Session: 13

Security

Objectives

- Describe enterprise application security
- Explain how to implement security at various levels in an application
- Explain how roles, users, and user groups are defined in an application
- Define authorization and authentication mechanisms used in enterprise applications
- Explain JASS architecture and its services
- Explain how to secure application clients

Introduction 1-2

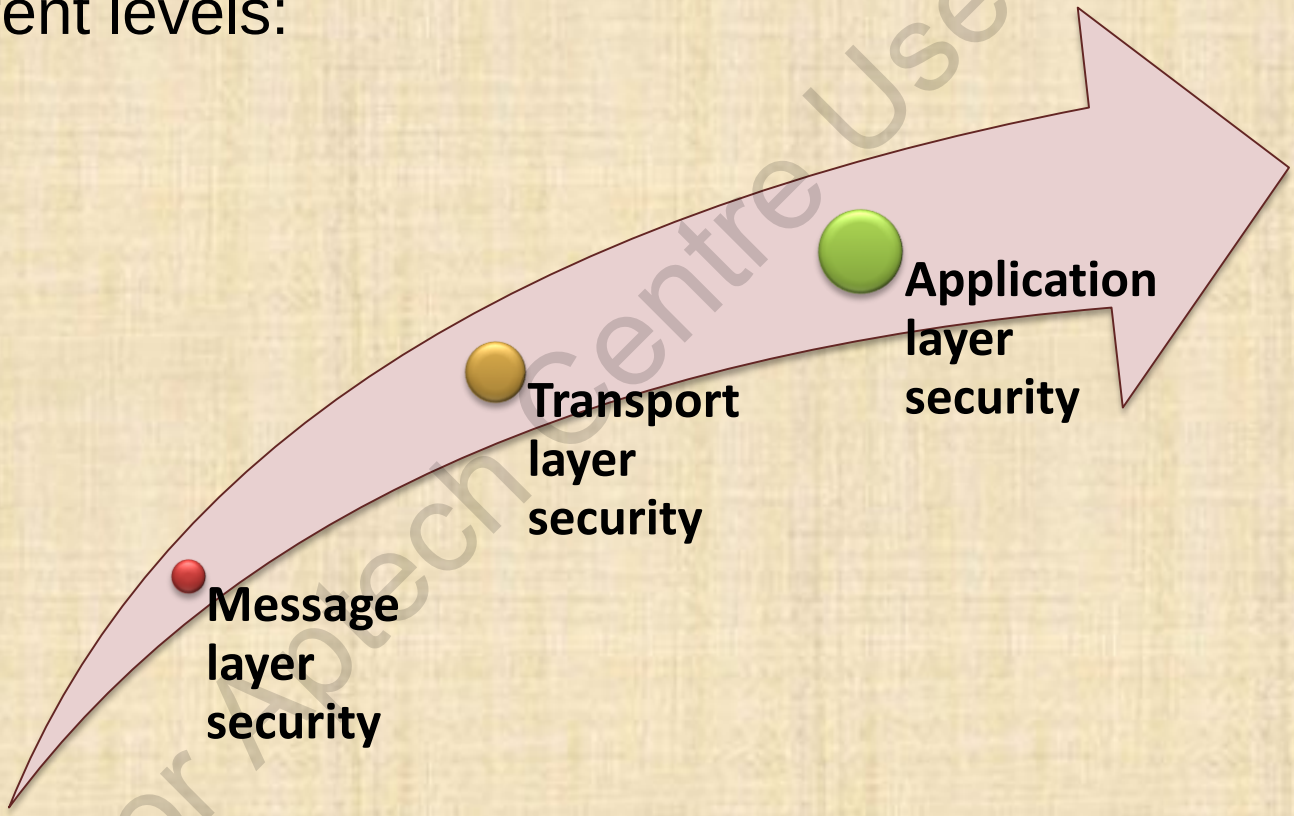
- **When an enterprise application is accessed through the Internet or any other open network:**
  - The users accessing the application components must be appropriately authenticated and authorized, before they can access the services from the application.
- **All the application components are deployed on the application server:**
  - Are logically managed through the container who is responsible for providing security services for the components deployed in it.

Introduction 2-2

- Security is applied to the application components through:
  - Container that implements the security policy defined in the application code.
  - Classes and interfaces to implement the security policy programmatically.

Implementing Security at Various Levels

- Enterprise application security is implemented at three different levels:



Application Layer Security

- Implemented by the container.
- Firewalls can be used at the application layer level to implement the security requirements.
- Defined both declaratively and programmatically.
- Declarative security definition is through deployment descriptors and annotations.
- Programmatic security definition is through interfaces such as `EJBContext` provided by Java EE.

Transport Layer Security 1-2

- Refers to the security mechanisms implemented while the application data is transmitted through the network.
- Based on Point to Point security mechanism ensuring message integrity, authentication, and confidentiality of data transmitted.
- Uses cryptographic techniques.

Transport Layer Security 2-2

- Following are the steps involved in implementing transport layer security:

Client and server agree upon the cryptographic algorithm

The secret key used for communication is exchanged using public key cryptography and certificate based authentication

The agreed upon secret key is used for exchange of data on the network

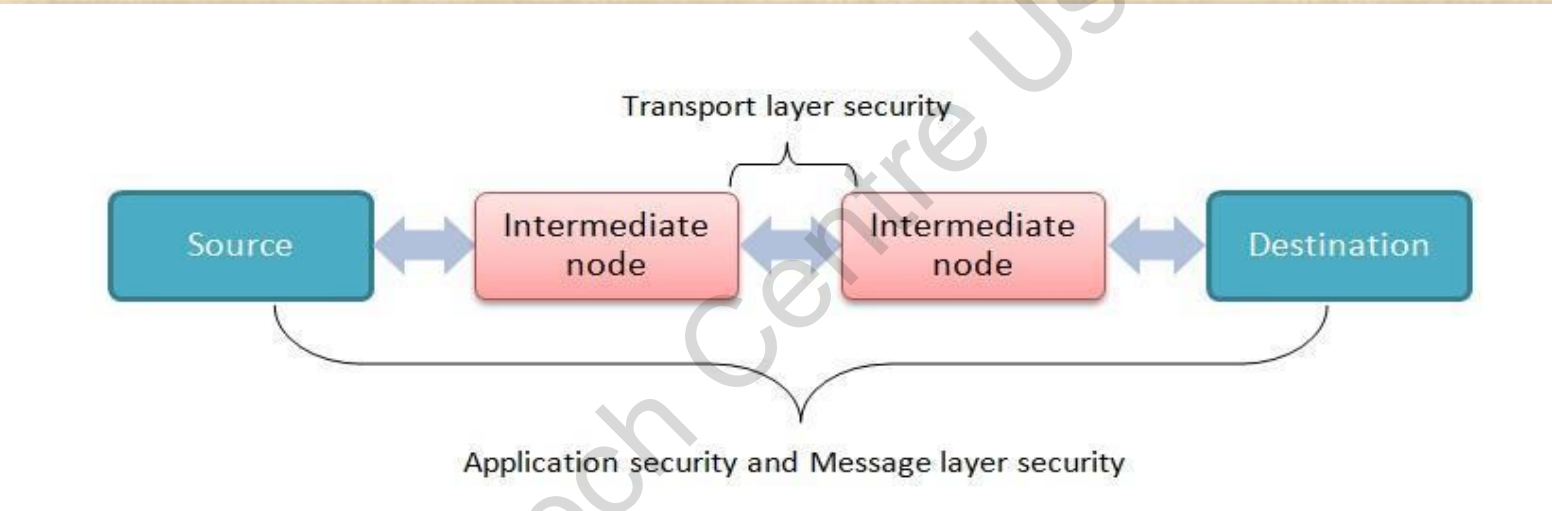
Transport layer security is unaware of the contents of message being transmitted.

Message Layer Security 1-2

- Security information is bundled along with Simple Object Access Protocol (SOAP) message.
- Security information travels to the destination along with the message.
- Is an end-to-end security.
- When the message with encrypted information is transmitted from the sender, it passes through several intermediate nodes and reaches the destination.
- Encrypted SOAP message is only decrypted by the receiver.

Message Layer Security 2-2

- Following figure demonstrates the implementation of security at various levels:



Unlike transport layer security, message layer security can be selectively applied on a part of the message.

Characteristics of Security Mechanisms

- Prevent unauthorized access to application data and components.
- Identity of an application user should be associated with each action performed on the enterprise application.
- Users cannot deny the operations performed.
- Protects the application from service failures such as server crash, network failure, and other interruptions.

Features of Application Security

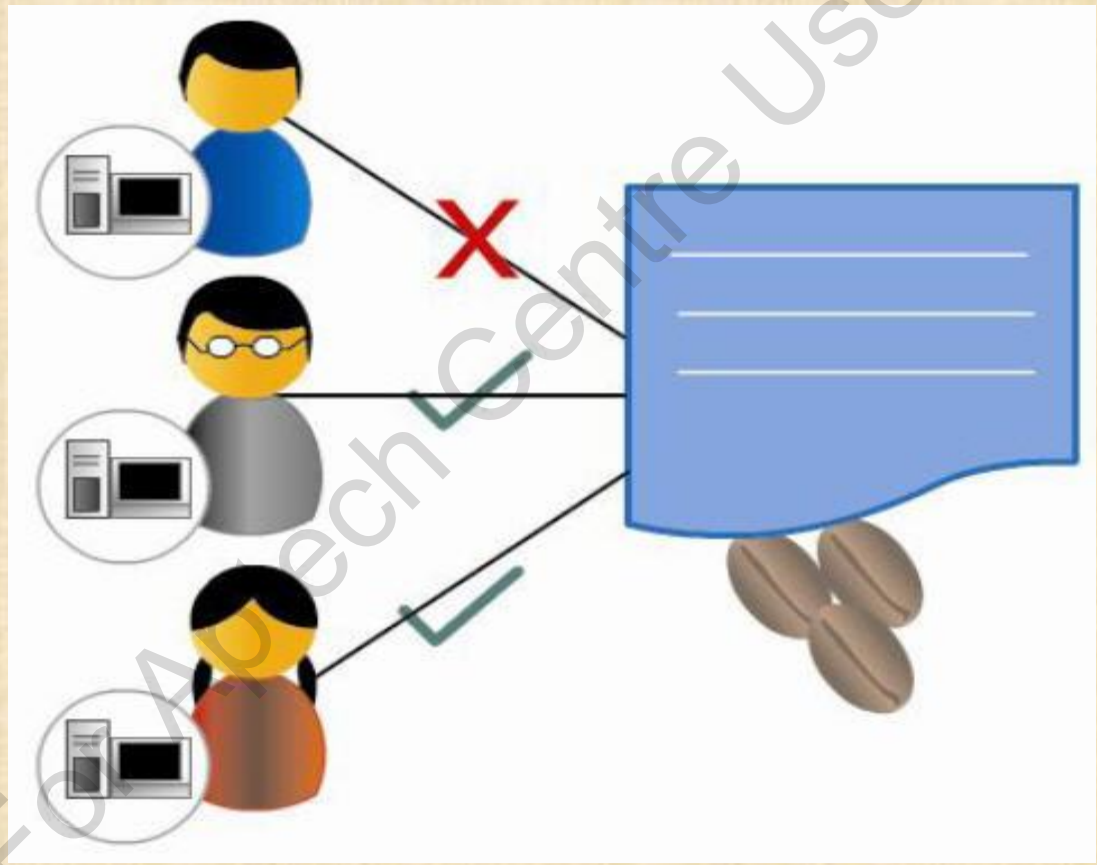
- Following features should be implemented for application security to reduce the risk of security threats to the application:

Authentication 1-2

- Is the process by which one entity in an interaction determines the identity of the other.
- In Java EE environment:
  - The EJB server determines the identity of all types of clients so, that it can determine the level of access to be granted.
  - The client may also want to authenticate the server, to ensure that it is interacting with the correct server.
- The most common form of authentication involves the use of username and password.
- The use of digital certificates offers a stronger form of authentication.

Authentication 2-2

- Following figure depicts authentication:

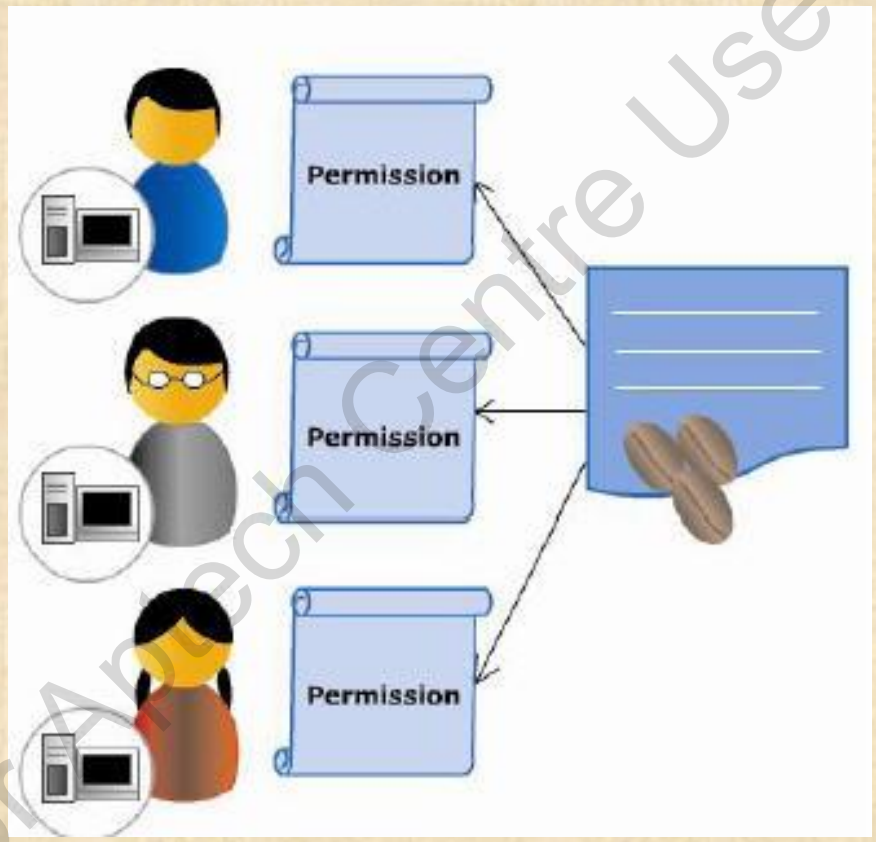


Authorization 1-2

- In an enterprise application, client authentication is usually followed by authorization.
- **Authentication** - Ensures only valid users get access to the application.
- **Authorization** - Controls what the authenticated user is allowed to do after he/she is granted access.

Authorization 2-2

- Following figure depicts authorization:



Data Integrity

- Data integrity is a characteristic which requires that the information is not modified by unwarranted users.
- Applications implement various checks on the data such as Cyclic Redundancy Checksum (CRC) codes and so on to detect whether the information is modified by any third party users.

Confidentiality and Non-repudiation

- Confidentiality implies secrecy, where by the security system allows access of data only to authorized users of the application.
- Non-repudiation is the security mechanism that associates the identity of the user with actions performed by them on the application.
- If a user performs a malicious operation on the application, the security mechanism ensures that the user does not deny the operations performed.

Quality of Service and Auditing

- The security mechanism implemented in the application increases the application execution time.
- For instance, when access to a resource requires username and password. The application execution cannot proceed until the user provides the appropriate information.
- Auditing of an application log is done to ensure that the application is performing as expected.

Simple Application Security Implementation 1-3

- Following are the steps to be performed to implement the security requirements of the application:
  - Every user is provided with a unique username.
  - Each username is linked with the account held by the user.
  - Users should be authenticated to access their account.
  - Authentication should be followed by authorizing the user.
  - The operations of checking the account balance and transfer funds are to be implemented in each account.
  - Enterprise beans should be invoked to perform the required operations.

Simple Application Security Implementation 2-3

- Following are the steps involved in a typical application execution with a security mechanism in place:

Request	• Application client or end user initiates an application request. • Application request can access EJB components.
Authentication	• Authenticates the application clients by prompting for the username and password.
URL Authorization	• Credentials provided are used to determine whether the given user is authorized to access resources or not.

Simple Application Security Implementation 3-3

Fulfilling original requests

- Application server sends requests to the security policy defined for the application to determine the resources to be accessed.
- The application request in turn initiates the authentication process to fulfil the request.

Invoking enterprise bean methods

- Application requests serviced through enterprise bean methods.
- EJB Container provides security to the bean methods.

Access Control Lists (ACLs) 1-2

- Permissions represent a right to access a particular resource or to perform some action on an application.
- An administrator:
  - Usually protects resources by creating lists of users and groups that have the permission to access a particular resource.
  - Lists are referred as Access Control Lists (ACLs).
- An ACL file is made up of entries, which contain a set of permissions for a particular resource and a set of users who can access those resources.

Access Control Lists (ACLs) 2-2

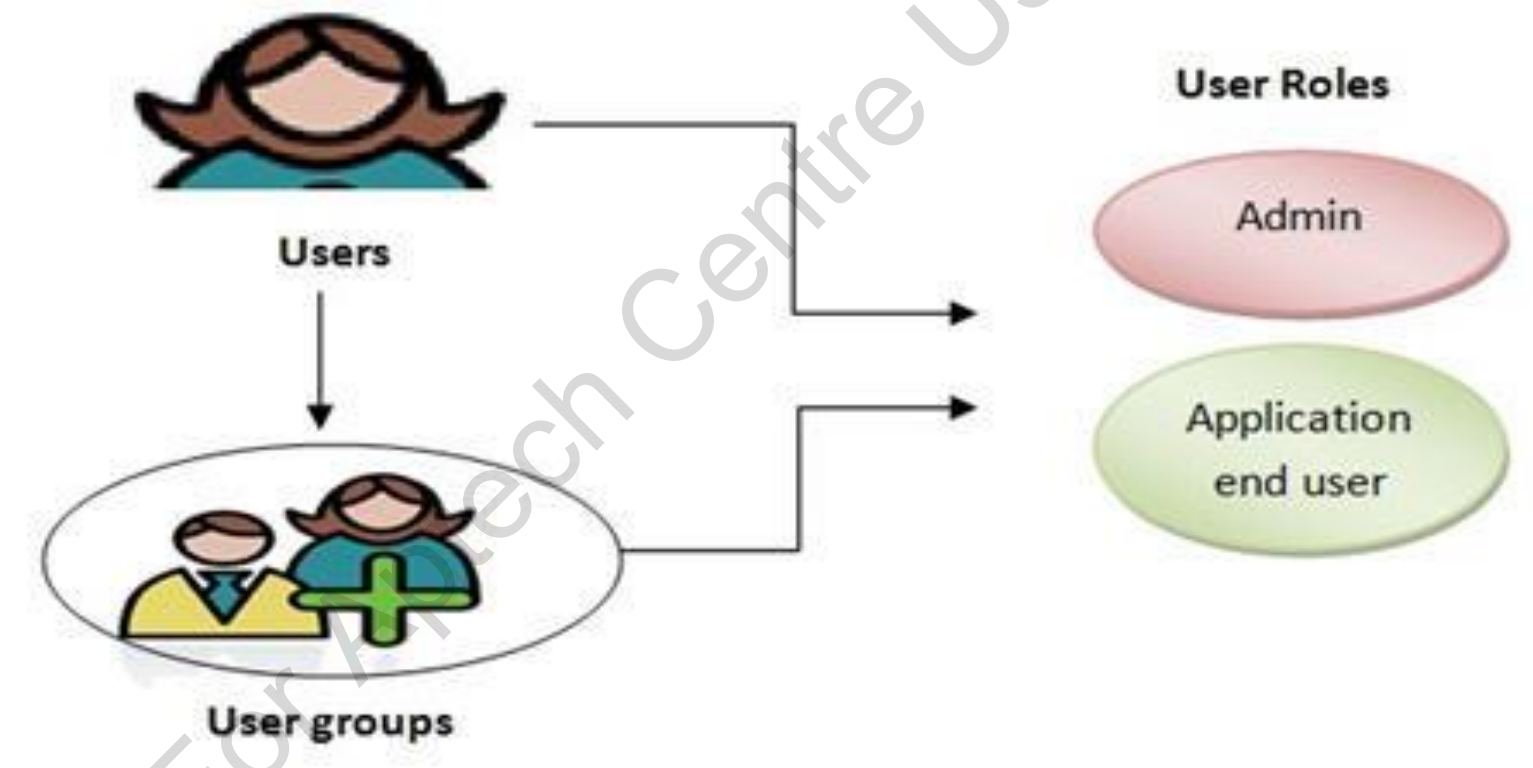
- Following figure shows Access Control Lists:

Users, Groups, Roles, and Realms 1-2

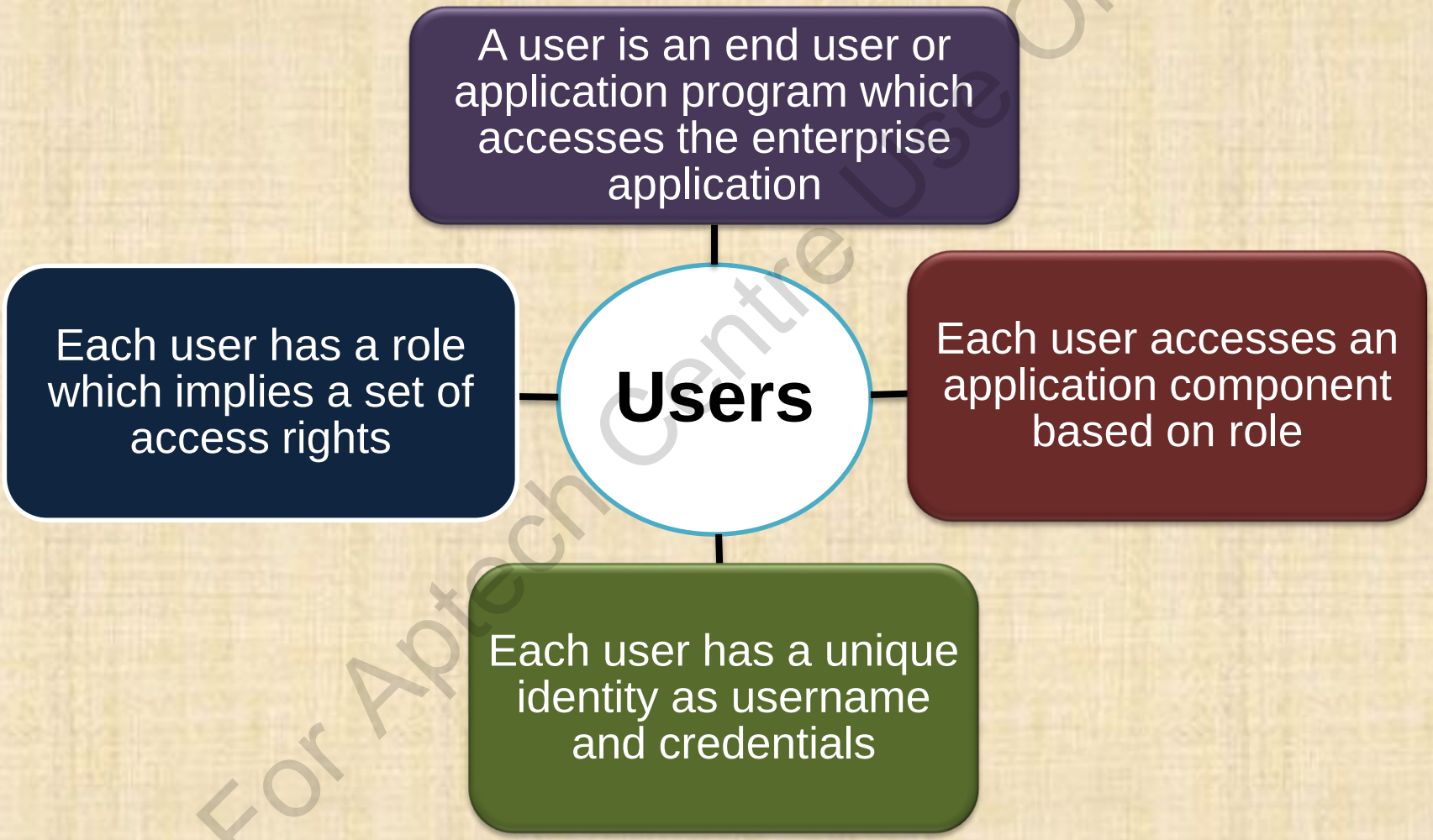
- An application domain has various end users.
- The users are logically grouped into user groups based on some common characteristics.
- Every end user is termed as a user of the application; however, different users may have different roles to play in the application.
- For instance, an employee's role in a bank application is different from the customer's role in the same domain.
- A realm is a single authentication policy that controls a set of users or user groups.
- An admin realm in an application therefore, grants administrative rights required by the application to the admin group.

Users, Groups, Roles, and Realms 2-2

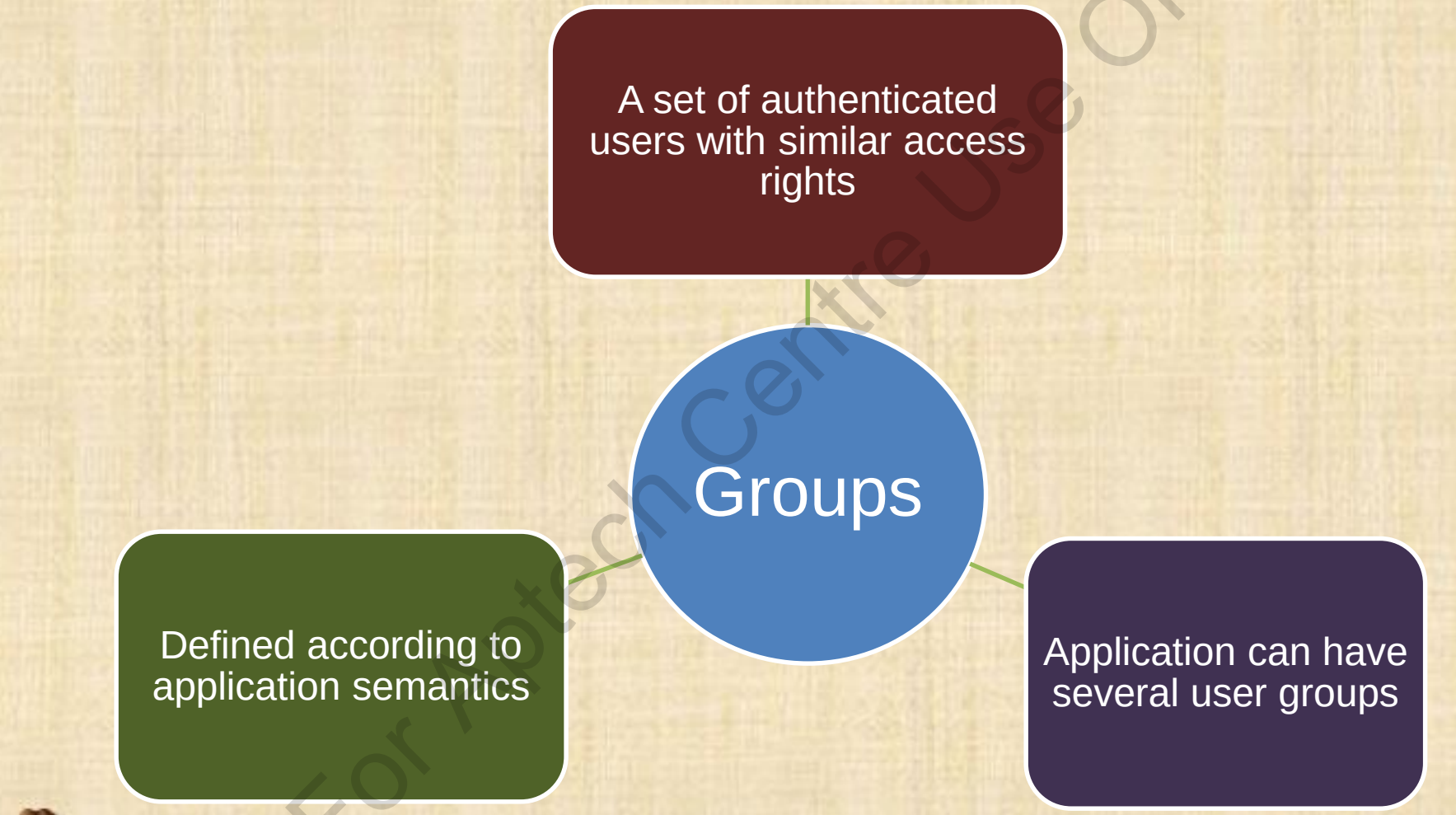
- Following figure shows Users, User groups, Roles, and their association:



Users



Groups

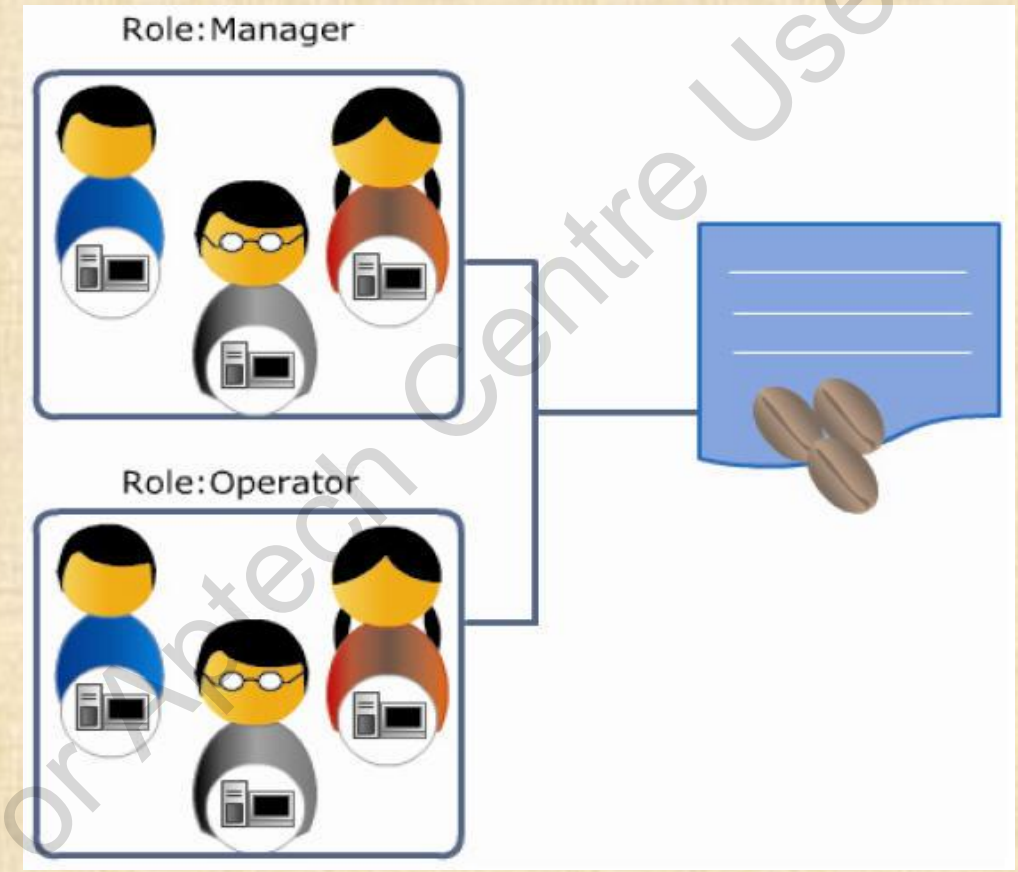


Roles 1-2

- Role in the application domain reflects the set of access rights held by a certain user.
- For instance, all employees in the bank application do not have the right to credit interest into the accounts of customers.
- The application has to define a role with the required privileges assigned to that role so that an employee can perform the task of crediting interest into the accounts of the customers.
- Each role contains a particular set of permissions.

Roles 2-2

- Following figure shows the groups and roles:



Realms 1-2

- Each application has a set of protected resources.
- Each realm is associated with an authentication scheme to access certain protected resources.
- Realm is the set of authorized users who can access those protected resources.
- Java EE supports three default realms:
  - Admin realm
  - Certificate realm
  - File realm

Realms 2-2

Admin realm

- Stores all the credential data in `admin-keyfile`.
- Authenticates information locally stored.

Certificate realm

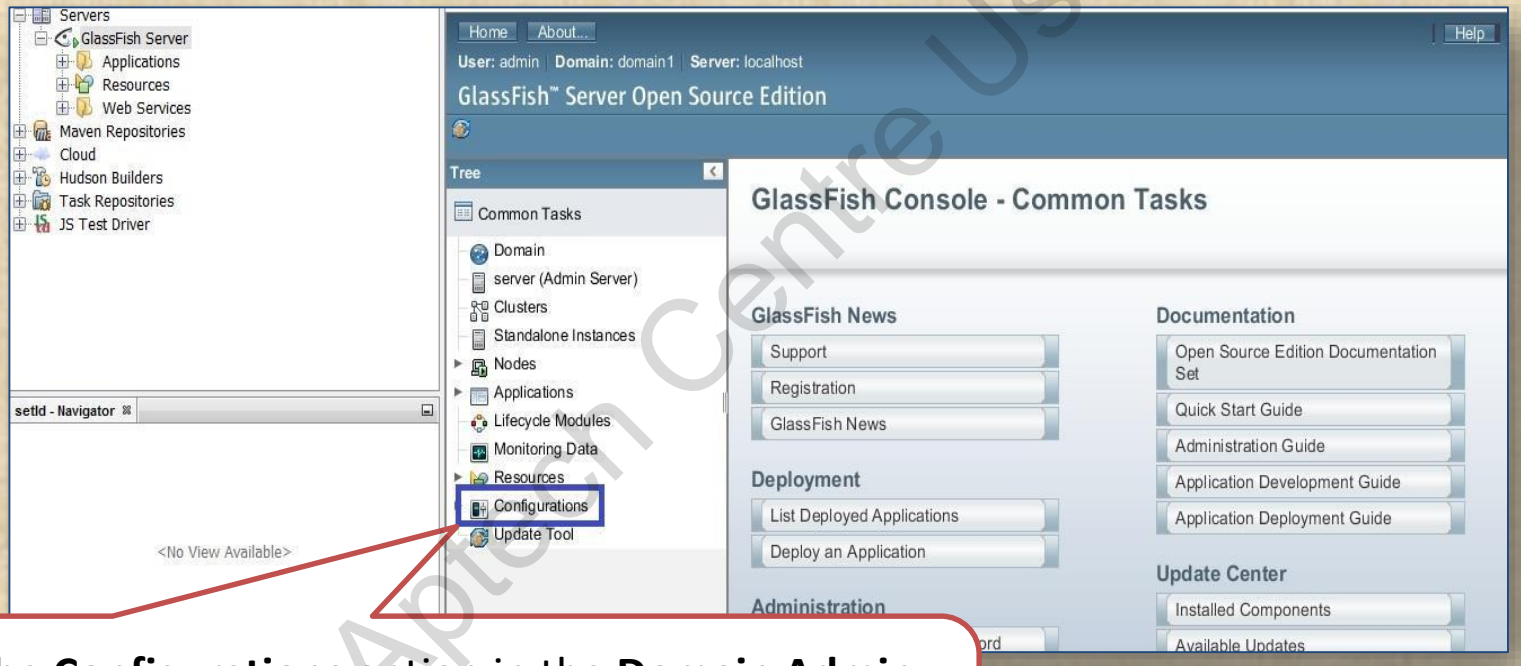
- Stores all the credential data in certificate database.
- Uses X.509 certificates for user authentication.

File realm

- Stores all the user credentials in `keyfile`.
- Can be used for all the enterprise clients but not for Web browser clients and those using HTTPs..

Managing Users and Groups on Glassfish Server 1-6

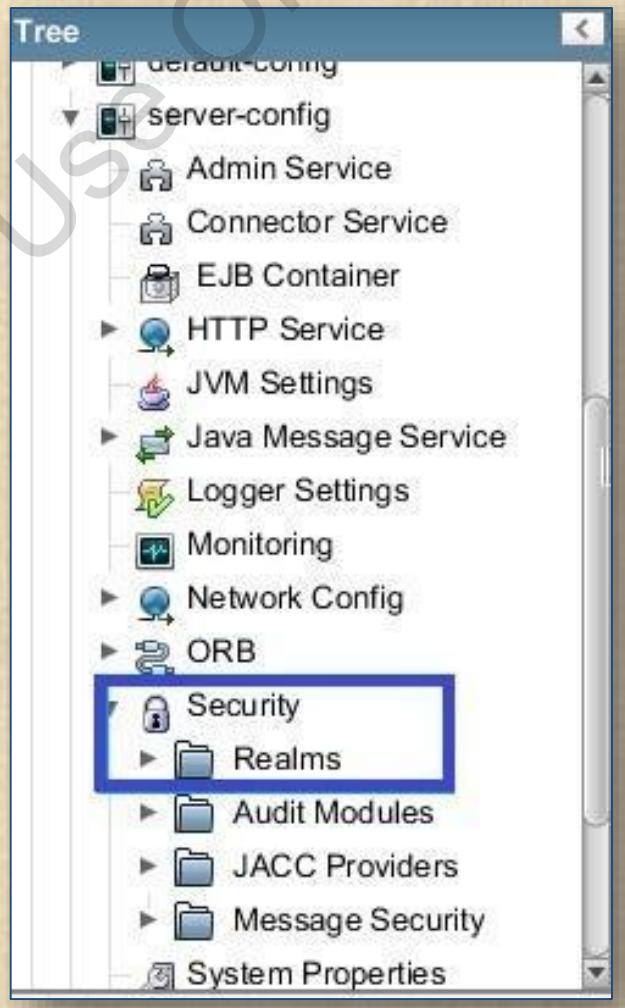
- Following figure shows the **Domain Admin Console** of the Glassfish server:



The **Configurations** option in the **Domain Admin Console** can be used to configure the security features of the application.

Managing Users and Groups on Glassfish Server 2-6

- The given figure shows the hierarchy through which the user can choose the realms for application.

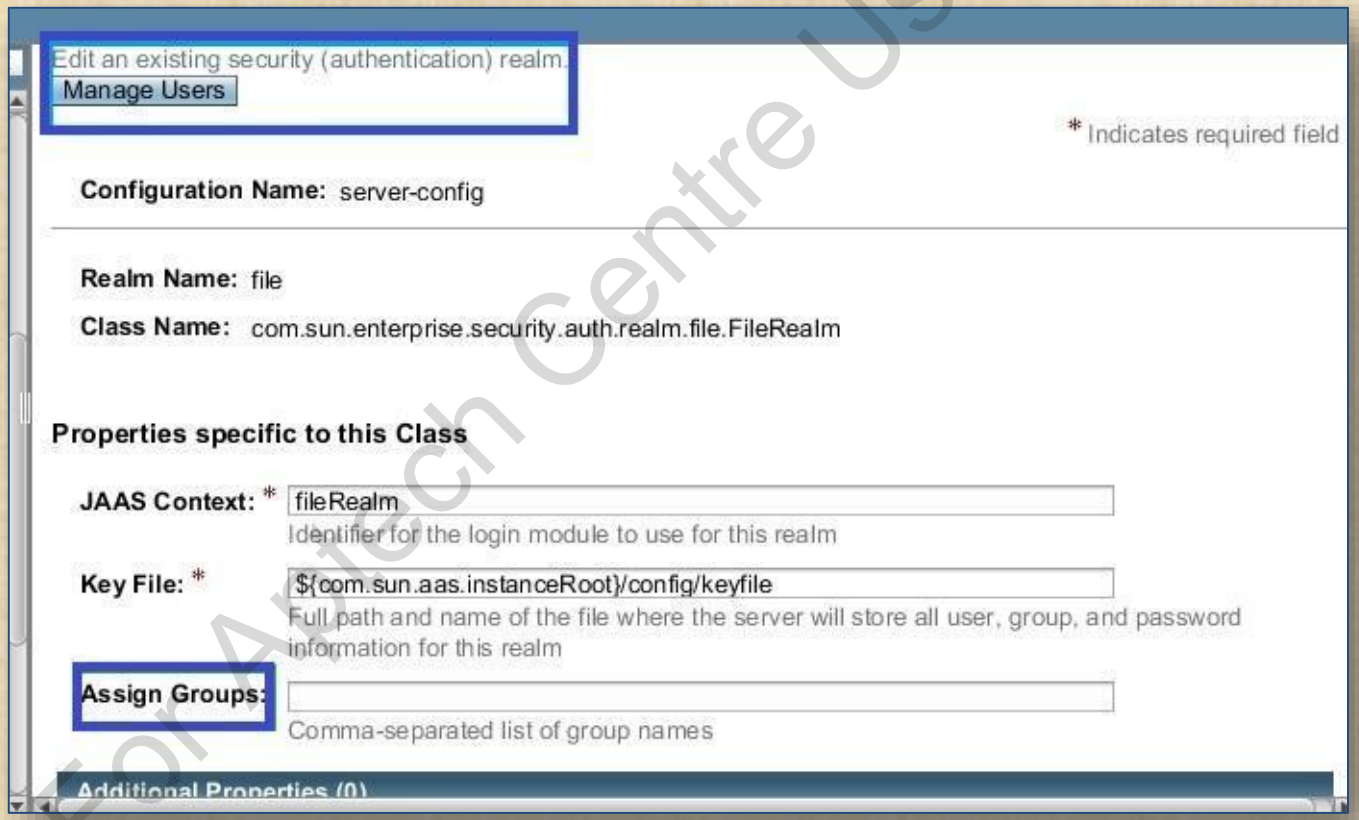


Managing Users and Groups on Glassfish Server 3-6

- Following figure shows the available realms on the server:

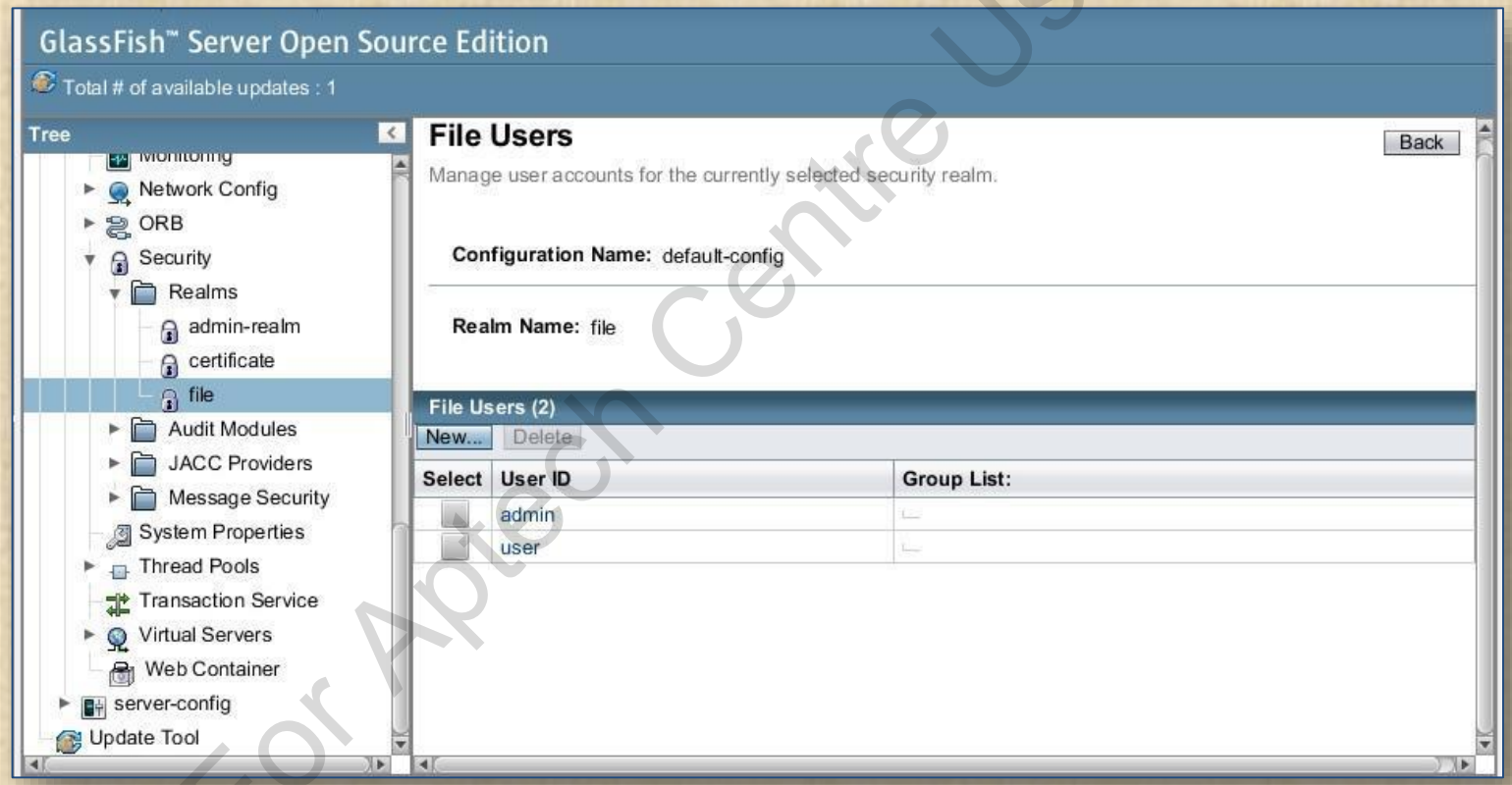
Managing Users and Groups on Glassfish Server 4-6

- Following figure demonstrates how to manage users for the application:



Managing Users and Groups on Glassfish Server 5-6

- Following figure shows on clicking ‘Manage Users’, leads to the interface on the server:



Managing Users and Groups on Glassfish Server 6-6

- Following figure demonstrates how new users can be created for the application:

Creating and Mapping Roles to Users in the Application 1-2

- Following are the annotations which can be used to define roles and their access rights in the application:

`@DeclareRoles`

`@RolesAllowed`

`@PermitAll`

`@DenyAll`

Creating and Mapping Roles to Users in the Application 2-2

- Following code snippet shows the usage of `@DeclareRoles` and `@RolesAllowed`:

```
import javax.annotation.security.DeclareRoles;
import javax.annotation.security.RolesAllowed;
...
 @DeclareRoles({"VALUATOR", "MANAGER"})
 @Stateless public class LoanApprovalBean  {
 @Resource SessionContext ctx;
 @RolesAllowed("VALUATOR")
 public void reviewPropertyValue(PropertyInfo info) {
    . . .
 }
 @RolesAllowed("MANAGER")
 public void ApproveLoan(PropertyInfo info) {
      ...
 }
...
}
```


Establishing a Secure SSL Connection

- Secure Sockets Layer (SSL) is used at transport layer for secure communication.
- Uses cryptographic techniques.
- Implements point-to-point security.
- Implements three important characteristics of security mechanism:
  - Authentication
  - Confidentiality
  - Integrity

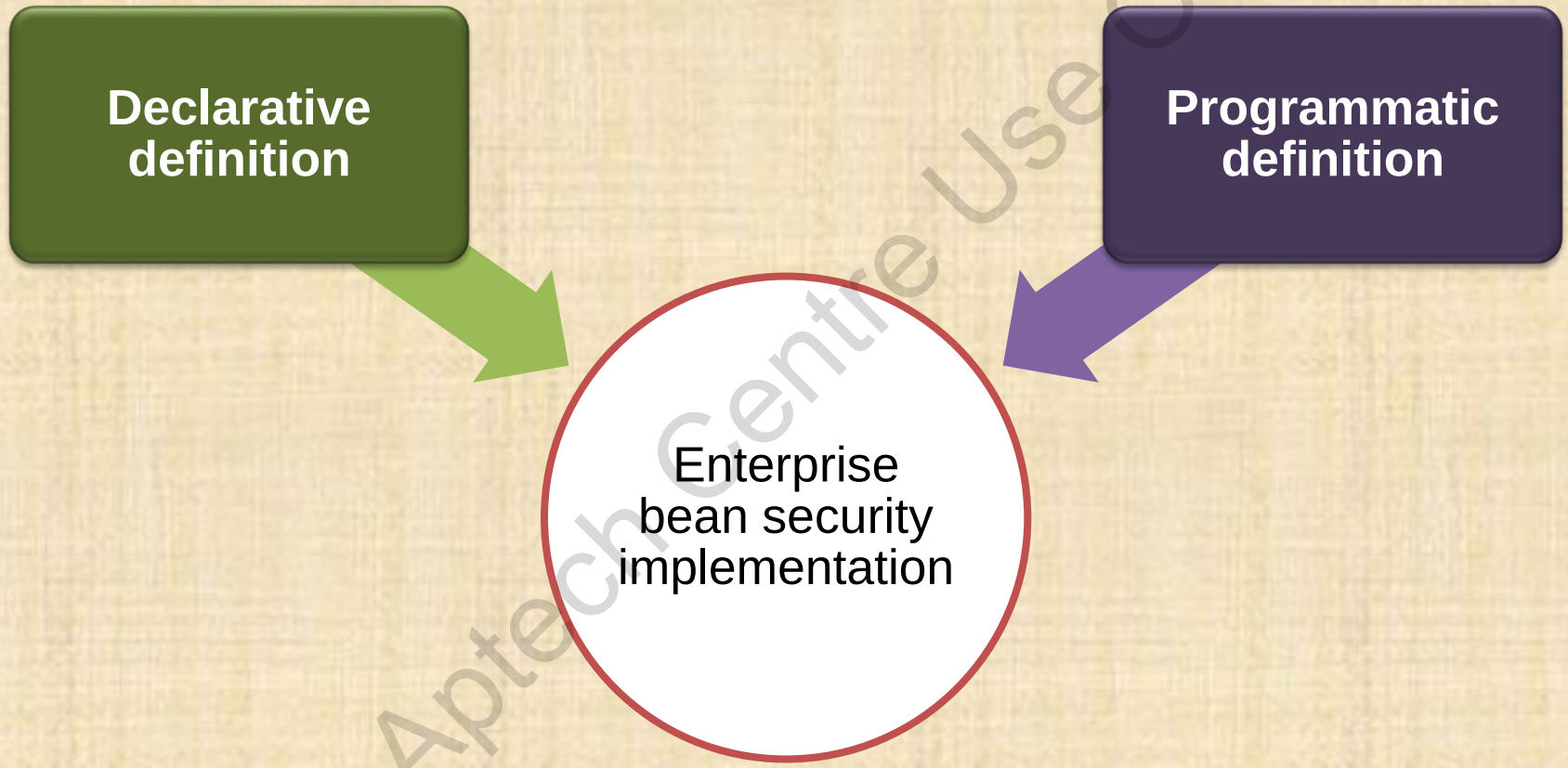
Security Tasks in Enterprise Applications 1-2

- Application security is implemented by:
  - **System administrators** - responsible for creating roles and users.
  - **Application developers** - provides access rights to different roles using annotations or deployment descriptors.
  - **Application deployers** - responsible for deploying the application on the server according to the security specifications provided in the deployment descriptor.
  - **Bean providers** - supports the security mechanisms required by the application.

Security Tasks in Enterprise Applications 2-2

- Following are the tasks that must be performed as part of security implementation for applications:
  1. Creating a database of users who will be accessing the application.
  2. Defining relevant user groups according to the application context.
  3. Assigning the users to appropriate groups.
  4. Propagating the user identity across all the application components.
  5. Configuring the application server with appropriate user and role mappings.
  6. Annotating the classes appropriately to declare roles and defining the access to be granted to different roles.

Securing Enterprise Beans



Securing an Enterprise Bean Method Declaratively

- The application deployer defines the security features on the application server based on the deployment descriptor and annotations.
- The deployer defines the users, user groups, and their respective roles on the application server.

Securing an Enterprise Bean Method Programmatically

- In this method of specifying the security mechanism, the developer uses the security APIs and methods to define the security mechanisms.

Accessing an Enterprise Bean Caller's Security Context 1-2

- `javax.ejb.EJBContext` provides methods to access security information about the user or entity who is invoking the enterprise bean method.
- Following are the methods provided by the `EJBContext` interface:
  - `getCallerPrincipal()`
  - `isCallerInRole()`

Accessing an Enterprise Bean Caller's Security Context 2-2

- Following code snippet demonstrates the usage of `isCallerInRole()` method:

```
…..

@Resource Session Context X
if(X.isCallerRole(admin)==true)
{
   System.out.println("Admin right assigned");
}
else
   System.out.println(" No Admin rights");
. . .
```


Security Mechanisms Provided by Java EE 1-2

Java Generic Security Services (Java-GSS API)	• Used for implementing security mechanisms during communication over the network. • Uses token based API for exchanging messages securely.
Java Cryptography Extension (JCE)	• Used to implement cryptographic structures within the application. • Define implementations of Message Authentication Code (MAC) algorithms, key generation, and so on.
Java Secure Sockets Extension (JSSE)	• Provides Java version of SSL and TLS implementation. • Provides encryption, server authentication, message integrity, and so on.

Security Mechanisms Provided by Java EE 2-2

Simple Authentication and Security Layer (SASL)

- Is an Internet standard which specifies a protocol for authentication and exchange of authentication data between the client and server applications.

Java Authentication and Authorization Service (JAAS)

- Set of APIs to define authentication and authorization mechanisms.
- Provides a pluggable and extensible framework for developers.

Java Authentication and Authorization Service (JAAS)

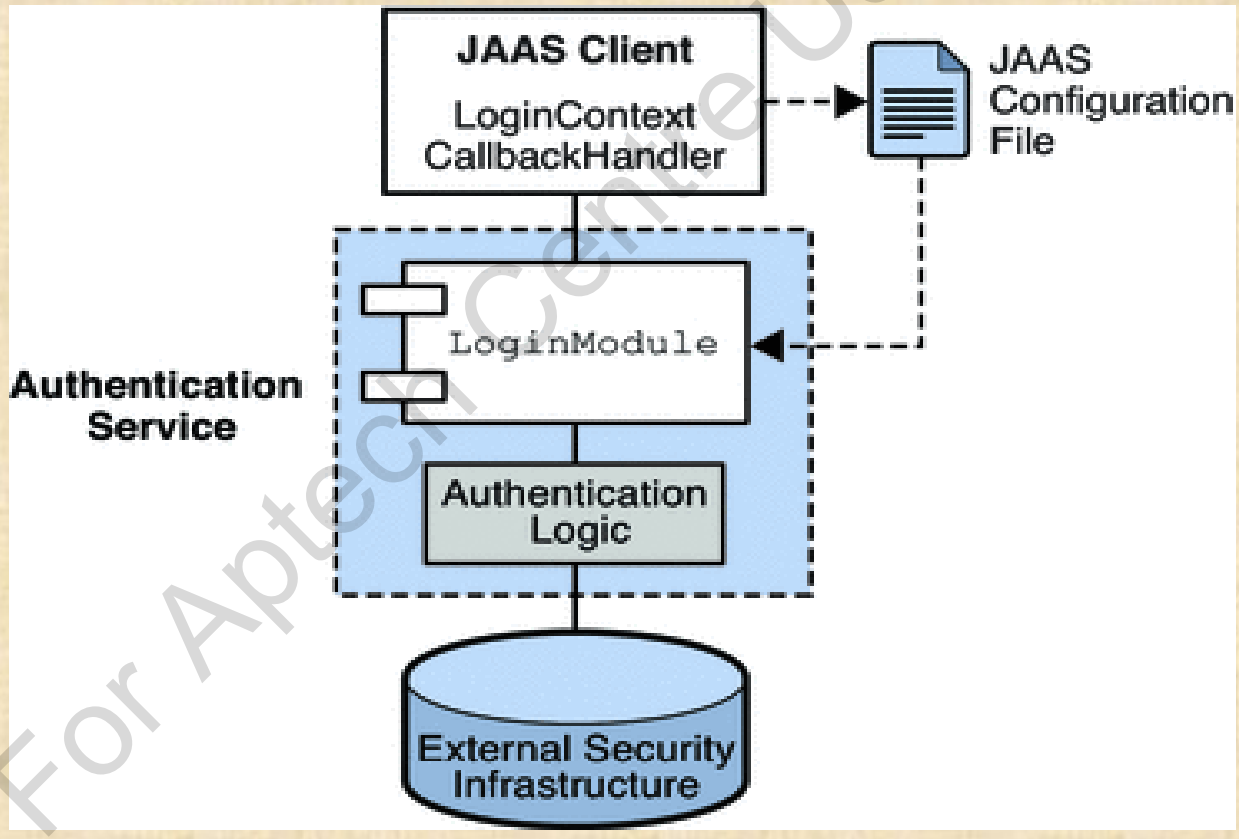
- JAAS implements Pluggable Authentication Model (PAM) framework.
- JAAS provides the following classes and interfaces to be used by developers for implementing security mechanisms.
- Following are the components of the core class library:
  - `LoginModule`
  - `LoginContext`
  - `Subject`
  - `Principal`
- Other classes include:
  - `CallBackHandler`
  - `Credentials`

JAAS Authentication 1-2

- JAAS authentication process involves the following steps:
  1. Create a `LoginContext`, the client application accesses the authentication mechanism through an instance of `LoginContext`.
  2. The `LoginContext` module accesses the `LoginModule`, which is defined in the configuration file.
  3. The authentication is performed through the `LoginModule`.
  4. In the authentication process, a `CallBackHandler` is used to communicate with the client and acquire authentication information such as username, password, and so on.
  5. If the authentication process fails or login process was unsuccessful, a `LoginException` is thrown.
  6. `LoginContext` is used to logout from the session.

JAAS Authentication 2-2

- Following figure shows the flow of control during the JAAS authentication process:



JAAS Authorization 1-2

- Uses a policy configuration file defined for the application.
- An authenticated user trying to access a protected resource is a **`Subject`**.
- Each **`Subject`** is associated with a `Permission` instance.
- Subject instances are managed by `SecurityManager`.
- The authorization process involves instances of `AccessController` and `AccessControlContext`.

JAAS Authorization 2-2

- Following are the steps involved in the authorization process:
  1. The `doAs()` method of `Subject` class is invoked to associate a role to the authenticated user.
  2. The `SecurityManager` checks the permissions associated with the Subject using `checkPermission()` method. It in turn invokes the `AccessController`.
  3. `AccessController` performs the required check and updates the `AccessControlContext` with the `Subject` and its associated permissions.

Propagating a Security Identity

- Following are different options for propagating the user identity:
  - The identity of the entity through which the user accessed the first entity can be propagated to the second entity by default.
  - By configuring a '**Run-as**' identity for the bean.

Securing Application Clients

- The security requirements of application clients are similar to that of EJB components.
- The application client authenticates the users accessing it either:
  - When the application client starts.
  - When the user is trying to access a protected resource in the application.
- The application client can use a `LoginModule` object to gather the user information.
- The `CallBackHandler` instances can further carry out the authentication process.

Summary

- Security mechanisms can be defined at three levels – application layer level, message layer level, and transport layer level.
- Security mechanisms in applications can be defined declaratively and programmatically.
- Security mechanisms are declaratively defined through annotations and deployment descriptors.
- Programmatically security mechanism is defined using EJBContext interface.
- Application users are logically defined as roles according to the application semantics.
- Users are categorized into logical groups known as user groups.
- Both users and user groups can be assigned to different roles in the application.
- JAAS provides various classes and interfaces for implementation of authentication and authorization process.