SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

Established in collaboration with MIT

ISDT 50.005 (Computer System Engineering)
Generating RSA certificate request

1 Objective

This document tells you how to generate an RSA certificate request for your secure file transfer project.

2 Background

In RSA cryptography, public keys are supposed to be public knowledge, i.e., everyone knows everyone else's public key. In reality, however, we need to assure the *authenticity* of the public key, i.e., that a certain public key presented to us indeed belongs to the entity (person, company, etc) that claims to own it. This issue is typically resolved with a key *certificate*. A certificate contains several fields that bind a public key to a subject (i.e., true owner).

Fig. 1 illustrates the data fields of an **X.509** certificate. X.509 is widely used for authentication in Internet communication, including secure IP (the IPsec protocol) and secure sockets (TLS or SSL).

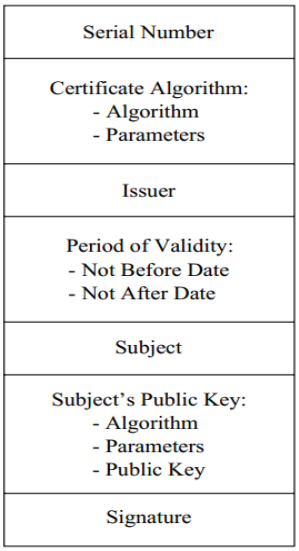

Fig. 1. X509 certificate.

The main idea is that after you have generated your RSA public key (e.g., using JCE like you did in last week's lab, or by OpenSSL according to this handout), you'll need to *register it with a trusted authority*. As mentioned in your project handout, this authority is typically a commercial provider of key certification like VeriSign, or a government agency like IDA.

In your project, however, you'll use your teaching staff (CSE-CA) for the certification. To use our service, you'll only need to generate a *certificate request* and upload it to eDimension. We'll then verify your information. If we're satisfied, we'll sign a certificate that binds your public key to your, and give you the certificate. If people indeed trust us (i.e., CSE-CA), they'll then trust the signed certificate and your public key contained therein.

You'll use OpenSSL to generate your certificate request. Section 3 gives the steps that you'll need to follow.

3 Generate certificate request

Use the following steps to generate your RSA key pair and a certificate request using OpenSSL.

- Generate an RSA key pair:

```
$openssl genrsa –out privateServer.pem 1024
```

The RSA key pair will be stored in the output file in `.pem` format. The number 1024 specifies the key size in bits.

- Using the key pair, generate the certificate request:

```
$openssl req -new -key privateServer.pem -out server.csr
```

The certificate request is now stored in the output file in `.csr` format. **Please upload this request to eDimension (Project: submit certificate request) by April 5, 2015 (Sunday).**

Also, since you generated your key pair using OpenSSL, to use it in a Java program, you'll need to convert the keys to `.der` format. Do the following to convert.

- Save the private key in `.der` format for your own use:

```
$openssl pkcs8 -topk8 -in privateServer.pem -outform der
-out privateServer.der –nocrypt
```

With the `-nocrypt` option, the command will output an unencrypted PrivateKeyInfo data structure, which is fine because the private key is for your own use. Please read the man page for `openssl` for the meanings of the other command line parameters.

- Export the public key in `.der` format:

```
$openssl rsa -in privateServer.pem -pubout -outform der
-out publicServer.der
```