

IBM® WebSphere® Application Server V7– LAB EXERCISE

# WebSphere Application Server security domains

What this exercise is about ..... 1

Lab requirements ..... 1

What you should be able to do ..... 2

Introduction ..... 2

Exercise instructions ..... 3

Part 1: Configure local operating system user registry ..... 4

Part 2: Create new security domain ..... 7

Part 3: Test new security domain ..... 10

What you did in this exercise ..... 12

## What this exercise is about

Security domains are a new element of WebSphere Application Server version 7 that allow administrators to define multiple security configurations for use in a single cell or application server. By default, all administrative and user applications in WebSphere Application Server use the same global security configuration. For example, a user registry defined within global security is used to authenticate users for every application in the cell. By default, this behavior is the same as it was in previous releases of WebSphere Application Server. Using security domains, you can create additional security configurations if you want to specify different security attributes for some or all of your user applications.

For example, you can define different settings (such as a different user registry) for user applications than for administrative applications. You can also define separate security configurations for user applications deployed to different servers and clusters. The application server in this exercise has administrative security enabled, and is configured to use the standard file based user registry in the federated repository. Using security domains, this exercise configures the user applications within the stand-alone application server to use a different user registry (local operating system) for application level security than the user registry defined for the global security settings (file based).

Although this exercise would be more interesting in a federated environment, the basic concepts of security domains can be easily demonstrated using a stand-alone application server. This is done by using the global security settings for administrative logins and configuring a new security domain for the user applications. The new security domain in the case includes a mapping of the application server to application security and a local operating system user registry. In a federated environment, for example, it would be possible to use the exact same concepts to enabling Java 2 security for some application servers and not others.

## Lab requirements

The list of system and software required for the student to complete the lab.

- A system that meets that requirements for running WebSphere Application Server Version 7.0, with approximately 500 MB of disk space for creating profiles
- The most current version of WebSphere Application Server V7.0
- An application server profiles with administrative security enabled, and with the administrative console and the default application deployed.

## What you should be able to do

At the end of this lab you should be able to:

- Create new administrative console users
- Map administrative console users to security roles
- Create and configure Administrative Authorization Groups
- Map Administrative Authorization Groups to both scopes and specific console users

---

## Introduction

WebSphere Application Server Version 7 introduces security domains, allowing administrators to define multiple security configurations for use in a single cell or application server.

This exercise goes through the process of configuring a new security domain using the administrative console. This security domain includes the definition for enabling application security and using the local operating system as the registry. It maps the security domain to the user applications within the application server. This has the effect of leaving the global security settings (including the file based user registry) active for the administrative console, but enabling the use of the local operating system registry for user application authentications.

This lab is divided into the following parts:

### **Part 1: Configure local operating system user registry**

This first part of the exercise creates a new local operating system user ID named “wslocalos”. This user exists only in the local operating system registry and does not exist in the file-based federated repository. Later in the lab, when the security domain is validated, this user is used to verify that the alternate user registry is used.

### **Part 2: Create new security domain**






This section creates the actual security domain called “ApplicationDomain” and scopes it to the application server. It then configures this security domain to enable application security and use the local operating system as the user registry.

### **Part 3: Test new security domain**

This part of the lab verifies that the new security domain is functioning as expected. This is done by verifying that although wsdemo can authenticate to the administrative console, it can not authenticate to the snoop servlet. Finally, this section verifies that wslocalos can authenticate to the snoop servlet but not to the administrative console.

## Exercise instructions

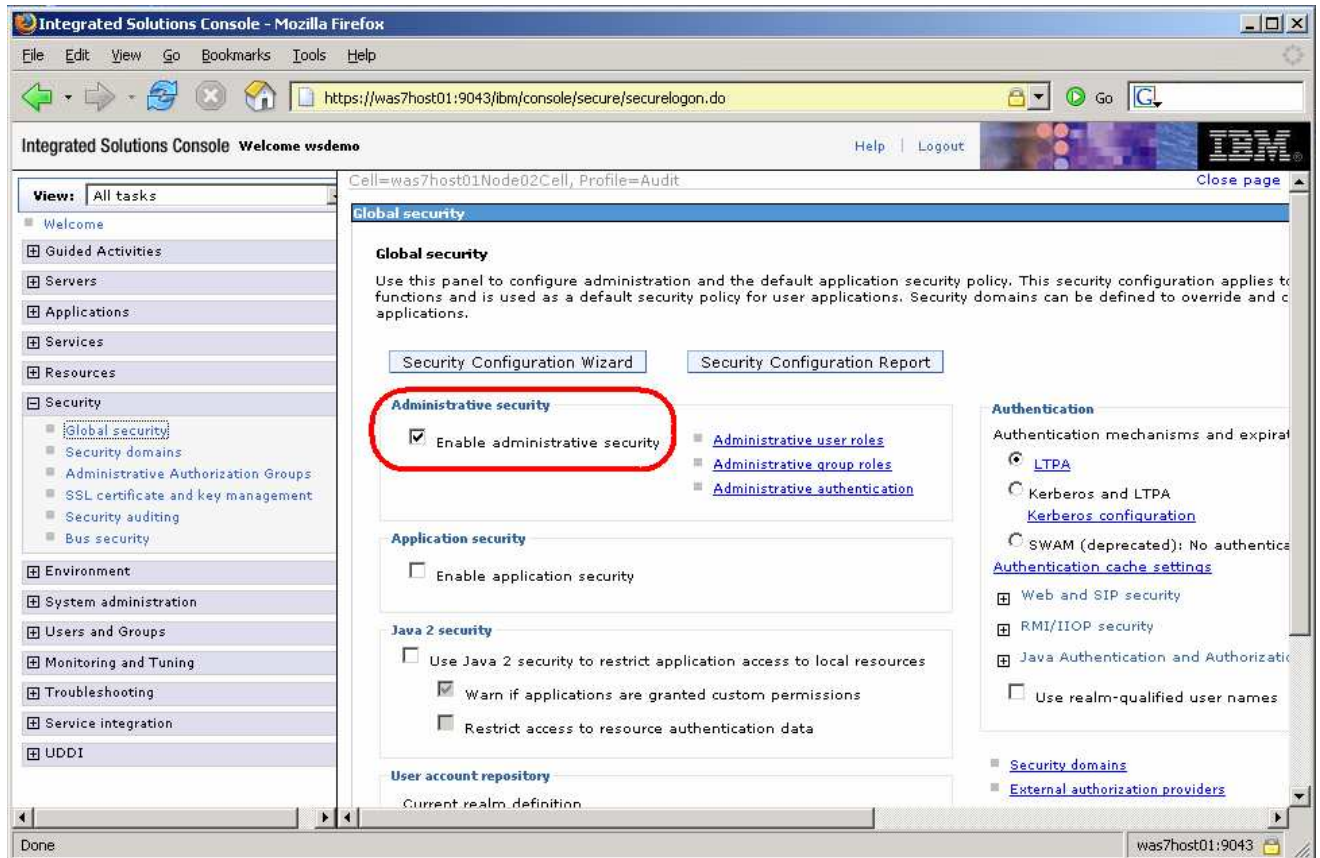
Instructions and subsequent documentation use symbolic references to directories which are listed as follows:

Reference Variable	 Location	  Location
<WAS_HOME>	C:\Program Files\IBM\WebSphere\AppServer	 /opt/WebSphere/AppServer  /usr/WebSphere/AppServer
<TEMP>	C:\temp	/tmp
<hostname>	Host name or host address for the machine where the profiles are being created	Host name or host address for the machine where the profiles are being created

## Part 1: Configure local operating system user registry

This exercise configures WebSphere Application Server to use the local operating system user registry as part of a security domain mapped to the user applications. As such, a local user will be needed to test access to the user applications. This part of the lab creates a user named **wslocalos** and defines it as the primary user for the local operating system registry.

- \_\_\_ 1. Start by ensuring that the application server is running.
- \_\_\_ 2. Open an administrative console and verify that administrative security is enabled.



- \_\_\_ a. If administrative security is not enabled, enable it and restart the server.
- \_\_\_ 3. Create a local operating system user ID.
  - \_\_\_ a. On **Windows**, open a command window and execute the following command:  

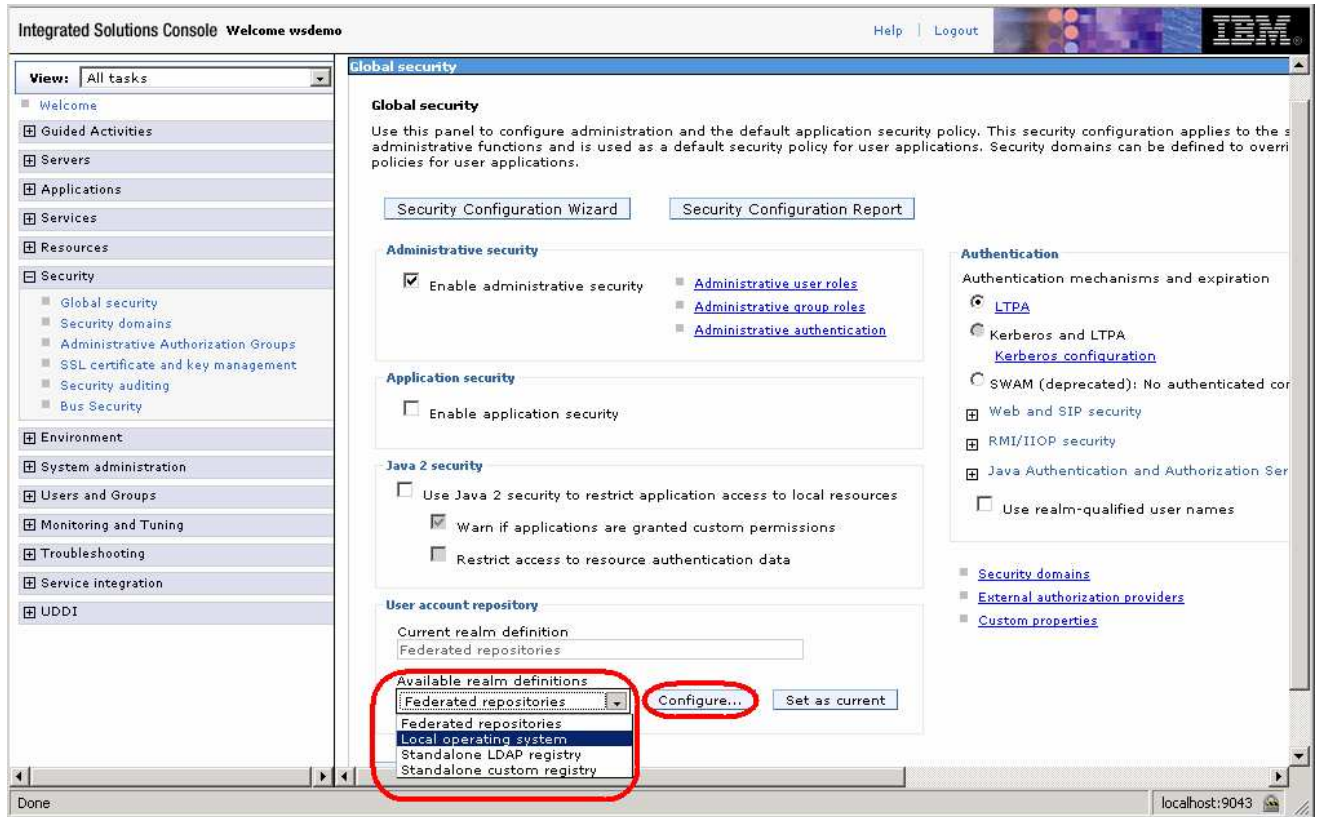
```
net user wslocalos wsdemopassword /add
```

---

**Note:** On **Linux** or **UNIX**, use the local operating system tools to create a user.

---

- \_\_\_ 4. Configure the local operating system user registry to use the new wslocalos user. This will be used for the application security once the security domain is defined in the next part of the exercise.
  - \_\_\_ a. In the administrative console, open the **Global security** page.
  - \_\_\_ b. Select **Local operating system** from the **Available realm definitions** pull down and click **Configure**.



\_\_\_ c. Enter **wslocalos** in the **Primary administrative user name** field and click **OK**.

**General Properties**

\* Primary administrative user name  
wslocalos

**Server user identity**

Automatically generated server identity  
 Server identity that is stored in the repository  
Server user ID or administrative user on a Version 6.0.x node  
\_\_\_\_\_  
Password  
\_\_\_\_\_

Custom properties

Select	Name	Value
<input type="checkbox"/>	_____	_____

New  
Delete

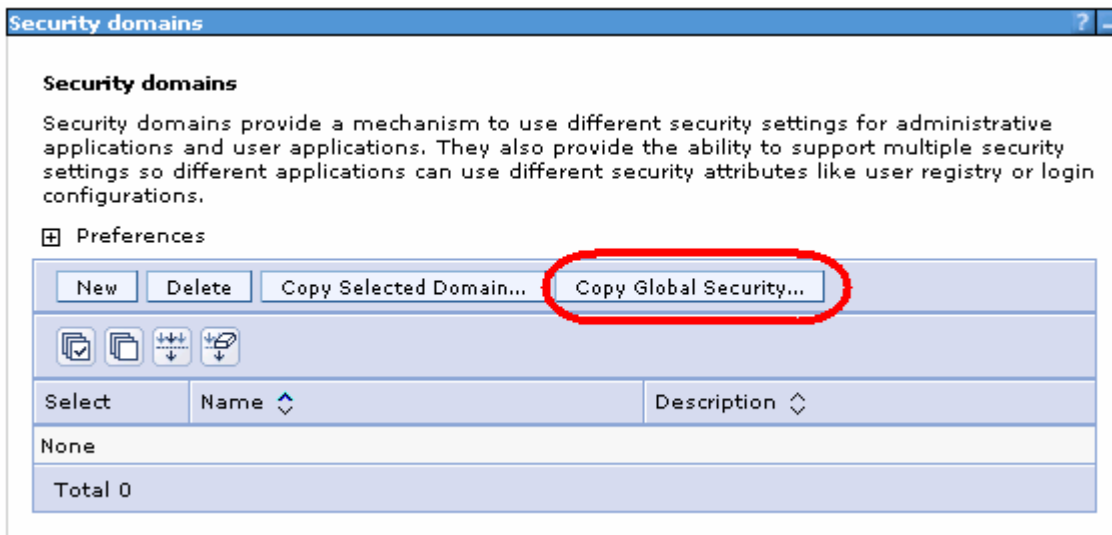
Apply **OK** Reset Cancel

\_\_\_ d. **Save** the changes.

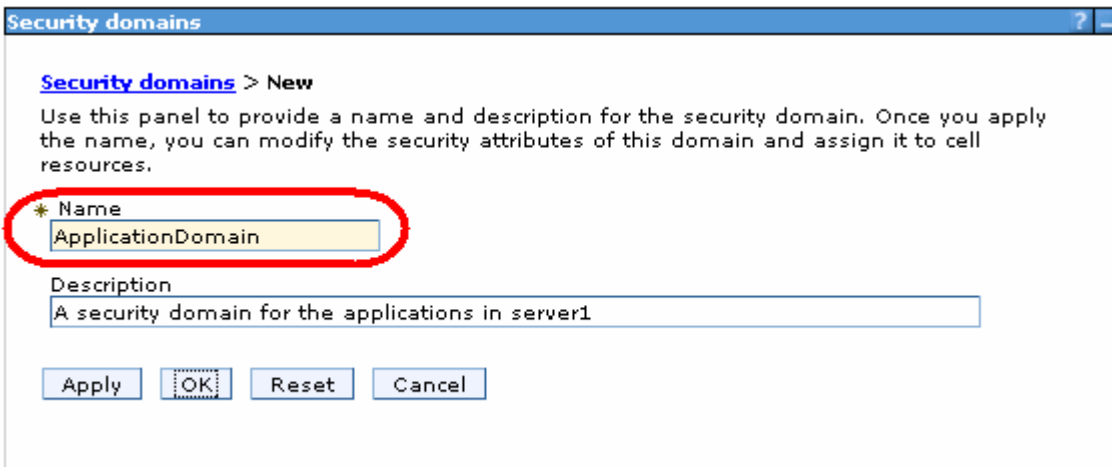
## Part 2: Create new security domain

The next step is to create the security domain, which defines server1 as having application security turned on as well as using the local operating system for the user registry. Note, that this security domain does not include the administrative functions like logging into the administrative console. The administrative security definitions still fall under the global security settings.

- \_\_\_ 1. Create a new security domain.
  - \_\_\_ a. In the console, under **Security**, click **Security domain**.
  - \_\_\_ b. Click **Copy Global Security** to create the new group which based on the Global Security settings.

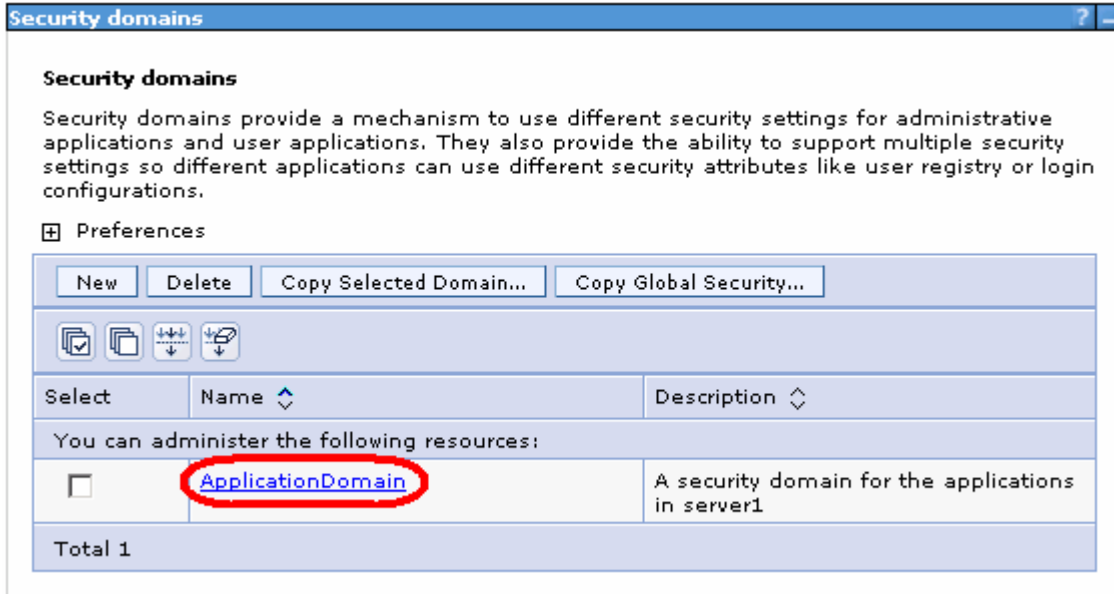


- \_\_\_ c. Enter **ApplicationDomain** for the **Name** and enter an appropriate **Description**.



- \_\_\_ d. Click **OK** and **Save** the changes.

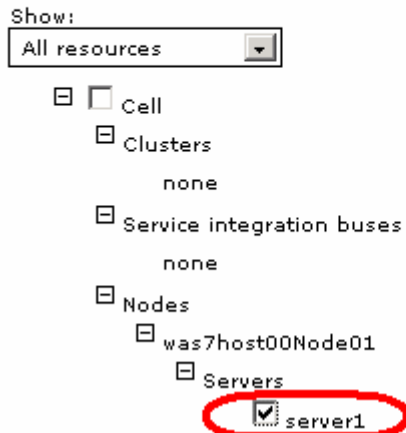
\_\_ e. Click **ApplicationDomain** within the list of security domains.



\_\_ f. Fully expand all of the assigned scopes and select **server1**.

Assigned Scopes

Assign the security domain to the entire cell or select the specific servers, clusters, and service integration buses to include in this security domain.




---

**Note:** There are numerous other possible scopes that would be much more interesting in a federated environment. It is possible to define separate security domains for different parts of a cell. For example, you can define only some resources to enforce Java 2 security while others have it disabled.

---



- \_\_\_ g. Under the Security attributes, expand **Application Security**. At this point you can choose to accept the settings define by the global security definitions, or you can customize the settings for this domain. In this case, ensure that **Customize for this domain** is selected and then check the box for **Enable application security**.

**Security Attributes**

**Application Security:** Customized

Use global security settings

Do not use realm-qualified user names

Do not enable application security

Do not use Java 2 security to restrict application access to local resources

**Customize for this domain**

**Enable application security**

- \_\_\_ h. Next, expand the **User Realm** portion. Ensure that **Customize for this domain** is selected and then select **Local operating system** from the pull down.

**User Realm:** Customized - defaultWIMFileBasedRealm

Use global security settings

Repository type: Federated repositories

**Customize for this domain**

Realm type

Local operating system

Configure...

- \_\_\_ i. Before clicking OK, verify that the **server1** scope is checked, **Enable application security** is also checked, and that **Local operating system** is selected. Then click **OK** and **Save** the changes.

- \_\_\_ 2. **Restart the application server** for the changes take effect.

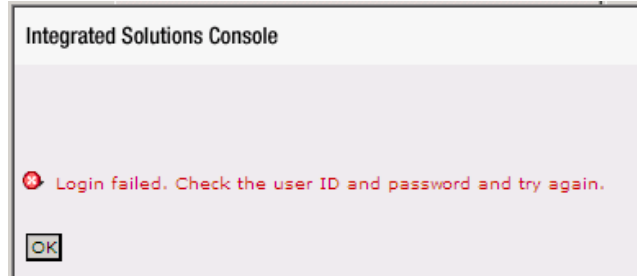
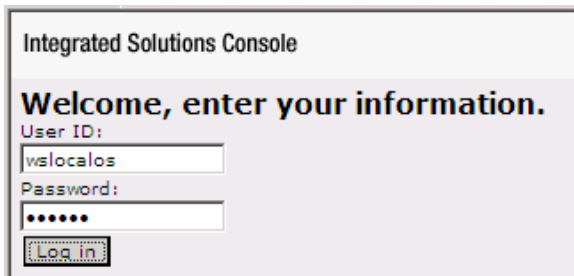
## Part 3: Test new security domain

With the new security domain configured, the `wsdemo` user still has the ability to log into the console. This is because `wsdemo` is defined in the file based federated user repository being used by the global security settings. Since the new security domain enabled application security for `server1`, accessing the `snoop` servlet requires the user to authenticate.

Using `wsdemo` (defined ONLY in the file based federated user repository) for authentication to `snoop` fails since it is not defined in the registry that is configured as part of the security domain (local operating system registry). Authenticating to `snoop` using `wslocalos` succeeds because that user exists in the local operating system registry. On the other hand, trying to authenticate to the console using `wslocalos` will fail since that user is not defined in the file based federated user repository (and it would also need to be further defined as a console user).

This section of the lab verifies that administrative logins use the global security user registry and that the application logins use local operating system registry as defined by the new security domain.

- \_\_\_ 1. Log into the administrative console as `wsdemo`.
  - \_\_\_ a. Once the application server has finished restarting, log in to the administrative console using the **`wsdemo`** user.
  - \_\_\_ b. Logout from the console and try logging in as **`wslocalos`**. This should fail since `wslocalos` does not exist in the federated repository. As a reminder, the password that you configured for the `wslocalos` ID is **`wsdemopassword`**.



\_\_\_ 2. Verify access to the snoop application.

\_\_\_ a. Open a new browser and enter the following URL:

<http://localhost:9080/snoop>

\_\_\_ b. In the authentication window, enter **wslocalos** in the **User name** and **wsdemopassword** for the **Password**.



## Snoop Servlet - Request/Client Information

**Requested URL:**

`http://localhost:9080/snoop`

**Servlet Name:**

`Snoop Servlet`

\_\_\_ c. Close the browser and open a new browser. Connect to the same URL and attempt login using the **wsdemo** user. This should fail since **wsdemo** does not exist in the local operating system.

## **What you did in this exercise**

In this lab you learned about the security domain feature in WebSphere Application Server Network Deployment V7. You created new local operating system users and configured the local operating system registry to use that user. You created the actual security domain and mapped it to the user applications. Finally, you verified that the access controls that were added did what was expected.