CS320 Web and Internet Programming

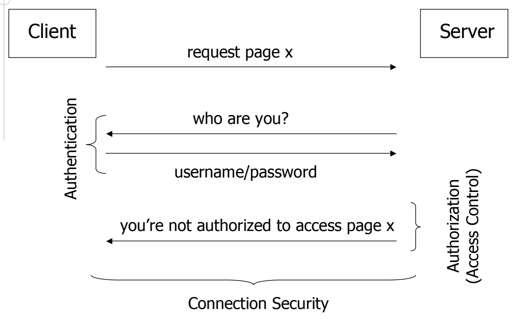
Introduction to Web Application Security

Chengyu Sun
California State University, Los Angeles

Users of Web Applications

- Multiple users
- Multiple *types* of users

Web Application Security



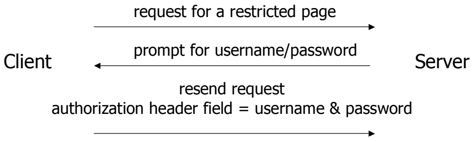
Authentication

- HTTP Basic
- HTTP Digest
- Form Based
- HTTPS Client

Chapter SRV.12, Java Servlet Specification Version 2.4

HTTP Basic Authentication

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


request for a restricted page
prompt for username/password
Client
Server
resend request
authorization header field = username & password

- Problem??

Cryptographic Hash Function…

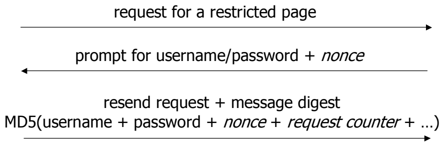
- String of arbitrary length → n bits *digest*
- Properties
  - Given a hash value, it's virtually impossible to find a message that hashes to this value
  - Given a message, it's virtually impossible to find another message that hashes to the same value
  - It's virtually impossible to find two messages 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)

HTTP Digest Authentication

- 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* + *request counter* + …)

Why nonce??

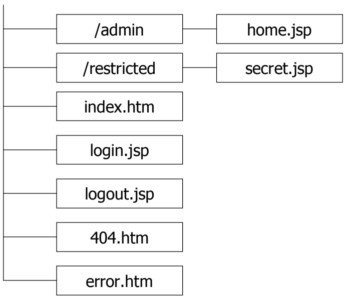
Form Based Authentication

- Both *Basic* and *Digest* authentications are supported by most of the HTTP servers
- *Form based* authentication is specific to J2EE application servers
  - 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



/admin

home.jsp

/restricted

secret.jsp

index.htm

login.jsp

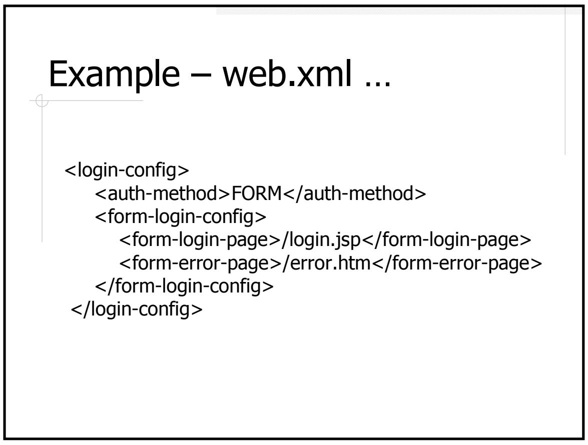
logout.jsp

404.htm

error.htm

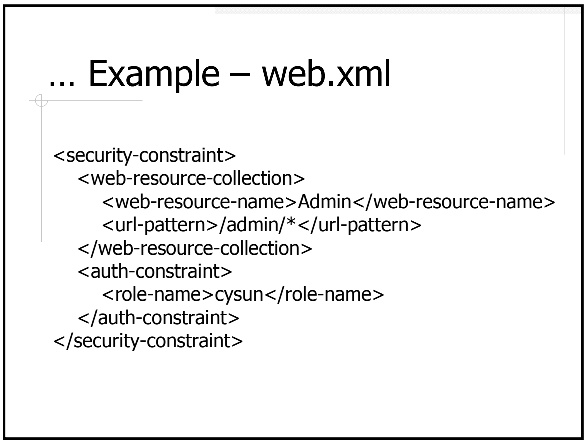
Example – Users and Roles

```
<?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>
```



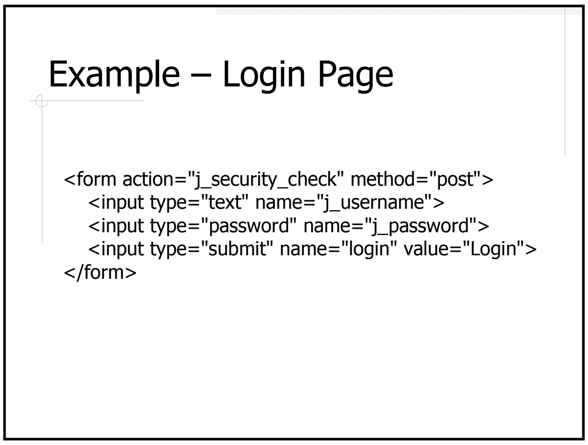
Example – web.xml …

```
<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>
```



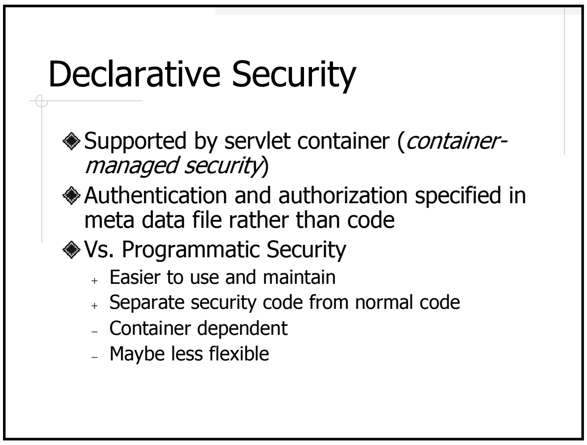
… Example – web.xml

```
<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>
```



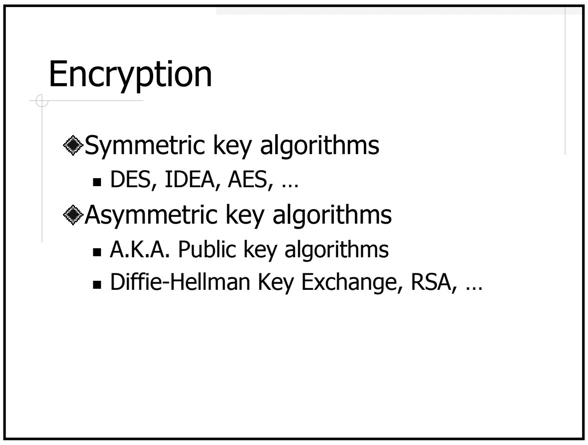
Example – Login Page

```
<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>
```



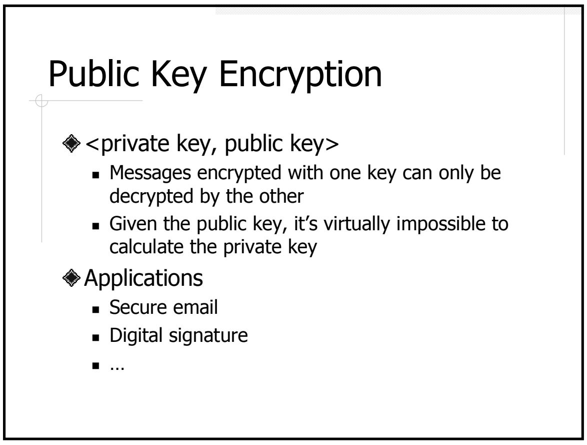
Declarative Security

- 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



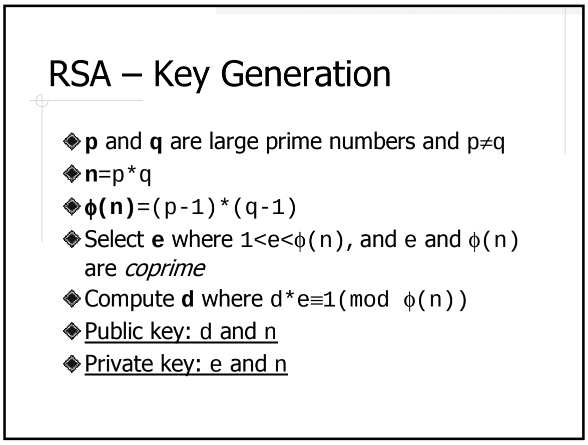
Encryption

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



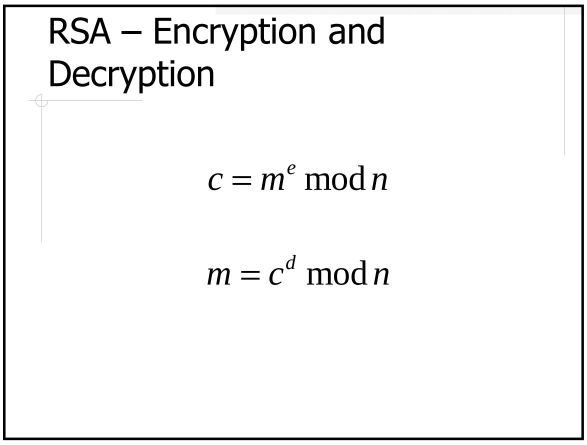
Public Key Encryption

- <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

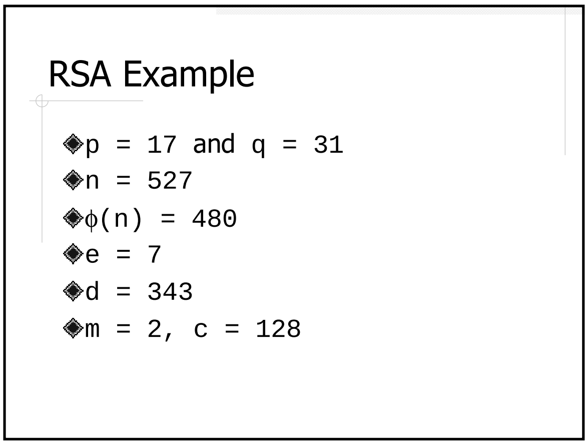
- **p** and **q** are large prime numbers and p≠q
- **n**=p*q
- **φ(n)**=(p-1)*(q-1)
- Select **e** where 1<e<φ(n), and e and φ(n) are *coprime*
- Compute **d** where d*e≡1(mod φ(n))
- Public key: d and n
- Private key: e and n



RSA – Encryption and Decryption

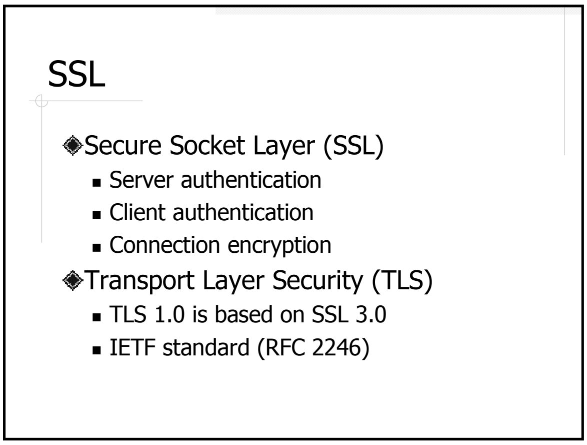
$$c = m^e \bmod n$$

$$m = c^d \bmod n$$



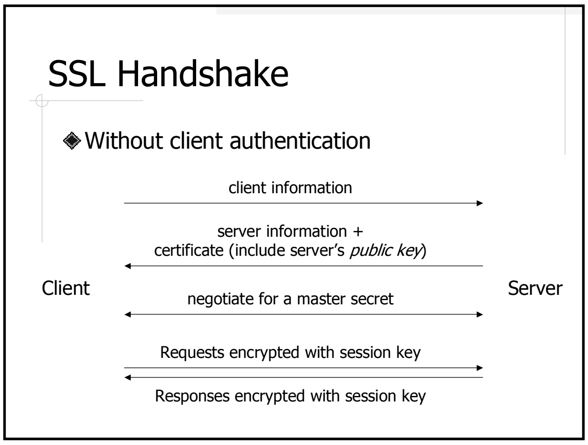
RSA Example

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



SSL

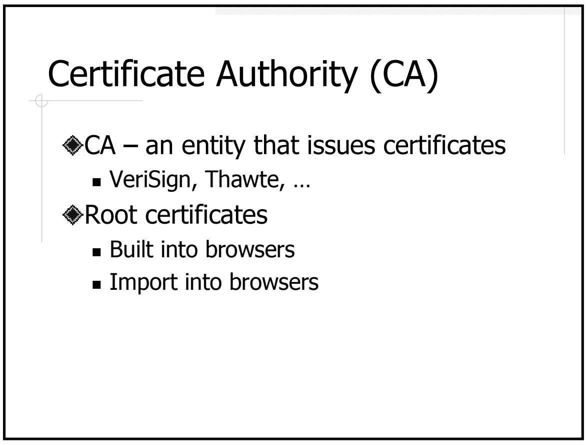
- Secure Socket Layer (SSL)
  - Server authentication
  - Client authentication
  - Connection encryption
- Transport Layer Security (TLS)
  - TLS 1.0 is based on SSL 3.0
  - IETF standard (RFC 2246)



SSL Handshake

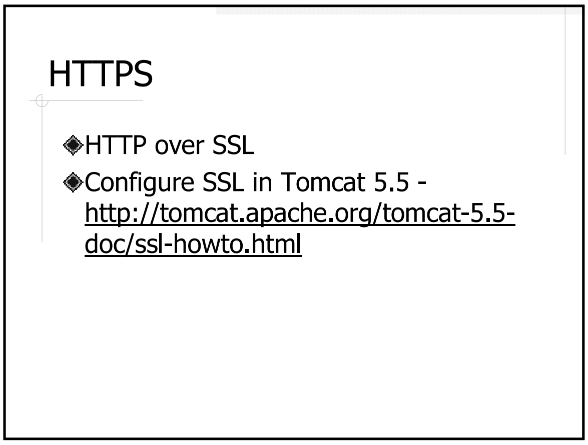
- Without client authentication

Client
Server
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