SWiN BUR NE

**SWINBURNE
UNIVERSITY OF
TECHNOLOGY**

SARAWAK CAMPUS

Disclaimer

© Swinburne University of Technology

This recording is provided as a Swinburne University of Technology unit resource for the use only by students enrolled in the unit for private study purpose.

Unless otherwise licensed this recording cannot be used by any other party or for any other purpose except as permitted under copyright legislation.

COS30041 Creating Secure and Scalable Software

Lecture 07 Securing Enterprise Applications

Learning Objectives

■ After studying the lecture material, you will be able to

☐ Understand and describe the security issues in Enterprise Application

☐ Understand and describe the advantages and disadvantages of using the JavaEE security model

☐ Understand the issues involved in implementing Enterprise Security using JavaEE

Pre-requisite

■ Basic Java Security concepts

Outline

- Characteristics of Application Security

- Overview of Enterprise Application Security

Roadmap

■ **Characteristics of Application Security**

■ Overview of Enterprise Application Security

Security Concerns

- Who is accessing the service?

- Are they who they say they are?

- Are they allowed to?

- Can anyone modify the data?

- Can anyone read this private data?

Characteristics of Secure Application

- Authentication – Who you say you are

- Authorization – Do you have access to the resources

- Data integrity – Authorized users can modify the data

- Confidentiality – Only authorized users can view the sensitive data

Other Characteristics

■ Auditing – Capture a tamper-resistant record of security-related events for the purpose of being able to evaluate the effectiveness of security policies and mechanisms

■ Non-repudiation – Proving the transaction has happened

■ Quality of Service – Provide better service to selected network traffic over various technologies

Roadmap

■ Characteristics of Application Security

■ **Overview of Enterprise Application Security**

Overview of Enterprise Application Security

■ Container provides the security for their software components

■ Security structure

　□ Authentication requirements (who you say you are)

　□ Access control requirements (who can access what)

　□ User-based / role-based security

　□ Security roles (what role you are with respect to the application)
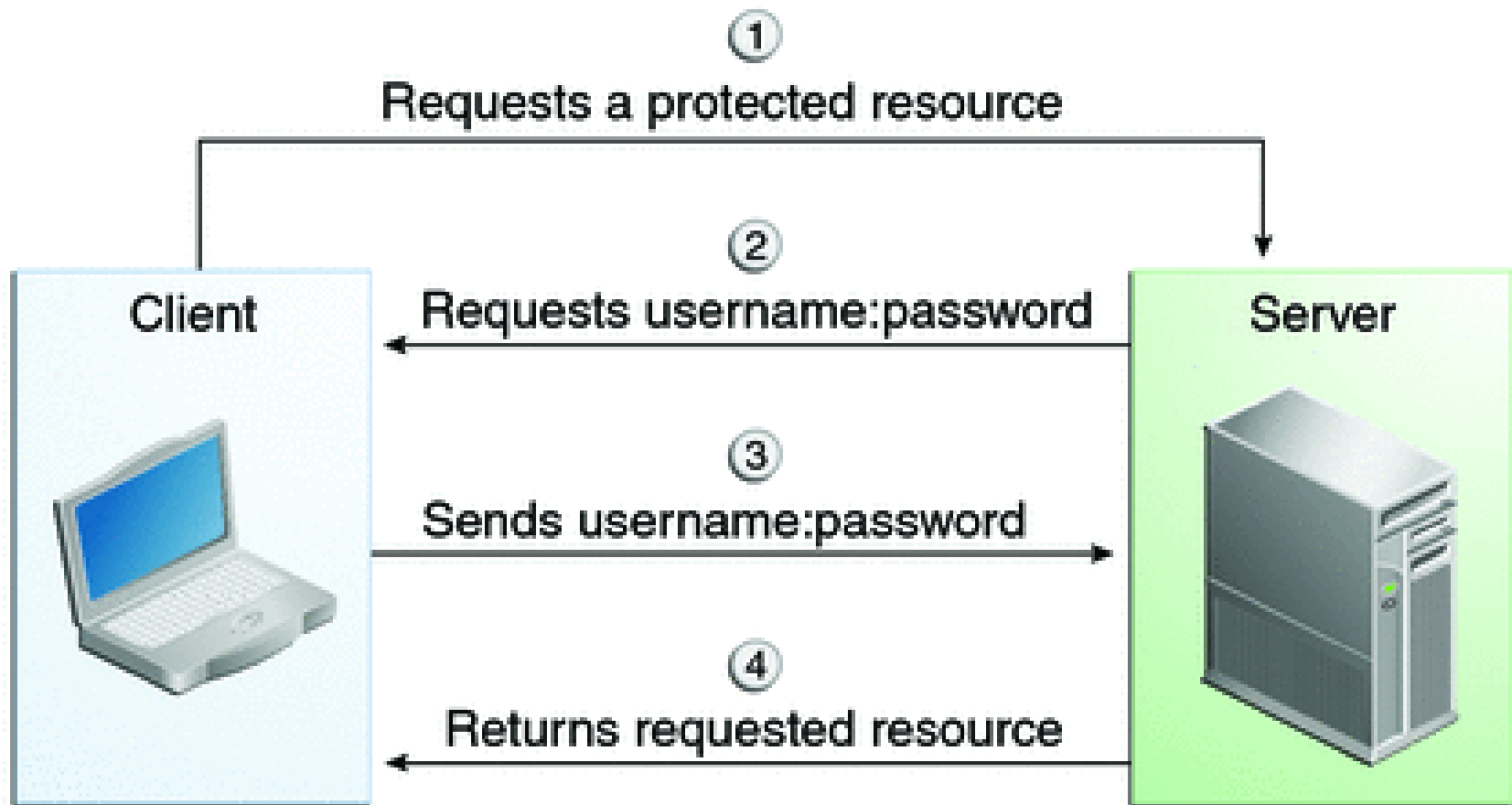
Password Security

- Password Authentication

- Storing passwords in database

  - Plain Text (e.g. "123456")

  - HASH (e.g. "123456" →hash→ "4982796920")

  - HASH + SALT (e.g. "123456" → hash → "4982796920" →salt→ "axez49827djfjl96920sljdfsl")

  - …

Default Security Behaviour in Ent. App.

Adapted from Fig. 40-2 [JEE6T]

Authentication

- Purpose: To identify who you say you are

- Many ways of authentication

  - Entity

    - UserId and Password (simplest, we use this in the unit)

    - …

    - UserId, Password and (Answer to Security Question | Sec Device)

    - …

    - Certificate based authentication (most sophisticated ?)

  - Secure vs Non-secure connection

  - Encrypted vs Plain

Modes of communication

Realm, User and Group [Java EE]

- A Realm = A security policy domain defined for a web or application server

    □ Contain a collection of users with or without groups

- A User = An individual that has been defined in the [GlassFish] server

- A Group = A set of authenticated users defined in the [GlassFish] server

Different types of Realm [JavaEE]

- File-realm*

    □ Stores user credentials in a file

- Admin-realm

    □ A file realm that stores administrators' user credentials

- JDBC-realm [Java EE]

    □ Store user credentials in database records

- Certificate

    □ Store user credentials in a certificate database

Note: We will only focus on file realm.

Authorization

- Purpose: to verify whether you have access to certain (protected) resources in the application

- Need to define the "access control requirements"

- Approaches
  - ☐ Declarative security
  - ☐ Programmatic security

Security Role

- simply, Role

- An abstract name for the permission to access a particular set of resources in an application

Group vs Security Role [Java EE specific]

Group

- Designated for the entire GlassFish Server

Security Role

- Associated only with a specific application in the GlassFish server

Associating Group with Role [Java EE specific]

- GlassFish server has a way to map security roles in an application to the groups

- Default – principal-to-role (group name = role name)

    - Principal = an entity that can be authenticated by an authentication protocol in a security service that is deployed in an enterprise

Note*: In case, group names and role names are not the same,
you need to use runtime deployment descriptor to specify the mapping

Declarative vs Programmatic

Declarative Security

- Specify the application component's security requirements by using either

  - ☐ Deployment descriptors, or

  - ☐ Annotations

- @DeclareRoles

- @RolesAllowed

- @PermitAll

- See EX-SLSB-SecInfo

Programmatic Security

- Security decisions were programmed in the source code

- Useful when declarative security is not sufficient to express the security model of an application

- SessionContext.getCallerPrincipal()

- SessionContext.isCallerInRole()

- Principal.getName()

- See EX-SLSB-SecInfo

Further Topics (for you to investigate)

- Different authentication mechanisms

- Defining different types of realm

- Configuring Security Using Deployment Descriptors

- Creating your own login module rather than the default provided by the application server

References

- Java EE Tutorial

END OF SLIDES