner, add him am a Pet and his Pet a visit.

# Keycloak - After

## Java EE 7 Petclinic

User: theo

User ID: 1de04847-d573-4a97-b39e-a4de161ac5e2


User Full Name: Theo Tester


User Roles: [admin, user]


User Custom Attributes: {}


- [account](#)
- [logout](#)

 Home

 Find Owners

 Veterinarians

 Specialties

 Pet Types

 Help

## Welcome

### First Steps:

- add some Pet Types like dog,cat,mouse,...
- add some Specialties for Veterinarians like dentist, anesthetist, radiology,...
- add a Veterinarian
- add an Owner, add him am a Pet and his Pet a visit.



# Keycloak - Recap

- Multi-Tenancy with dedicated Realms
- Realm Management
  - Login, Email, Tokens, Web Security
  - Authentication, Flows, Policies (OTP, Password)
  - Themes
  - Roles
- Client Management
- User Management
  - User Account
  - User Group
  - Role Assignments
- Identity Brokering
- User Federation
- Monitoring
  - Sessions
  - Events

# Keycloak

## Integration and Customizing

# Keycloak - Integration Options

- Integration via Adapter
  - JBoss EAP & Wildfly Adapter
  - Tomcat / Jetty / Undertow Adapter
  - Java EE Servlet-Filter Adapter
  - Spring Security Adapter
  - Spring Boot Adapter
- Support for Non Java Apps
  - JavaScript “Adapter”, AngularJS, Aurelia
  - NodeJS
  - Mobile (Cordova / Native)
  - Reverse Proxy with Header Injection
- OpenID Connect
  - Apache mod\_oidc, (generic, for legacy Apps: PERL, CGI etc.)
  - Bundles for Portals / CMS (Drupal, Joomla, etc.)
  - Bundles for other Technology Stacks (PHP, Python, etc.)
- SAML
  - Apache mod\_auth\_mellon

- IdM User Experience
  - Themes for Login, Registration, User Account, (Admin Console)
  - HTML / Text based Emails
  - i18n for Template Strings, e.g. Product Names
- Authentication Mechanisms
  - E.g. Conditional OTP - only for certain HTTP Requests (based on HTTP Headers)
- Protocol Mappers
  - Allows to encode Client / Realm specific Information in Access Token
  - Can be based on Realm, Client, User-Attributes
- Client Templates
  - Supports common settings across multiple Clients
  - Protocol Mappers
- Custom Attributes
  - Generic Key-Value Pairs for User, Groups (and Clients)
  - Can be used for storing custom User information in Keycloak

# Keycloak - Extension Points

- Many defined Extension Points
  - Service Provider Interface (SPI)
  - based on Java Service Loader
- Themes
  - Look & Feel
  - Additional Pages / Fragments
  - Resources: Freemarker, HTML, CSS, JS, properties
- Highlights
  - Custom Authentication Mechanisms
  - Federation Provider (JDBC, REST, etc.)
  - Event Listener (Provisioning, JMS)
  - Credentials / Hashing Mechanisms
  - "Required Actions"
- Custom REST Endpoints (Health Checks!)
- Custom Persistent Entities

```
Spi (org.keycloak.provider)
  ImportSpi (org.keycloak.exportimport)
  FormAuthenticatorSpi (org.keycloak.authentication)
  IdentityProviderSpi (org.keycloak.broker.provider)
  UserSessionSpi (org.keycloak.models)
  IdentityProviderMapperSpi (org.keycloak.broker.provider)
  PasswordHashSpi (org.keycloak.hash)
  MongoConnectionSpi (org.keycloak.connections.mongo)
  EventStoreSpi (org.keycloak.events)
  SocialProviderSpi (org.keycloak.broker.social)
  EmailTemplateSpi (org.keycloak.email)
  HttpClientSpi (org.keycloak.connections.httpclient)
  MongoUpdaterSpi (org.keycloak.connections.mongo.updater)
  RequiredActionSpi (org.keycloak.authentication)
  MessagesSpi (org.keycloak.messages)
  TruststoreSpi (org.keycloak.truststore)
  BruteForceProtectorSpi (org.keycloak.services.managers)
  ClusterSpi (org.keycloak.cluster)
  UserSpi (org.keycloak.models)
  ClientDescriptionConverterSpi (org.keycloak.exportimport)
  TimerSpi (org.keycloak.timer)
  InfinispanConnectionSpi (org.keycloak.connections.infinispan)
  EmailSenderSpi (org.keycloak.email)
  LoginFormsSpi (org.keycloak.forms.login)
  WellKnownSpi (org.keycloak.wellknown)
  UserFederationMapperSpi (org.keycloak.mappers)
  UserFederationSpi (org.keycloak.models)
  ThemeSpi (org.keycloak.theme)
  AuthenticatorSpi (org.keycloak.authentication)
  JpaUpdaterSpi (org.keycloak.connections.jpa.updater)
  ProtocolMapperSpi (org.keycloak.protocol)
  FormActionSpi (org.keycloak.authentication)
  AccountSpi (org.keycloak.forms.account)
  LoginProtocolSpi (org.keycloak.protocol)
  ClientAuthenticatorSpi (org.keycloak.authentication)
  ClientRegistrationSpi (org.keycloak.services.clientregistration)
  MigrationSpi (org.keycloak.migration)
  UserSessionPersisterSpi (org.keycloak.models.session)
  JpaConnectionSpi (org.keycloak.connections.jpa)
  CacheRealmProviderSpi (org.keycloak.models.cache)
  ExportSpi (org.keycloak.exportimport)
  ClientInstallationSpi (org.keycloak.protocol)
  EventListenerSpi (org.keycloak.events)
  CacheUserProviderSpi (org.keycloak.models.cache)
  RealmSpi (org.keycloak.models)
```

# Keycloak - Custom Dashboard Extension

localhost:8081/auth/admin/master/console/#/realms/master/dashboard

Apps What Do WebLo Comcast/jrugger How To Add Buil Read Now Nicht verfügbar Neutron - OpenS Open vSwitch Introducing flan TodoMVC Java Neural Netv Get Started Now

## KEYCLOAK

Master

Configure

Manage

Realm Settings

Clients

Client Templates

Roles

Identity Providers

User Federation

Authentication

Dashboard

Groups

Users

Sessions

Events

Import

### Dashboard

10 Total Users

7 Active Users

181 Logins  
4 1

3 Registrations  
1

### Logins along the year

April May June July August September October November December January February March

### Latest Logins

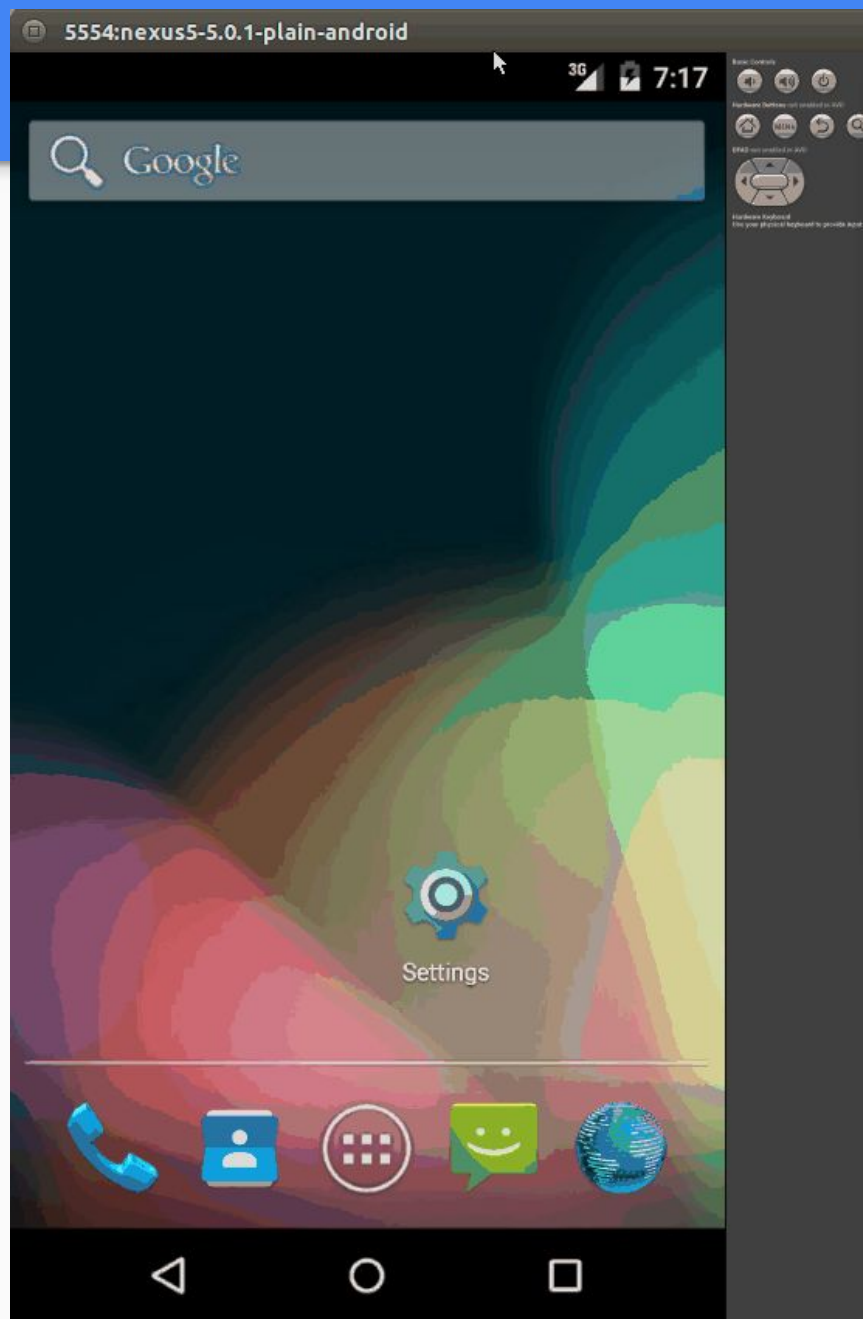
| Username | login at   | last visit |
|----------|------------|------------|
| user777  | 6:43:00 PM | never      |
| admin    | 6:42:08 PM | 6:41:57 PM |
| admin    | 6:41:57 PM | 6:41:43 PM |
| admin    | 6:41:43 PM | 6:35:33 PM |
| admin    | 6:35:33 PM | 6:35:31 PM |

### New Registrations

| Username | Registered at |
|----------|---------------|
| user5    | 11:29:39 PM   |
| user777  | 6:43:00 PM    |
| user6    | 9:54:52 PM    |

Please vote :) <https://issues.jboss.org/browse/KEYCLOAK-1840>

# Keycloak Android



- 42 PRs to Keycloak master
- Some examples (Android, Drupal) and PoCs (Dashboard)
- Revised Spring Boot Integration (Jetty, Undertow)
- Talks at Java User Groups / Developer Meetups



- Easy to get started
- Works quite well in practice
- Very active project
- Useful documentation <http://www.keycloak.org/documentation.html>
- Responsive Community
- Blog / User- and Developer-Mailing list
- Many examples on github
- Official Docker Container for Quick Start
  - `docker run -e KEYCLOAK_USER=admin -e KEYCLOAK_PASSWORD=admin -p 8080:8080 jboss/keycloak`
  - Browse to `http://localhost:8080/auth`
- Wildfly Swarm Module!
- 2.3.0.CR1 passes OpenID Connect Certification :)

## Keycloak - What's not so cool?

- Hardly any comments in Code :-)
- Very slow tests (Mostly end-to-end tests with Selenium)
- Keycloak for political reasons (Red Hat) only with Wildfly
  - though it can be wrapped in a Spring Boot app ;-)
  - See: <https://github.com/thomasdarimont/spring-boot-keycloak-server-example>
- Sometimes hard to debug
- No clear guidelines how to package custom extensions

## Keycloak - Lessons Learned

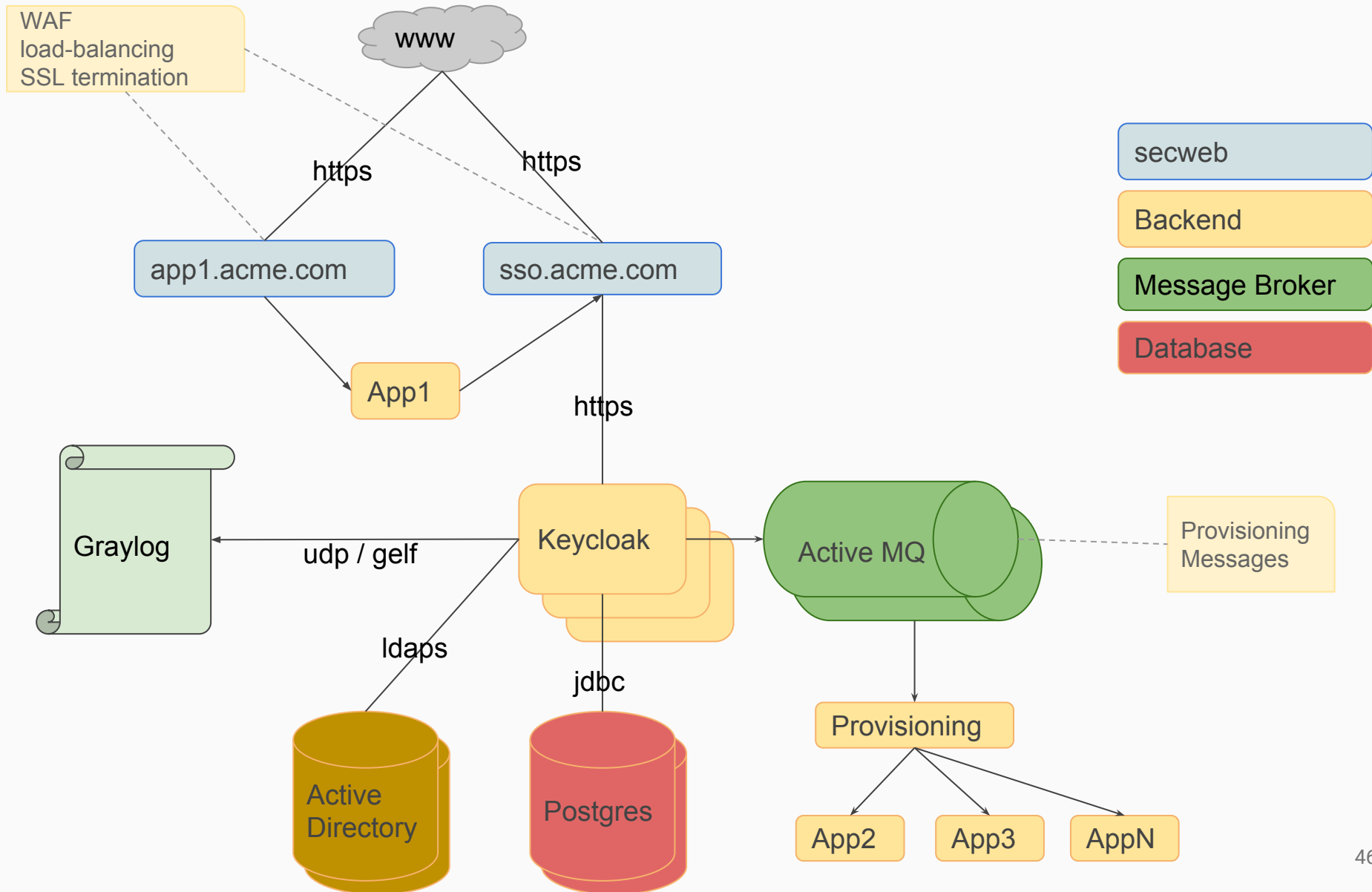
- First contact 1.4.0.Final (Sep 2015)
- New version approx. every 6 Weeks (quite stable so far)
- Good Mix of Feature Completeness & Usability
- Very flexible and powerful tool
- Based on proven standards
- Wide range of extension points
- One can live with Wildfly...
- You need Wildfly know-how!
- You need to be able to go deep (Keycloak src, tcpdump, wireshark)
- Clustering with Docker is tricky - but doable
- Production setup can get very complex
  - Proxy (X-Forwarded-\* Header!!!)
  - HTTPS / TLS
  - Clustering
  - Load-Balancer
  - Web Application Firewall (WAF)
  - HA Postgres
  - Logging
  - Monitoring

# Identity Management

Example production setup

- **Apache**
  - Web Application Firewall (secweb, mod\_security)
  - Load-Balancer (mod\_proxy\_balancer)
- **Virtual Machines (VMWare)**
  - Keycloak / Postgresql
- **Docker**
  - Container for Keycloak
- **Keycloak (Wildfly)**
  - Clustered Infinispan
- **Message Broker (ActiveMQ)**
  - Publishing of Provisioning Events
- **Monitoring (Graylog, Nagios)**
  - Log Monitoring
  - System-Health Checks & Anomaly Detection
- **Database (HA Postgres)**
  - Keycloak Metadaten
  - User Accounts (Profile, Roles, Credentials)
  - Clients, Groups, Roles, Events

# Simplified IdM Setup



[Keycloak Website](#)

[Keycloak Docs](#)

[Keycloak REST API](#)

[Keycloak User Guide](#)

[Keycloak Blog](#)

[Keycloak User Mailinglist](#)

[Keycloak Developer Mailinglist](#)

[JSON Web Tokens](#)

[Awesome Keycloak](#)