15

Signing an Application and Authentication

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Objectives

After completing this lesson, you should be able to:

- Describe cryptography, encryption, and cipher
- Differentiate a symmetric key from an asymmetric key
- Describe public/private key encryption
- Describe digital certificates
- Describe hash algorithms
- Sign an application by using Java tools
- Describe how SSL works
- Explain the security concepts that are applied in BrokerTool

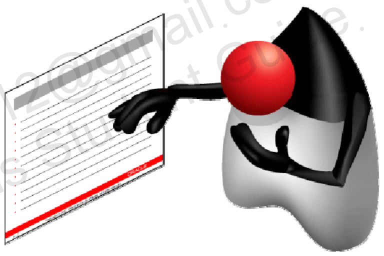
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodela2012@gmail.com) has a non-transferable license to use this Student Guide.

Topics

- Key security concepts
- Applying encryption technologies
- Code signing
- SSL
- Explain the security concepts that are applied in BrokerTool



Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Key Security Concepts

- Cryptography (from Latin *cryptographia*, “hidden writing”)
  - The practice and study of securing communications between two parties
  - Dates back as early as Julius Caesar
- Encryption
  - Process of converting plain text into ciphertext (unintelligible text)
- Cipher
  - The algorithm used to encrypt and decrypt the text
- Key
  - A piece of information that determines the output of the cipher

ORACLE

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Cryptography is the practice and study of securing communication between two parties so that third parties cannot read the communication.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Caesar Cipher

- Convert plain text message
  - Hi, how are you?
- Substitution cipher
  - `ABCDEFGHIJKLMNOPQRSTUVWXYZ`
  - `DEFGHIJKLMNOPQRSTUVWXYZABC`
- Encrypted message
  - Kl, krz duh bry?

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Cryptography was known to be used by Julius Caesar to send messages of military significance. He used a simple substitution cipher to encrypt his messages.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Types of Encryption

- Symmetric key encryption
  - The same key is used to encrypt and decrypt data.
- Asymmetric key encryption
  - Separate keys can be used to encrypt and decrypt the data.
- Public key encryption
  - An asymmetric key encryption system
  - A private key encrypts the data that can be decrypted with a public key.
  - Anyone with the public key can read the message.
  - A public key is used to encrypt data that can be decrypted by the private key.
  - Used for digital signatures and code signing

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

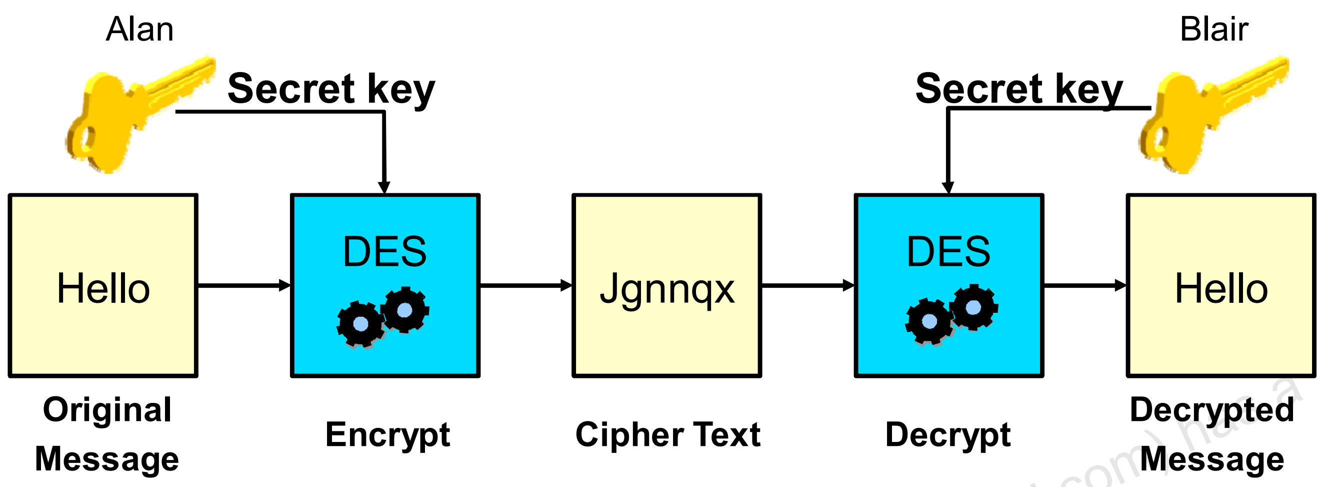
Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Symmetric key encryption is a faster, more efficient way to encrypt and decrypt data. However, given how powerful computers have become today, it is not very secure and can be defeated using a brute force attack (that is, by trying every possible combinations of letters).

Public key encryption uses two keys to encrypt and decrypt data. One key, called the private key, is encrypted using a password known only to the owner and is never transmitted by any means. The other key, called the public key, is provided freely to anyone with whom the key pair owner wishes to communicate.

Any message encrypted by the private key can be decrypted by the public key, and vice versa. If the private key is lost, then any message encrypted using the public key is lost as well.

Symmetric Key Encryption



The same key is used for encryption and decryption.

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

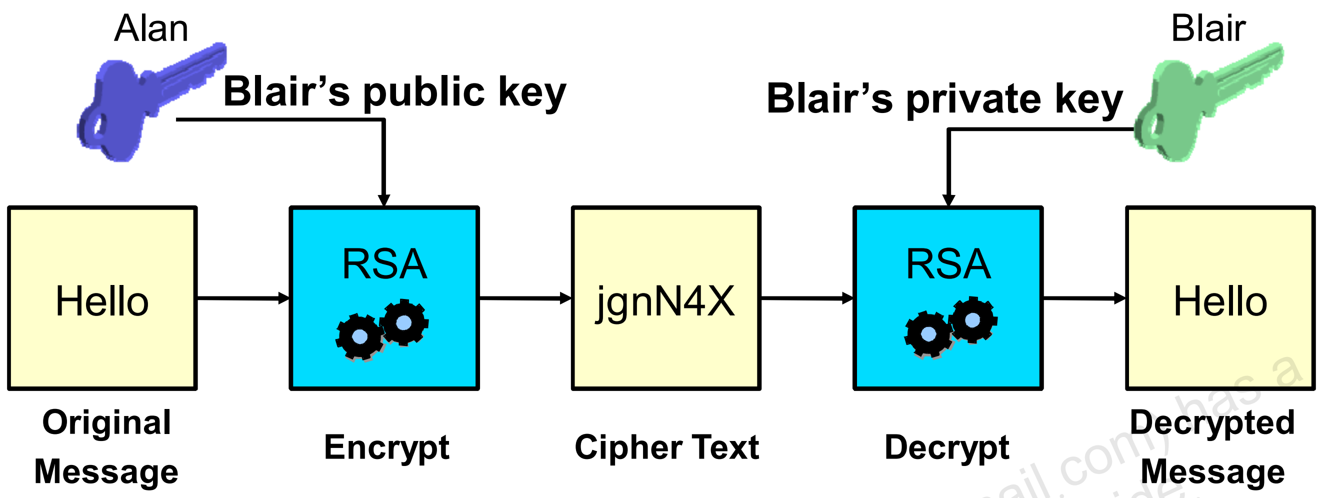
Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

In the example, data is encrypted using the DES cipher. Data Encryption Standard (DES) was a symmetric key encryption standard developed by Horst Feistel at IBM around 1974. However, eventually it became evident that the 56-bit key used for DES was insufficient and the cipher was replaced with other techniques like Triple DES.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Public Key Encryption



The receiver's public key is used for encryption and the receiver's private key is used for decryption.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

This example shows how a message is sent from Alan to Blair. But how do we know that we really have Blair's public key?

Quiz

A message encrypted with a private key can be decrypted with:

a. A symmetric key
b. Both the public and private key
c. A private key
d. A public key

ORACLE
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Answer: d

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.
Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Topics

- Key security concepts
- Applying encryption technologies
- Code signing
- SSL
- Explain the security concepts that are applied in BrokerTool

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Applying Encryption Technologies

Given these great tools, what can we do with them?

- Code signing
  - Verifying that software comes from you
  - Lets customer knows that software is safe
- SSL/TLS
  - Secure information between client and server
  - Once again provides trust to customers

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

So we have all these wonderful encryption tools, but what can we use them for?

First, we can use the technologies to digitally sign software. The allows customers to trust that the software came from you and is safe to use. The steps for signing software is covered next.

Finally, when using a web service, it is important that we secure communications point to point. This can be done using Secure Socket Layer (SSL), also called Transport Layer Security (TLS). The protocol uses a combination of techniques to keep your data private.

Digital Certificates

- Digital ID
- Proves your identity
- Typically includes
  - Owner’s name
  - Owner’s public key
  - Public key expiration date
  - Certificate authority name
  - Serial number
  - Certificate authority digital signature

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

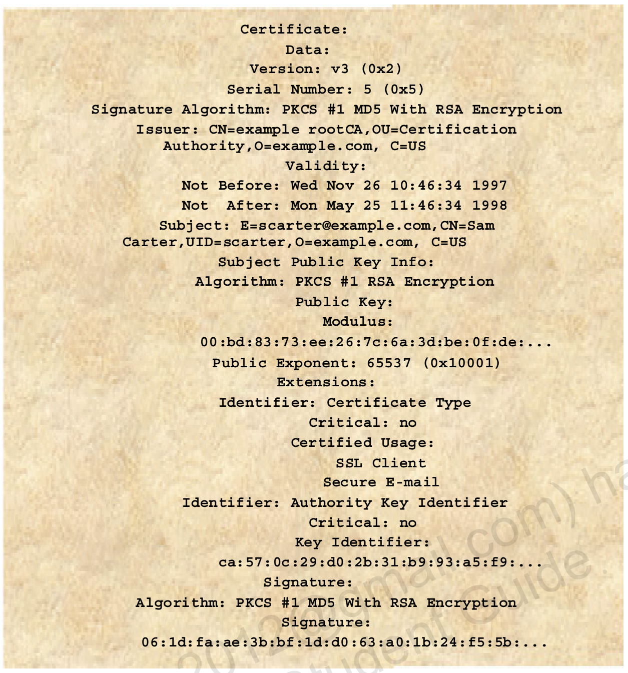
Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Digital certificates are the electronic counterparts to driver licenses, passports, and other forms of identification. You present your digital certificate electronically to prove your identity, or your right to access information or services online.

Digital certificates bind an entity’s identity to a pair of digital keys that can encrypt and sign information. When used with encryption, digital certificates provide a more complete security solution, assuring the identity of all parties involved in electronic transactions.

Certificate Sample



```
Certificate:
    Data:
        Version: v3 (0x2)
        Serial Number: 5 (0x5)
        Signature Algorithm: PKCS #1 MD5 With RSA Encryption
        Issuer: CN=example rootCA,OU=Certification
Authority,O=example.com, C=US
        Validity:
            Not Before: Wed Nov 26 10:46:34 1997
            Not  After: Mon May 25 11:46:34 1998
        Subject: E=scarter@example.com,CN=Sam
Carter,UID=scarter,O=example.com, C=US
        Subject Public Key Info:
            Algorithm: PKCS #1 RSA Encryption
            Public Key:
                Modulus:
                    00:bd:83:73:ee:26:7c:6a:3d:be:0f:de:...
                Public Exponent: 65537 (0x10001)
        Extensions:
            Identifier: Certificate Type
                Critical: no
                Certified Usage:
                    SSL Client
                    Secure E-mail
            Identifier: Authority Key Identifier
                Critical: no
                Key Identifier:
                    ca:57:0c:29:d0:2b:31:b9:93:a5:f9:...
    Signature:
        Algorithm: PKCS #1 MD5 With RSA Encryption
        Signature:
            06:1d:fa:ae:3b:bf:1d:d0:63:a0:1b:24:f5:5b:...
```

A certificate contains a public key and information about its owner, with a certificate authority's guarantee about the accuracy of the information.

How does Alan know that the certificate was not altered?

ORACLE

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

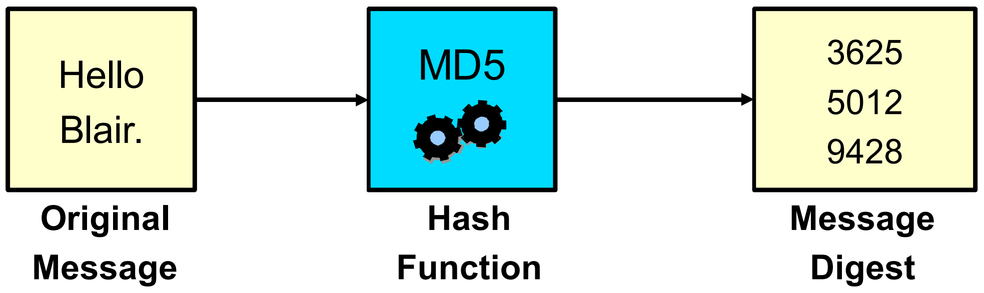
Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide

Certificates used in Java code signing follow the X.509 standard. Pictured is some of the sample information contained in the certificate. But how do we know that the information has not been altered?

Sample Certificate Information:

```
Certificate:
   Data:
       Version: v3 (0x2)
       Serial Number: 5 (0x5)
       Signature Algorithm: PKCS #1 MD5 With RSA Encryption
       Issuer: CN=example rootCA,OU=Certification
Authority,O=example.com, C=US
       Validity:
           Not Before: Wed Nov 26 10:46:34 1997
           Not  After: Mon May 25 11:46:34 1998
       Subject: E=scarter@example.com,CN=Sam
```

Message Digests/Hashes



Message digests are unique hash values such that:

- Two documents are not likely to hash to the same value
- Given a hash, you cannot determine what the original message was (one-way hash)

How do I know that the message digest was not altered?

ORACLE
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.
Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

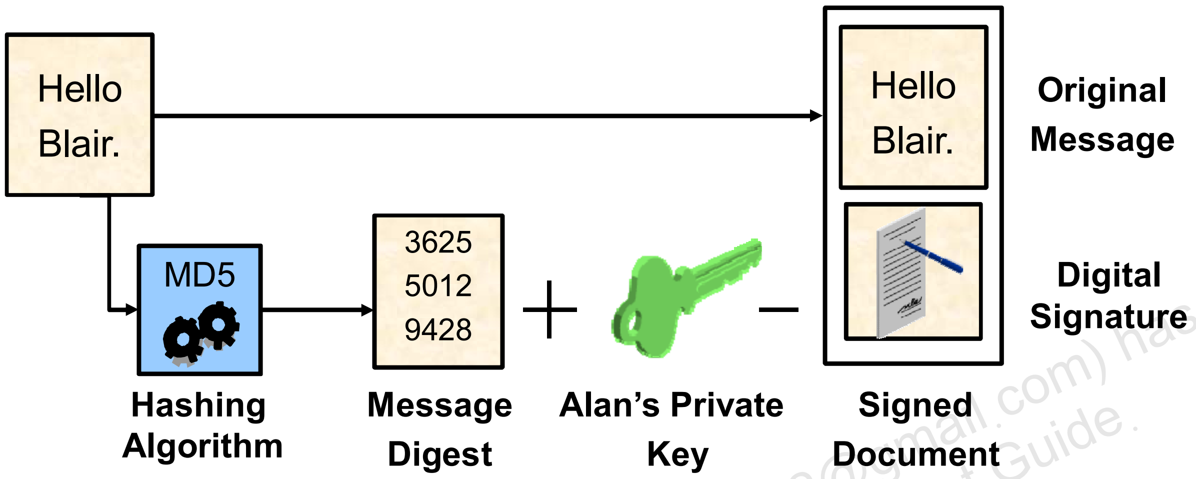
Message integrity is achieved by calculating a numerical value, called a message digest, which provides the original document's digital fingerprint. A message digest is created using a hash function that derives a unique numerical value from message text.

So, if you could compute a message digest before a message is sent and if you computed another message digest after the message is received, then if the BEFORE digest and the AFTER digest are the same, you can be sure the message has not been changed.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Digital Signatures I

Alan sends a message to Blair.



The message digest encrypted with the sender's private key is the digital signature.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

ORACLE
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

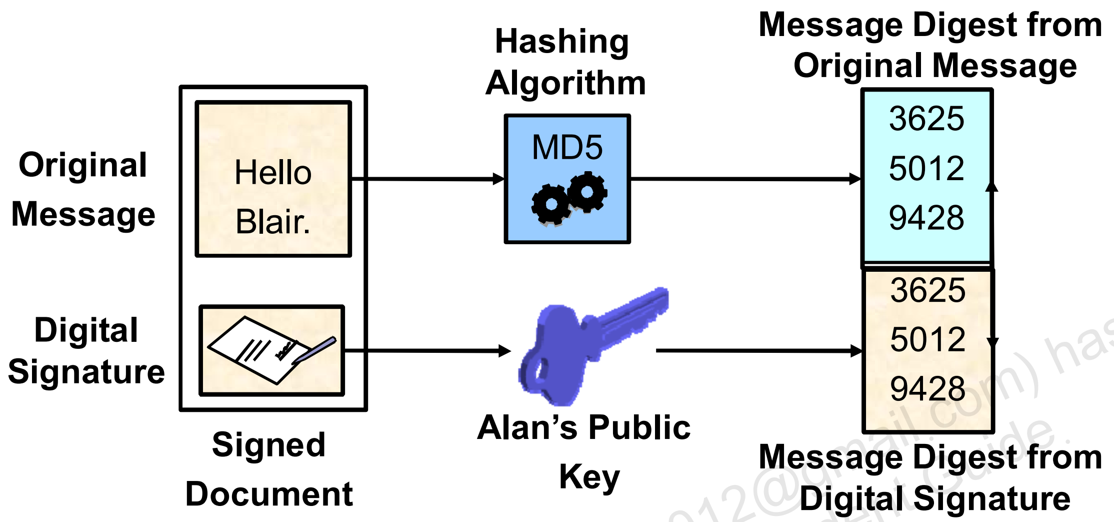
A message digest provides the original document's digital fingerprint. When the message digest is encrypted using the sender's private key, it becomes a digital signature, which can be appended to an encrypted or plain-text message to guarantee the message's source and integrity.

A digital signature, in theory, is of greater validity than a physical signature on paper. If a ten-page paper document is signed on the last page, there is no guarantee that one of the previous pages was not altered. In contrast, any change to a digitally signed document changes the message digest.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Digital Signatures 2

Blair receives the message.



The recipient hashes the message and compares the resultant message digest with the one from the digital signature.
Whose certificate should I trust?

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

To verify the data's integrity, the receiving software first uses the sender's public key to decrypt the message digest. It then uses the same hashing algorithm that generated the original hash to generate a new one-way hash of the same data. (Information about the hashing algorithm used is sent with the digital signature, although this information is not shown in this slide.)

Finally, the receiving software compares the new hash against the original hash. If the two hashes match, the data has not changed since it was signed. If they do not match, the data might have been tampered with because it was signed, or the signature might have been created with a private key that does not correspond to the public key presented by the signer.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Certificate Authority

- A trusted entity that issues and manages certificates
- Three ways to handle certificates:
  - External: CAs sell individual certificates
  - Outsourced: CAs manage certificate issuance and revocation
  - Internal: Use a certificate server

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

ORACLE

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

An X.509 certificate is issued by a third party called a certificate authority (CA), which takes responsibility for identifying a specific public key's owner. The method used by the CA to establish the owner's identity is the most important part of CA security policy. The effectiveness of Internet security depends on the care taken when issuing certificates.

Quiz

Which technology can you use to ensure that a message has not been tampered with?

a. A digital signature
b. A certificate
c. A symmetric key
d. Compression

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Answer: a

Topics

- Key security concepts
- Applying encryption technologies
- Code signing
- SSL
- Explain the security concepts that are applied in BrokerTool

ORACLE
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Code Signing with a CA

- For an application or applet distributed over the Internet
- Must purchase certificate from the CA
- Example Certificate Authorities
  - VeriSign
  - Thawte
  - Go Daddy
- Check with the vendor for specific steps

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

If you have a public application available over the Internet that you wish to have signed, you must purchase a certificate from a certificate authority. Each vendor has its own requirements, but the steps should be pretty similar.

Note: The companies listed are examples of vendors who offer these services. It is in no way an endorsement by Oracle.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Steps for Getting a Certificate

To get a certificate from a CA, you most perform the following steps:

- Create a keystore.
  - This will create a public and private key.
  - Use keytool, which comes with Java.
- Generate a certificate signing request.
  - Create a request form using your public key.
- Go to the vendor's website to purchase and submit the request.
- The vendor sends you a certificate.
- Store the certificate in your keystore.
  - Now you can use the certificate to sign your applications.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

ORACLE

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Once you have done all the steps outlined in the slide, you are ready to go. When you sign a .jar, your private key is used. The certificate is included with the `.jar` so that a user can verify that the file comes from you and has not been tampered with.

Generating Public and Private Keys with keytool

- keytool
  - It is a key and certificate management utility.
  - Users can create public/private key pairs.
  - Users can create certificates for use in self-authentication.
  - It allows users to cache the public keys (in the form of certificates).
- Sample keytool command:
  - `keytool -genkey -alias signFiles -keystore examplestore`

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

```
keytool -genkey -alias signFiles -keystore examplestore
```

What do each of the keytool subparts mean?

`-genkey`: Generates the keys

`-alias signFiles`: The alias used to refer to the keystore entry containing the keys that are generated

`-keystore examplestore`: The name (and optionally path) of the keystore that you are creating

You will also be prompted for a password for the keystore (the storepass) and for the private key (keypass). You can optionally make them the same password.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Additional keytool Information

In addition to the storepass and keypass values, you will be prompted for distinguished name information such as the following:

```
What is your first and last name?
  [Unknown]:  John Adams
What is the name of your organizational unit?
  [Unknown]:  Purchasing
What is the name of your organization?
  [Unknown]:  ExampleCompany
What is the name of your City or Locality?
  [Unknown]:  Mountain View
What is the name of your State or Province?
  [Unknown]:  CA
What is the two-letter country code for this unit?
  [Unknown]:  US
Is <CN=John Adams, OU=Purchasing, O=ExampleCompany,
    L=Mountain View, ST=CA, C=US> correct?
  [no]:  y
```

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

In addition to the passwords, you are prompted for additional information to uniquely identify this keystore.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Sign a jar

- Jarsigner
  - Is a command line tool
  - Can sign or verify a jar file
  - Creates a new jar and does not change the source jar

```
jarsigner -keystore examplestore -signedjar sCount.jar
Count.jar signFiles
```

- NetBeans
  - Project properties
  - Build > Deployment > Request Unrestricted Access

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

ORACLE

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

To create a self-signed jar file, you can use the jarsigner command line tool. In addition, NetBeans includes support for creating self-signed `.jar` files.

```
jarsigner -keystore examplestore -signedjar sCount.jar Count.jar
signFiles
```

`-keystore`: Specifies the keystore file

`-signedjar`: Specifies the jar that will be created from the command (`sCount.jar`)

The `.jar` file and alias for the keystore are the last two parameters in the command.

Quiz

Which tool can be used to create a keystore?

a. Keystoretool
b. keytool
c. jarsigner
d. keystore

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Answer: b

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Topics

- Key security concepts
- Applying encryption technologies
- Code signing
- SSL
- Explain the security concepts that are applied in BrokerTool

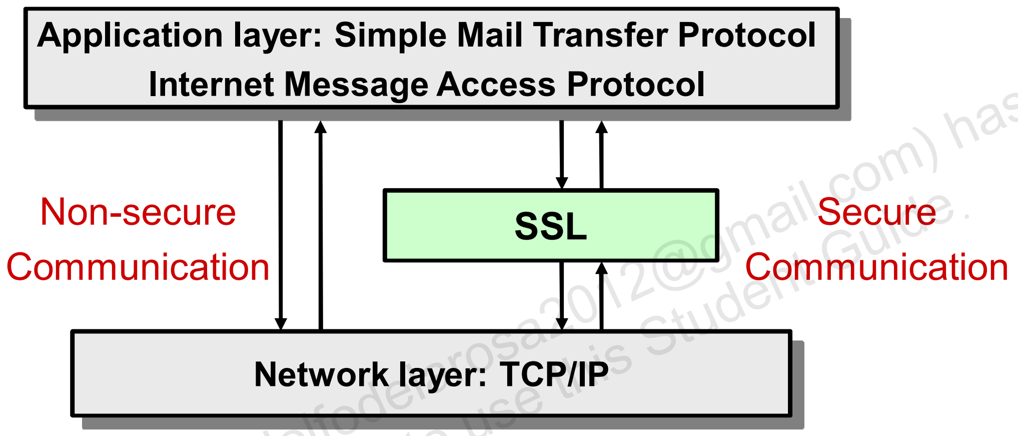
ORACLE
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.
Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

SSL

Secure Socket Layer (SSL)

- Ensures safe and secure client-server transactions
- Data sent over the network is point-to-point encrypted.
- Also called Transport Layer Security (TLS)


Application layer: Simple Mail Transfer Protocol
Internet Message Access Protocol
Non-secure
Communication
SSL
Secure
Communication
Network layer: TCP/IP

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

To secure our web service connection, we would use secure sockets layer (SSL). SSL provides the following:

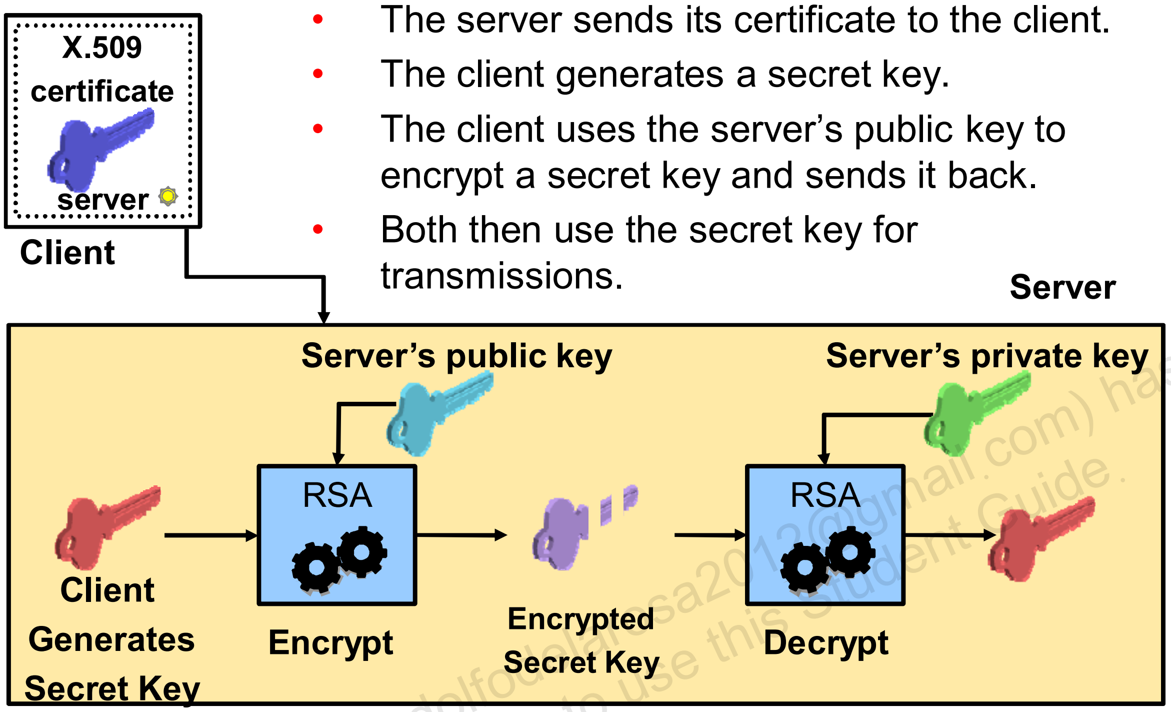
- **Authentication:** Using X.509 certificates and digital signatures encrypted using RSA public-key technology
- **Message privacy:** Using DES, RC2, or RC4 symmetric encryption
- **Message integrity:** Using MD5 or SHA-1 message digests

SSL forms an application-independent layer between TCP/IP and the network application layer. As a result, SSL can be used to provide secure communications for different network applications. For HTTP connections, Hypertext Transport Protocol, Secure (HTTPS) secures the connections between clients and servers.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Generating Secure Connections

- The server sends its certificate to the client.
- The client generates a secret key.
- The client uses the server's public key to encrypt a secret key and sends it back.
- Both then use the secret key for transmissions.



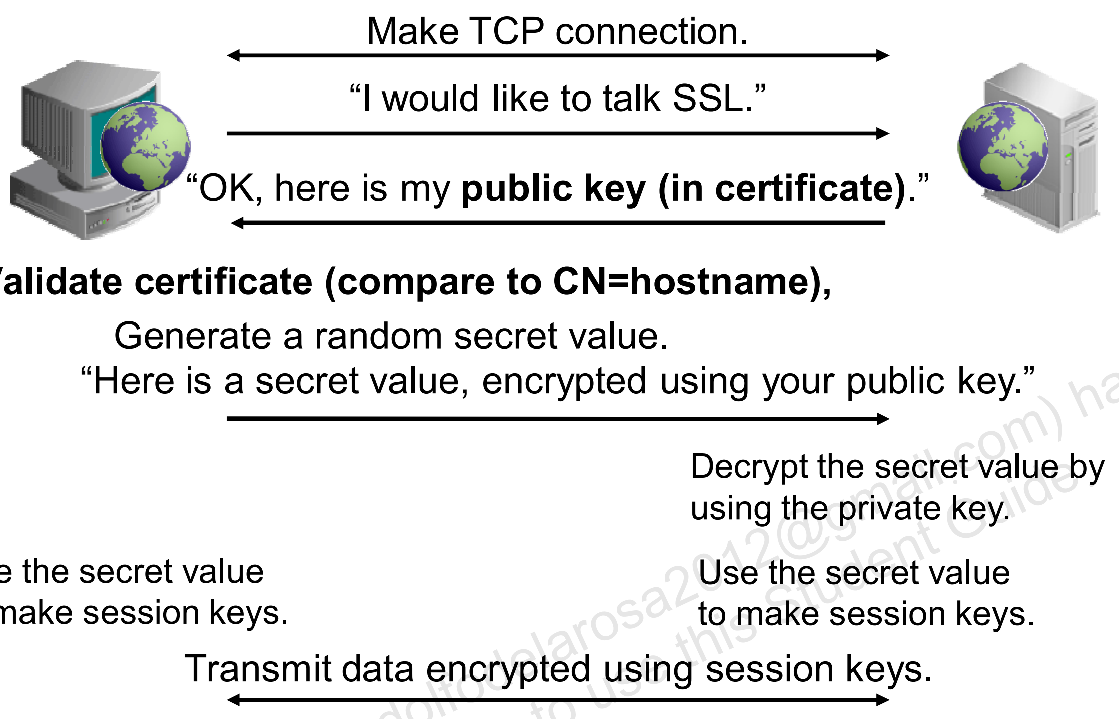
ORACLE
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

By combining symmetric and asymmetric-key encryption and digital certificates, you can exchange symmetric keys. A secure connection is generated when using the following steps:

1. A client generates a temporary symmetric key. This is sometimes called the session key because it expires after communication between the client and the server is complete. It is used only for one session.
2. The client encrypts the temporary symmetric key with a server's public key. This means that only the server can decrypt this information.
3. The encrypted symmetric key is sent to the server in much the same way that an encrypted message is sent.
4. The server then decrypts the secret key by using its own private key, so now both the client and the server can use the newly generated symmetric key.
5. During the rest of the session, encrypted communication takes place using the faster, more efficient symmetric key.
6. When the session is over, the symmetric key is discarded, because it no longer has any use to either system.

SSL Server Authentication


Make TCP connection.
"I would like to talk SSL."
"OK, here is my **public key (in certificate)**."
Validate certificate (compare to CN=hostname),
Generate a random secret value.
"Here is a secret value, encrypted using your public key."
Decrypt the secret value by using the private key.
Use the secret value to make session keys.
Use the secret value to make session keys.
Transmit data encrypted using session keys.

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

The following summarizes the steps in the SSL process when only the server is authenticated.

1. **Client hello:** The client sends a message that begins the handshake portion of the SSL protocol. The client tells the server which encryption algorithms it recognizes.
2. **Server hello:** The server responds with an X.509 certificate signed with its private key and tells the client which encryption algorithm is the strongest form of encryption that both of them recognize.
3. **Client key exchange:** The client calculates a master secret, encrypts it with the server's public key, and sends it to the server.
4. **Change cipher spec:** The client and the server create session keys from the master key and begin communicating securely.
5. **HTTPS request:** The client sends the HTTPS request, encrypted with the session key.
6. **HTTPS response:** The server responds, encrypting its response with the session key, and notifies the client that it is done.

Quiz

What is the purpose of SSL?

a. To sign documents
b. To create a secure connection between a client and a server
c. To create unbreakable certificates
d. To decrypt any document that has been encrypted using a public key

ORACLE

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Answer: b

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Topics

- Key security concepts
- Applying encryption technologies
- Code signing
- SSL
- Explain the security concepts that are applied in BrokerTool

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Java EE Application Security Mechanisms

- Security of Java EE components are provided by their containers.
- A container provides two kinds of security:
  - Declarative
  - Programmatic
- Declarative security uses either deployment descriptors or annotations.
- Programmatic security is embedded in an application and is used to make security decisions.

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Enterprise-tier and web-tier applications are made up of components that are deployed into various containers.

These components are combined to build a multitier enterprise application.

A deployment descriptor is an XML file that contains information about security roles, access control, and authentication requirements. NetBeans IDE provides tools for creating and modifying deployment descriptors.

Web components may use a web application deployment descriptor named `web.xml`.

Enterprise JavaBeans components may use an EJB deployment descriptor named `META-INF/ejb-jar.xml`, contained in the EJB JAR file.

Annotations, also called metadata, are used to specify information about security within a class file. When the application is deployed, this information can be either used by or overridden by the application deployment descriptor. Annotations save you from having to write declarative information inside XML descriptors. Instead, you simply put annotations on the code, and the required information gets generated.

Programmatic security is useful when declarative security alone is not sufficient to express the security model of an application.

Securing Web Applications

By using annotations or deployment descriptors, you can secure a web application by:

- Specifying security constraints
- Specifying authentication mechanisms
  - HTTP basic
  - Form-based
  - Digest
  - Client
  - Mutual
- Declaring security roles

ORACLE

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

A *security constraint* is used to define the access privileges to a collection of resources by using its URL mapping. Its sub-elements are:

- Web resource collection
- Authorization constraint
- User data constraint

A user-authentication mechanism specifies:

- The way a user gains access to web content
- With basic authentication, the realm in which the user will be authenticated
- With form-based authentication, additional attributes

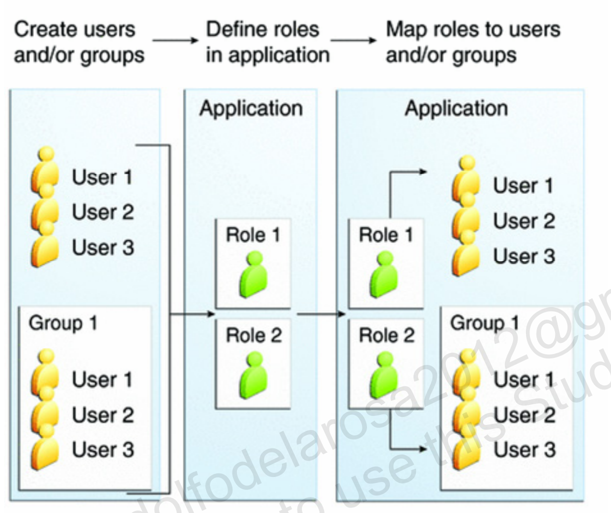
When an authentication mechanism is specified, the user must be authenticated before access is granted to any resource that is constrained by a security constraint. There can be multiple security constraints applying to multiple resources, but the same authentication method will apply to all constrained resources in an application.

Before you can authenticate a user, you must have a database of user names, passwords, and roles configured on your web or application server.

You can declare security role names used in web applications by using the security-role element of the deployment descriptor. Use this element to list all the security roles that you have referenced in your application.

Applying Security in GlassFish Server

- GlassFish Server supports the Java EE 6 security model.
- You can configure GlassFish Server for adding, deleting, or modifying authorized users and existing or custom realms.



ORACLE
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

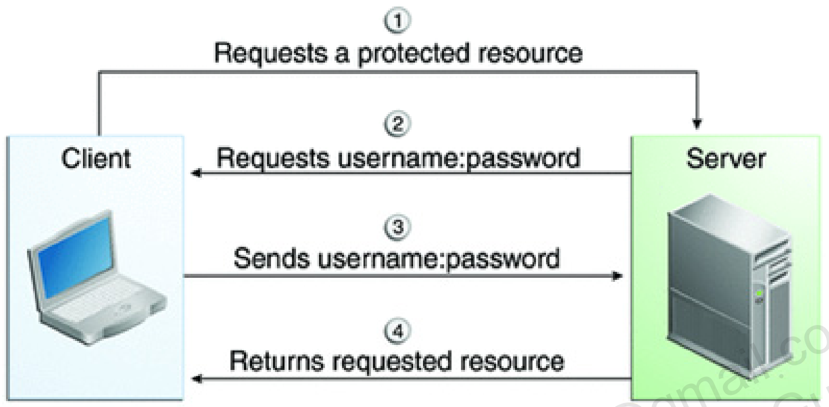
A realm is a security policy domain defined for a web or application server. A realm contains a collection of users, who may or may not be assigned to a group.

The figure in the slide shows how mapping roles to users and groups are done in GlassFish.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.
Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

HTTP Basic Authentication Mechanism


1
Requests a protected resource
2
Requests username:password
3
Sends username:password
4
Returns requested resource
Client
Server

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

ORACLE
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

A user authentication mechanism specifies the way a user gains access to web content.

With basic authentication:

1. A client requests access to a protected resource.
2. The web server returns a dialog box that requests the user name and password.
3. The client submits the user name and password to the server.
4. The server authenticates the user in the specified realm and, if successful, returns the requested resource.

Specifying HTTP basic authentication requires that the server request a user name and password from the web client and verify that the user name and password are valid by comparing them against a database of authorized users in the specified or default realm.

Authentication in an Application

You protect resources to ensure that only authorized users have access.

To authenticate a user, you need to perform the following basic steps:

1. The application developer writes code to prompt for a user name and password.
2. The application developer communicates how to set up security for the deployed application by use of a metadata annotation or deployment descriptor.
3. The server administrator sets up authorized users and groups on the GlassFish Server.
4. The application deployer maps the application's security roles to users, groups, and principals defined on the GlassFish Server.

ORACLE

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

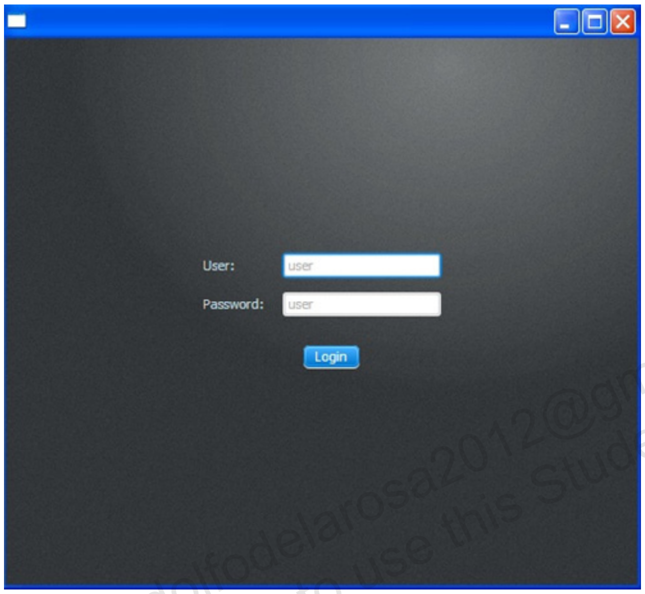
Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

An application will often prompt for a user name and password before allowing access to a protected resource.

After the user name and password have been entered, that information is passed to the server, which either authenticates the user and sends the protected resource or does not authenticate the user, in which case access to the protected resource is denied.

Authentication in the BrokerTool Application

`Login.fxml` of `BrokerToolClient3` prompts for a user name and password.



ORACLE
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

The Login screen that will prompt for a user name and password.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.
Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Deployment Descriptor of BrokerToolServer

```
<security-constraint>
        <display-name>UserConstraint</display-name>
        <web-resource-collection>
            <web-resource-name>User</web-resource-name>
            <description/>
            <url-
    pattern>/resources/com.brokertool.model.shares/*</url-pattern>
        </web-resource-collection>
        <auth-constraint>
            <description/>
            <role-name>UserRole</role-name>
        </auth-constraint>
    </security-constraint>
    <login-config>
        <auth-method>BASIC</auth-method>
        <realm-name>file</realm-name>
……
…. .
```

ORACLE

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

View the web.xml configuration file of the BrokerToolServer project where you can specify authentication mechanism.

The code snippet in the slide displays the security constraints declared in `web.xml` deployment descriptor file.

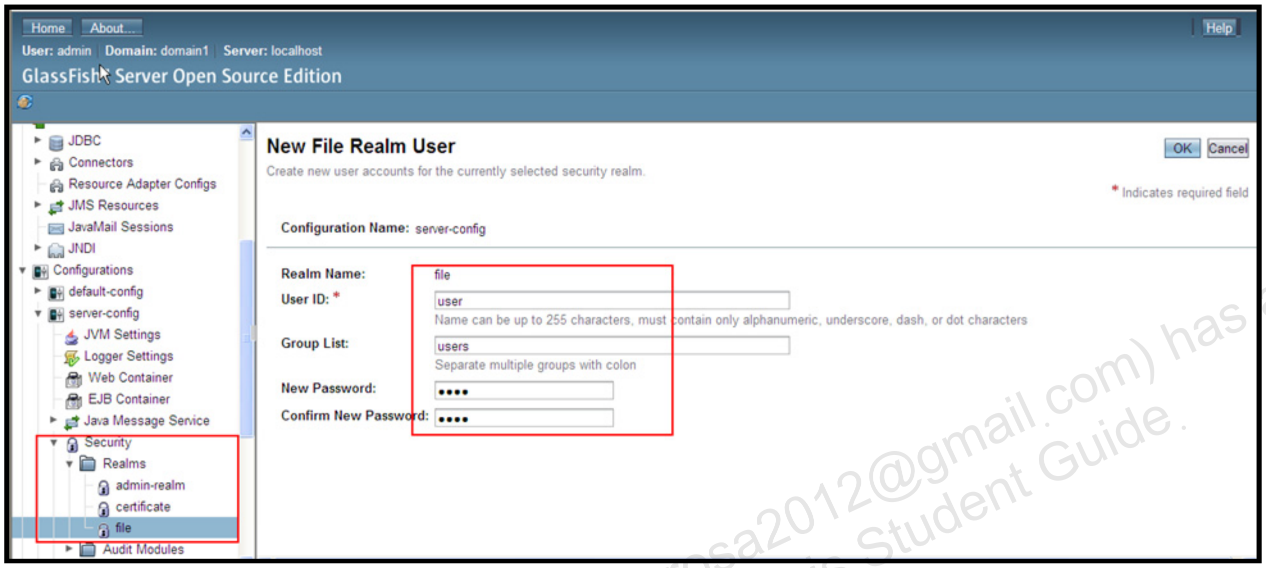
The auth-constraint element specifies which of the roles is authorized to access protected resources.

To specify an authentication mechanism, use the login-config element. It can contain the following sub-elements:

- The auth-method sub-element configures the authentication mechanism for the web application. The element content must be either NONE, BASIC, DIGEST, FORM, or CLIENT-CERT.
- The realm-name sub-element indicates the realm name to use when the basic authentication scheme is chosen for the web application.

You can declare security role names used in web applications by using the security-role element of the deployment descriptor. Use this element to list all the security roles that you have referenced in your application.

Set Up Users and Groups on the GlassFish Server



New File Realm User

Create new user accounts for the currently selected security realm.

* Indicates required field

Configuration Name: server-config

Realm Name: file

User ID: * user

Name can be up to 255 characters, must contain only alphanumeric, underscore, dash, or dot characters

Group List: users

Separate multiple groups with colon

New Password: ••••

Confirm New Password: ••••

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

BrokerToolClient3 includes basic authentication for "user"/"user“; therefore, in order to run this client with BrokerToolServer, your GlassFish server must be configured to include this specific user.

1. Open the GlassFish Admin Console in NetBeans.
   a. In the Services tab, open Services > GlassFish Server 3.1.1. Right-click and select View Domain Admin Console.
2. In the Common Tasks tree, open Configurations > server-config > Security > Realms > file. Select "file“. Edit Realm appears.
3. Select Manage Users on the Edit Realm page.
4. Add one user:
   User ID = user
   Group List = Users
   Password = user

The above is probably all you need, but another user has been added anyway:

User ID = admin
Group List = Users
Password = admin

Deployment Descriptor: `glassfish-web.xml`

You will use `glassfish-web.xml` to map the application's security roles to users, groups, and principals defined on the GlassFish Server.

```
<glassfish-web-app error-url="">
  <security-role-mapping>
<role-name>AdminRole</role-name>
    <principal-name>admin</principal-name>
  </security-role-mapping>
  <security-role-mapping>
    <role-name>UserRole</role-name>
    <principal-name>user</principal-name>
  </security-role-mapping>
```

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

The role-name must match the role-name in the security-role element of the corresponding deployment descriptor `web.xml`.

Principal: An entity that can be authenticated by an authentication protocol in a security service that is deployed in an enterprise. A principal is identified by using a principal name and authenticated by using authentication data.

Security policy domain, also known as **security domain** or **realm**: A scope over which a common security policy is defined and enforced by the security administrator of the security service

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Programmatic Security

```
@Override
 protected List<Customer> call() throws Exception {
                Client client = Client.create(CONFIG);
    client.addFilter(new
    HTTPBasicAuthFilter(BrokerToolClientApp.getInstance().getUs
    er(),BrokerToolClientApp.getInstance().getPassword()));
                WebResource productsWebResource =
    client.resource("http://localhost:8080/BrokerToolServer/res
    ources").path("com.brokertool.model.shares");
                ClientResponse response =
    productsWebResource.get(ClientResponse.class);
                int status = response.getStatus(); //status =
    401 if login error
                if (status == 401) {
                    throw new AuthenticationException();
                }
                return null;
            }
```

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

ORACLE

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

The code from `login.java` of the BrokerToolClient3 project demonstrates the use of programmatic security using Jersey API (com.sun.jersey.api.client.filter.HTTPBasicAuthFilter).

The code checks whether you are trying to access a web service resource without providing the authorized user credentials. If the user name and password is correct, the next screen appears; otherwise, the login screen appears.

This code snippet does the following:

- It retrieves the user name and password as provided by the user in the login screen.
- The resource `("http://localhost:8080/BrokerToolServer/resources").path("com.brokertool.model.shares")` is accessed.
- The status of `ClientResponse` determines whether the user has been authenticated.
- If its value is 401, an authentication exception is thrown.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Quiz

HTTP basic is an:

a. Authentication mechanism
b. Authority mechanism
c. Agreement mechanism
d. Amazing mechanism

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

ORACLE
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Answer: a

Summary

In this lesson, you should have learned how to:

- Describe cryptography, encryption, and cipher
- Differentiate a symmetric key from an asymmetric key
- Describe public/private key encryption
- Describe digital certificates
- Describe hash algorithms
- Sign an application by using Java tools
- Describe how SSL works
- Explain the security concepts that are applied in BrokerTool

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Practice 15: Overview:

- Practice 15-1: Signing an Application by Using `keytool` and `jarsigner`
- Practice 15-2: Signing and Deploying `TextViewer` by Using NetBeans

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.