Users of Web Applications

- Multiple users
- Multiple types of users

Authentication and Authorization

Client request page x

Who are you?

- username/password

Authorization (Access Control)

You're not authorized to access page x

Connection Security

Authentication

- Basic
- Digest
- Form
- SSL

Authentication – Basic

HTTP 1.0, Section 11.1:
http://www.w3.org/Protocols/HTTP/1.0/draft-ietf-http-spec.html

Client request for a restricted page

Prompt for username/password

Server resend request

Authorization header field = username & password

Problem??

Cryptographic Hash Function...

- String of arbitrary length  
- bits digest

Properties

- Given a hash value, it's virtually impossible to find a message that hashes to this value
- Given a message, it's virtually impossible to find another message that hashes to the same value
- It's virtually impossible to find two messages that hash to the same value

A.K.A.

- One-way hashing, message digest, digital fingerprint
Cryptographic Hash Function

- **Common usage**
  - Store passwords, software checksum ...
- **Popular algorithms**
  - MD5 (broken, sort of)
  - SHA-1 (expected to be broken soon)
  - SHA-256 and SHA-512 (recommended)

Authentication – Digest

- RFC 2617 (Part of HTTP 1.1) - http://www.ietf.org/rfc/rfc2617.txt

  - Request for a restricted page
  - Prompt for username/password + nonce
  - Resend request + message digest
  - MD5(username + password + nonce + other stuff)

  Why nonce??

Authentication – Form

- Both Basic and Digest authentications are implemented by the HTTP server
- Form authentication is implemented by the Servlet/JSP engine
  - Username/password are passed as clear text
  - Login page instead of login prompt

Form Authentication using Tomcat

- $TOMCAT/conf/tomcat-users.xml
  - Users and roles
- $APPLICATION/WEB-INF/web.xml
  - Authentication type (FORM)
  - Login and login failure page
  - URLs to be protected

Example – Directory Layout

```
<table>
<thead>
<tr>
<th>/admin</th>
<th>home.jsp</th>
</tr>
</thead>
<tbody>
<tr>
<td>/restricted</td>
<td>secret.jsp</td>
</tr>
<tr>
<td>index.htm</td>
<td>login.jsp</td>
</tr>
<tr>
<td>login.jsp</td>
<td>logout.jsp</td>
</tr>
<tr>
<td>logout.jsp</td>
<td>404.htm</td>
</tr>
<tr>
<td>404.htm</td>
<td>error.htm</td>
</tr>
</tbody>
</table>
```

Example – Users and Roles

```
<?xml version='1.0' encoding='utf-8'?>
<tomcat-users>
  <role rolename="tomcat"/>
  <role rolename="cysun"/>
  <role rolename="manager"/>
  <role rolename="guest"/>
  <user username="tomcat" password="tomcat" roles="tomcat"/>
  <user username="cysun" password="abcd" roles="cysun,manager"/>
  <user username="test" password="test" roles="tomcat"/>
  <user username="guest" password="guest" roles="guest"/>
</tomcat-users>
```
Example – web.xml ...

```xml
<login-config>
  <auth-method>FORM</auth-method>
  <form-login-config>
    <form-login-page>/login.jsp</form-login-page>
    <form-error-page>/error.htm</form-error-page>
  </form-login-config>
</login-config>
```

... Example – web.xml

```xml
<security-constraint>
  <web-resource-collection>
    <web-resource-name>Admin</web-resource-name>
    <url-pattern>/admin/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <role-name>cysun</role-name>
  </auth-constraint>
</security-constraint>
```

Example – Login Page

```xml
<form action="j_security_check" method="post">
  <input type="text" name="j_username">
  <input type="password" name="j_password">
  <input type="submit" name="login" value="Login">
</form>
```

Declarative Security

- Supported by servlet container (container-managed security)
- Authentication and authorization specified in meta data file rather than code
- Vs. Programmatic Security
  - Easier to use and maintain
  - Separate security code from normal code
  - Container dependent
  - Maybe less flexible

Encryption

- Symmetric key algorithms
  - DES, IDEA, AES, ...
- Asymmetric key algorithms
  - A.K.A. Public key algorithms
  - Diffie-Hellman Key Exchange, RSA, ...

Public Key Encryption

- Private key, public key
  - Messages encrypted with one key can only be decrypted by the other
  - Given the public key, it's virtually impossible to calculate the private key
- Applications
  - Secure email
  - Digital signature
  - ...
RSA – Key Generation

- \( p \) and \( q \) are large prime numbers and \( p \neq q \)
- \( n = p \times q \)
- \( \phi(n) = (p-1) \times (q-1) \)
- Select \( e \) where \( 1 < e < \phi(n) \), and \( e \) and \( \phi(n) \) are coprime
- Compute \( d \) where \( d \times e \equiv 1 \pmod{\phi(n)} \)
- Public key: \( d \) and \( n \)
- Private key: \( e \) and \( n \)

RSA – Encryption and Decryption

- \( c = m^e \mod n \)
- \( m = c^d \mod n \)

RSA Example

- \( p = 17 \) and \( q = 31 \)
- \( n = 527 \)
- \( \phi(n) = 480 \)
- \( e = 7 \)
- \( d = 343 \)
- \( m = 2, c = 128 \)

SSL

- Secure Socket Layer (SSL)
  - Server authentication
  - Client authentication
  - Connection encryption
- Transport Layer Security (TLS)
  - TLS 1.0 is based on SSL 3.0
  - IETF standard (RFC 2246)

SSL Handshake

- Without client authentication
  - Client information
  - Server information + certificate (include server’s public key)
  - Negotiate for a master secret
  - Requests encrypted with session key
  - Responses encrypted with session key

Certificate Authority (CA)

- CA – an entity that issues certificates
  - VeriSign, Thawte, ...
- Root certificates
  - Built into browsers
  - Import into browsers
HTTPS

- HTTP over SSL
- Configure SSL in Tomcat 5.5 -
  http://tomcat.apache.org/tomcat-5.5-
doc/ssl-howto.html