X.509 certificates and keys

Basic features and managing tools

What are X.509 certificates?

- Part of X.509 Public Key Infrastructure standard
- Assumes strict hierarchical system of certificate authorities
- Infrastructure based on trust
- Used for authentication, integrity and privacy assurance

How does X.509 certificate look like?

- Certificate
  - Version Number
  - Serial Number
  - Signature Algorithm ID
  - Issuer Name
  - Validity period
    - Not Before
    - Not After
  - Subject name
  - Subject Public Key Info
    - Public Key Algorithm
    - Subject Public Key
  - Issuer Unique Identifier (optional)
  - Subject Unique Identifier (optional)
  - Extensions (optional)
    - ...
- Certificate Signature Algorithm
- Certificate Signature

Overview

Basic features of X.509 certificate and key managing tools

General features

Tools	General				
	Type	View information about certificate	License	Operating System	Tool Type
OpenSSL	PKCS12	Yes	Public	All	Command Line
Keytool	JKS	Yes	Public	All	Command Line
GnuPG	None	No	Public	All	Command Line
Keystore Explorer	Both	Yes	Public	Win | OSX	Graphical Interface
XCA	PKCS12	Yes	Public	All	Graphical Interface
GnuTLS	PKCS12	Yes	Public	UNIX	Command Line
Makecert and pvk2pfx	PKCS12	No	Public	Windows	Command Line
Windows PowerShell PKI Module	PKCS12	Yes	Public	Windows	Command Line

Generation of keys and Basic Constraints

Tool	Generate keys				Basic Constraints	
	self-signed certificate	CSR	Specify length	Specify algorithm	Specify Type	Specify path length
OpenSSL	Yes	Yes	Yes	Yes	Yes	Yes
Keytool	Yes	Yes	Yes	Yes	Yes	Yes
GnuPG	No	Yes	No	No	No	No
Keystore explorer	Yes	Yes	Yes	Yes	Yes	Yes
XCA	Yes	Yes	Yes	Yes	Yes	Yes
GnuTLS	Yes	Yes	Yes	Yes	Yes	Yes
Makecert and pvk2pfx	Yes	No	Yes	Yes	Yes	Yes
Windows PowerShell PKI Module	Yes	No	No	No	No	No

Specifications

Tool	CSR signing	Privkey + signed chain	Specify certificate validity	SAN for end certificates	Support for CSP
OpenSSL	Yes	Yes	Yes	Yes	Yes
Keytool	Yes	Yes	Yes	Yes	Yes
GnuPG	No	No	No	No	No
Keystore explorer	Yes	Yes	Yes	Yes	No
XCA	Yes	Yes	Yes	Yes	No
GnuTLS	Yes	No	Yes	Yes	No
Makecert and pvk2pfx	No	No	Yes	No	Yes
Windows PowerShell PKI Module	No	No	Yes	Yes	Yes

Exporting and conversion

Tools	Conversions			
	Exporting		Direct JKS and PKCS12	Import certificate and private key into a file
	Certificate or chain from file	Private key only		
OpenSSL	Yes	Yes	No	Yes
Keytool	Yes	No	Yes	Yes
GnuPG	No	No	No	No
Keystore explorer	Yes	Yes	Yes	Yes
XCA	Yes	Yes	No	Yes
GnuTLS	Yes	No	No	Yes
Makecert	No	No	No	Yes
Windows PowerShell PKI Module	Yes	Yes	No	Yes

OpenSSL

Practical example

Viewing certificates

- PEM encoded:
  - `openssl x509 -in [cert_file] -text -noout`
- DER encoded
  - `openssl x509 -in [cert_file] -inform der -text -noout`

Generate self-signed certificate and new key pair

- Generating RSA of certain length and using SHA-256 hash
  - ```
    openssl req -x509 -sha256 -nodes -days [validity] \
                -newkey rsa:[length] -keyout [output_key].key \
                -out certificate_name.crt
    ```
- Creating certificate signing request (CSR)
  - ```
    openssl req -out CSR.csr -new -newkey rsa:[length] \
                -nodes -keyout privateKey.key
    ```

Basic constraints and extra options

- To use OpenSSL effectively, directories hierarchy and configuration file are needed
- To sign CSR
- Choose whether certificates issued will be CA or end certificates
- Choose maximum path length
- Set Subject Alternative Name

Exporting

- Certificate or Certificate chain from a file:
  - `openssl pkcs12 –in [pkcs12_file] [-nokeys] [-clcerts|-cacerts] \`
    `-out [output_file]`
  - Use `–clcert` for certificate only and `–cacerts` for chain
- Exporting keys
  - `openssl pkcs12 –in [pkcs12_file] –nocerts [-nodes] –out [output_file]`
    - Using `-nodes` exports unencrypted keys
- Importing certificates and keys into PKCS#12 file:
  - `openssl pkcs12 -export -out [output_file.p12] \`
    `-inkey [private_key] -in [certificate_file] \`
    `-certfile [ca_certificate]`

Java keytool

Practical example

Viewing certificates

- PEM or DER encoded:
  - `keytool -printcert -file [certificate_file]`
- Information about Java Keystore or PKCS#12 file:
  - `keytool -list -v -keystore [store_file]`

Generate self-signed certificate and new key pair

- Generating RSA of certain length and using SHA-256 hash
- ```
  keytool -genkey -keyalg [RSA|DSA] -alias [aliasname] \
          -keystore [keystore_file.jks] -storepass [password] \
          -validity [days] -keysize [length]
  ```
- Creating certificate signing request (CSR)
  - First, generation of new key pair is required:
    - ```
      keytool -genkey -alias namealias -keyalg [RSA|DSA] \
      -keysize [length] -keystore [keystore_path | new_keystore_name]
      ```
  - Then you create a CSR with new key pair
    - ```
      keytool -certreq -keyalg [keyalg_from_genkey] -alias [alias_from_genkey] \
      -file certreq.csr -keystore [jks_from_genkey]
      ```
      ```

Conversion between JKS and PKCS#12

- Conversion JKS -> PKCS#12
  - ```
    keytool -importkeystore -srckeystore [existing_keystore.jks] \
    -destkeystore [destination_filename.p12|pfx] -srcstoretype JKS \
    -deststoretype PKCS12 -srcstorepass [.jks password] \
    -deststorepass [same as .jks password] -srcalias [source_alias] \
    -destalias [source_alias] -srckeypass [source_key_password] \
    -destkeypass [source_key_password] –noprompt
    ```
- Conversion PKCS#12 -> JKS:
  - ```
    keytool -importkeystore -srckeystore [PKCS12_file.pfx|.p12] \
    -srcstoretype pkcs12 -destkeystore [dest_keystore_name.jks] \
    -deststoretype JKS
    ```

Exporting

- Certificate with alias `[alias]`
  - `keytool -export -alias [alias] -file [output.crt] -keystore [source_keystore]`
    - Exported certificate will be DER encoded
- Exporting keys
  - Exporting private key only is not possible with keytool, conversion to PKCS#12 and using other tool (such as OpenSSL) is needed
- Importing certificates and keys into PKCS#12 file:
  - `keytool -import -alias [certificate_chosen_alias] \-file [certificate_file.ext] -keystore [target_keystore]`

Questions?