What are the security requirements for web apps?

- **Authentication**: To verify the identity of the parties involved
- **Authorization**: To limit access to resources to users
- **Confidentiality**: To ensure that information passed only to parties involved
- **Integrity**: To ensure that information transmitted is not changed or tampered
How to achieve these requirements?

- Secure web applications using a web server
  - HTTP authentication
  - Authorization of users/groups
  - Authorization of domains
  - Secure HTTP, an extension of HTTP
  - SSL capabilities

How to achieve these requirements? (cont.)

- Secure web applications using a servlet container
  - HTTP authentication (basic, digest)
  - Form-based authentication
  - Authorization of users/groups
  - SSL capabilities
  - When to apply securities to web servers vs. servlet container?
HTTP Basic Authentication

- HTTP message header type: AUTHENTICATION
  - Authentication: basic user:password
  - user:password is base64 encoded
- Once configured, a login is required before allowed to access
- Lowest level of security, ID/password easy to hack
- No multiple sessions for the same site
- No logout mechanism

Configuring servlet container for HTTP authentication

- Add security-constraint, login-config and security-role element (in this order) to the deployment descriptor
  - Security-constraint: Define resources to be protected, and the role of authorized users.
  - Login-config: Define the authentication method and the associated realm
  - Security-role: Define all the security roles for this application (optional for simple apps)
- Add username/password/roles into the realm definition
Configuring Security Realm

- What is a realm?
  - A resource containing username/password/roles information
  - Vendor-specific, not a J2EE standard in the servlet specification
  - For instance, the tomcat-users.xml file in $CATALINA_HOME/conf directory
  - Various implementation available in Tomcat:
    - JDBC, JDBC DataSource, Memory, and JNDI
    - Configure via servlet.xml in the conf directory

HTTP Digest Authentication

- Username and password encrypted using standard digest (one-way hash) algorithms
- Standard HTTP message header type AUTHENTICATION with information of digest algorithm
- Configuration needed in both web.xml and server.xml
- Not widely used. Optional for container vendors. Not supported by all browsers
Form-based Authentication

- Using HTML form instead browser prompt for entering username/password
- More secure than HTTP based authentication because username/password only to be transmitted once
- Login form to use standard form action and input field names:
  - j_security_check (action), j_username, j_password
- Specify the user-defined login form and error form in the login-config element of web.xml

Customized Authentication

- Perform user authentication in user-defined servlet class
- Using the servlet API to access username and authentication type
  - httpRequest.getUserPrincipal().getName()
  - httpRequest.isUserInRole(String role)
  - httpRequest.getAuthType()
- More flexible. Useful not just for authentication, may also be used for application control
Secure Socket Layer (SSL) based Authentication

- Invented by Netscape in the mid 90’s
- Secure/encrypt every HTTP message to/from the web server using standard PKI technology
- De-facto standard used for secure web-based transactions
- Default URL https://some.domain.com with default port number 443
- Don’t get confused with S-HTTP (secures/encrypts only the http msg body)

How does PKI work? Cryptography 101

- How does Alice send a message to Bob that no one can see and prove it was indeed sent by Alice?
  - Confidentiality and Non-repudiation
  - Symmetric key encryption
    - one key to encrypt and decrypt
  - Asymmetric key encryption
    - one key to encrypt and second key to decrypt
    - Freely distributed public key and safe-guarded private key
    - trusted third parties, digital certificates
    - PKI – public key infrastructure with CA
  - Standard encryption algorithms: RSA, DES
  - Digital signature: message digest + PKI encryption
Digital Certificate

- Public key signed by using a private key and presented in standard format (X.509)
- In real word, usually signed by a CA: user’s public key signed by the CA’s private key
- Certificate chain: a certificate signed by multiple CA’s in the trust
- X.509 Content:
  - Public key value, cert holder’s unique name (DN), issuer's DN, version number, serial number, CA name, valid period, CA's signature

SSL for the Web

- Using authorized certificates for authentication
- Session-based (SSL session)
- Two step process, use public key to establish the session, and use symmetric key to exchange messages
- SSL server authentication: Used by client/browser to authenticate that the web server connected is legitimate. Used in most e-commerce apps
- SSL client authentication: Used by the server to authenticate the user. Used in highly secure apps where unique user certificates are required
The SSL Protocol

Source: The SSL Protocol version 3

1. Client Hello
7. Certificate *
8. Client Key Exchange *
9. Change Cipher Spec
10. Finished
14. Encrypted Data

2. Server Hello
3. Certificate *
4. Certificate Request *
5. Server Key Exchange *
6. Server Hello Done
11. Change Cipher Spec
12. Finished
13. Encrypted Data

SSL Server Authentication

- Web site/server to obtain digital certificate (public key signed by a CA) from a CA
- Server sends its certificate to the client upon the initial client request
- Client validates the server's DC using CA's DC (usually stored in the browser)
- If validates, client generates a symmetric key (DES) for the session, encrypt it using server's public key, and use the DES key to encrypt all subsequent transactions
- Support confidentiality, but not no-repudiation
SSL Client Authentication

- After the server is authenticated by the client via SSL, the server asked for the client certificate.
- The client sends its signed certificate to the server to be validated by the server.
- If validated, the client generates a symmetric key for the session, and encrypt it using server’s public key, and use the DES key to encrypt subsequence transactions.
- Support both confidentiality and non-repudiation.

Applicability of Client SSL Authentication

- Highest level of security. Possibility to integrate with smart-card and biometric technologies.
- Not supported by standard browsers (additional plug-ins required).
- Additional server module required to validate clients, usually using a vendor-specific security server.
- Very expensive because large PKI resources (hardware/software/personnel) needed to create, maintain, distribute user certificates.
Configure SSL for Tomcat server

- Create a user-defined certificate using standard JDK `keytool`
  ```
  %JAVA_HOME%/bin/keytool -genkey -alias tomcat -keyalg RSA
  ```
- Configure the SSL Http Connector of Tomcat by editing the `server.xml`
  ```
  <Connector className="org.apache.coyote.tomcat4.CoyoteConnector"
    port="8443" minProcessors="5" maxProcessors="75"
    enableLookups="true"
    acceptCount="100" debug="0" scheme="https" secure="true"
    useURIValidationHack="false" disableUploadTimeout="true">
    <Factory className="org.apache.coyote.tomcat4.CoyoteServerSocketFactory"
      clientAuth="false" protocol="TLS" />
  </Connector>
  ```
- Restart the server and run servlet apps on port 8443

Summary

- Web applications requires proper security at various levels for different purposes
  - HTTP Authentication, Basic/Digest (lowest level)
  - Form-based authentication
  - Customized authentication
  - SSL server authentication
  - SSL client authentication (highest level)
- Other security concerns
  - Database security
  - Network security
  - Human factors
Quiz

What types of web security (or combination) should be used in your servlet application, if your application requires only
- Simple authentication
- Simple authentication with protection of username/password
- Simple authentication with protection of the content of transactions
- Web documents to be protected, user access to the documents to be identified, and the validation of access to be accepted in the court of law