Oracle WebLogic Server Application Security

Implementing the Superstition in JDeveloper

Duncan Mills
"The Answer Man"

Peter Koletzke
Technical Director & Principal Instructor

ORACLE quovera

Believe It or Not

Security is mostly a superstition.
It does not exist in nature,
nor do the children of men
as a whole experience it.
Avoiding danger is no safer
in the long run than outright exposure.
Life is either a daring adventure
or nothing.

—Helen Keller (1880-1968)

Agenda

- Overview
- Setting up ADF Security
- Securing resources

Slides will be on the OOW website.

Mon
5:00 - Oracle ADF On-Ramp: What You Need to Know to Use the ADF Technology Stack (Peter)

Thurs
10:30 - Real-World Performance Tuning for Oracle ADF Applications (Duncan)
12:00 - Achieving the Perfect Layout with Oracle ADF Faces Rich Client (Peter)

Application Security Objectives

- Ultimate security may just be superstition, however, data must be protected
  - Need to make breaking in as difficult as possible
- Web apps are more accessible to hackers
- Protections needed for
  - Application access
  - Application functions (no SQL injection, cross-site scripting)
  - Data access
  - Data visibility
  - Tracking user activity

Assumes the server and file systems are protected.

Two Primary Operations

- Authentication
  - Validate that the user is who she/he claims to be
    - Normally done with passwords
    - With extra equipment, could be something else
      - Retinal scan, thumbprint, biometric scanners? DNA?
- Authorization
  - Allow authenticated user access to specific resources
  - Usually done with security roles
    - Like database roles
    - Application components (pages, functions) and data are made available to named roles
    - Users are enrolled in roles
      - User has access to whatever the role is granted

How to Implement the Superstition

- Use recognized, prebuilt, proven, supported security technologies
- Java Authentication and Authorization Services (JAAS)
  - Java API library in the Java SE Development Kit (JDK or J2SDK))
  - Accessible through Oracle Platform Security Services (OPSS) – a service of WebLogic Server
- Oracle ADF Security
  - Built on top of OPSS
  - Uses standard ADF declarative techniques
  - Once you turn it on, you need to define access for all pages in the application

The Security Policy

- A definition of privileges in ADF Security
  - Contained in a *Security Policy Store*
- Create one or more in an application
- Principals
  - One or more roles (groups of users) who are granted access
- Resources
  - Bounded task flow – including all flows under it
  - Web pages that use ADF bindings
  - Entity objects and entity object attributes
- Permissions
  - Privileges such as View, Customize, Grant, Personalize

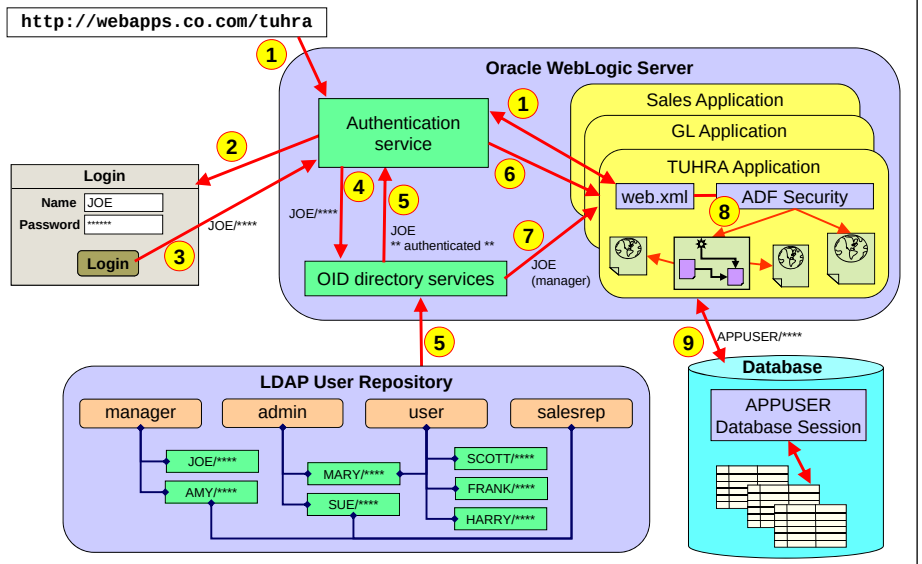
The User Repository

- The storehouse of user and enterprise role information
  - A.k.a., *credentials store* or *identity store*
- OPSS can tap into multiple LDAP repositories
  - LDAP (Lightweight Directory Access Protocol)
    - A communications protocol
    - Internal WebLogic LDAP
    - Oracle Internet Directory (OID)
      - Used for Single Sign-On (SSO)
    - Can read other LDAP providers
    - E.g., Microsoft Active Directory

What's a Role?

- Users have a "role" within the enterprise
  - AKA "Enterprise Roles"
  - "Warehouse Clerk", "HR Manager", "Chief Bottlewasher"
  - A single user will usually have multiple roles
  - Totally dependent on the business organization
  - May change over time for a single user
- Applications also have the concept of "role"
  - **Not** the same thing
  - Application roles define functional areas within the application's "responsibilities"
    - "approver", "page manager", "user" etc...

Application Security Flow



Application Security Flow

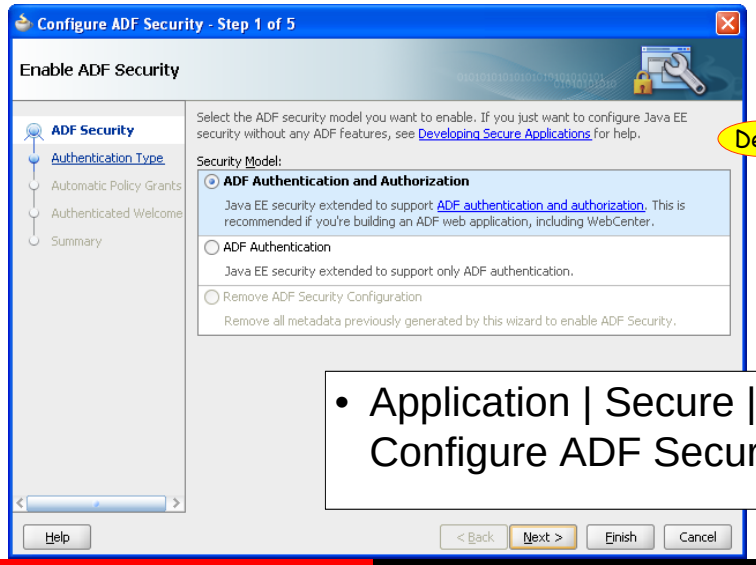
1. User sends HTTP request including a context root indicating a particular application. The request determines that ADF security is active in the application.
2. The authentication service determines the method (XML or LDAP) and presents a login page.
3. The user enters an ID and password and submits the login page.
4. The authentication service requests OID to verify the user and password.
5. OID verifies the password from the LDAP source and indicates pass or fail to the authentication service.
6. The authentication service accesses the application and places the user name into the HTTP session state.
7. The application can request the username or group (role, in this example, "manager") to which the user belongs.
8. web.xml activates ADF Security for authorization to specific resources like pages and task flows.
9. The application connects to the database using the application database user account (APPUSER) written into a configuration file.

Agenda

- Overview
- Setting up ADF Security
- Securing resources

Demo

Enable ADF Security

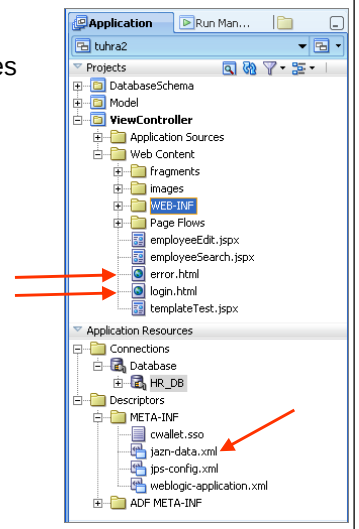


Demo

- Application | Secure | Configure ADF Security

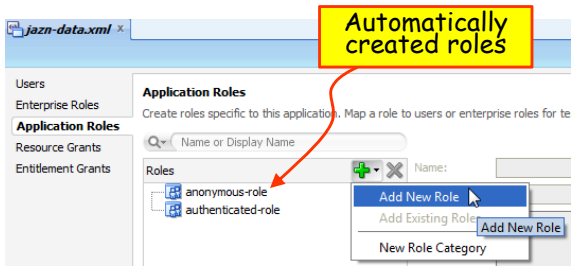
Configure ADF Security Wizard

- Form-based authentication
  - Can create custom login and error pages
- No automatic grants
- Redirect upon successful authentication
- Creates
  - login.html
  - error.html
  - jazn-data.xml
- Updates
  - web.xml (auth type and page names)
  - weblogic.xml
    - Look at it for security-role-assignment
  - Maps principals (users) to roles



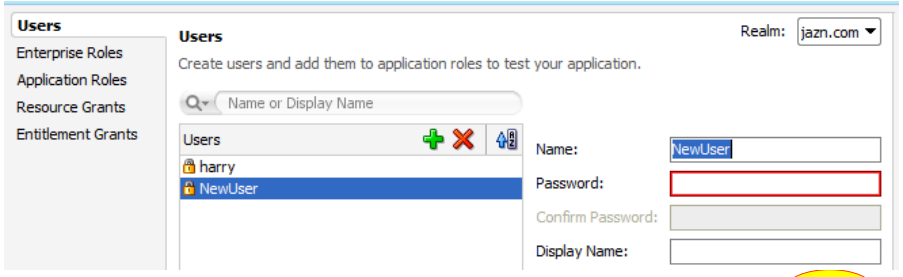
Create Application Roles

- Application menu, **Secure | Application Roles**
  - Opens editor for jazn-data.xml
    - In the META-INF directory
- Define application roles
  - Application Roles tab
  - Add New Role (+)
  - Name, display name



Define Application Users

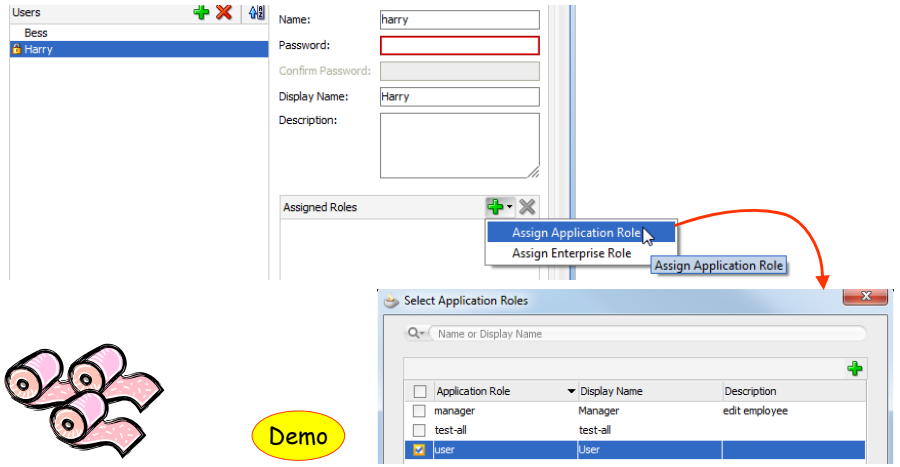
- Users tab
  - Define name
  - Password (internal to XML file, not for enterprise security) – 8 chars, include number



Demo

Users in Roles

- Assigned Roles area of Users tab



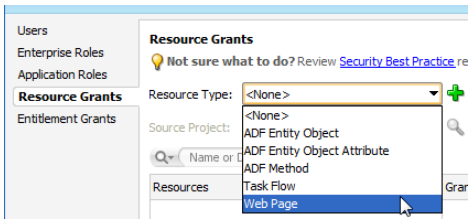
Demo

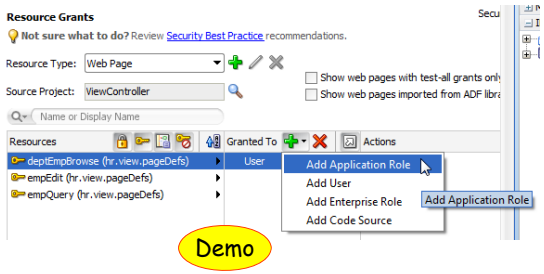
Agenda

- Overview
- Setting up ADF Security
- Securing resources

Set Up Grants to Resources

- Resource Grants tab
- Resource Type: Web Page
- Select a page
  - Add (+)
  - Roles (preferred) or users (not as flexible)
- Pages will require authentication





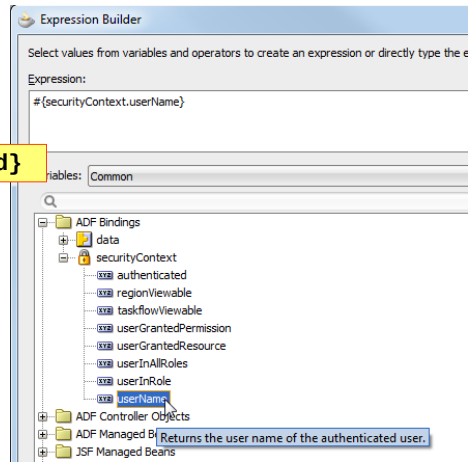
Demo

Add User Name Display

- Output Text (Formatted), Expression Builder for *Value* property
  - Display the name
- Set *Rendered*

```
#{securityContext.authenticated}
```

(display if authenticated)



Add Login Link

- Link (Go), *Text* property

```
#{securityContext.authenticated ? "Logout" : "Login"}
```

  - Logout if not authenticated, else Login
  - *Destination* property
    - Call the ADF authentication servlet
    - If already authenticated, pass logout = true to log out the user and return to menu.jspx
    - If not authenticated, pass success URL of main.jspx (which requires authentication and will display the login page

```
#{securityContext.authenticated ?
"/adfAuthentication?logout=true&end_url=/faces/menu.jspx" :
"/adfAuthentication?success_url=/faces/main.jspx"}
```


Hiding Items

- Suppose Salary is sensitive data and only viewable by managers
  - *Set the rendered* property on Salary field

```
#{securityContext.userInRole['manager']}
```

- Hiding a link or button based on the availability of a resource
  - e.g, edit menu item only viewable to those allowed to edit

```
#{securityContext.regionViewable['view.pageDefs.editPageDef']}
```

```
#{securityContext.taskFlowViewable[
                  '/WEB-INF/userEdit.xml#userEdit']}
```

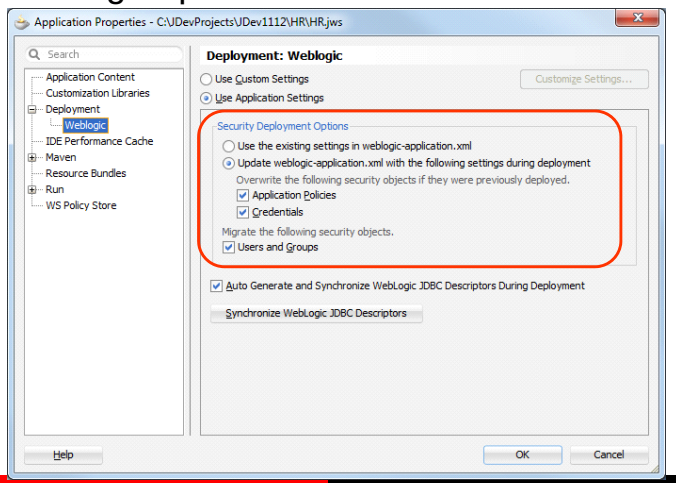

You can Ask the Same Questions in Code..

- Create the relevant permission class instance
- Pass it to the security context for evaluation

```
String tfID = "/WEB-INF/userEdit.xml#userEdit"
TaskFlowPermission permission = new
TaskFlowPermission(
     tfID,TaskFlowPermission.VIEW_ACTION);
SecurityContext sctx  = ADFContext.getCurrent().
     getSecurityContext();
 if sctx.hasPermission(permission) {
    …
 }
```


Application Deployment Settings

- Deploying the application automatically deploys users and groups



Deploying Security

- Deploying the application automatically deploys users and groups

```
Deployment - Log
[04:07:38 PM] ----  Deployment started.  ----
[04:07:38 PM] Target platform is  (Weblogic 10.3).
[04:07:39 PM] Retrieving existing application information
[04:07:39 PM] Running dependency analysis...
[04:07:39 PM] Building...
[04:07:43 PM] Deploying 2 profiles...
[04:07:43 PM] Wrote Web Application Module to C:\JDevProjects\JDev112\HR\ViewController\deploy\HR_ViewController_webapp.war
[04:07:43 PM] Wrote Enterprise Application Module to C:\JDevProjects\JDev112\HR\deploy\HR_Project1_HR.ear
[04:07:45 PM] Deploying Application...
[04:07:49 PM] [Deployer:149192]Operation 'deploy' on application 'HR_Project1_HR' is in progress on 'AdminServer'
[04:07:56 PM] [Deployer:149194]Operation 'deploy' on application 'HR_Project1_HR' has succeeded on 'AdminServer'
[04:07:56 PM] Application Deployed Successfully.
[04:07:56 PM] The following URL context root(s) were defined and can be used as a starting point to test your application:
[04:07:56 PM] http://[2001:0:4137:9e76:142b:1826:bc43:6ff7]:7001/HR-ViewController-context-root
[04:07:56 PM] Uploading jazn-data users.
[04:07:56 PM] Creating user "harry".
[04:07:56 PM] Creating user "bess".
[04:07:56 PM] Elapsed time for deployment:  18 seconds
[04:07:56 PM] ----  Deployment finished.  ----
```

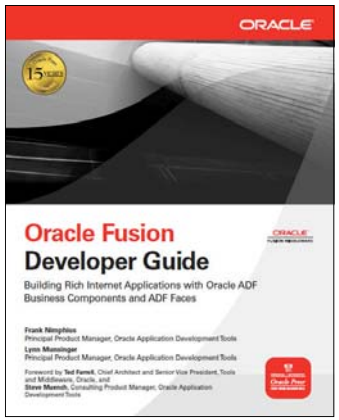
- For an LDAP server, configure the “WLS authentication provider”

Deployment Effects

- Permissions (defined in jazn-data) are deployed to and merged with the permission store in WLS
  - Permission jazn-data appears in the EAR that you create with OJDeploy
- For the embedded WLS running in development mode
  - JDeveloper will create the users in WLS that you've defined in the IDE.
  - JDeveloper, not OJDeploy, does this by calling the WLS MBeans to create the new users on the fly
- Users defined in jazn-data do not migrate into external LDAP repositories

Other Resources

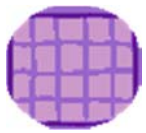
- *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework 11g Release 1 (11.1.2)*
  - PDF at OTN, online in JDev
  - Chapter 29
- *Oracle Fusion Middleware Security Guide 11g Release 1*
  - PDF at OTN
- *Oracle Fusion Developer Guide*
  - Nimphius and Munsinger, McGraw-Hill Professional, Oracle Press (2010)
  - Chapter 21
- OTN Tutorial



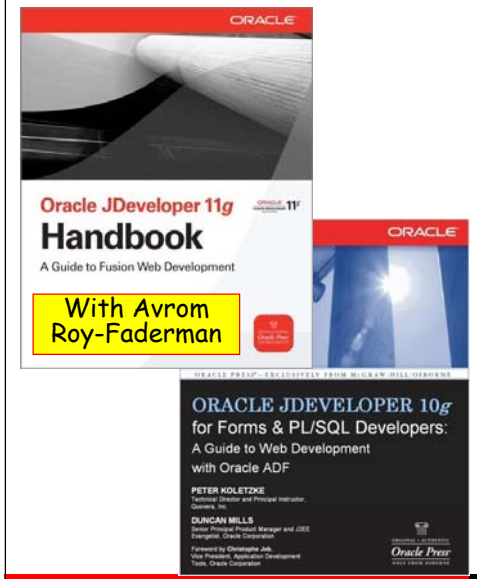
http://download.oracle.com/docs/cd/E18941_01/tutorials/jdtut_11r2_29/jdtut_11r2_29.html

Summary

- You need to design application security
- OPSS offers easy access to standard JAAS security features
- ADF Security provides declarative definition of security policies for task flows and pages
- Binding expressions on the page can hide or disable items
- Design and test for all security breach scenarios



The Books



The Coauthors

- Duncan Mills
  - Widely published on OTN, ODTUG, JDJ etc.
  - groundside.com/blog/DuncanMills.php
  - www.oracle.com
- Peter Koletzke
  - Six other Oracle Press books about Oracle tools
  - www.quovera.com
- Book examples
  - www.tuhra.com