Implementing the Superstition with ADF Security

Peter Koletzke

Technical Director &
Principal Instructor

ORACLE ACE Director

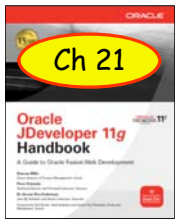
Co-author in absentia:
Duncan Mills,
Senior Architect, ADF Product Development

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



ADF only: not applicable to MAF

Slides and white paper **(Editor's Choice Winner: 2012; Top 5: 2013, 2015)** are on the UTOUG and Quovera websites

The white paper contains a hands-on practice.

Upcoming
3:30 – Designing ADF for Mobile

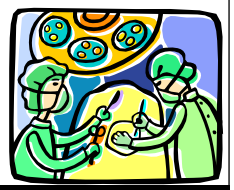
Application Security Objectives

- 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
  - Can you spell N.S.A.?
- 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

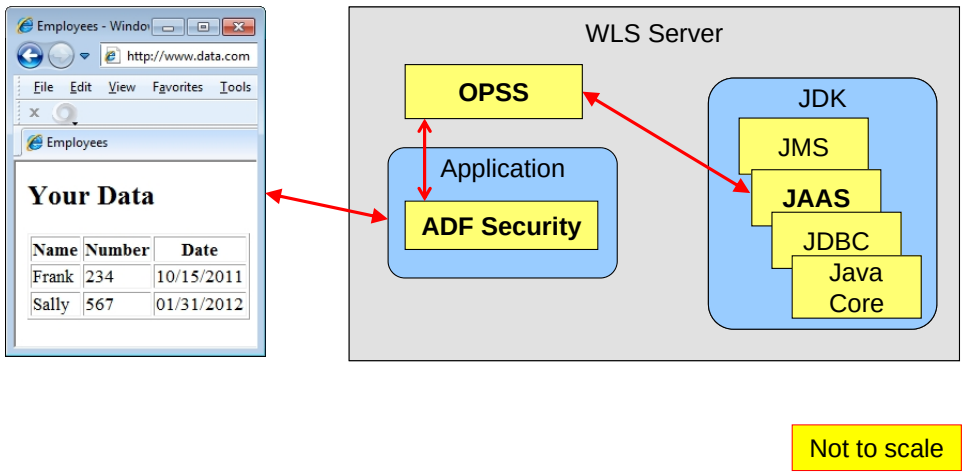
- Authentication
  - Validate that the user is who she/he claims to be
    - Normally done with passwords
    - With extra equipment, could be something else
      - Retinal scan, thumbprint, biometric scanners? DNA?
- Authorization
  - Allow authenticated user access to specific resources
  - Usually 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 to use OPSS
  - Uses standard ADF declarative techniques
  - Once you turn it on, you need to define access for all pages in the application

Summarizing That

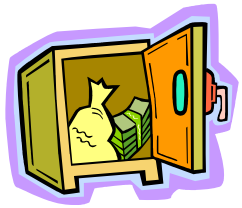


The Security Policy

- 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
- Can use other user repositories such as Oracle database accounts
  - Tie in using a *Login module* (JAAS code)

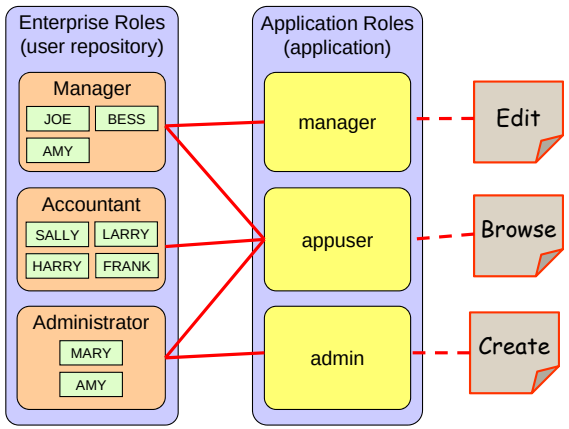


What's a Role?

- Users have a "role" within the enterprise
  - A.k.a, "*enterprise role*"
  - "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.

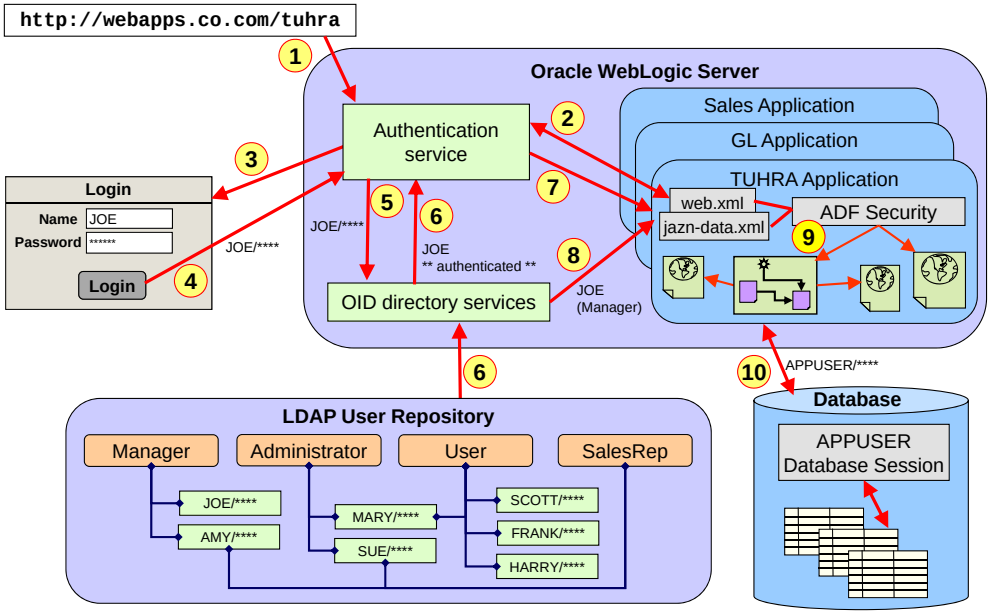
Sample Role Mapping

- Users are members of enterprise roles
- Application roles are granted application functions
- Users access application functions through their membership in a mapped role



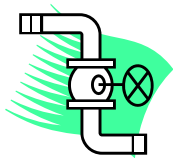
- Quick quiz: Privs for Joe, Sally, Amy, George?

Application Security Process



Application Security Flow

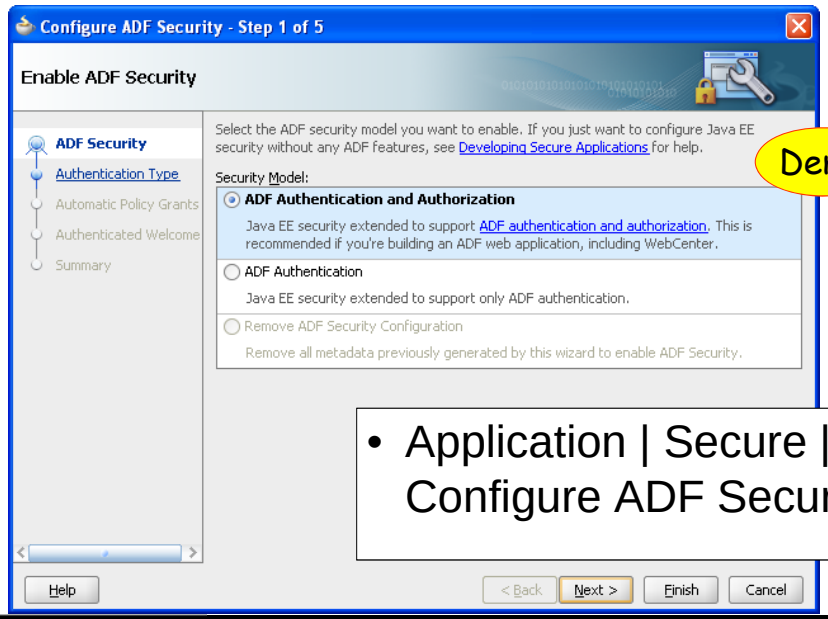
1. User sends HTTP request including a context root indicating a particular application.
2. The authentication service determines the type of authentication needed from web.xml
3. The authentication service presents the login page.
4. The user enters an ID and password and submits the login page.
5. The authentication service requests OID to verify the user and password.
6. OID verifies the password from the LDAP source and indicates pass or fail to the authentication service. Failure returns 401 error.
7. If authentication passes, the service passes control to the application and places the user name into the HTTP session.
8. The application can request the username or group (role, in this example, “Manager”) to which the user belongs.
9. web.xml and jazn-data.xml defines ADF Security for authorization to specific resources like pages.
10. The application connects to the database using the application database user account (APPUSER) written into a data source on the application server.



Agenda

- Overview
- Setting up ADF Security
- Securing resources

Enable ADF Security

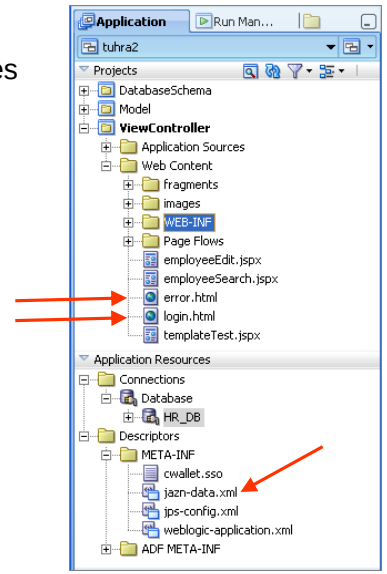


Demo

- Application | Secure | Configure ADF Security

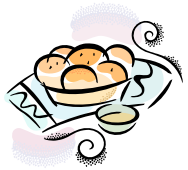
Configure ADF Security Wizard

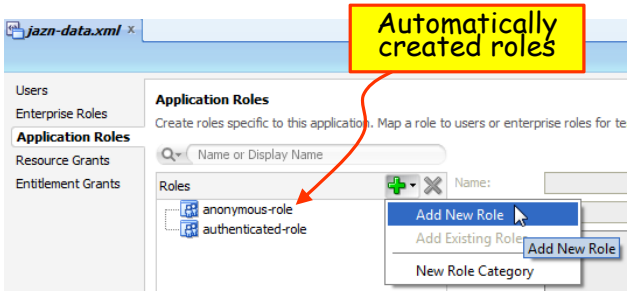
- Form-based authentication
  - Can create custom login and error pages
- No automatic grants
- Redirect upon successful authentication
- This creates
  - login.html
  - error.html
  - jazn-data.xml
- This updates
  - web.xml (auth type and page names)
  - weblogic.xml
    - Look at it for security-role-assignment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



Create Application Roles

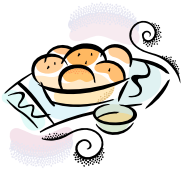
- Application menu, **Secure | Application Roles**
  - Opens editor for jazn-data.xml
    - In the META-INF directory
- Define application roles
  - Application Roles tab
  - Add New Role (+)
  - Name, display name

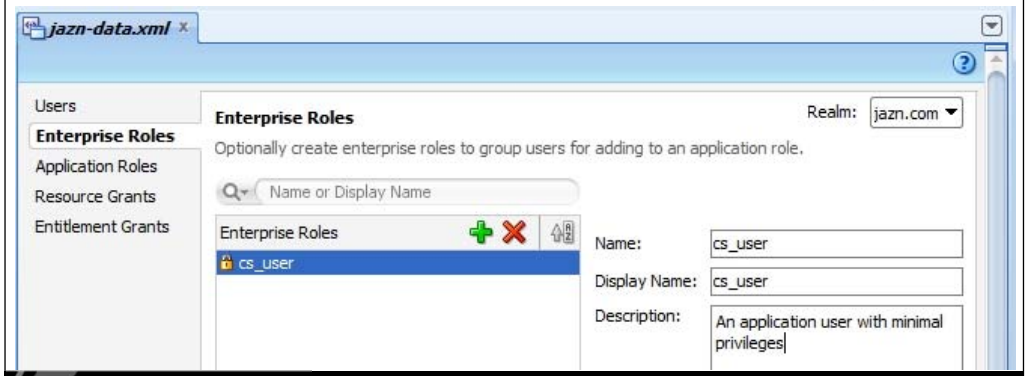




Create Test Enterprise Roles

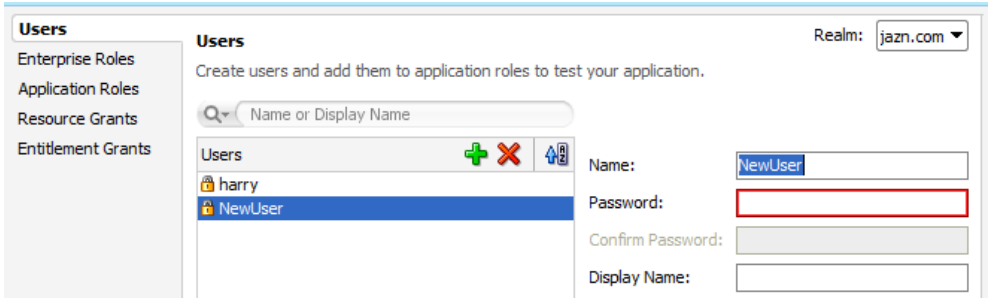
- In jazn-data.xml, Enterprise Roles tab
  - Click +, fill out properties





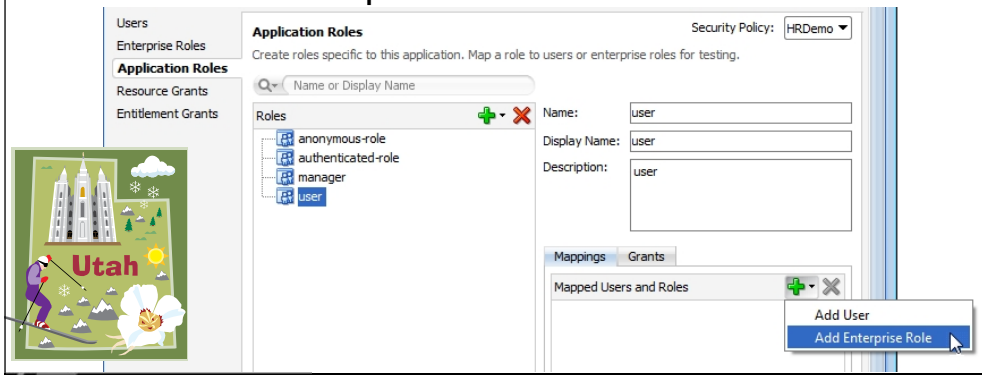
Define Test Users

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



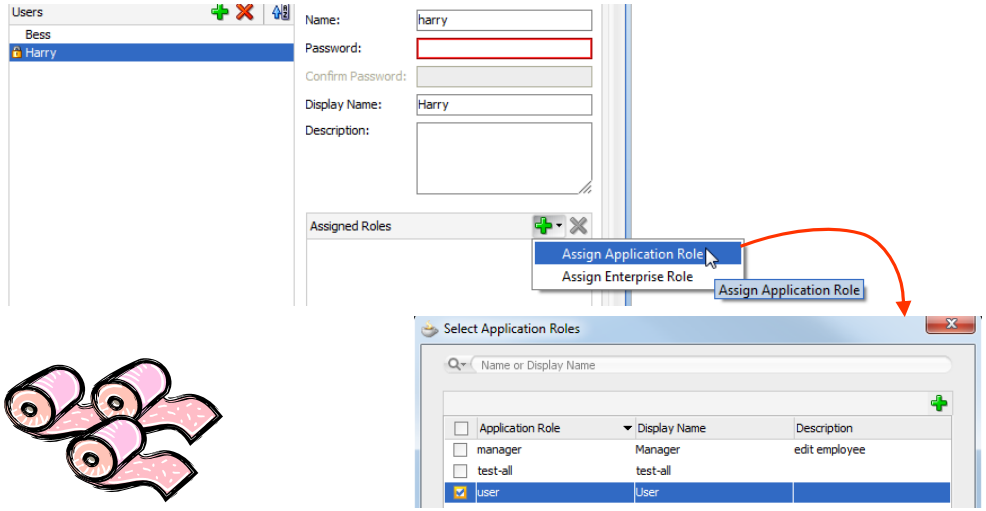
Map App Roles to Enterprise Roles

- In jazn-data.xml, Application Roles tab
- Mappings tab at bottom
  - Click +, Add Enterprise Role
  - Select enterprise roles to include



Users in Roles

- Assigned Roles area of Users tab

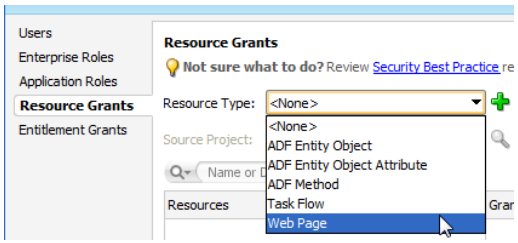


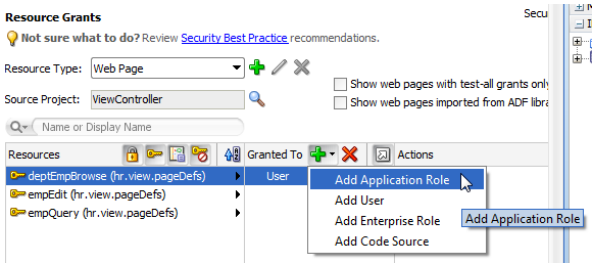
Agenda

- Overview
- Setting up ADF Security
- Securing resources

Set Up Grants to Resources

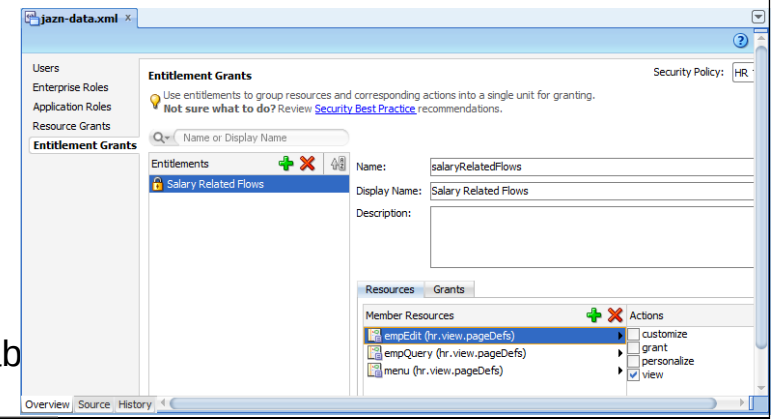
- Resource Grants tab
- Resource Type: Web Page
- Select a page
  - Add (+)
  - Roles (preferred) or users (not as flexible)
- Pages will require authentication





Entitlement Grants

- Set of permissions
  - Groups resources like task flows
- Apply the set not individual permissions
  - Faster to apply
  - Easier changes
  - Grant the set on the Grants tab

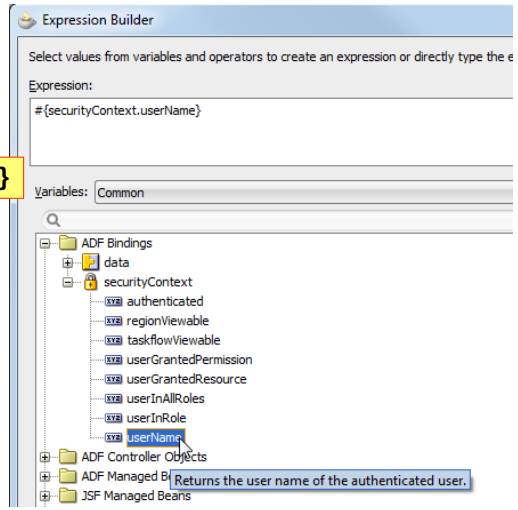


Add User Name Display

- 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']}
```

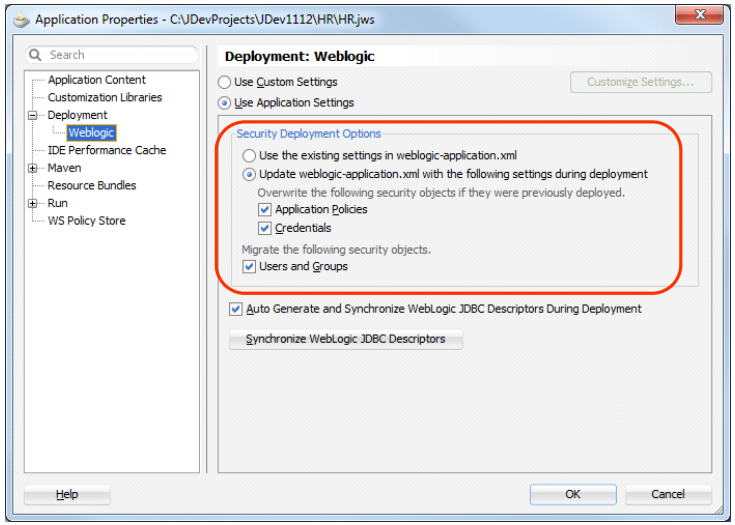

You can Ask the Same Questions in Code...

- 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) {
    …
 }
```

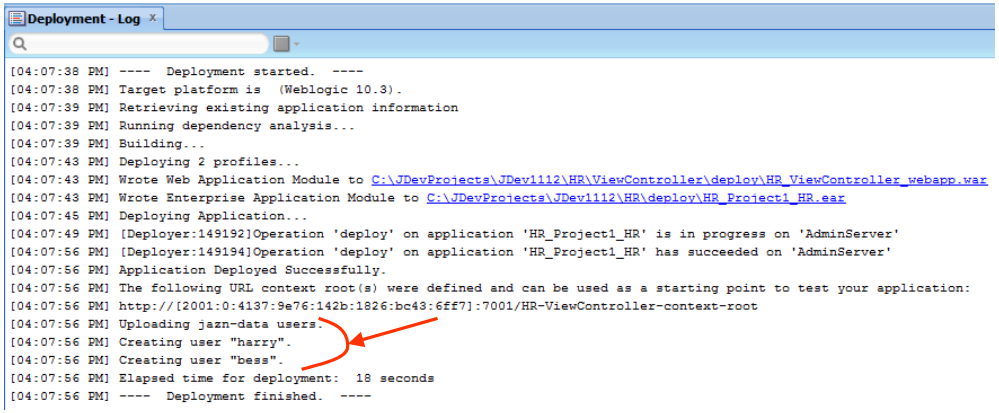
Application Deployment Settings

- Deploying the application automatically deploys users and groups



Deploying Security

- Deploying the application automatically deploys users and groups



- 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 users on the fly
- Users defined in jazn-data do not migrate into external LDAP repositories

Other Resources

- Hands-on practice in the white paper
- Oracle Fusion Middleware 12c 12.1.3n – doc set
  - “Secure the Environment”
  - docs.oracle.com/middleware/1213/index.html
  - *Oracle Fusion Middleware Security Guide 11g Release 1*
  - *Oracle Fusion Middleware* Securing Applications with Oracle Platform *Security* Services, *12c*
- *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework 11g Release 1 (11.1.2)*
  - Chapter 29
- ADF Insider Essentials (google this)
- *Oracle Fusion Developer Guide, Ch 21*
- fusionsecurity.blogspot.com
- OTN Tutorial

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

Tres Cher

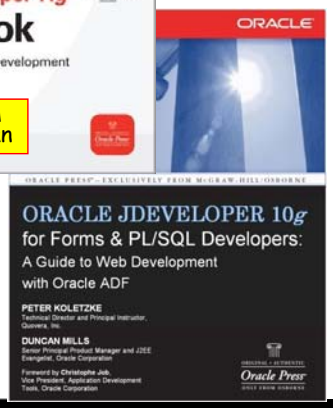
We will bankrupt ourselves in the vain search for absolute security.

—Dwight David Eisenhower, (1890-1969)

Summary

- You need to design and test 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
- Give the hands-on practice a spin

The Books



The Coauthors

- Peter Koletzke
  - Six other Oracle Press books about Oracle tools
  - www.quovera.com
- Duncan Mills
  - Widely published on OTN, ODTUG, JDJ etc.
  - blogs.oracle.com/groundside
- Book examples
  - tuhra2.java.net