

# Securing ORDS

APEX World 2018 Rotterdam, 2018-03-22



QUALOGY



## Securing access to REST Data Services

## About me

### Menno Hoogendijk

- ✓ Fulltime APEX developer
- ✓ Working with Oracle since 2008
- ✓ Tries to be a fullstack developer

### Speaking at:

- ✓ APEX World 2018
- ✓ APEX Alpe Adria 2018
- ✓ APEX Connect 2018
- ✓ Kscope18



@mennooo



mennooo



www.menn.ooo



... colleagues work in the  
Netherlands



... colleagues work in the  
Caribbean

# Today's menu

Our use case

What is REST & ORDS?

Creating the REST API

Securing the REST API

Using the REST API

# Our use case

## We offer an online Sales App for Third Parties

The screenshot displays a web application interface for a 'Sample Database Application'. The top navigation bar is blue and contains a hamburger menu icon, the title 'Sample Database Application', and links for 'Mobile', 'Help', and a user profile 'admin'. A dark blue sidebar on the left lists navigation options: Home, Customers (8), Products (10), Orders (14), Reports, and Administration. The main content area features a search bar with the placeholder text 'Search customers, orders & product'. Below the search bar, the title 'Sample Database Application' is followed by the subtitle 'Track and Manage Customers, Orders and Products'. The dashboard is titled 'Dashboard' and contains four key performance indicator (KPI) cards: 'Monthly Sales' at \$2,215, 'Monthly Orders' at 6, 'Total Products' at 10, and 'Total Customers' at 8.

| Metric          | Value   |
|-----------------|---------|
| Monthly Sales   | \$2,215 |
| Monthly Orders  | 6       |
| Total Products  | 10      |
| Total Customers | 8       |

## And one of our clients wants more..



“I want to expand my business”

“I want to open an online fashion store!”

“Can I manage my sales through my own online App?”

---

And we say: “Sure, we will provide a REST API”

# Requirements

Our REST API needs to provide the following

**Access to global resources:**

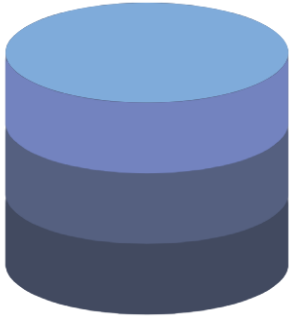
- Get product information
- Create new user accounts

**Access to user/ customer resources:**

- Get user account details
- Place an order
- Get order history



## What do we need?



Database



ORDS



HTTP Server

# What is REST & ORDS

The background features a solid red color with several large, overlapping, curved shapes in shades of orange and dark red. These shapes create a sense of depth and movement, framing the central text.

# REST – Representational State Transfer

REST is a set of rules to:

- Provide access to resources via URIs
- Use CRUD operations on these resources via HTTP methods

| CRUD operation | HTTP method |
|----------------|-------------|
| CREATE         | POST        |
| READ           | GET         |
| UPDATE         | PUT         |
| DELETE         | DELETE      |

## REST – Example

Getting a list of products from our Sales App

```
GET http://localhost:8080/ords/demo/api/products
```

```
{  
  "items": [{  
    "product_id": 6,  
    "product_name": "Ladies Shoes"  
  },  
  {  
    "product_id": 7,  
    "product_name": "Belt"  
  }  
]  
}
```

{ JSON }  
JavaScript Object Notation

# ORDS – Oracle REST Data Services

Enables us to develop REST interfaces for relational data in the Oracle Database



And run APEX..

# Creating the REST API (Demo)

## Getting a list of products from our Sales App

```
GET http://localhost:8080/ords/demo/api/products
```

# Securing the REST API

The background features several overlapping, curved, semi-transparent shapes in shades of orange and red, creating a dynamic, layered effect. The text is centered horizontally and vertically within the frame.



# How to secure a REST resource with ORDS

Follow these two steps



Client application

Products, Customers

Customer

Accountdetails

# What kind of users can access the REST API

Users known to the HTTP server (most common):

- Local users
  - File based entries in HTTP Server configuration
  - Useful during development
- Database users stored in our tables
  - Accessed by HTTP server through JDBC
  - Passwords should be stored with SHA256 hash
- LDAP users
- OAuth clients (using client credentials flow)

## Which HTTP Server can work with external user data store

| HTTP Server             | Local users | JDBC users | Downside                 |
|-------------------------|-------------|------------|--------------------------|
| ORDS Standalone (Jetty) | Y           | N          | Only local users         |
| Glassfish               | Y           | Y          | Will be desupported      |
| Tomcat                  | N           | N          | Current bug in ORDS 17.4 |
| Weblogic                | Y           | Y          | Paid license             |

We will use Glassfish for the demo because it still works but will get desupported.

# Choosing an authentication scheme

# You have to choose an authentication scheme

The two options are:

- Basic Authentication
  - Client has username & password credentials
- Bearer Authentication
  - Client has an access token

# Basic authentication

## Can be used in the following scenario:

- Internal REST API within your organization

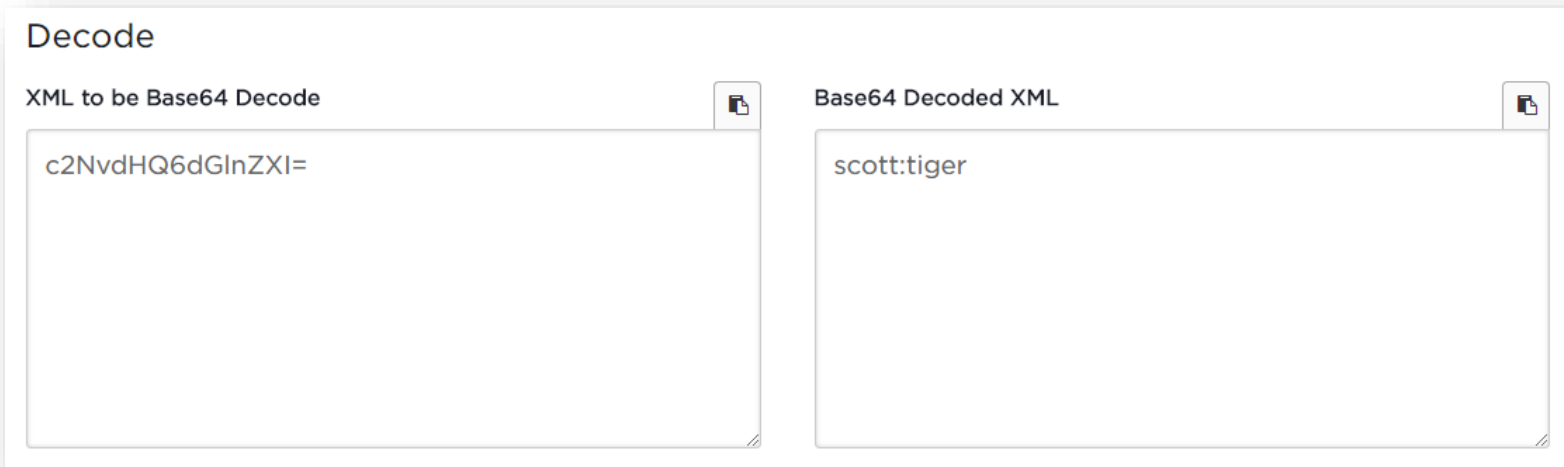
## Why avoid external usage:

- The credentials are send in the header of every request
- Somebody might intercept your network request
- If you do want to use it externally, make sure the requests are send via HTTPS

# How a hacker could get your credentials

## Request header

```
GET http://localhost:8080/ords/demo/api/products  
Authorization: Basic c2NvdHQ6dGlnZXI=
```



The screenshot shows a web-based Base64 decoder interface. It consists of two main text input fields. The left field is titled 'XML to be Base64 Decode' and contains the text 'c2NvdHQ6dGlnZXI='. The right field is titled 'Base64 Decoded XML' and contains the text 'scott:tiger'. Both fields have a small document icon in the top right corner. The interface is clean and minimalist, with a white background and light gray borders.

# Bearer authentication

Is part of OAuth2 specification

**Safer choice because:**

- Requires HTTPS
- Does not send credentials in request header
- Access token is short-lived
  - Expires in one hour by default in ORDS



## Step 1: Create a OAuth client in ORDS

The client app always makes REST requests using an OAuth client

Via a database procedure:

```
OAUTH.CREATE_CLIENT (  
  p_name          VARCHAR2 IN,  
  p_grant_type    VARCHAR2 IN,  
  p_owner         VARCHAR2 IN DEFAULT NULL,  
  p_description   VARCHAR2 IN DEFAULT NULL,  
  p_allowed_origins VARCHAR2 IN DEFAULT NULL,  
  p_redirect_uri  VARCHAR2 IN DEFAULT NULL,  
  p_support_email VARCHAR2 IN DEFAULT NULL,  
  p_support_uri   VARCHAR2 IN DEFAULT NULL,  
  p_privilege_names VARCHAR2 IN)
```

## Step 2: Assign role to client

The client app must have the appropriate role to access resources

Via a database procedure:

```
OAUTH.GRANT_CLIENT_ROLE (  
    p_client_name VARCHAR2 IN,  
    p_role_name   VARCHAR2 IN);
```

## Main goal of OAuth2: obtain an access token for a user

There are three different flows to obtain a token

| Oauth flow         | Users need to grant access? | Refresh token included? |
|--------------------|-----------------------------|-------------------------|
| Client credentials | N                           | Y                       |
| Authorization code | Y                           | Y                       |
| Implicit           | Y                           | N                       |

# Demo: The fashion store app

## Some important notes

- OAuth is not the same as Social Authentication
- OAuth clients are created for applications, not for end users
- End users must be able to login on HTTP server (except client credentials flow for OAuth)
- Use client credentials flow or authorization code flow for server side applications
- Use implicit flow for client side applications

# The demo and other resources are on GitHub

<https://github.com/mennooo/securing-ords>

## Securing Oracle REST Data Services

How to secure a REST API build with Oracle REST Data Services (ORDS)

### Content

- Prerequisites
- Creating the REST API
- Configure user authentication on Glassfish
- Configure user authentication on Tomcat (does not work for current ORDS release 17.4.1)
- Configure user authentication on ORDS Standalone (only for local ORDS users)
- Configure user authentication on Weblogic (requires a paid licence)
- Implementing basic authentication
- Implementing Oauth2 authorization



Thank you