SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

Established in collaboration with MIT

ISTD 50.005, Spring 2015
Signed certificate verification

1 Objective:

This supplemental document shows you how to (i) verify the server's signed certificate using CA's certificate and (ii) extract public key from signed certificates.

2 Background

You know from the lectures that public key will never be sent alone, the owner of this public key always sends his certificate, and the recipient will extract sender's public key from the received certificate. In addition, a certificate needs to be signed by trusted Certificate Authorities (CAs) before it can be used. Roughly speaking, to sign a certificate request, a CA just needs to encrypt the certificate request using his private key. Therefore, it's quite straightforward for you to verify a signed certificate: (1) extract CA's public key from CA's certificate, and (2) verify server's signed certificate using a CA's public key obtained from (1). To complete (1) and (2) you need to use X509Certificate class in Java.

3 X509Certificate Class

X509Certificate class provides basic functions for (signed) certificate verification, checking (signed) certificate validity and extracting public key from a signed certificate. You can refer http://docs.oracle.com/javase/7/docs/api/javax/security/cert/X509Certificate.html for documentation. To use it, you first need to create X509Certificate object.

3.1 Create X509Certificate object

To create X509Certificate object, you can use one of the following ways:

1. Instantiates an X509Certificate object, and initializes it with the specified byte array.

```
public static final X509Certificate getInstance(byte[] certData);
```

2. Instantiates an X509Certificate object, and initializes it with the data read from the input stream inStream.

```
public static final X509Certificate getInstance(InputStream inStream)
```

Assume that the certificate is stored in your computer with name: CA.crt, the following example shows you how to create X509Certificate object from file CA.crt.

```
InputStream inStream = new FileInputStream("CA.crt");
X509Certificate CAcert = X509Certificate.getInstance(inStream);
```

3.2 Extract public key from X509Certificate object

To extract public key from X509Certificate object, you need to use method getPublicKey(), a method inherited from Certificate class. Its prototype is:

```
public abstract PublicKey getPublicKey();
```

Given a X509Certificate object (CAcert) which has been created in section 3.1, here is how we extract public key from this object:

```
PublicKey key = CAcert.getPublicKey();
```

3.3 Check validity and verify signed certificate.

To check if a certificate is currently valid or not, use the following method:

```
public abstract void checkValidity();
```

It is valid if the current date and time are within the validity period given in the certificate.
To verify a signed certificate we use function verify():

```
public abstract void verify(PublicKey key);
```

The PublicKey parameter is the public key of CA (Refer to 3.2 to know how to extract public key). Assume that public key of CA is CAkey, the X509Certificate object created from the server signed certificate is ServerCert, here is an example of how to verify and check validity:

```
ServerCert.checkValidity();
ServerCert.verify(CAkey);
```

4 What you need to do

You can download your signed certificate and CA's certificate (Project Files) from eDimension (Project: signed certificate; Project Files) and complete the project.