BUILD BETTER SOFTWARE SD WEST 2003

J2EE Security for Servlets, EJBs and Web services

Pankaj Kumar
Software Architect, HP

Date: March 28, 2003

Presentation Goal

Learn about security issues of relevance to Java programmers and things/APIs to know while designing and implementing secure programs using J2EE™ platform.

Contents

- 10,000 ft. view of Security
- APIs for Java Security
- J2EE and software security
- RMI Security
- Web Application Security
- EJB Security
- Web Services Security

A Brief Self Introduction

- Author of a book titled "*J2EE Security for Servlets, EJBs and Web services*" [To be published by Prentice Hall in the second half of the year].
- Have been member of a number of J2EE JSR Expert Groups (JAX-RPC, JSR109).
- Have been an Architect with HP Application Server [now discontinued] development team.
- Presently, Software Architect with HP OpenView Group.
- More than 12+ years of enterprise solution development experience. *Not a security expert*.
- Personal Home Page at: http://www.pankaj-k.net

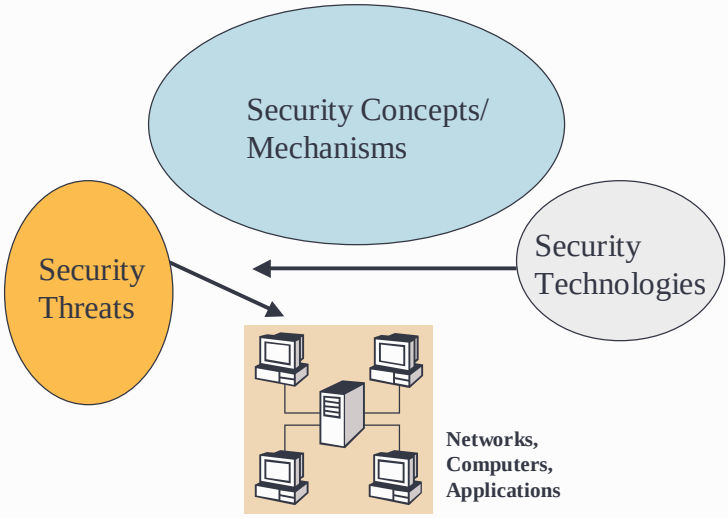
Section

- 10,000 ft. view of Security
- Java Security
- J2EE and software security
- RMI Security
- Web Application Security
- EJB Security
- Web Services Security

The Security Problem

- *July 12, 2002.* Hackers broke into USA Today's website and replaced legitimate news stories with phony articles,
- *June 13, 2002.* A Middleton, Massachusetts, woman was charged for hacking into her former boss's computer system
- *April 5, 2002.* Computer hackers cracked into the California state's personnel database
- *First week of September, 2001.* CryptoLogic Inc., a Canadian software company that develops online casino games, said a hacker had cracked one of the firm's gaming servers.
- *August 25, 2000.* Shares of Emulex Corporation fell more than sixty percent after a fake press release was posted to Internet Wire, an online news service.

10,000 ft. View of Security

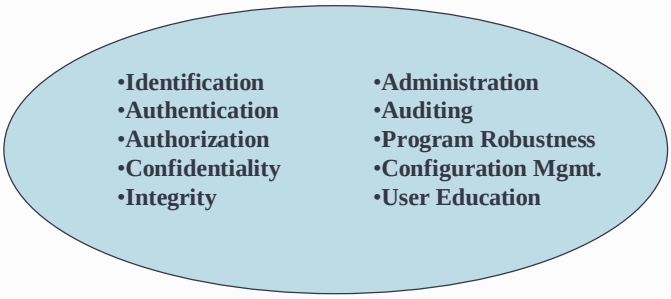


Security Threats

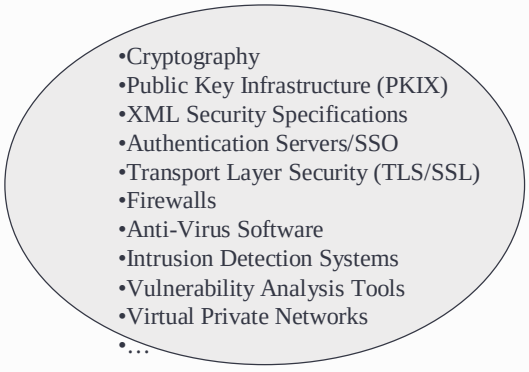
- Protocol Weaknesses
- Implementation flaws
- Insecure configuration
- Insecure design

Viruses
Worms
Trojan Horses
DoS/DDos
Password cracking
Session Hijacking
Privilege Escalation
Unauthorized Access
Network snooping
Person-in-the-middle
Spoofing
Cross Site scripting
Command Injection
…

Security Concepts/Mechanisms



Security Technologies



Contents

Java Security

- Cryptographic APIs and Tools
  - Java Cryptographic Architecture
  - Java Cryptographic Extension (JCE)
  - PKI Support
  - **`keytool`**
- Transport Layer Security (or SSL)
  - Java Secure Socket Extension (JSSE)
- Access Control
  - Access Control through policies
  - Java Authentication and Authorization Service (JASS)

Cryptographic Services
(Not an exhaustive list)

Service	Type/Algorithm
SecureRandom	SHA1PRNG
MessageDigest	SHA1, MD5
Mac	HmacMD5, HmacSHA1
Signature	SHA1withDSA, SHA1withRSA
Cipher	DES, TripleDES
KeyGenerator	DES, TripleDES
KeyPairGenerator	DiffieHellman, RSA
KeyStore	JKS, JCEKS, PKCS12
CertStore	LDAP, Collection
CertificateFactory	X509

Cryptographic API Architecture

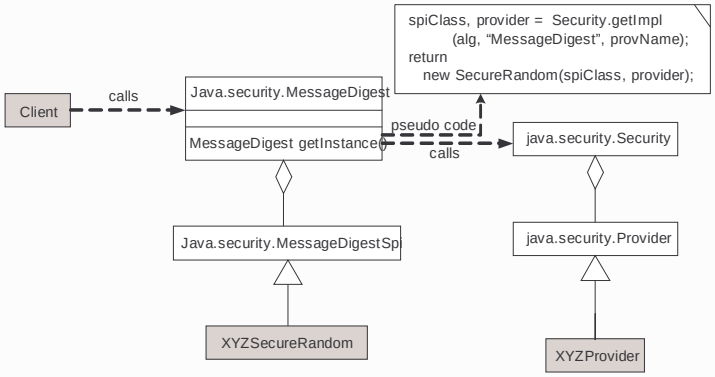
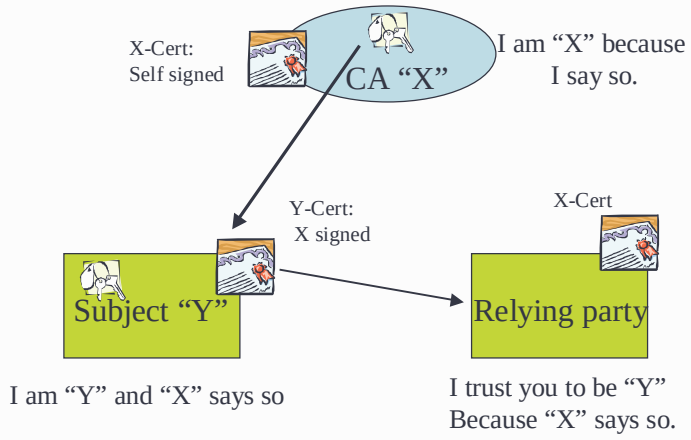


Figure 3-1: Provider Architecture Illustration with MessageDigest

Code to create MessageDigest

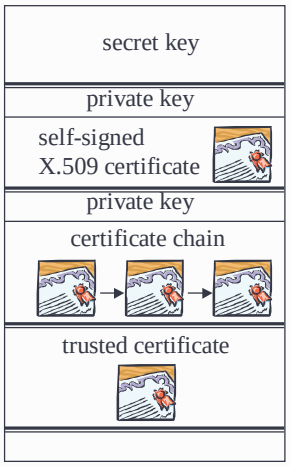
```
byte[] databytes;

//fill databytes with the
//message bytes …

MessageDigest md =
  MessageDigest.getInstance("SHA1");

md.update(databytes);
byte[] mdbytes = md.digest();
```

Public Key Infrastructure



keystore

- A password protected file to store
  - Secret keys
  - Private and Public Key pairs with self-signed X.509 certificates or a CA signed certificate with certificate chain
  - Trusted CA certificates or certificate chains
- Supported formats
  - JKS, JCEKS, PKCS12 (read-only)
- Each entry identified by an alias



`keytool`

- A command line tool to manage keystores
  - Create a keystore with a private key and self-signed certificate (`-genkey`)
  - Generate a certificate signing request (`-certreq`)
  - Import a CA signed certificate (`-import`)
  - Export a certificate (`-export`)
  - List entries (`-list`), delete entry (`-delete`), change keystore password (`-storepasswd`), change key entry password(`-keypasswd`), …

Limitations

- No tool support for cryptographic services
- Can’t sign certificates (using keytool or any API)
- Passwords entry displays password on screen (for keytool)
- Can’t export private key and certificate chain in PKCS12 format (required for use by MS IE or Netscape Navigator)
- Can’t use the certificate store of Windows
- Cipher Service not for asymmetric cryptography
- …

Java Secure Socket Extension

- Supports development of TLSv1/SSLv3 client and server programs
- SSL is a secure online communication protocol with following properties
  - Message Integrity
  - Message Confidentiality
  - Server Authentication (through X.509 certificate)
  - Optional Client Authentication
- Programming API is similar to that of Socket APIs

JSSE (Contd.)

- A client program can access HTTP over SSL simply by changing `http://`... to `https://`... using `java.net.URL` class.
- RMI communication can be setup to use SSL.
- Most of the configuration is through system properties
  - `javax.net.ssl.keyStore`
  - `javax.net.ssl.keyStoreType`
  - `javax.net.ssl.keyStorePassword`
  - `javax.net.ssl.trustStore`
  - `javax.net.ssl.trustStoreType`
  - `javax.net.ssl.trustStorePassword`

Limitations

- Limited number of cipher suits are supported
- Doesn’t work with NIO channels

Access Control

- Granular access control of specific operations on specific entities by granting permission through policy files
  - Permissions defined for SDK classes with names and action strings.
  - User programs can create their own Permissions.
- Example:

```
grant {
    permission java.io.Permission
      "${user.dir}${/}*", "read,write";
    permission java.net.SocketPermission
      "www.hp.com:1024-", "connect, resolve"
};
```

Access Control (Contd.)

- Access Control Criteria
  - Access to code loaded from a specific location (URL)
  - Access to code signed with the private key corresponding to a X.509 certificate
  - Access to code running on behalf of a user with specific identity (user id. Or role) [added by JAAS]
  - None or more of the above
- Example:

```
keystore "file:${user.dir}${/}test.ks"
grant codeBase "http://www.hp.com/-"
    signedBy "pankaj"
    Principal javax.security.auth.x500.X500Principal
      "CN=Pankaj Kumar, …" {
    };
```

Access Control (Contd.)

- Policy based checks are performed only when a Security Manager is installed
  - By default, applets run with Security Manager enabled
  - By default, standalone JVM runs without Security Manager

Section

- 10,000 ft. view of Security
- Java Security
- J2EE and software security
- RMI Security
- Web Application Security
- EJB Security
- Web Services Security

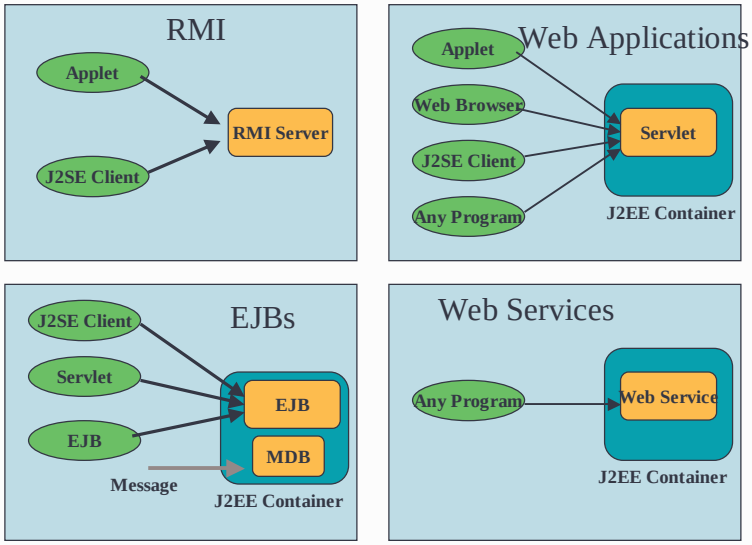
J2EE & Security

- J2EE is a platform for building *distributed*, Enterprise Solutions
- Focus is on supporting design, development and deployment of secure solutions
- Can't solve all security problems
- Contains relevant APIs for Programmers
- Contains SPIs for Security Product Vendors
- Deployment time security configuration for administrators

How does J2EE Secure Applications?

- Protects applications and users from interacting with unknown entities by supporting authentication mechanisms.
- Protects resources (URLs, EJBs, Files, …) from unsanctioned use by supporting authorization.
- Protects communication between two entities through SSL
  - Confidentiality
  - Tamper detection
  - Appropriation

Java Based Distributed Architectures



Contents

- 10,000 ft. view of Security
- Java Security
- J2EE and software security
- RMI Security
- Web Application Security
- EJB Security
- Web Services Security

RMI Security

- By default, RMI has limited security
  - Downloading of stub code from a URL requires security manager.
  - No transport level security for RMI messages but SSL can be used.
  - JAAS can be used to authenticate the client but requires significant attention to application design.

Contents

Web Application Security

Top ten Web application flaws published by OWASP (http://www.owasp.org)

1. Un-validated parameters
2. Broken Access Control ←
3. Broken Account and Session Management ←
4. Cross Site Scripting
5. Buffer Overflows ←
6. Command-line injection flaws
7. Error handling problems
8. Insecure use of cryptography ←
9. Remote Administrations Flaws
10. Web Application and Server Mis-configuration

J2EE Security for Web. Apps.

- Declarative
  - Declarative statements in deployment descriptor file web.xml
  - Adequate for most purposes
- Programmatic
  - Information about the user made available to the program through APIs
  - Program makes access control decisions
- It is common to combine these.

Access Control through Deployment Descriptor

```
<security-constraint>
  (<display-name>descriptive name</display-name>)?
  (<web-resource-collection>
    <web-resource-name>desc-name</web-resource-name>
    (<description>descriptive text</description>)?
    (<url-pattern>url pattern</uri-pattern>)*
    (<http-method>http method</http-method>)*
  </web-resource-collection>)+
```

Which URLs to protect?

Which HTTP Methods?
GET, POST,
PUT, DELETE,
HEAD

Access Control through Deployment Descriptor (Contd.)

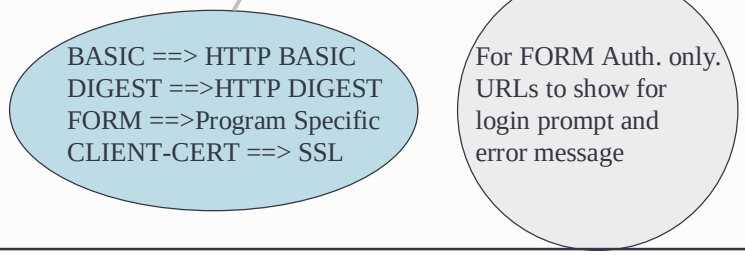
```
(<auth-constraint>
    (<description>descriptive text</description>)?
    (<role-name>user role</user-role>)*
  </auth-constraint>)?
  (<user-data-constraint>
    (<description>descriptive text</description>)?
    (<transport-guarantee>tg</transport-guarantee>)
  </user-data-constraint>)?
</security-constraint>
```

Which users?

What transport?
NONE ==> plain HTTP
INTEGRAL, CONFIDENTIAL
==> HTTP over SSL

User Login

```
<login-config>
  (<auth-method>auth. mechanism</auth-method>)?
  (<realm-name>realm id. String</realm-name>)?
  (<form-login-config>
    <form-login-page>login-url</form-login-page>
    <form-error-page>error-url</form-error-page>
  </form-login-config>)?
</login-config>
```



Programmatic Security

- Methods in `HttpServletRequest` class
  - `String getRemoteUser()`
  - `boolean isUserInRole(String role)`
  - `Java.security.Principal getUserPrincipal()`
- Example:

```
...
if (!req.isUserInRole(“payinguser”)){
  return;
}
...
```

Contents

- 10,000 ft. view of Security
- J2EE and software security
- APIs for Java Security
- RMI Security
- Web Application Security
- EJB Security
- Web Services Security

EJB Security

- Separation of security responsibilities – Bean Provider, Application assembler, Deployer and System Administrator
- Authenticate the Caller
- Access Control per EJB, per operation
- Allow Caller Identity Propagation
- Allow Caller Identity Delegation
- Protect Message on the Wire
- Interoperate with CORBA !!

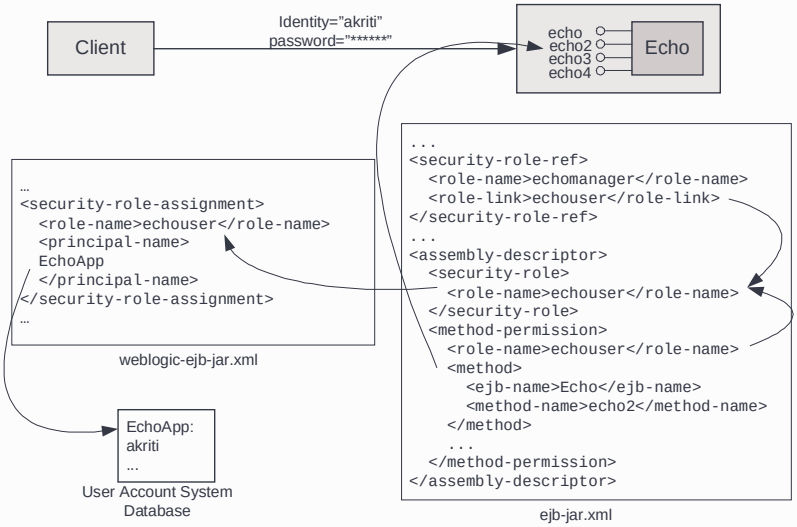
Separation of Security Responsibilities

- Programmatic Security (Bean Provider)
  - `java.security.Principal getCallerId();`
  - `isCallerInRole(String roleName)`
- Declarative Security in ejb-jar.xml file (Application Assembler)
  - `<security-role-ref>`
  - `<security-role>`
  - `<method-permission>`
- Mapping to Container Specific mechanisms (Deployer)
- User Creation/Modification/Removal (Administrator)

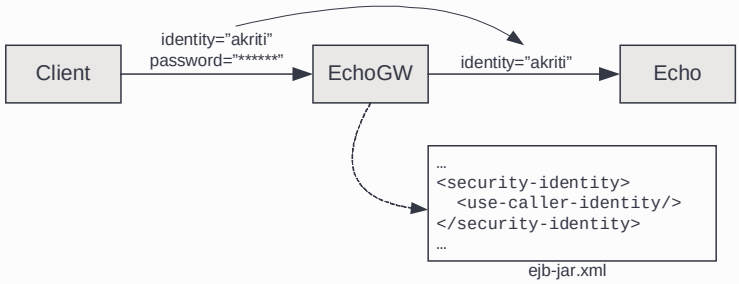
Caller Authentication

- JNDI based authentication
  - Caller specifies user credentials as properties to JNDI context
    - Username and password
    - X.509 certificate
- JAAS Authentication
  - Using LoginModules
- Authentication by Web Application

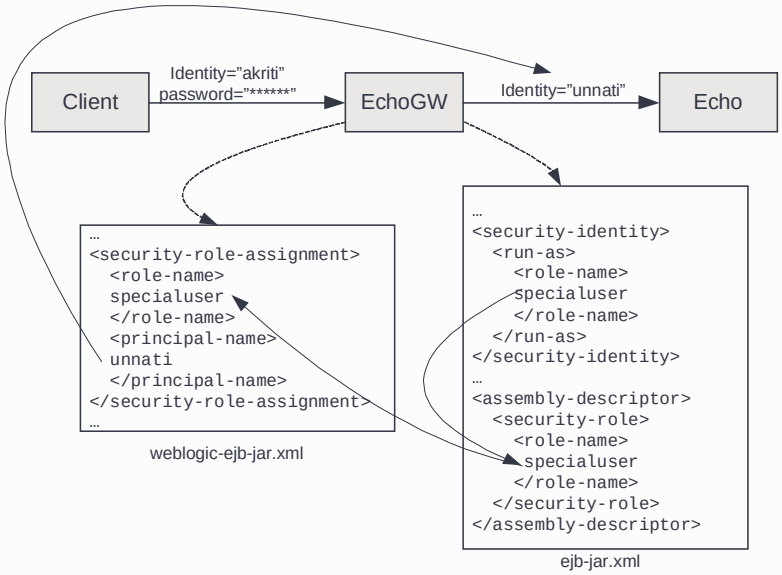
A Secured EJB



Identity Propagation



Identity Delegation



Other aspects of EJB Security

- Uses SSL for network security
- Uses CSIv2 for identity assertion across EJB containers and to interoperate with CORBA
- Need not use JAAS for Access Control !

Contents

- 10,000 ft. view of Security
- J2EE and software security
- APIs for Java Security
- RMI Security
- Web Application Security
- EJB Security
- Web Services Security

Web Services Security

- Transport level security same as Web Applications (not EJBs !)
  - Amounts to using HTTP over SSL
  - No message level protection
- Not Adequate for end-to-end security

Web Services Security

- A number of XML based security standards are now available
  - XML Signature – Message Level Authentication and tamper-evident. (W3C)
  - XML Encryption – Message Level Privacy (W3C)
  - XML Trust Services – Key Management (W3C)
  - SAML – Security Assertion Markup language (OASIS)
  - XACML – Extensible Access Control Markup Language (OASIS)

Web Services Security

- Java APIs are getting defined
  - JSR 104: XML Trust Service API
  - JSR 105: XML Digital Signature APIs
  - JSR 106: XML Digital Encryption APIs
  - JSR 155: Web Services Security Assertions
  - JSR 183: Web Services Message Security APIs.

Q&A