Intro. to Computer Security: Exercise 1

Deadline: 22/1/2016 (Friday) 12:00pm noon

1 Introduction

The exercises for this module will be based on analysing the security of a virtual machine. For this exercise you must download the VM, get it working and perform a number of cryptographic operations.

2 Exercises

2.1 Get the VM working

You first need to download the VM image from the module's exercises page. The direct link is:

`https://www.cs.bham.ac.uk/internal/courses/intro-comp-sec/ICS_VM.ova`

To run this you will need a copy of VirtualBox, which you can download from: `https://www.virtualbox.org/wiki/Downloads`.

Install VirtualBox (using the default options recommended by the installer), open it, and and go to *File* → *Import Appliance...* to import the image. Press the green *Start* button to start the VM. After it finishes booting, you should see a login screen. I *strongly* recommend using Linux (or Mac) as the host operating systems, if you currently use Windows I recommend that you install Linux.

There are many user accounts on the VM; you begin the module with access to just two:

Username: `alice`, password: `alice`
Username: `bob`, password: `bob`

You must use your own copy of the VM for this (and every other) exercise. You must not share your VM with other students.

In Alice's home directory you will find a token in the file `theFirstToken`. Submit this token to on the website:

`https://www.cs.bham.ac.uk/internal/courses/comp-sec/tokens`

[1 marks]

2.2 Using Crypto Tools

Submit your answers for this part of the exercise using *Quiz 1* on the Canvas module page.

1. In the `gpg` directory, there are four files: `message.txt.asc`, `jts.txt.asc`, `bob.txt.gpg` and `afnom.gpg`. Using the public key certificates in the `cert` directory and the `gpg` tool, say which of these files are:

   - signed,
   - clear signed,
   - signed and encrypted,
   - incorrectly signed (i.e. forged),

   and who signed the file.

   [4 marks]

2. The directory `detached` contains 3 pdfs and a detached signature. Use gpg to find out which file the detached signature is for and who signed it.

   [1 marks]

3. Set up gpg to work with your e-mail. I recommend using Thunderbird (`http://www.mozilla.org/en-US/thunderbird/`) and the Enigmail plug in (`https://www.enigmail.net/`) on your own laptop or desktop (you can find full installation instructions here: `https://www.enigmail.net/documentation/quickstart-ch1.php`). You will be able to find the public keys of all members of the module staff in public keyservers such as `https://sks-keyservers.net/i/`.

   - Get some of the other students on the module to sign your public key.
   - Upload your key to a public keyserver (e.g. `https://sks-keyservers.net/i/`).
   - Get the key for `ics-ex1@hackinggroup.net` from a keyserver.
   - Send a signed, encrypted e-mail to `ics-ex1@hackinggroup.net` with your student ID number as the e-mail title.

   [3 marks]

You do not have to submit anything on Canvas or the token website for this Question 3, only sending the e-mail is necessary. You will receive an automated confirmation. Please wait at least 2 hours between uploading your key to the keyserver and sending the e-mail. We will only count the last e-mail you sent.

2.3 Encryption in Java

2.3.1 Introduction

This part of the exercise is about encryption and decryption in Java. Log into the `bob` account; in Bob's home directory will find the files `Ex1AESencrypted` and `Ex1RSAencrypted`, these are files that have been encrypted with AES in CTR mode, AES in CCM mode and RSA. You will also find the Java file `EncTool.java` which is a command line tool that can be used to encrypt files.

This tool uses the "Bouncy Castle" crypto libraries. These can be found in the bcprov-jdk15on-151 jar file in Bob's home directory, and Java needs to be pointed to this. So you can compile the code using:

```
javac -cp bcprov-jdk15on-151.jar EncTool.java
```

and to run it using:

```
java -cp bcprov-jdk15on-151.jar:. EncTool <mode>
          <key:128 bits in as hex>  <inputFile> <outputFile>
```

E.g. the `ex1CTR.enc` file was created using the command:

```
java EncTool -encAESCTR 3eafda76cd8b015641cb946708675423
                              plainText.txt ex1CTR.enc
```

and the `ex1CCM.enc` file was created using the command:

```
java EncTool -encAESCCM 3eafda76cd8b015641cb946708675423
                              plainText.txt ex1CTR.enc
```

RSA encryption must also includes the name of the keyStore:

```
java EncTool -encRSA <keyStore> <keyName> <inputFile> <outputFile>
```

where `keyStore` is the file name of a key store and `keyName` is the name of the RSA key within the store. The tool will ask you for the password for the keystore (Bob is using "password").

The file `myKeyStore` contains an RSA key pair called `mykey` and is protected with the password: `password`. The `Ex1RSAencrypted` file was created with the command:

```
java EncTool -encRSA myKeyStore mykey plainText.txt ex1RSA.enc
```

2.3.2 Exercise

4. Complete the method `decryptAESCTR()`, so that the program can decrypt messages encrypted with AES in CTR mode. Decrypt the `ex1CTR.enc` file (with the key given above) and submit the token it contains to the website:

   `https://www.cs.bham.ac.uk/internal/courses/comp-sec/tokens`

   [1 marks]

5. As discussed in lectures, CTR mode encryption does not authenticate the encrypted data, therefore it can be altered by the attacker. The plaintext of the message encrypted in the file `ctr.enc` is `Pay Tom 1000 pounds`, you do not have access to the encryption key. Change the ciphertext so that the decrypted message would read `Pay Bob 9999 pounds`. I strongly suggest that you make your own cipher texts with EncTool and experiment with them, and using the hex editor ghex2, which is installed on the VM. Submit your edited `ctr.enc` file to Canvas.

   [3 marks]

6. Complete the method `decryptAESCCM()`, so that the program can decrypt message encrypted with AES in CCM mode. Decrypt the `ex1CCM.enc` file (with the key given above) and submit the token it contains to the website. (You might also like to try editing some cipher texts and see that CCM mode detects the changes).

   [1 marks]

7. Complete the method `decryptRSA()`, so that the can decrypt message encrypted with RSA mode. Decrypt the `ex1RSA.enc` file (with the private key in the keystore) and submit the token it contains to the website. N.B. the RSA key used (and therefore the size of the encrypted AES key) is 1024 bits long.

   [2 marks]

Submit your finished version of the `EncTool.java` file via Canvas. *Make sure you use sensible variable names, and your code is comment and correctly indented, and you* (N.B. I really, really hate incorrectly indented code).

2.4 Getting help:

There is documentation for all of the tools on their respective websites and many "how to" guides can be found via Google. You can come to my office hour 11:00 until 13:00 Tuesday, with questions and to get extra help with the exercise. You can also post questions on the Facebook group, or send me questions in PGP encrypted e-mails.