Prof. Dr. Christian Schindelhauer
Shou-Yu Chao, Lulu Cai

Freiburg, 2009-12-16

Practical Exercises
Communication Systems (Rechnernetze II)
Topic 15: SSL

Exercise 1:
SSL uses a hybrid approach consist of a Asymmetric-key RSA algorithm at the initial stage and symmetric encryption for communicating actual data.

1. Could you explain why not just use RSA but instead using a complex hybrid approach?
2. Could you compare the speed of RSA and AES with the same amount of data?

Exercise 2:
In the 1990s, United States export policy forbid certain encryption technologies to be export overseas. RC4 128-bit cipher are among the list. At that moment, International version of browsers could have maximum 40-bit key length. Could you tell how easy is it to crack this level of encryption?

Exercise 3:
Establish a SSL connection with a remote host.

1. Under Linux terminal type:
   openssl s_client -connect meine.deutsche-bank.de:443 and

   openssl s_client -connect www.bankofamerica.com:443

   You will notice that Bank of America is using RC4 and Deutsche Bank is using AES256.
2. RC-4 is the one of the most frequently used symmetric encryption cipher of SSL. Certain wireless network encryption also uses RC-4. However after several attacking scheme being proposed, AES(256 bit) is becoming more in favor. Try to enumerate some weakness of RC-4.

Exercise 4:
A lot of protocols could be wrapped by SSL(TLS) layer but they all need a certificate to work. We now try to generate a homebrew certificate.

1. We first generate a Private key by following command:
   openssl genrsa -des3 -out server.key 1024
   You will be asked for a password. Try to memorize it and you will use it again in the following step.

2. Now we need to generate a Certificate Signing Request(CSR) by following command:

   ```
   openssl req -new -key server.key -out server.csr
   ```

   You will be asked for the password in the previous step and some company profile. Normally we will upload this CSR to a certificate authority(CA) and all data must be accurate in order to pass the review process. But since we are making a homebrew certificate, you could fill whatever you like.

3. We will need to remove the password protection from the key in order to import it into the server application:

   ```
   cp server.key server.key.org
   openssl rsa -in server.key.org -out server.key
   ```

4. We are now generating a Self-Signed Certificate:

   ```
   openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt
   ```

   Now we have server.crt and server.key. You can import these pair of key into server application such as Apache with mod_ssl

5. VeriSign charges US$399 per year for a one site SSL certificate or US$995 per year for a extended validation SSL certificate(Although there are some cheaper alternatives). Everyone could generate certificates at home but why these companies could print certificate and make money from it? Discuss who gave them the right to sell these certificate.

6. Open your Firefox browser and go to: Edit/Preference/Advanced/Encryption/View Certificates/Authorities to see which certificate authorities do you trust.