



WebSphere security



Unit objectives

After completing this unit, you should be able to:

- Explain basic security concepts
- Describe the WebSphere Application Security architecture
- Describe enhancements to certificate management
- Configure fine-grained administrative security
- Configure application security
- Describe SSL concepts and configuration
- Describe support for multiple security domains
- Describe auditing features and functions
- Describe support for Java Platform, Enterprise Edition 6 (Java EE 6) security annotations

Topics

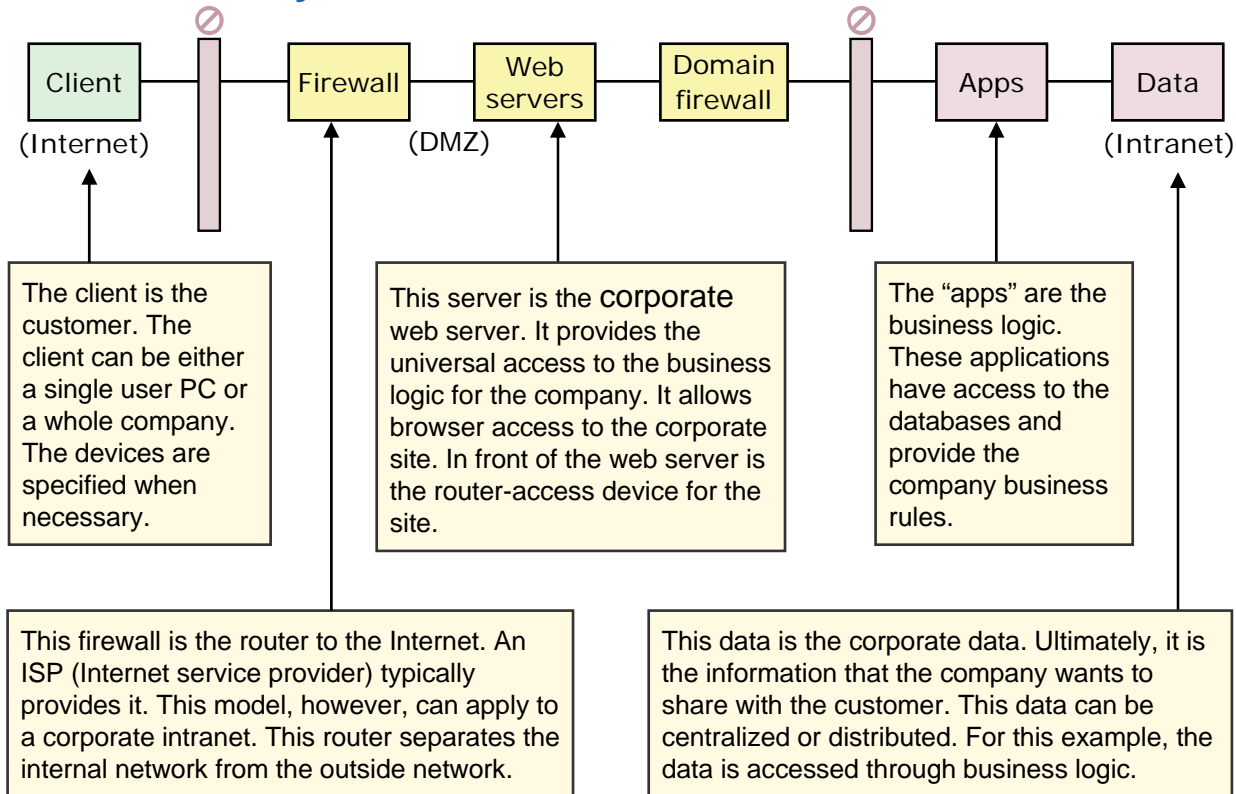
- WebSphere security basics
- WebSphere user registries
- Administrative security
- Application security
- Security domains
- Java 2 security
- SSL basics
- Certificates and certificate authorities
- SSL within a WebSphere cell
- Security auditing



WebSphere security basics



Basic security end-to-end model





WebSphere Application Server security overview

- Security can be applied at different levels

WebSphere
resources

*HTML
*JSPs, servlets, EJB

WebSphere security

Security APIs

File system security

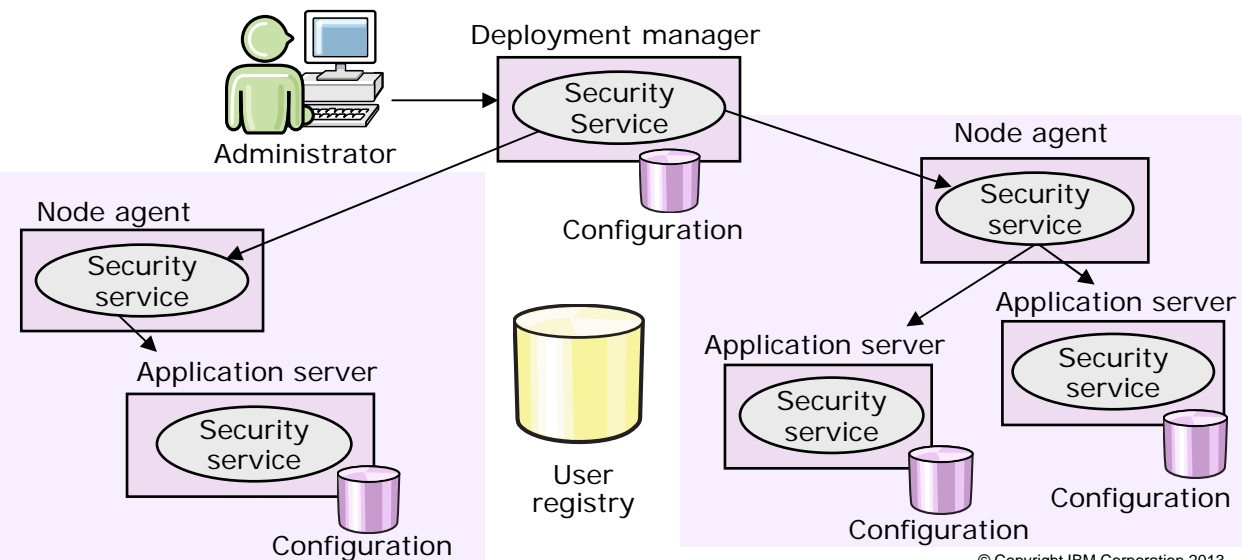
OS security

Network security

Physical security

WebSphere security service: Big picture

- Security service runs locally in each process (deployment manager, node agent, and application server)
 - Security workload not bottlenecked to a single process
 - Security service failure affects only a single process
- Separation of authentication mechanism and user registry



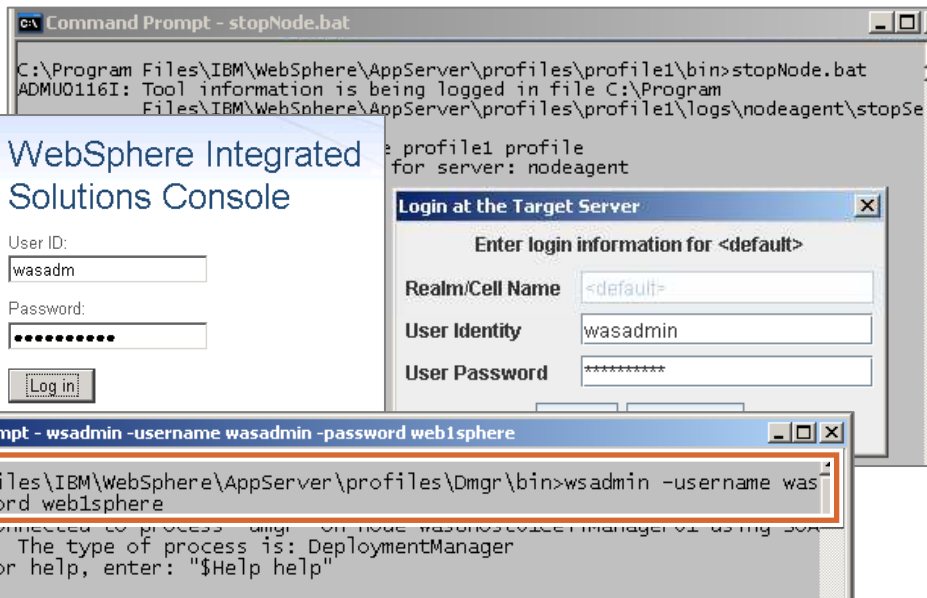
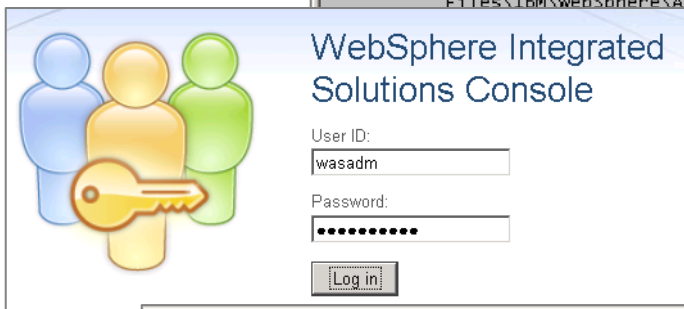
Types of security

- Administrative security
 - Protects things such as administrative console, wsadmin, scripts
- Application security
 - Protects access to the applications
- Java 2 security
 - Protects the local systems

A screenshot of the WebSphere security configuration interface. It shows three sections: 'Administrative security', 'Application security', and 'Java 2 security'. In the 'Administrative security' section, the checkbox 'Enable administrative security' is checked and highlighted with an orange box. To its right are three links: 'Administrative user roles', 'Administrative group roles', and 'Administrative authentication'. In the 'Application security' section, the checkbox 'Enable application security' is unchecked and highlighted with an orange box. In the 'Java 2 security' section, the checkbox 'Use Java 2 security to restrict application access to local resources' is unchecked and highlighted with an orange box. Below it are two more unchecked options: 'Warn if applications are granted custom permissions' and 'Restrict access to resource authentication data'.

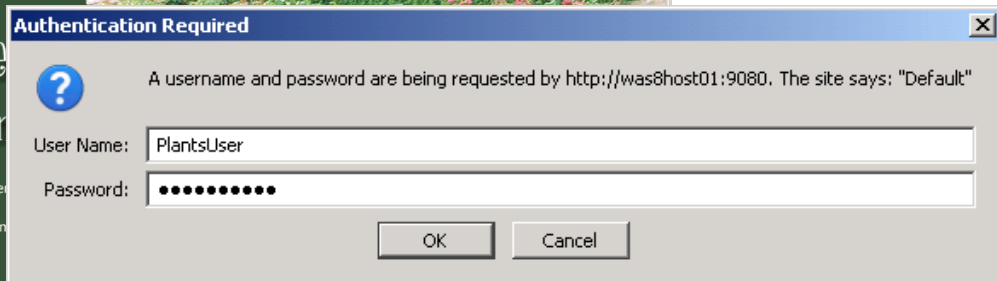
Administrative security

- Protects administrative console, scripts, wsadmin, and others
- Access can be restricted through:
 - Administrative roles
 - Fine-grained access



Application security

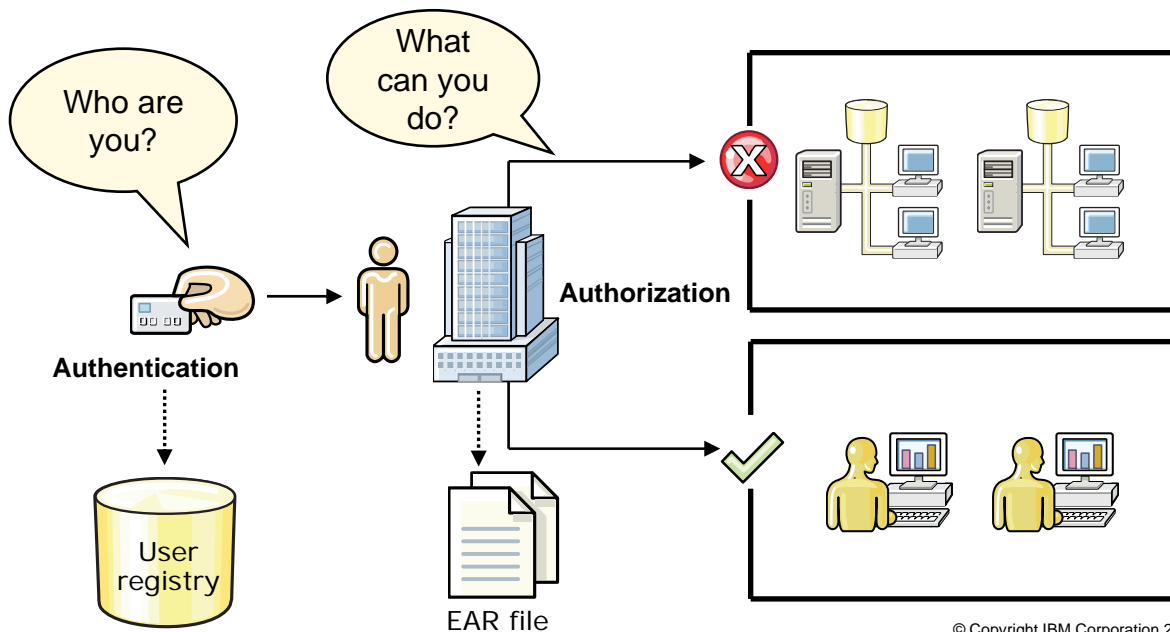
- Enables security for the applications in your environment
- Provides application isolation and requirements for authenticating application users
 - Security constraints protect servlets
 - Method permissions protect EJBs



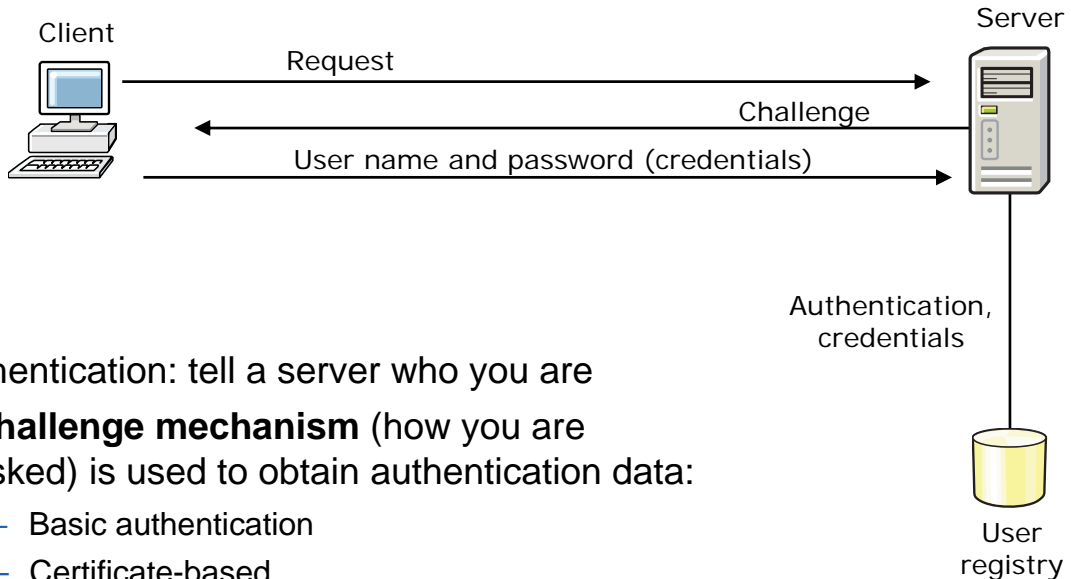


Authentication and authorization: What is the difference?

- There is a distinction between authentication and authorization
 - Authentication → Who are you?
 - Authorization → When authenticated, what are you allowed to do?



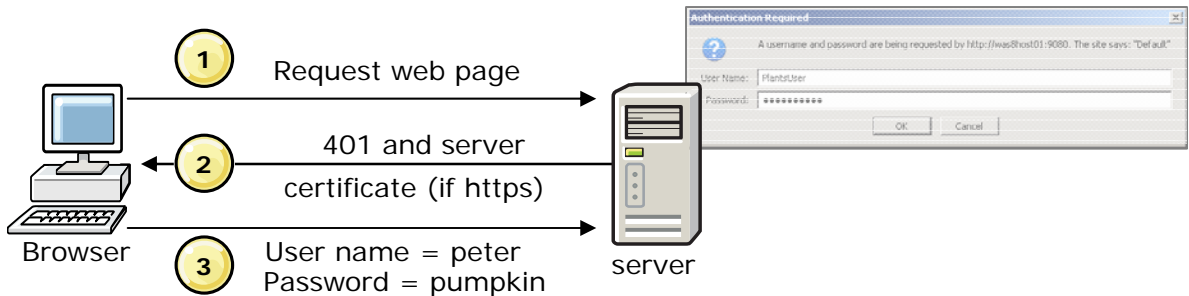
Challenge mechanism: Authentication basic steps



Authentication: tell a server who you are

- **Challenge mechanism** (how you are asked) is used to obtain authentication data:
 - Basic authentication
 - Certificate-based
 - Form-based authentication (preferred)
- Challenge mechanism is defined in the deployment descriptor
- Credentials are validated against user registry

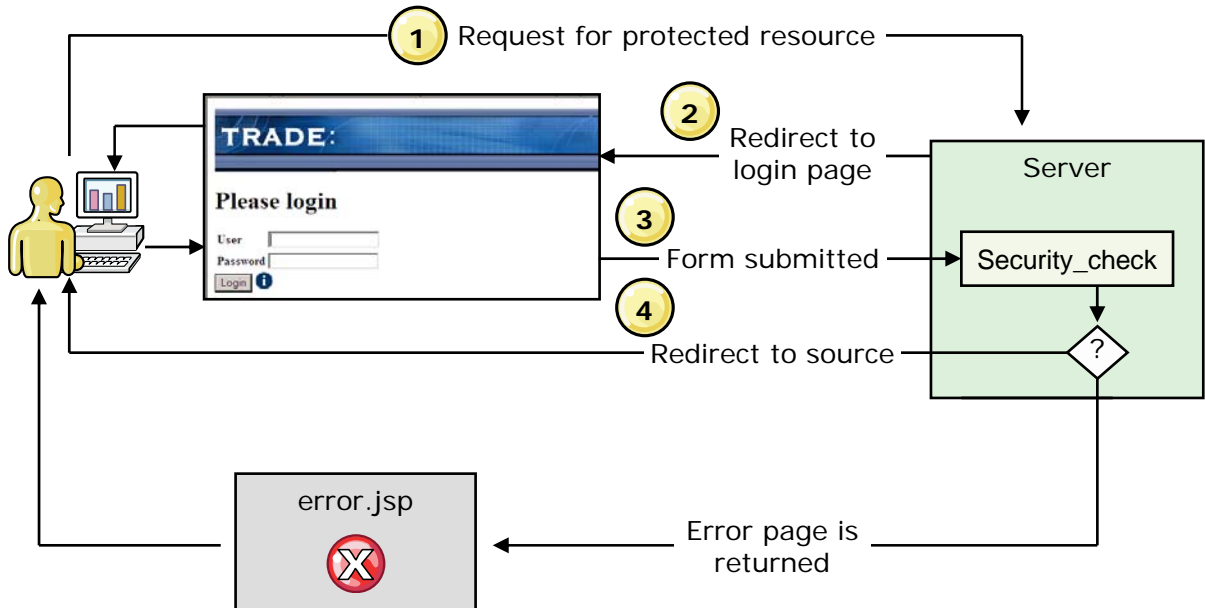
Challenge mechanism: Basic authentication



- Warning: password is not encrypted, merely encoded
 - Make sure that this channel is over HTTPS
- Warning: basic authentication token is sent across in the HTTP header
 - Danger: there is no way for the server to remove the token during a logout
 - If you walk away from a public browser, even if you log off, your credentials are still stored in the browser
 - Another person can walk up and be automatically logged in to your site
 - To remove the basic authentication token, close the browser or explicitly tell the browser to delete authentication credentials

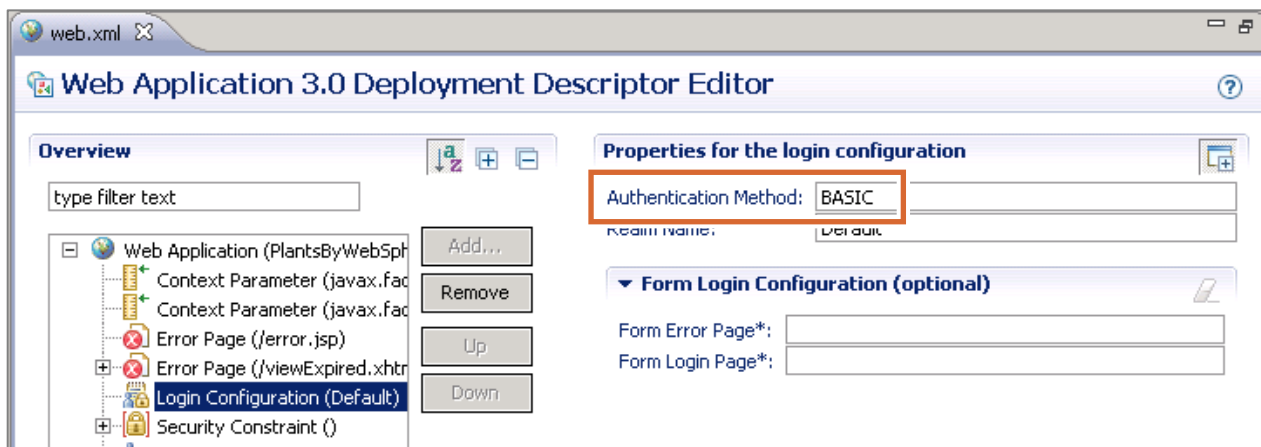
Challenge mechanism: Form-based authentication

- Defined within the application
- Suggested approach



Defining the challenge type

- Challenge types are set in the EAR deployment descriptors
 - The default that Rational Application Developer uses is basic authentication
 - Form-based is defined by adding a login configuration

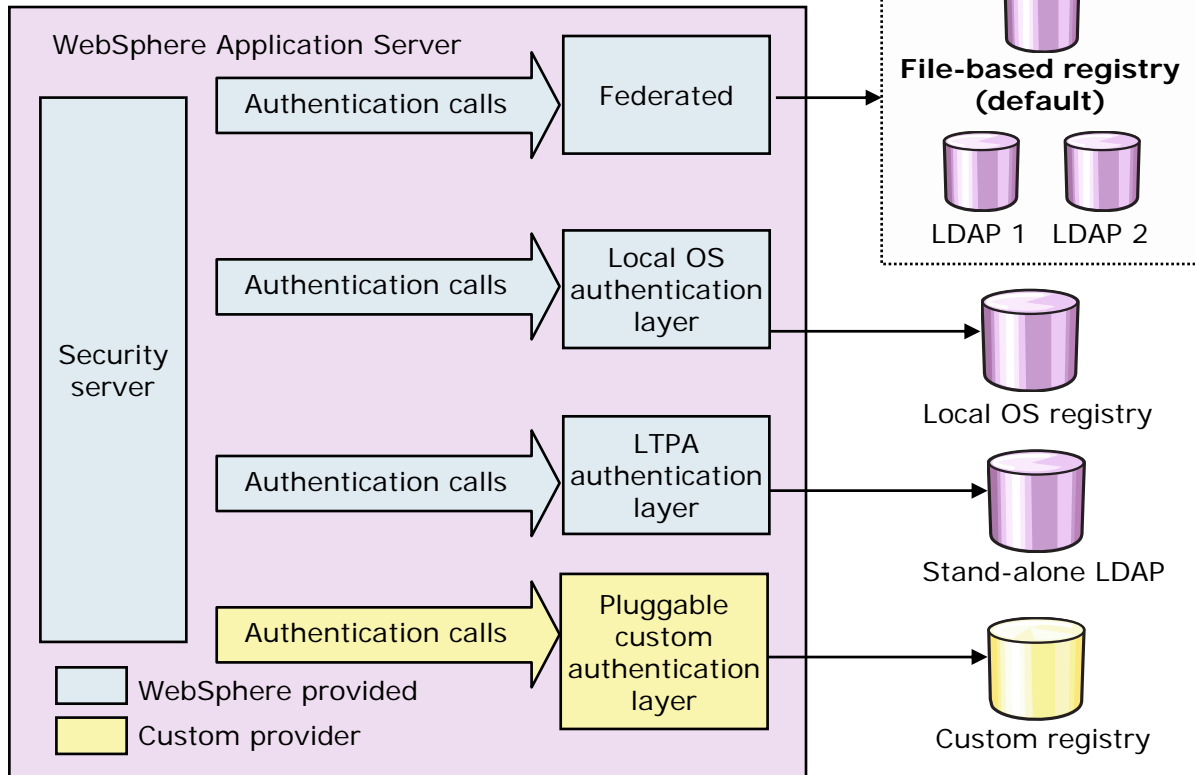




WebSphere user registries



Registries and authentication mechanisms



Defining user registries

- Use the administrative console to configure user registries
 - Manually (preferred)
 - Wizard (considered too simplistic)

Global security

Use this panel to configure administration and the default application security policy. This security policy for all administrative functions and is used as a default security policy for user applications. You can override and customize the security policies for user applications.

Security Configuration Wizard Security Configuration Report

Administrative security

☒ Enable administrative security

- [Administrative user roles](#)
- [Administrative group roles](#)
- [Administrative authentication](#)

Application security

☐ Enable application security

Java 2 security

☐ Use Java 2 security to restrict application access to local resources

- ☒ Warn if applications are granted custom permissions
- ☐ Restrict access to resource authentication data

User account repository

Realm name
defaultWIMFileBasedRealm

Current realm definition

Available realm definitions

- Federated repositories
- Federated repositories
- Local operating system
- Standalone LDAP registry
- Standalone custom registry

Configure Set as current

Authentication

Authentication methods

- ☒ LTPA
- ☐ Kerberos authentication
- ☐ Kerberos authentication
- ☐ Authentication configuration
- ☐ Web and SIP
- ☐ RMI/IIOP security
- ☐ Java Authentication and Authorization
- ☐ Enable Java Authentication and Authorization
- ☐ Use realm-defined authentication
- ☐ Security domain
- ☐ External authentication
- ☐ Programmatic authentication
- ☐ Custom properties

Manual security configuration

User account repository

Realm name
defaultWIMFileBasedRealm

Current realm definition
Federated repositories

Available realm definitions

Federated repositories (selected)
Local operating system
Standalone LDAP registry
Standalone custom registry

Configure...

- The steps depend on the specific environment and which repository is being configured
- Much more detailed than the security wizard

Global security

Global security > Federated repositories

By federating repositories, identities stored in multiple repositories can be managed in a single, virtual realm. The realm can consist of identities in the file-based repository that is built into the system, in one or more external repositories, or in both the built-in repository and one or more external repositories.

General Properties

* Realm name
defaultWIMFileBasedRealm

* Primary administrative user name
wasadmin

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

☒ Ignore case for authorization

☐ Allow operations if some of the repositories are down

Repositories in the realm:

Select	Base Entry	Repository Identifier	Repository Type
<input type="checkbox"/>	o=defaultWIMFileBasedRealm	InternalFileRepository	File

You can administer the following resources:

Security wizard: Step 1

- Not detailed enough to be used for anything but simple environments

Configure security

This wizard assists you in securing your application serving environment. The application serving infrastructure can store administrative users and passwords or can use an existing registry with stored administrative users, application users, or both.

→ **Step 1: Specify extent of protection**

(The next step of the wizard depends on decisions made in the current step)

Specify extent of protection

This wizard assists you in securing your application serving environment. The application serving infrastructure can store administrative users and passwords or can use an existing registry with stored administrative users, application users, or both.

If you are using an existing registry such as the local operating system, LDAP, or a custom registry, you need the following information:

- Configuration information to connect to the existing registry
- An existing user name in the registry to act as the primary administrative user

At a minimum, this task provides for secure administration. However, administrative security alone does not provide full security. In most environments, it is recommended that you also enable application and resource security.

☒ Enable application security

☐ Use Java 2 security to restrict application access to local resources

Next

Cancel

Security wizard: Step 2

Configure security

Secure the application serving environment

Step 1: Specify
extent of protection

→ **Step 2: Select
user repository**

*(The next step of the
wizard depends on
decisions made in the
current step)*

Step 3: Summary

Select user repository

The user account repository stores users and group names that are used for authentication and authorization. The default repository is built into the application serving system and can be federated with one or more external Lightweight Directory Access Protocol (LDAP) repositories. You can also select a standalone external repository.

- ☒ Federated repositories
- ☐ Standalone LDAP registry
- ☐ Local operating system
- ☐ Standalone custom registry

Previous

Next

Cancel

Security wizard: Step 3

Configure security

Secure the application serving environment

Step 1: Specify extent of protection

Step 2: Select user repository

→ **Step 3: Configure federated repository**

Step 4: Summary

Configure federated repository

A secure, file-based user repository is built into the system for storing administrative users or environments with a small number of users. The file-based user repository can be federated with one or more external LDAP repositories. If this is the first time security has been enabled using this repository, provide a new user name and password to act as an administrator. If security was previously enabled using this repository, provide the name of a user with administrator privileges that is in the built-in repository.

Note: Use this panel to configure a federated repository with a built-in, file-based repository in the realm. To configure a federated repository with a non file-based repository in the realm, you must use the User accounts repository section on the Global security panel.

* Primary administrative user name

Password

Confirm password

Previous

Next

Cancel

Security wizard: Step 4

Configure security

Secure the application serving environment

Step 1: Specify
extent of protection

Step 2: Select user
repository

Step 3: Configure
federated repository

→ **Step 4: Summary**

Summary

Displays the list of values that are selected during the wizard and are used to enable security.

Options	Values
Enable administrative security	true
Enable application security	true
Use Java 2 security to restrict application access to local resources	false
User repository	Federated repositories
Primary administrative user name	wasadmin

Previous

Finish

Cancel

User registry support

- WebSphere Application Server supports some user registries

Local OS

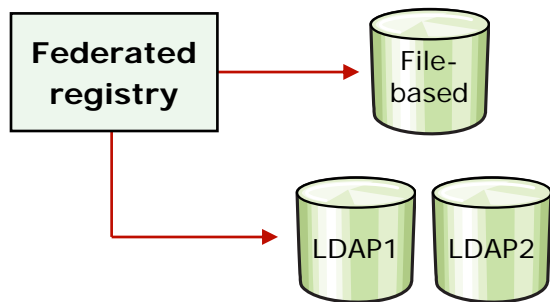
NT Domain, NT WorkGroup, Windows
AIX
Solaris
HP-UX
Linux
OS/400

LDAP

IBM Tivoli Directory Server
IBM SecureWay Directory Server
Sun Java System Directory Server
IBM Lotus Domino
Microsoft Active Directory
Novell eDirectory
Custom (requires addition configuration)

Federated repositories

- The installation wizard and profile management tools have a default of enabling administrative security
 - The default repository type is a file-based federated repository
- Federated repositories provide for the use of multiple repositories with WebSphere Application Server
- Can be:
 - File-based
 - Single LDAP
 - Custom registry
 - Database
 - Multiple LDAPs
 - Subtree of an LDAP
- Defined and theoretically combined under a single realm
- All of the user repositories that are configured under the federated repository are invisible to WebSphere Application Server



Custom registry: Configuration

[Global security](#) > Standalone custom registry

Specifies a custom registry that implements the UserRegistry interface in the com.ibm.websphere.security package. For backward compatibility, the application server also supports a custom registry that implements the CustomRegistry interface in the com.ibm.websphere.security package. When security is enabled and any of the properties on this panel are changed, go to the Security > Global security panel. Click Apply or OK to validate the changes.

General Properties

* Primary administrative user name

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 registry class name

☐ Ignore case for authorization

Custom properties

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

New

Delete

Related Items

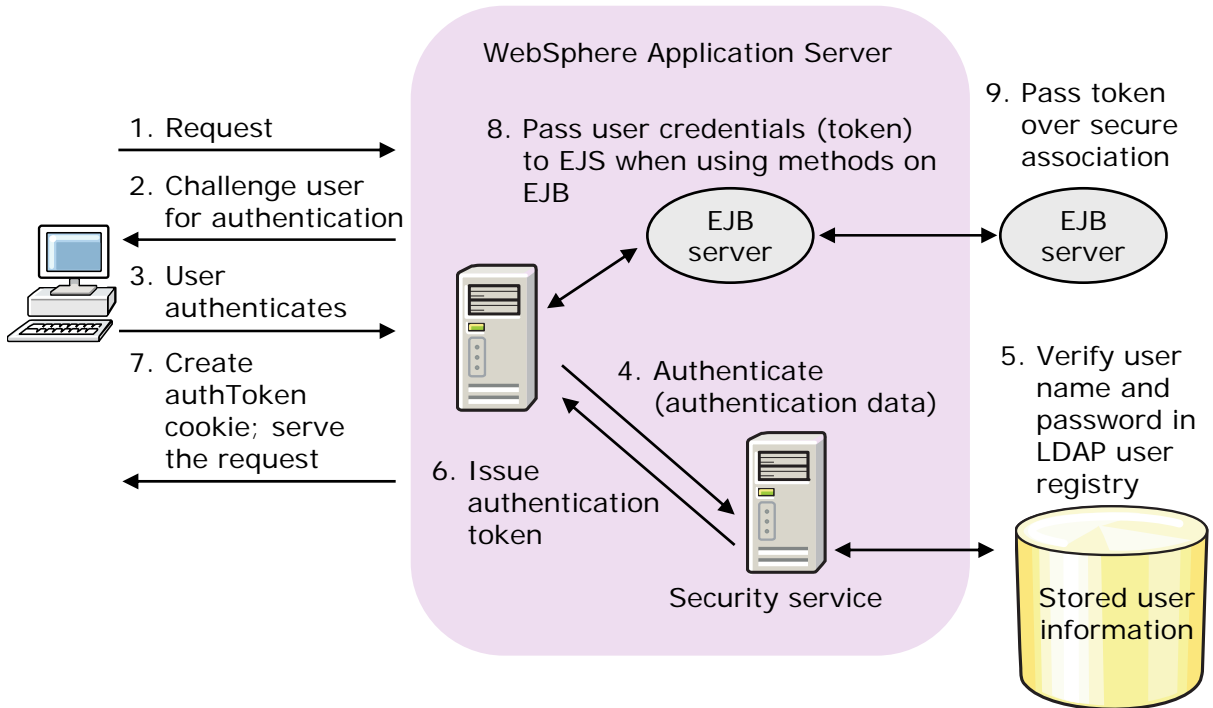
- [Trusted authentication realms - inbound](#)

Configured from administrative console:

- **Security > Global security**
- Select **Stand-alone custom registry** from the **Available realm definitions**
- Click **Configure**
 - User name and password must exist
 - Class name must be implemented and in class path

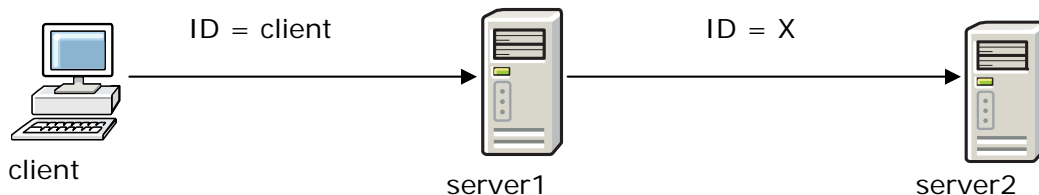
Authentication mechanism: LTPA

- Allows the identity of a user to be passed around the distributed network





LTPA provides delegation



X can run as:

Option 1. Client

Option 2. Server 1

Option 3. Specified identity

- Using an LTPA token supports delegation

LTPA provides single sign-on (SSO)

Global security

Global security > Single sign-on (SSO)

Specifies the configuration values for single sign-on.

General Properties

☒ Enabled

☐ Requires SSL

Domain name

☐ Interoperability mode

LTPA V1 cookie name

LTPA V2 cookie name

☒ Web inbound security attribute propagation

☒ Set security cookies to HTTPOnly to help prevent cross-site scripting attacks

- As soon as clients have a valid LTPA token, they do not need to reauthenticate within a cell (until the LTPA token expires)
- SSO is on by default
- Issues cookies to web browser to track user authentication information
- Provides for SSO within or even between WebSphere cells



Administrative security



Administrative security

Turning on administrative security enables many features, including:

- Authentication of HTTP and IIOP clients
- Administrative console security
- Naming security
- Use of SSL transports
- Role-based authorization checks of servlets, EJBs, and MBeans
- Propagation of identities (RunAs)
- The common user registry

Console and other administrative tools: access is initially restricted to only the primary user

- You must create your administrative users and groups
- As of version 7, fine-grained access can be defined for console users
 - For example, Bob can be configured to have administrative access to application servers A and B
 - Fred can be configured to have operator access to only servers C and D

Console security

Defines which roles have access to the administrative tools

- **Monitor:** least privileged; allows a user to view the WebSphere configuration and current application server state
- **Configurator:** monitor privilege plus the ability to change the WebSphere configuration
- **Operator:** monitor privilege plus the ability to change runtime state, such as starting or stopping servers
- **Administrator:** operator, configurator, and iscadmins privilege, plus more privileges that are granted solely to the administrator role, such as:
 - Modifying the primary administrative user and password
 - Mapping users and groups to the administrator role
 - Enabling or disabling administrative and Java 2 security

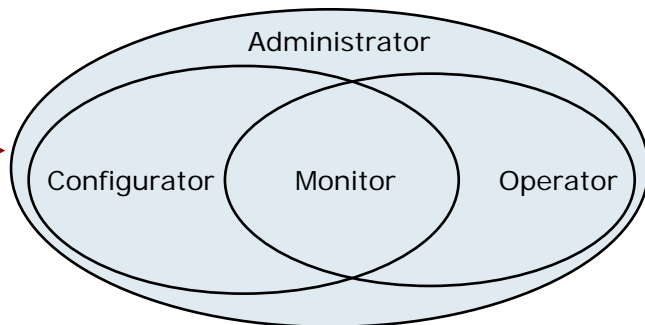


Administrative console

```
C:\> wsadmin
```

Security check

Application server

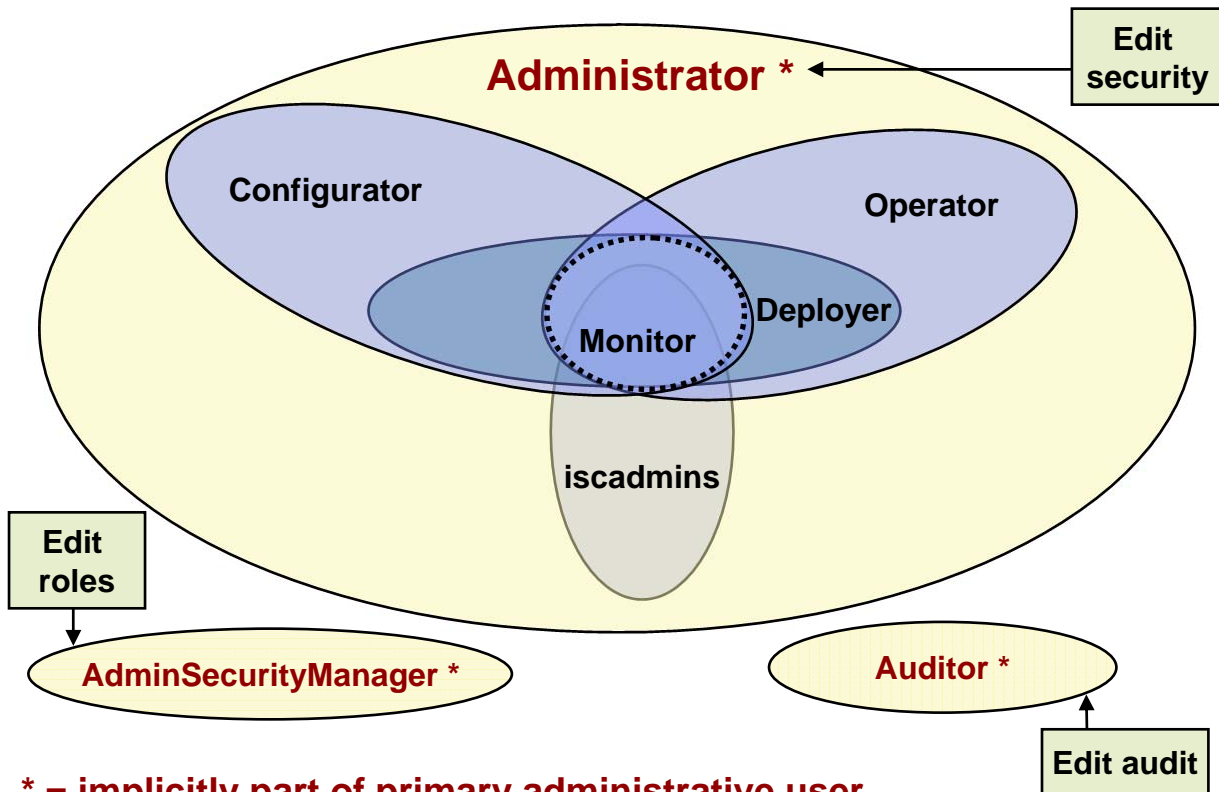




Additional console security roles

- **iscadmins** (Integrated Solutions Console)
 - *Only available for administration console users*
 - Allows a user to manage users and groups in the federated repositories
- **Deployer**
 - Allows a user to change configuration and runtime state on applications that use wsadmin
- **Admin Security Manager**
 - Allows a user to map users to administrative roles by using wsadmin
 - When restricted access to resource authentication data is in effect, users can also manage authorization groups
- And others

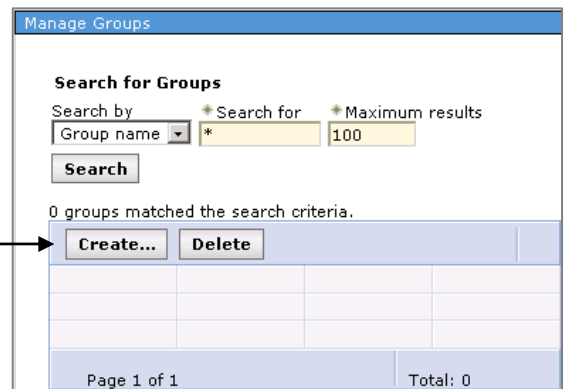
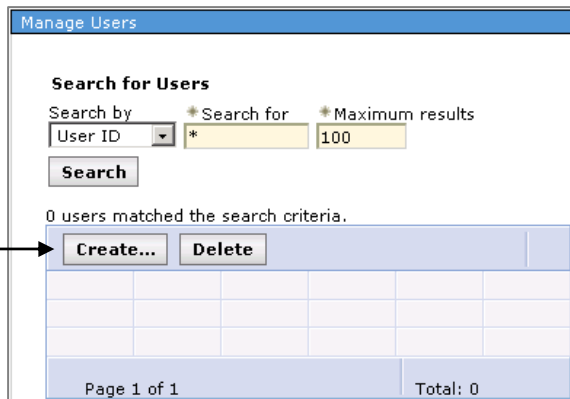
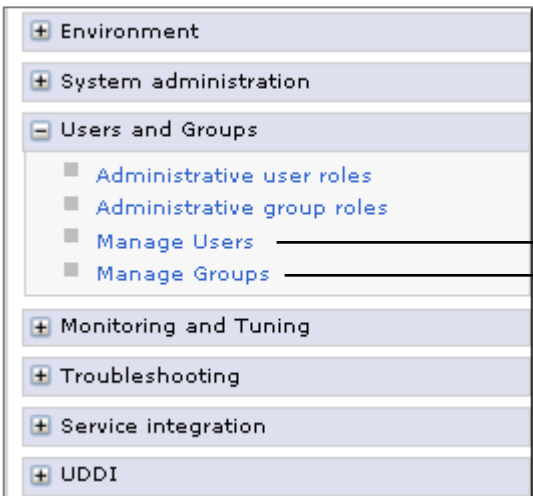
Administrative roles



*** = implicitly part of primary administrative user**

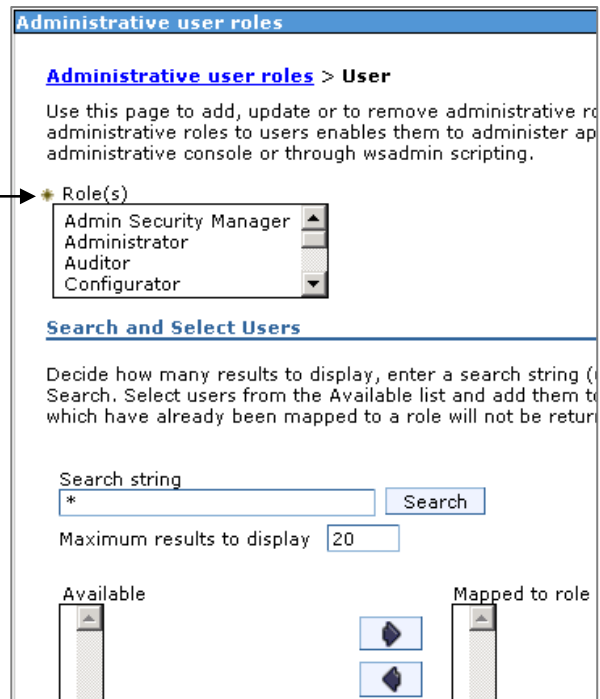
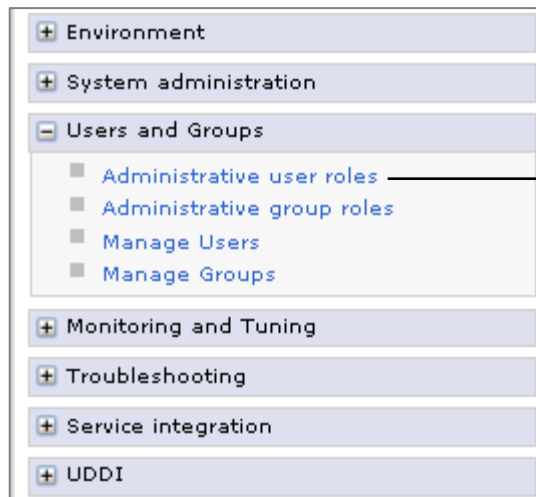
Console security: Creating users and groups

- To set up console security
 - Turn on administrative security
 - **Create console users and groups**
 - Done in active user registry



Console security: Mapping users and groups

- To set up console security
 - Turn on administrative security
 - Create console users and groups
 - Map users and groups to administrative roles**





Application security

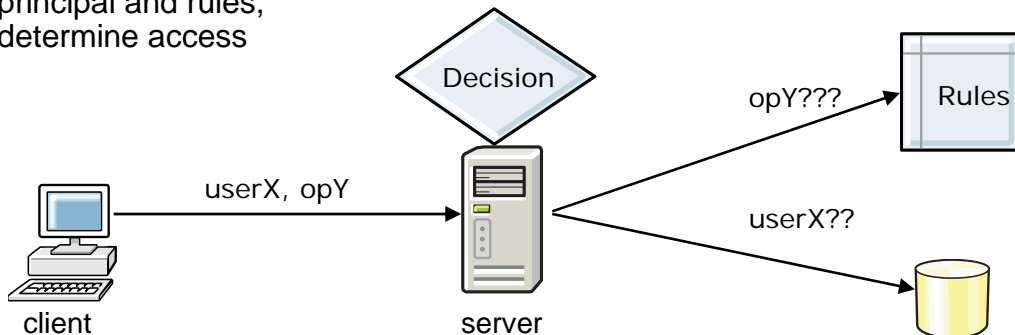


Authorization

Authorization involves granting trusted principals permission to perform actions on resources (web pages, servlets, JSPs, and EJB components)

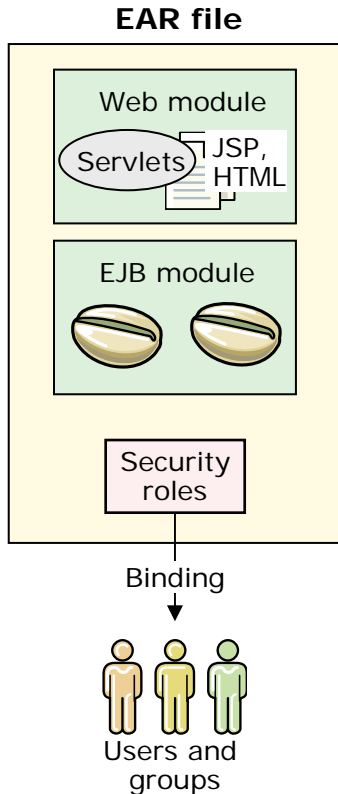
Control access to resources

- Security lookup (by server)
 - Determine security privileges for principal
 - Access information that is stored in registry
- Rule enforcement (by server)
 - Obtain rules from registry
 - Given privileges of principal and rules, determine access

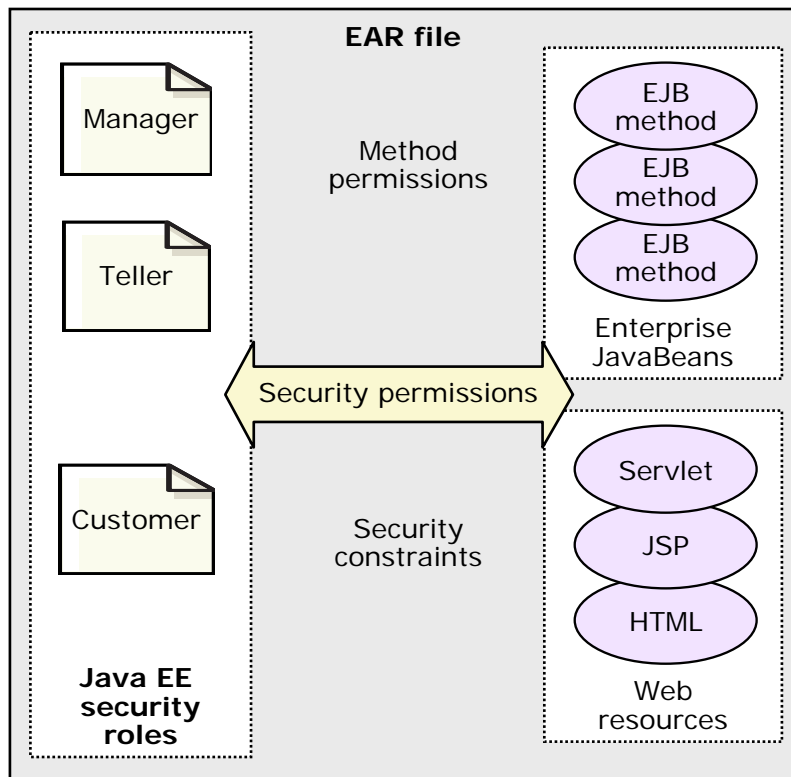


Security roles: Application authorization

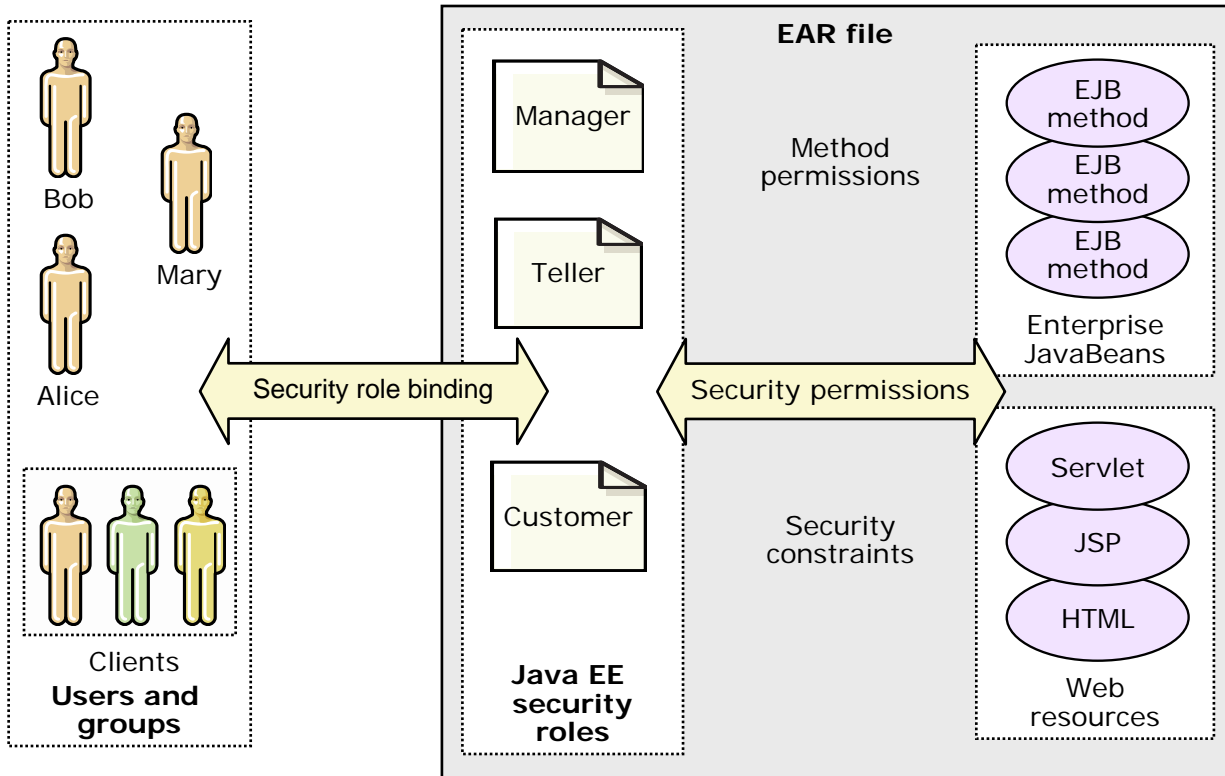
- Use security roles for authorization
 - Specify security at an abstract level without knowledge of actual users and groups
- Security roles are then applied to the web and EJB application components
 - Web URIs or EJB methods
- Binding of the users and groups to the security roles is generally done at the application installation time
 - Can be done post-installation as well



Securing Java EE application artifacts: Part 1



Securing Java EE application artifacts: Part 2



Applying application security

- Application security can be applied to resources within an EAR
 - Security roles are defined in the application deployment descriptor
 - Servlets and JSPs are protected with security constraints, which are mapped to the security roles
 - EJBs are protected with method permissions, which are mapped to the security roles
- The security roles are then mapped to actual users and groups during installation of the application

The screenshot shows the 'Web Application 3.0 Deployment Descriptor Editor' window. The 'Overview' tab is active, displaying a tree view of the application's components. The 'Security Constraint ()' is highlighted in the tree. To the right of the tree are buttons for 'Add...', 'Remove', 'Up', and 'Down'. The 'Details' tab is also visible, showing the 'Authorization Constraint (optional)' section with a 'Role Name' field set to 'SampAdmin' and buttons for 'Add', 'Remove', 'Up', and 'Down'.

web.xml

Web Application 3.0 Deployment Descriptor Editor

Overview

type filter text

- Web Application (PlantsByWebSpot)
 - Context Parameter (javax.faces.context.FacesContext)
 - Context Parameter (javax.faces.context.FacesContext)
 - Error Page (/error.jsp)
 - Error Page (/viewExpired.xhtml)
 - Login Configuration (Default)
 - Security Constraint ()**
 - Security Role (SampAdmin)

Add... Remove Up Down

Details

Display Name:

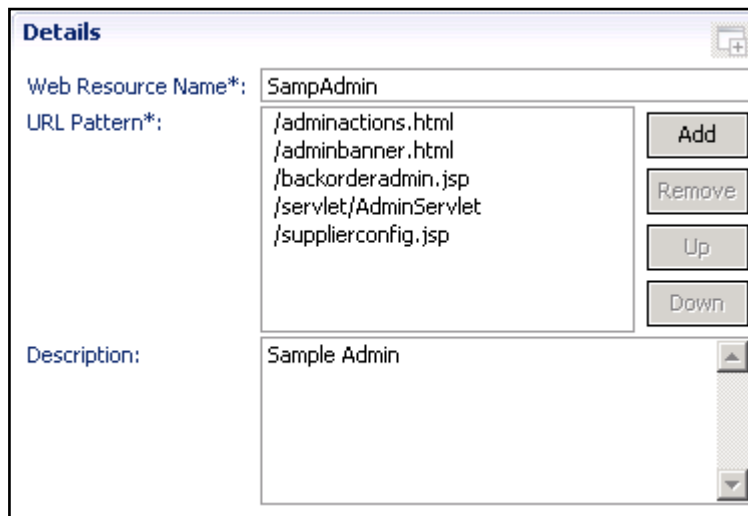
Authorization Constraint (optional)

Role Name: SampAdmin

Add Remove Up Down

Creating security constraints

- After the security role is created, it can be mapped in a security constraint to protect web application artifacts

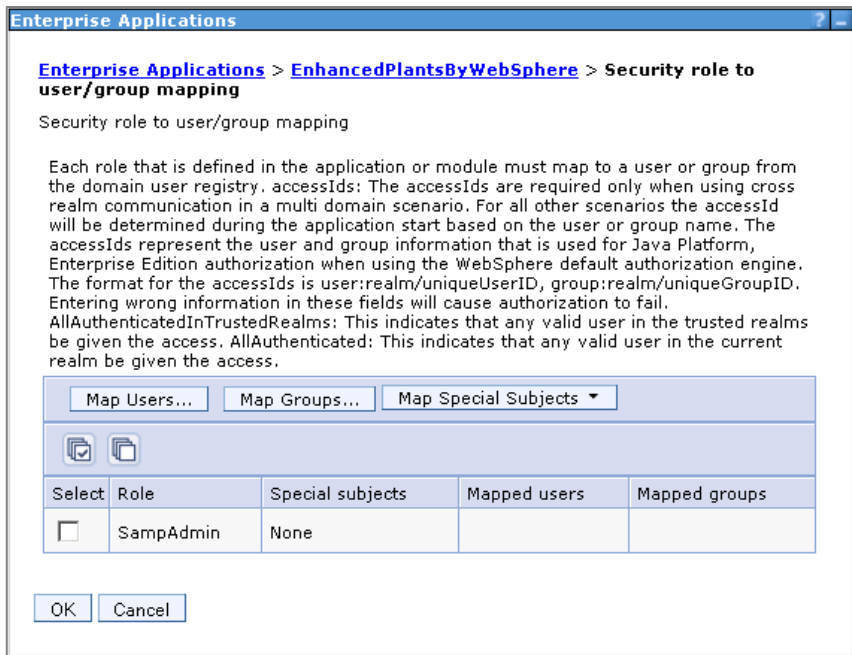


The screenshot shows a 'Details' dialog box for configuring a security constraint. It contains the following fields and controls:

- Web Resource Name*:** A text field containing 'SampAdmin'.
- URL Pattern*:** A list box containing the following patterns:
 - /adminactions.html
 - /adminbanner.html
 - /backorderadmin.jsp
 - /servlet/AdminServlet
 - /supplierconfig.jspNext to the list box are four buttons: 'Add', 'Remove', 'Up', and 'Down'.
- Description:** A text area containing 'Sample Admin'.

Using the console to map security roles

- The mapping of users and groups to security roles can take place during or after application installation
 - After installation, use the administrative console to go to the application and under Detailed Properties, select **Security role to user/group mapping**





Security domains

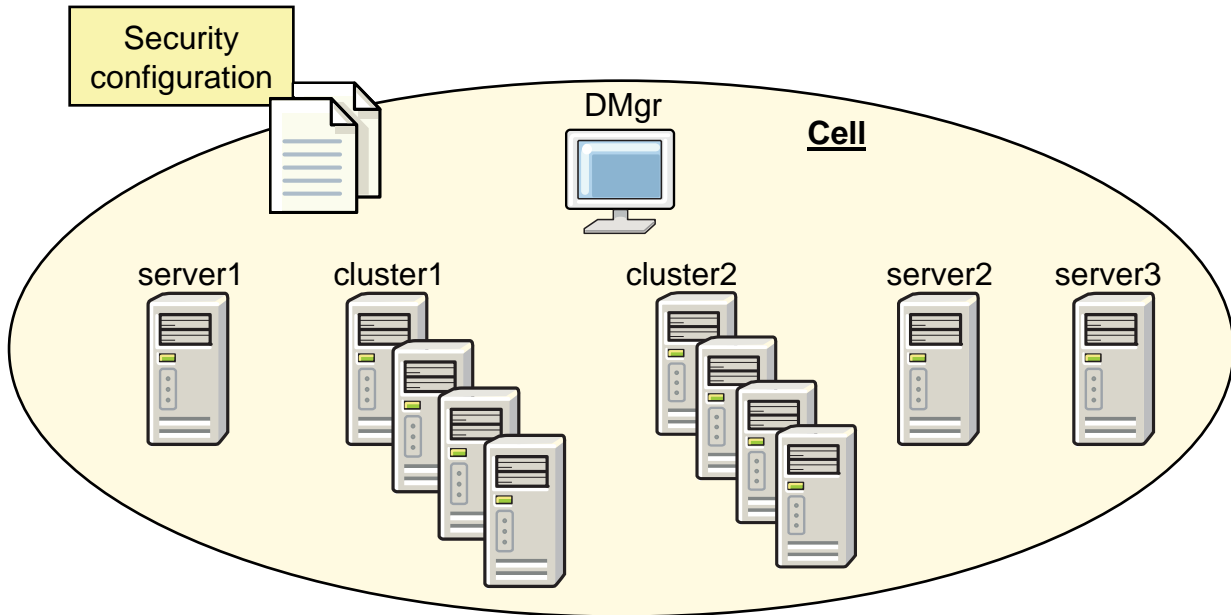


Security domains

- Multiple security domains are supported as of WebSphere Application Server version 7
 - Can create different security configurations and assign them to different applications
 - Can configure different security attributes for both administrative and user applications within a cell environment
 - Can configure different applications to use different security configurations by assigning the servers, clusters, or service integration buses to the security domains
- Only users that are assigned to the administrator role can configure multiple security domains
- For example, with security domains, it is possible to have different user registries that are configured for distinct parts of the cell

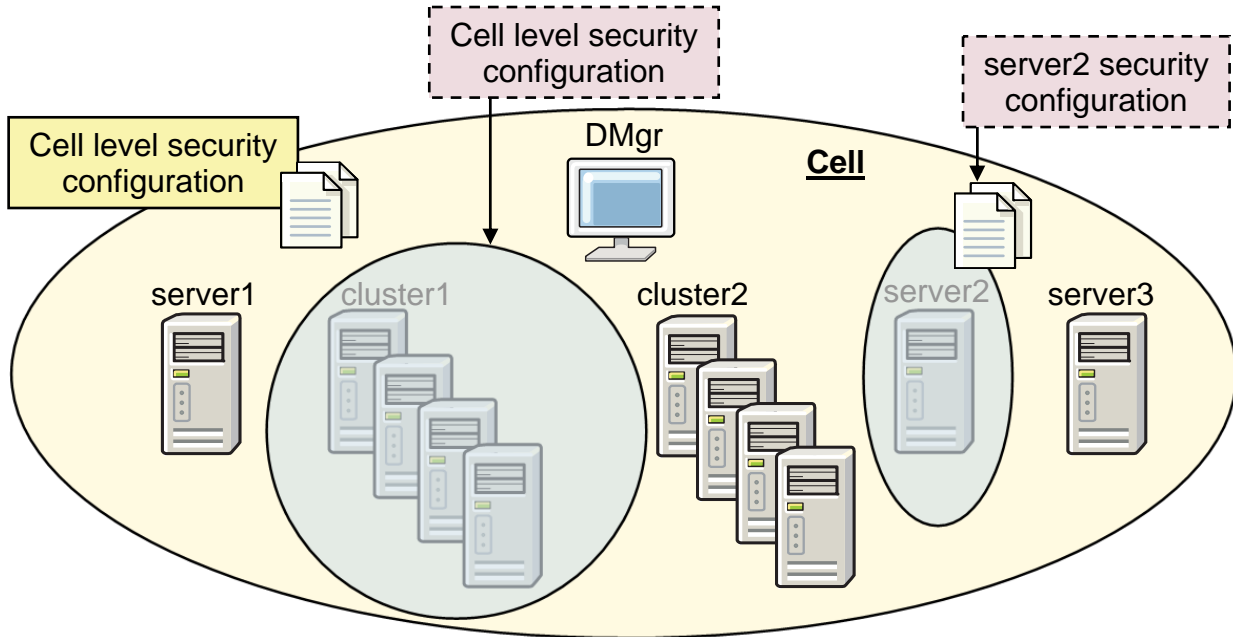
Security configurations

- Traditionally, the security configuration was defined at a cell level
 - A side effect was all elements of the cell shared the exact same security configuration



Security domains

- With security domains, it is possible to have a cell level security configuration, and multiple other security configurations at different scopes



Creating a security domain

- Use the console to create a security domain
 - **Security > Security domains > New**

The image shows two overlapping screenshots from the WebSphere console. The top screenshot shows the main console interface with a left-hand navigation pane. The 'Security' section is expanded, and 'Security domains' is highlighted with a red box. The right-hand pane shows the 'Security domains' overview page, with the 'New...' button highlighted by a red box. The bottom screenshot is a zoomed-in view of the 'Security domains > New...' dialog box. It contains a 'Name' field with the text 'PlantsSecurityDomain', an empty 'Description' field, and buttons for 'Apply', 'OK', 'Reset', and 'Cancel'.

Security domains

Security domains provide a mechanism to use different security settings for admin applications. They also provide the ability to support multiple security settings so different security attributes like user registry or login configurations.

Preferences

New... Delete Copy Selected Domain... Copy Global Security...

Security domains > New...

Use this panel to provide a name and description for the security domain. Once you apply the name, you can configure the security attributes of this domain and assign it to cell resources.

Name
PlantsSecurityDomain

Description

Apply OK Reset Cancel

Configuring a security domain

- Define a scope and configure the attributes
 - It is possible to enable application security for only the PlantsByWebSphere

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.

Show:

All scopes

- ☐ Cell
 - ☒ Clusters
 - ☒ Service integration buses
 - ☐ Nodes
 - ☒ was8host01CellManager01
 - ☐ was8host01Node01
 - ☐ Servers
 - ☒ server1
 - ☒ was8host01Node02

Security Attributes

- ☐ **Application Security:** Disabled
 - ☒ Use global security settings
Do not enable application security
 - ☐ Customize for this domain
 - ☐ Enable application security
- ☒ **Java 2 Security:** Disabled
- ☒ **User Realm:** Administrative realm
- ☒ **Trust Association:** Disabled



Java 2 security



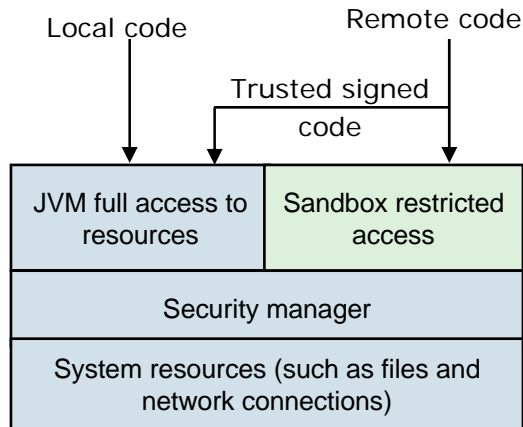
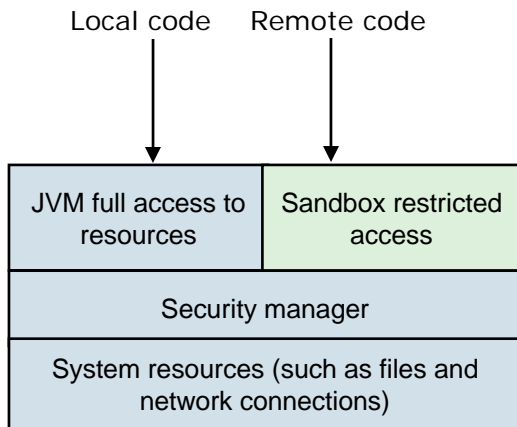
Java security model

- Java 1.0 (sandbox model):

- Downloaded code (untrusted) runs in a sandbox (restricted environment)
- Application code (local Java classes) has full access to resources (trusted and no protection)

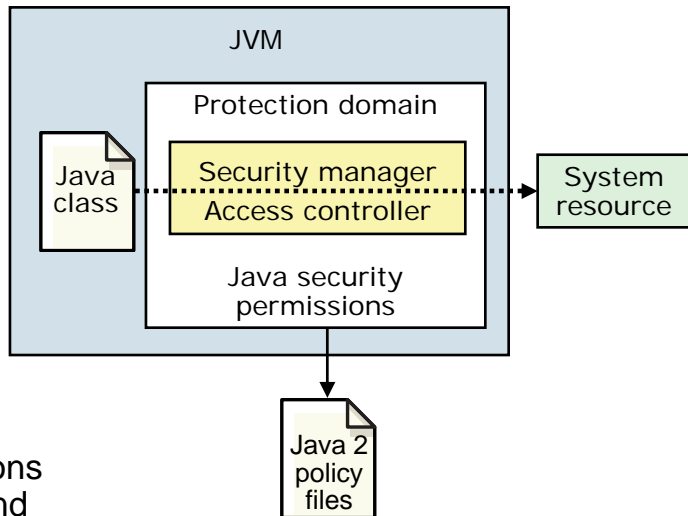
- Java 1.1 (signed code):

- Extends 1.0 sandbox model. Introduces signed code. Digitally signed remote code is treated like local code if the public key used to verify that the signature is trusted



Java security overview

- Protects the system from the applications
- Provides an access control mechanism to manage the application access to system level resources
 - File I/O, network connections (sockets), property files
 - Policy-based
- Policies define a set of permissions available from various signers and code locations
 - Stored in policy files
- All Java code runs under a security policy
 - Grants access to certain resources
- Can be turned on or off independently of administrative security



- Java code needs access to certain system resources
- Java code must get the permission from Java 2 access control
- Access control looks at the Java 2 policy files to determine whether the requesting Java code has the appropriate permission

Enabling Java 2 security

- Can be enabled and disabled independently of administrative and application security
- Java 2 security provides a policy-based, fine-grain access control mechanism that increases overall system integrity.
- Java 2 security checks for permissions before allowing access to certain protected system resources

Global security

Use this panel to configure administration and the default application security policy. This security configuration applies to the policy for all administrative functions and is used as a default security policy for user applications. Security domains can be overridden and customized the security policies for user applications.

Security Configuration Wizard
Security Configuration Report

Administrative security

☒ Enable administrative security

- [Administrative user roles](#)
- [Administrative group roles](#)
- [Administrative authentication](#)

Application security

☒ Enable application security

Java 2 security

☒ Use Java 2 security to restrict application access to local resources

- ☒ Warn if applications are granted custom permissions
- ☐ Restrict access to resource authentication data

Authentication

Authentication mechanisms and expiration

- ☒ [LTPA](#)
- ☐ Kerberos and LTPA
 - [Kerberos configuration](#)
 - [Authentication cache settings](#)
- ☐ Web and SIP security
- ☐ RMI/IIOP security
- ☐ Java Authentication and Authorization S...
- ☐ Enable Java Authentication SPI (JASPI)
 - [Providers](#)
- ☐ Use realm-qualified user names

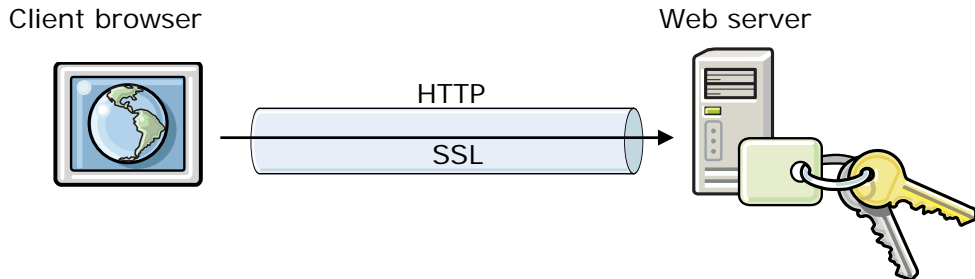


SSL basics



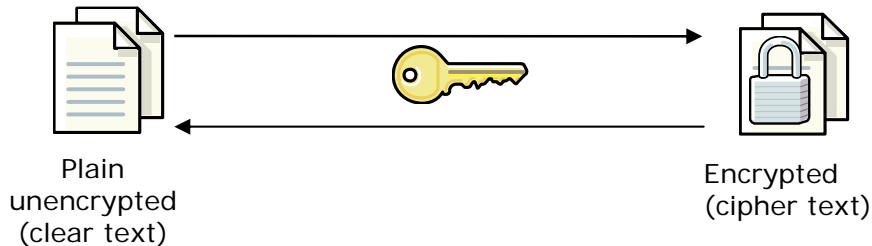
What is SSL?

- SSL stands for Secure Sockets Layer
- SSL provides connection security through:
 - Communication privacy: the data on the connection can be encrypted
 - Communication integrity: the protocol includes a built-in integrity check
 - Authentication: the client knows who the server is
- Creates a VPN
 - Uses both symmetric and asymmetric key encryption



Symmetric key encryption

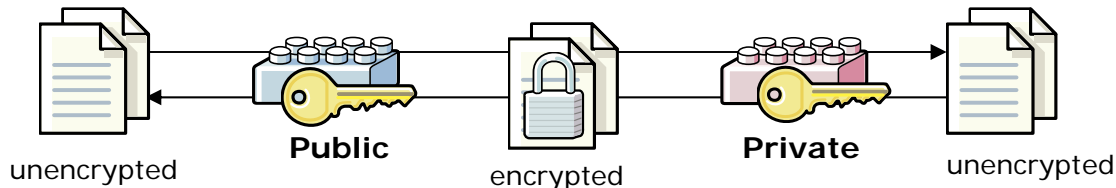
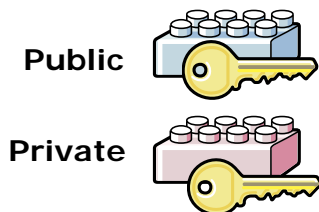
- Symmetric or secret key technology is a model in which two parties have a shared secret
- The same key is used for both encryption and decryption



Asymmetric key encryption

Public key cryptography

- Two keys that are cryptographically related:
 - Public key (can share with everyone)
 - Private key (must never be shared; possession is proof)
- Keys are asymmetric:
 - Given message is encrypted with one key and decrypted with another
 - Symmetric, secret key technology uses the same key for encryption and decryption





How does SSL work?

SSL uses a combination of asymmetric and symmetric encryption to create a session between the client and server

- Asymmetric encryption is used to negotiate a session key (shared secret)
 - Asymmetric encryption is slow but does not require a shared secret
- Symmetric encryption is used to transfer data between the client and server
 - Symmetric encryption is fast but requires a shared secret

Client



3. Client verifies server certificate

Server



1. Client requests SSL connection

2. Server presents certificate

4. Client generates a **session key**, encrypts it with the public key for the server

5. Using the session key, client and server switch to symmetric key encryption

6. HTTPS communications





Certificates and certificate authorities





What is a certificate?

Simple answer:

- It is an electronic document that identifies you, and a third-party vouches for both you and the certificate itself
- Examples:
 - Employee badge (vouched for by your employer)
 - Drivers license (vouched for by your state)
 - Passport (vouched for by your country)



More information:

- Includes information about you
- Includes public key
- A certificate authority digitally signs it

Types of certificates

There are different types of certificates:

- **Certificate**

- Contains a public key that is signed
- Contains information about the owner of the certificate
- Contains certificate expiration date

- **Personal certificate:**

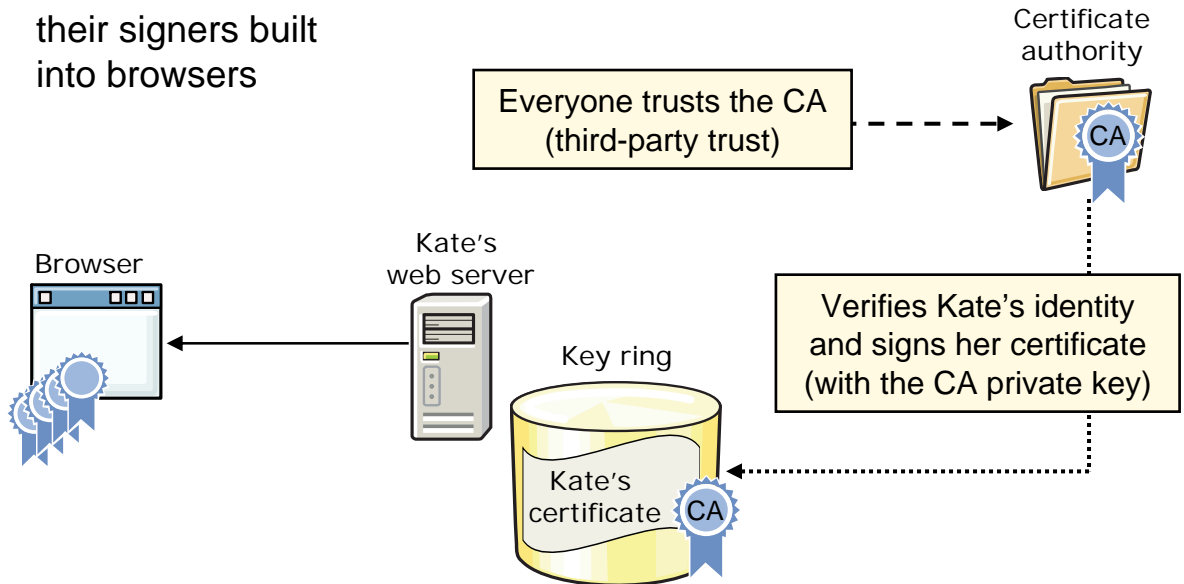
- Typically meant as the certificate along with private key data

- **Signer certificate:**

- The certificate that corresponds to the private key used to digitally sign another certificate

What is a certificate authority (CA)?

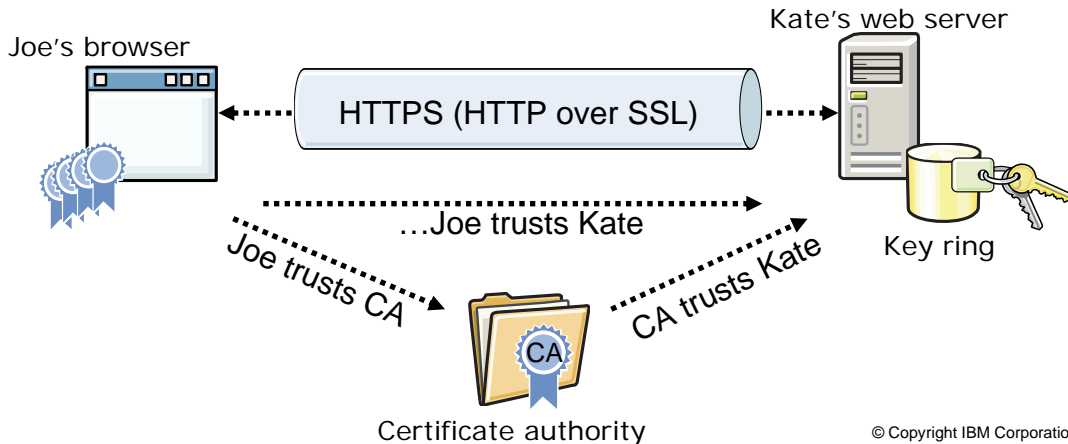
- An entity that signs public keys, thus creating certificates
 - A CA validates Kate's identity before vouching for her (signing her certificate)
 - A special type of signer (a trusted signer)
- Well known CAs have their signers built into browsers



SSL: Putting it all together

The SSL handshake establishes:

- The identity of the server (based on trusting the CA)
 - The server provides a CA signed certificate
 - The server proves that it has the corresponding private key
 - Therefore, Joe trusts that Kate really is Kate
- Encrypted (with symmetric key) channel between the browser and server



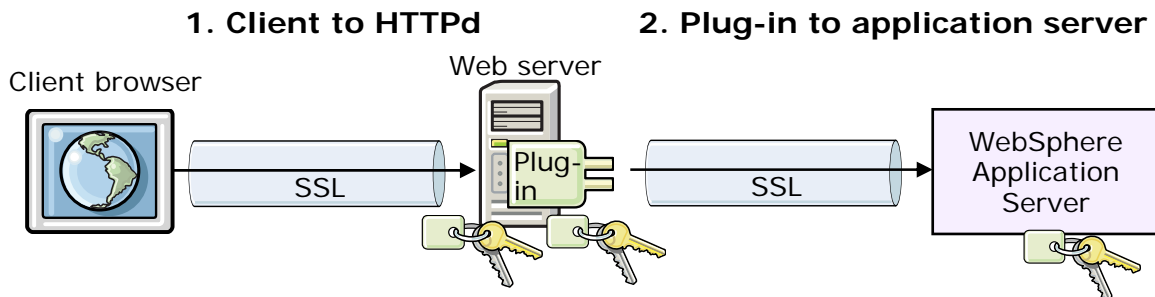


SSL within a WebSphere cell



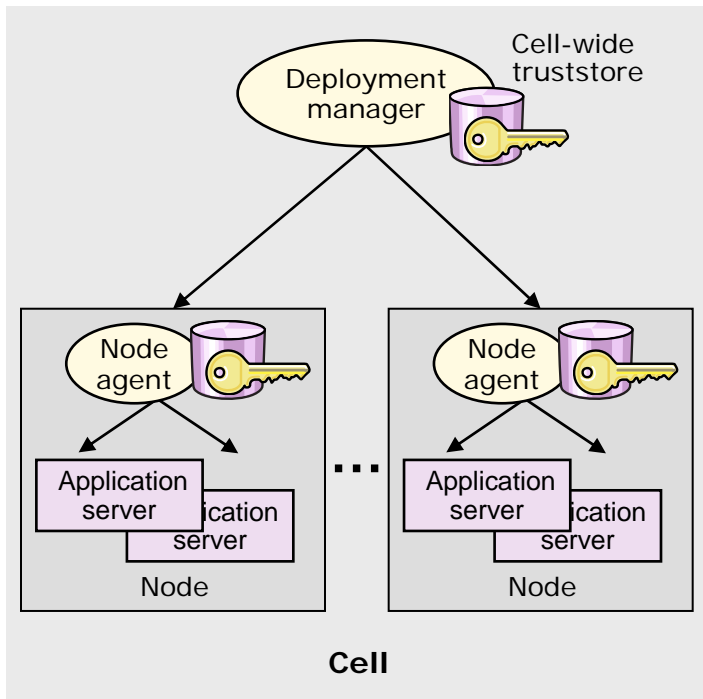
SSL within WebSphere Application Server

- SSL can be used to secure network traffic for a number of links
 - From the client to the web server
 - From the plug-in to the application server
 - Other network links can also be secured (LDAP and others)
- The **administrative console** (or iKeyman) can be used to create and manage the necessary keys and keystores
 - Keystores contain digital certificates that are needed for SSL to establish secure communication between two points

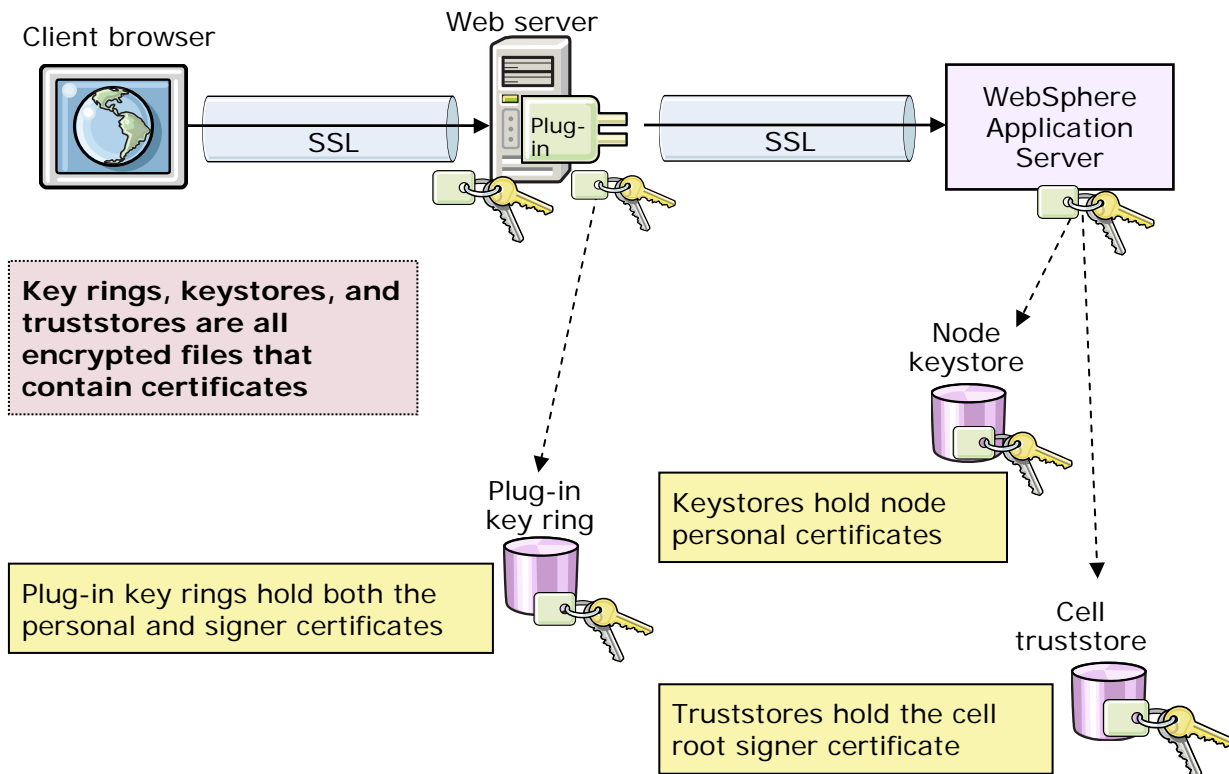


WebSphere SSL management

- WebSphere automatically creates node certificates
 - The cell root certificate (called a “chained certificate”) signs the node certificates
 - Node certificates are stored in a node-specific keystore
- Cell-wide truststore includes cell root signer certificate
 - Each node can therefore validate certificates that other nodes present
- An expiration manager automatically renews expiring keys (default behavior)
- The keystores and truststores are stored within the cell configuration and therefore distributed to the nodes through file synchronization

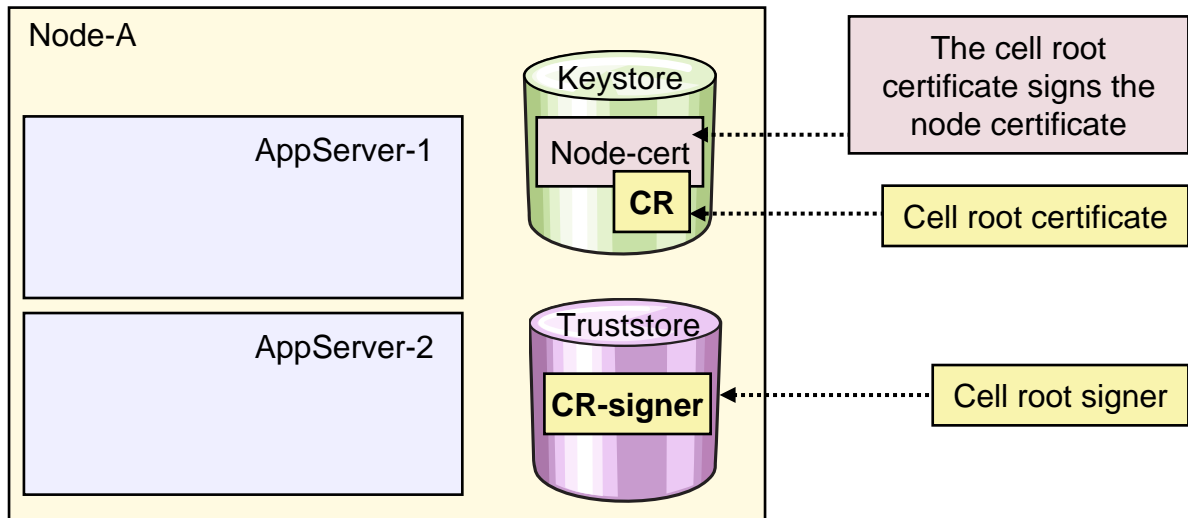


What are key rings, keystores, and truststores?



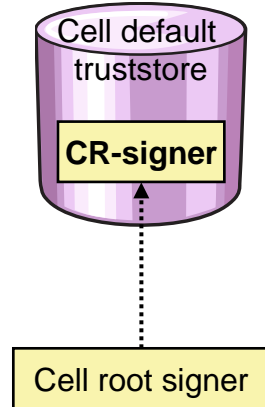
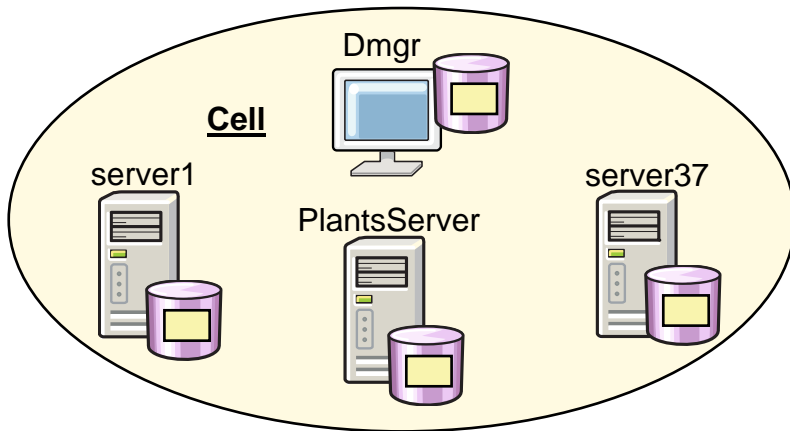
Node certificates

- Each node has a node certificate
 - The cell root certificate (a chained certificate) signs the node certificate
 - The cell root signer is therefore needed to validate the node certificate
 - The application servers, by default, all use the local node certificate



Cell default truststore

- The cell default truststore contains signers, include the cell root signer
 - It is synchronized to all the members of the cell
 - All the nodes use the common cell default truststore
 - Although there are node truststore files, they are not used by default



Managing WebSphere keystores

- Keystores and certificates for the cell, nodes, and plug-ins can be managed directly from the console
- Expiration management
- Keystores
- Trust files
- Certificates

SSL certificate and key management

SSL configurations

The Secure Sockets Layer (SSL) protocol provides secure communications between remote server processes or endpoints. SSL security can be used for establishing communications inbound to and outbound from an endpoint. To establish secure communications, a certificate and an SSL configuration must be specified for the endpoint.

In previous versions of this product, it was necessary to manually configure each endpoint for Secure Sockets Layer (SSL). In this version, you can define a single configuration for the entire application-serving environment. This capability enables you to centrally manage secure communications. In addition, trust zones can be established in multiple node environments by overriding the default, cell-level SSL configuration.

If you have migrated a secured environment to this version using the migration utilities, the old Secure Sockets Layer (SSL) configurations are restored for the various endpoints. However, it is necessary for you to re-configure SSL to take advantage of the centralized management capability.

Configuration settings

[Manage endpoint security configurations](#)

[Manage certificate expiration](#)

☐ Use the United States Federal Information Processing Standard (FIPS) algorithms. Note: This option requires the TLS handshake protocol, which some browsers do not enable by default.

☒ Dynamically update the run time when SSL configuration changes occur

Apply

Reset

Creating keystores and certificates

<div> <div>Create</div> <div>Delete</div> <div>Receive from a certificate authority...</div> <div>Replace...</div> <div>Extract...</div> <div>Import...</div> <div>Export...</div> <div>Revoke...</div> <div>Renew</div> </div>						
Select	Alias	Issued To	Issued By	Serial Number	Expiration	
You can administer the following resources:						
<input type="checkbox"/>	default	CN=192.168.136.128, OU=was8host01Node02Cell, OU=was8host01Node03, O=IBM, C=US	CN=was8host01, OU=Root Certificate, OU=was8host01Cell01, OU=was8host01CellManager01, O=IBM, C=US	32492542466710	Valid from Oct 20, 2011 to Oct 19, 2012.	
		CN=was8host01, OU=Root Certificate, OU=was8host01Cell01, OU=was8host01CellManager01, O=IBM, C=US	CN=was8host01, OU=Root Certificate, OU=was8host01Cell01, OU=was8host01CellManager01, O=IBM, C=US	577457852324	Valid from Aug 14, 2011 to Aug 10, 2026.	

General Properties

Alias

default

Version

X509 V3

Key size

2048 bits

Serial number

32492542466710

Validity period

Valid from Oct 20, 2011 to Oct 19, 2012.

Issued to

CN=192.168.136.128, OU=was8host01Node02Cell,
OU=was8host01Node03, O=IBM, C=US

Issued by

CN=was8host01, OU=Root Certificate, OU=was8host
OU=was8host01CellManager01, O=IBM, C=US

Fingerprint (SHA digest)

97:A6:3A:19:68:F8:13:13:7E:3E:C3:8D:D5:83:1D:A

- Keystores for WebSphere are managed through the administrative console
 - Creating keystores
 - Requests and imports CA certificates
 - **iKeyman** can also be used

What is a chained certificate?

- A chained certificate is merely a certificate that another certificate signs
- The cell root certificate (sometimes called a mini-CA) signs the node certificate

Chained certificate

- The cell root certificate signs it

key stores and certificates > [NodeDefaultKeyStore](#) > Personal certificates

Preferences

Create Delete Receive from a certificate authority... Replace... Extract... Import... Export... Revoke... Renew

Select	Alias	Issued To	Issued By	Serial Number	Expiration
<input type="checkbox"/>	default	CN=was8host01.localdomain, OU=was8host01Node01Cell, OU=was8host01Node01, O=IBM, C=US	CN=was8host01.localdomain, OU=Root Certificate, OU=was8host01Cell01, OU=was8host01CellManager01, O=IBM, C=US	4100991604240	Valid from Nov 16, 2011 to Nov 13, 2021.
<input type="checkbox"/>		CN=was8host01.localdomain, OU=Root Certificate, OU=was8host01Cell01, OU=was8host01CellManager01, O=IBM, C=US	CN=was8host01.localdomain, OU=Root Certificate, OU=was8host01Cell01, OU=was8host01CellManager01, O=IBM, C=US	3264426922974	Valid from Nov 16, 2011 to Nov 12, 2026.

Node certificate

Cell root certificate

Notice: these two are the same



Expiration manager scheduling

The screenshot shows the 'Expiration manager scheduling' configuration page. A yellow box labeled 'Manual start' has an arrow pointing to the 'Start now' button. Another yellow box labeled 'Schedule' has an arrow pointing to the 'Repeat interval' field, which is set to 4 weeks. The 'Check by calendar' radio button is selected, and the 'Weekday' is set to Sunday. The 'Expiration notification threshold' is set to 60 days. The 'Next start date' is Sunday, March 13, 2011 9:30 PM. The 'Expiration check notification' is set to MessageLog. The 'Automatically replace expiring self-signed and chained certificates' and 'Delete expiring certificates and signers after replacement' checkboxes are both checked.

Start now ← Manual start

General Properties

* Expiration notification threshold
60 days

☒ Enable checking

Expiration checking

Scheduled time of day to check for expired certificates
21 : 30 ☐ A.M. ☐ P.M. ☒ 24-hour

☒ Check by calendar

Weekday * Repeat interval weeks ← Schedule

☐ Check by number of days

* Repeat interval
 days

Next start date

Expiration check notification

☒ Automatically replace expiring self-signed and chained certificates

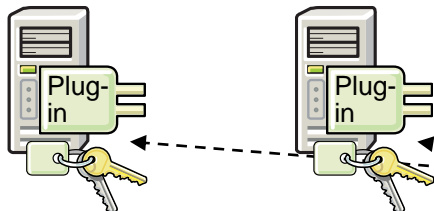
☒ Delete expiring certificates and signers after replacement

- The expiration manager can be run manually or through a schedule
- Running manually can be useful since you actively monitor the log file and thus generate a list of certificates that are going to expire soon

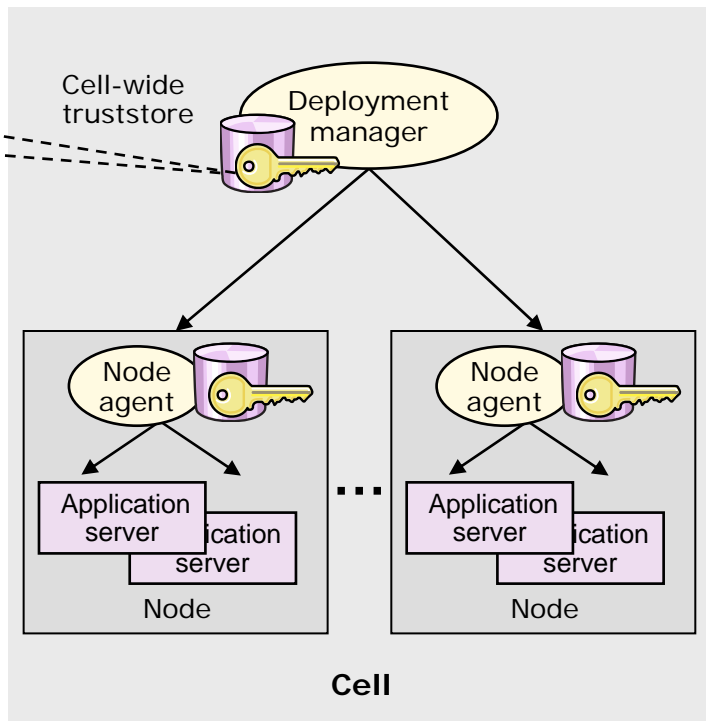
Keys for web servers

Web server

Web server

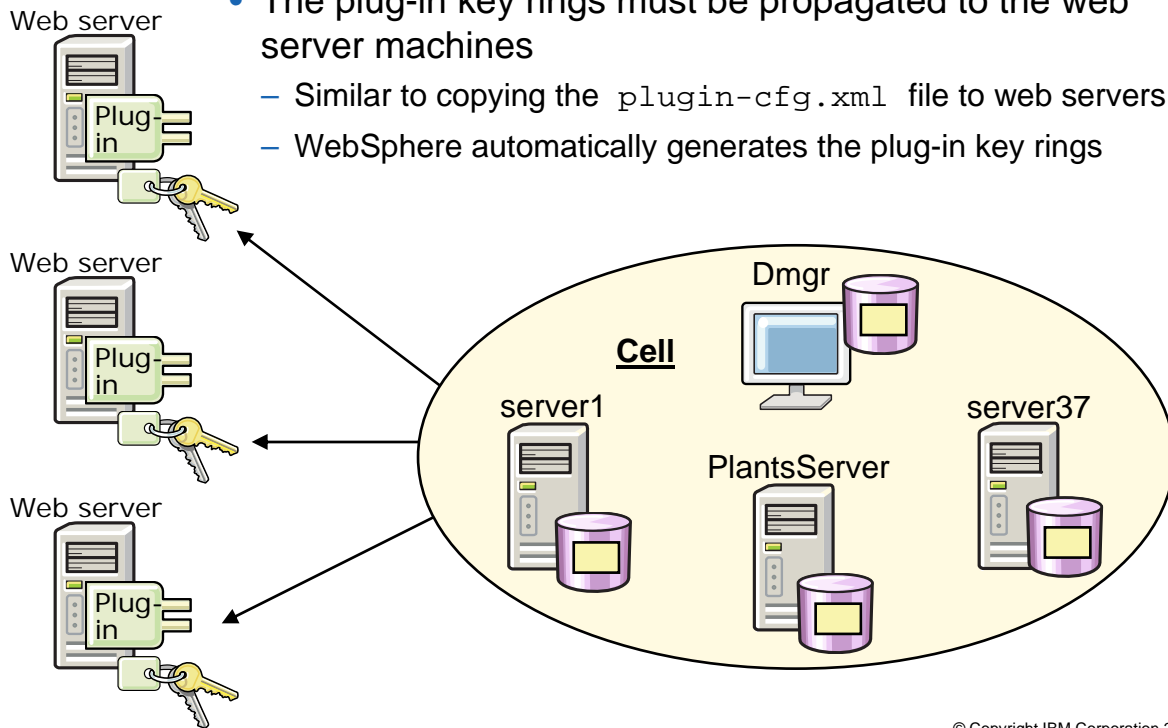


- Single keystores are generated for each unmanaged web server node:
 - Contains signed personal certificate for the unmanaged node (which the cell root certificate signs)
 - Includes the cell root signer certificate, allowing the plug-ins to communicate with the nodes securely
- **Important:** These key rings must be distributed to the web servers



Web server plug-in keystores propagation

- The plug-in key rings must be propagated to the web server machines
 - Similar to copying the `plugin-cfg.xml` file to web servers
 - WebSphere automatically generates the plug-in key rings



IBM HTTP Server key ring propagation

Web servers

[Web servers](#) > [webserver1](#) > **Plug-in properties**

Use this page to configure a web server plug-in. The plug-in passes HTTP requests from servers.

Runtime **Configuration**

Plug-in properties

☐ Ignore DNS failures during Web server startup

* Refresh configuration interval
60 seconds

Repository copy of Web server plug-in files:

* Plug-in configuration file name
plugin-cfg.xml [View](#)

☒ Automatically generate the plug-in configuration file

☒ Automatically propagate plug-in configuration file

* Plug-in key store file name
plugin-key.kdb

[Manage keys and certificates](#)
[Copy to Web server key store directory](#)

- Web server keystores are automatically generated
 - Can be managed from the administrative console
- The keystore for IBM HTTP Server servers can be remotely propagated



Security auditing



Security auditing

- The security auditing subsystem was introduced in WebSphere Application Server Version 7 and has two primary goals:
 - Confirm the effectiveness and integrity of the existing security configuration
 - Identify areas where improvement to the security configuration might be needed

- The security auditing subsystem can capture the following types of auditable events:
 - Authentication
 - Authorization
 - Principal and credential mapping
 - Audit policy management
 - User registry and identity management
 - Delegation
 - Administrative configuration management

Enabling security auditing

- Configuration is necessary before auditing can be enabled
 - Create an audit-specific set of console users or groups and map to Auditor role
 - Define notification mechanism (log file, email)
 - Enable monitoring
- Enabling auditing

Security auditing

Security auditing

Security auditing provides a means to gather and store auditable event records to help assure the integrity of the business computing environment.

General Properties

☒ **Enable security auditing**

Audit subsystem failure action
No warning

Primary auditor user name
wasadmin

☐ Enable verbose auditing

Apply Reset

Related Items

- [Event type filters](#)
- [Audit service provider](#)
- [Audit event factory configuration](#)
- [Audit encryption key stores and certificates](#)
- [Audit record encryption configuration](#)
- [Audit record signing configuration](#)
- [Audit monitor](#)

Viewing audit data

- Audit data can be viewed as:
 - Text
 - An HTML report (through wsadmin)

```
***** Start Display Current Environment *****
WebSphere Platform 7.0.0.0 [ND 7.0.0.0 r0835.03] running with process name was7host01Cell1
Host Operating System is Windows XP, version 5.1 build 2600 Service Pack 3
Java version = J2RE 1.6.0 IBM J9 2.4 Windows XP x86-32 jvmwi3260-20080816_22093 (JIT enab
J9VM - 20080816_022093_1hdsMr
JIT - r9_20080721_1330ifx2
GC - 20080724_AA, Java Compiler = j9jit24, Java VM name = IBM J9 VM
was.install.root = C:\Program Files\IBM\WebSphere\AppServer
user.install.root = C:\Program Files\IBM\WebSphere\AppServer\profiles\DmgrProfile
Java Home = C:\Program Files\IBM\WebSphere\AppServer\java\jre
ws.ext.dirs = C:\Program Files\IBM\WebSphere\AppServer\java\lib;C:\Program Files\IBM\WebS
Classpath = C:\Program Files\IBM\WebSphere\AppServer\profiles\DmgrProfile\properties;C:\P
Java Library path = C:\Program Files\IBM\WebSphere\AppServer\java\jre\bin;.;C:\Program Fi
Current trace specification = *-info
***** End Display Current Environment *****
Seq = 0 | Event Type = SECURITY_RESOURCE_ACCESS | Outcome = SUCCESSFUL | OutcomeReason = :
Seq = 1 | Event Type = SECURITY_RESOURCE_ACCESS | Outcome = SUCCESSFUL | OutcomeReason = :
Seq = 2 | Event Type = SECURITY_RESOURCE_ACCESS | Outcome = SUCCESSFUL | OutcomeReason = :
Seq = 3 | Event Type = SECURITY_RESOURCE_ACCESS | Outcome = SUCCESSFUL | OutcomeReason = :
Seq = 4 | Event Type = SECURITY_RESOURCE_ACCESS | Outcome = SUCCESSFUL | OutcomeReason = :
Seq = 5 | Event Type = SECURITY_RESOURCE_ACCESS | Outcome = SUCCESSFUL | OutcomeReason = :
Seq = 6 | Event Type = SECURITY_RESOURCE_ACCESS | Outcome = SUCCESSFUL | OutcomeReason = :
```

Audit Records - Windows Internet Explorer

C:\basicAuditReport.html

Audit Records

ResourceName=placeReport	ResourceType=SM_MBEAN	ResourceUniquel=0
243	SECURITY_RESOURCE_ACCESS	SUCCESS
CreationTime=Wed Jan 28 15:21:21 EST 2009	Action=preinvoke MBean	ProgName=StatusCache
RemoteAddr=192.168.58.135	RemotePort=3570	RemoteHost=192.168.58.135
ResourceName=placeReport	ResourceType=SM_MBEAN	ResourceUniquel=0
244	SECURITY_RESOURCE_ACCESS	SUCCESS
CreationTime=Wed Jan 28 15:22:01 EST 2009	Action=preinvoke MBean	ProgName=Server (module)
RemoteAddr=192.168.58.135	RemotePort=3570	RemoteHost=192.168.58.135
ResourceName=getProcessType	ResourceType=SM_MBEAN	ResourceUniquel=0
245	SECURITY_RESOURCE_ACCESS	SUCCESS
CreationTime=Wed Jan 28 15:22:14 EST 2009	Action=preinvoke MBean	ProgName=StatusCache
RemoteAddr=192.168.58.135	RemotePort=3570	RemoteHost=192.168.58.135
ResourceName=placeReport	ResourceType=SM_MBEAN	ResourceUniquel=0

Securing audit records

- Access to audit configurations is restricted
 - To change audit settings, Auditor access is required (Administrator access is not sufficient)
- Audit data can be digitally protected
 - Can be digitally signed
 - Can be encrypted with a separate audit certificate

Security auditing

[Security auditing](#) > **Audit record signing configuration**

Signing audit records provides a means of tamper-proofing audit records.

General Properties

☒ Enable signing

Managed keystore containing the signing certificate

CellDefaultKeyStore ((cell):was8host01.Cell01)

☐ Certificate in keystore

Certificate alias

default

Security auditing

[Security auditing](#) > **Audit record encryption configuration**

By encrypting the audit records, only a user given the Auditor role can view the audit records.

General Properties

☒ Enable encryption

The Audit keystore containing the encryption certificate.

AuditKeyStore New...

☐ Certificate in keystore

Certificate alias

auditcert

Unit summary

Having completed this unit, you should be able to:

- Explain basic security concepts
- Describe the WebSphere Application Security architecture
- Describe enhancements to certificate management
- Configure fine-grained administrative security
- Configure application security
- Describe SSL concepts and configuration
- Describe support for multiple security domains
- Describe auditing features and functions
- Describe support for Java Platform, Enterprise Edition 6 (Java EE 6) security annotations

Checkpoint questions

1. Which type of security restricts access to the application?
 - A. Administrative security
 - B. Application security
 - C. Java 2 security
 - D. File system security
2. Which type of security restricts access to the operating system?
 - A. Administrative security
 - B. Application security
 - C. Java 2 security
 - D. File system security
3. Which type of security restricts access to the console?
 - A. Administrative security
 - B. Application security
 - C. Java 2 security
 - D. File system security

Checkpoint answers

1. Which type of security restricts access to the application?
B. Application security
2. Which type of security restricts access to the operating system?
C. Java 2 security
3. Which type of security restricts access to the console?
A. Administrative security



Exercise 12



Configuring WebSphere security



Exercise objectives

After completing this exercise, you should be able to:

- Enable WebSphere security
- Configure administrative security by configuring access to administrative functions
- Configure fine-grained administrative security



Exercise 13



Configuring application security



Exercise objectives

After completing this exercise, you should be able to:

- Define Java EE security roles
- Define access for resources in an application
- Enable and verify application security



Exercise 14

Configuring SSL for WebSphere



Exercise objectives

After completing this exercise, you should be able to:

- Define the certificate life span of a profile
- Use the administrative console to find and view certificates within the cell
- Configure and run the certificate expiration service
- Propagate the generated plug-in keystore out to the plug-in
- Create a keystore for a web server
- Generate a self-signed key
- Configure IBM HTTP Server to load and use HTTPS