Deployment

Topics

- Basic Security
  - Keys
  - Signing
- JAR
- WebStart

Public Key Cryptography

- Relies on a keypair
  - Private key - must be carefully guarded
  - Public key - should be freely distributed
- Generated at the same time
- Can be used to prove identity or make communications private
- `java.security` and its sub-packages

Keys

- Keys have:
  - An algorithm
    - DSA, RSA
  - An encoding
    - The encoded key
  - A format
    - Name of the principal encoding, i.e. X.509

Keystore

- A database for storing:
  - Keys
    - Your private keys
  - Trusted certificates
    - Public key certificates of another party
- Entries have an alias
- JDK provided tool `keytool` manages `Keystore`

Digital Signature Algorithms (DSA)

- Similar to message digests, but…
- Can't get away with altering signature
  - Send
    - Message
    - Signature
    - Public key
  - Validate
    - Verify signature using public key

Generating a Key Pair

- **Programmatically**

```
KeyPairGenerator keygen;
keygen = KeyPairGenerator.getInstance( "DSA" );
keygen.initialize( 512 );  // or 1024
KeyPair keyPair = keygen.genKeyPair();
PublicKey publicKey = keyPair.getPublic();
PrivateKey privateKey = keyPair.getPrivate();
```

Creating a Key and Keystore

```
>keytool -genkey -keystore russKeyStore -alias russ
Enter keystore password:  password
What is your first and last name?
  [Unknown]:  Russ Moul
What is the name of your organizational unit?
  [Unknown]:  UW Extension
What is the name of your organization?
  [Unknown]:  University of Washington
What is the name of your City or Locality?
  [Unknown]:  Seattle
What is the name of your State or Province?
  [Unknown]:  WA
What is the two-letter country code for this unit?
  [Unknown]:  US
Is CN=Russ Moul, OU=UW Extension, O=University of Washington,
   L=Seattle, ST=WA,
C=US correct?
  [no]:  yes

Enter key password for <russ>
        (RETURN if same as keystore password):  russmoul
```

Generating a Signature

- **Programmatically**

```
Signature signature =Signature.getInstance( "DSA" );
signature.initSign( privateKey );
signature.update( contentBytes );
byte[] signatureBytes = signature.sign();
```

Verifying a Signature

- **Programmatically**

```
Signature verify = Signature.getInstance( "DSA" );
verify.initVerify( publicKey );
verify.update( contentBytes );
boolean ok = verify.verify( signatureBytes );
```

JAR

- Java Archive
  - Cross platform
  - Handles any file type
    - audio, images, classes,…
  - Open standard

JAR

- Benefits
  - Security
  - Reduced download time
  - Compression
  - Extension packaging
  - Package sealing
  - Package versioning
  - Portable

Manifest

- Contains meta data about the other contents of the JAR file
- Specific name and location
  - `META-INF/MANIFEST.MF`
- Specific format
  - *`Element-Name: value`*

Fundamental

- Version of the manifest file format
- The class to run when executing the jar
- Manifest entries
  - Manifest-Version
  - Main-Class

```
Manifest-Version: 1.0
Created-By: Apache Ant 1.5.1
Main-Class: org.apache.tools.ant.Main
...
```

Extensions

- Manifest entry
  - Lists supporting jar files using relative paths
  - Class-Path header field
- Extension JARs may reference each other
- May use multiple `Class-Path` headers
- When executing with `–jar` option no other class path is recognized

```
Manifest-Version: 1.0
Created-By: Apache Ant 1.5.1
Main-Class: org.apache.tools.ant.Main
Class-Path: xml-apis.jar xercesImpl.jar optional.jar xalan.jar
...
```

Using ant to Create a Manifest

```
<jar destfile="${path.dist}/stockquoter.jar">
  <fileset dir="${path.classes}">
    .
    .
    .
  </fileset>
  <manifest>
    <attribute name="Main-Class“
               value="edu.uw.rgm.quote.StockQuoter"/>
    <attribute name="Class-Path" value="xerces.jar"/>
  </manifest>
</jar>
```

Package Sealing

- Sealed packages require all files in the package to be in the same JAR file
- May seal all packages
- Manifest entry
  - Name header field, names the class or package
    - All entries after the Name header, without an interceding blank line, apply to the named package
  - Sealed header field

Sealing Examples

- **Seal all packages**

```
Manifest-Version: 1.0
Sealed: true
```

- **Seal a specific package**

```
Manifest-Version: 1.0

Name: edu/washington/ext/cp130/framework/exchange
Sealed: true
```

Package Versioning

- **Provides version information for a package**
- **Manifest entry**
  - **Must immediately follow the** `Name` **entry**
  - `Specification-Title`
  - `Specification-Version`
  - `Specification-Vendor`
  - `Implementation-Title`
  - `Implementation-Version`
  - `Implementation-Vendor`

Versioning Example

```
Manifest-Version: 1.0
Created-By: Apache Ant 1.5.1
Main-Class: org.apache.tools.ant.Main
Class-Path: xml-apis.jar xercesImpl.jar optional.jar xalan.jar

Name: org/apache/tools/ant/
Extension-name: org.apache.tools.ant
Specification-Title: Apache Ant
Specification-Version: 1.5.1
Specification-Vendor: Apache Software Foundation
Implementation-Title: org.apache.tools.ant
Implementation-Version: 1.5.1
Implementation-Vendor: Apache Software Foundation
```

Signing a JAR File

- **Need signing certificate in keystore**
- **Signing**
  - **Adds** `<signer>.ds` **and** `<signer>.sf` **files to JAR file**
  - **Generates** `Name` **and** `SHA1-Digest` **manifest entries for each file in the jar**

```
Name: edu/uw/rgm/quote/NasdaqQuoter$NasdaqException.class
SHA1-Digest: 0Iw2dJcrKWFPOzFgLDW0uSbOXKw=
```

Creating a Signing Certificate

```
>keytool -selfcert -alias russ -keystore russKeyStore
Enter keystore password:  password
Enter key password for <russ>russmoul

>keytool -list -keystore russKeyStore
Enter keystore password:  password

Keystore type: jks
Keystore provider: SUN

Your keystore contains 1 entry

russ, Aug 17, 2003, keyEntry,
Certificate fingerprint (MD5): AC:4F:7A:EC:AE:
   1E:D9:59:A6:E1:29:65:9B:50:64:BE
```

Signing with `jarsigner`

```
> jarsigner –keystore russKeyStore framework.jar russ
```

Signing with ant

```
<signjar jar="${path.dist}/stockquoter.jar"
            keystore="russKeyStore" storepass="password"
            alias="russ" keypass="russmoul"/>
```

Loading Resources From JAR

- Reads files/resources using classloader
- Can read any file in the CLASSPATH
- Two `Class` methods:

```
URL getResource( String name )
InputStream getResourceAsStream( String name )
```

  - Paths are relative to the class' package
- Two `ClassLoader` methods

```
URL getSystemResource( String name )
InputStream getSystemResourceAsStream( String name )
```

Break

WebStart

- Supports the deployment of an application using the web infrastructure
- Applications are deployed/downloaded using a web browser
- Applications run within a sandbox unless they are signed
- WebStart must be installed on the client
  - May have been installed with JDK installation
  - May be installed from the web site

Sandbox

- Restrictions
  - No access to hard disk
  - All JAR files must be downloaded from same host
  - Network connection allowed back download host only
  - No security manager may be installed
  - No native libraries
  - Limited access to system properties
    - Read/write properties defined in JNLP file
    - Read-only access to “applet” property set

WebStart Development

- Configure web server for Java WebStart MIME type
- Create JNLP file for application
- Make application accessible on web server
- Create link to JNLP file

```
<a href="app.jnlp">The Application</a>
```

Web Server Configuration

- **Files with** `.jnlp` **extension are set to the** `application/x-java-jnlp-file` **MIME type**
- **Configuration varies from server to server**

JNLP File

```
<?xml version="1.0" encoding="utf-8"?>
  <!-- JNLP File for SwingSet2 Demo Application -->
  <jnlp spec="1.0+" codebase="http://my_company.com/jaws/apps"
        href="swingset2.jnlp">
  <information>
    <title>SwingSet2 Demo Application</title>
    <vendor>Sun Microsystems, Inc.</vendor>
    <homepage href="docs/help.html"/>
    <description>SwingSet2 Demo Application</description>
    <description kind="short">Swing demo.</description>
    <icon href="images/swingset2.jpg"/>
    <icon kind="splash" href="images/splash.gif"/>
    <offline-allowed/>
  </information>
  <security>
      <all-permissions/>
  </security>
  <resources>
    <j2se version="1.4.2"/>
    <jar href="lib/SwingSet2.jar"/>
  </resources>
  <application-desc main-class="SwingSet2"/>
</jnlp>
```

JNLP Element

- `spec`
  - **Must be `1.0` or higher, `"1.0+"` is default**
- `codebase`
  - **All relative URLs in `href` attributes use this as base**
- `href`
  - **URL to the JNLP file itself**

Information Element

- `title`
  - **The name of the application**
- `vendor`
  - **Name of the applications vendor**
- `homepage`
  - **A single `href` attribute referencing the applications home page**

Information Element

- `description`
  - **A short statement about the application**
  - **An optional** `kind` **attribute**
    - `one-line`, **used where one line is available**
    - `short`, **used where a paragraph is available**
    - `tooltip`, **used for tool tip**
- `icon`
  - **An** `href` **attribute referencing a GIF or JPEG**
  - **Optional** `width` **and** `height` **attributes**
  - **Optional** `kind="splash"` **attribute**

Information Element

- `offline-allowed`
  - **Optional, indicates if the application can be launched offline**
  - **Influences how WebStart checks for an update**

Security Element

- `all-permissions`
  - **If this sub-element is present the application will have full access to client machine and network, all JAR files must be signed**

Resources Element

- `os`
  - **Optional, specifies what O/S this resource is valid for**
- `arch`
  - **Optional, specifies what architecture this resource is valid for**
- `j2se`
  - **Optional sub-element, with attributes**
    - `version` **, the required JRE version (e.g.** `"1.4"`, `"1.4+"` **)**
    - `initial-heap-size` **, optional**

Resources Sub-Elements

- `jar`
  - **Identifies application's JAR files**
    - `href`, **URL to the JAR file**
    - `Download="lazy"`, **optional, resources are downloaded as needed, default is all resources are downloaded before application is launched**
- `nativelib`
  - **Optional sub-element, with attributes**
    - `href`, **URL to the library file**

Application-Desc Elements

- `main-class`
  - Optional attribute identifying the “main” class
- `argument`
  - Optional sub-element, specifying arguments to the application’s “main” method