**CS255: Introduction to Cryptography** **Winter 2013**

# Project 2: SSL MiTM

Botao Hu (botaohu@stanford.edu), Borui Wang (borui@stanford.edu) *March 16, 2013*

## 1 Design

This project implements a man-in-the-middle attack on SSL by implementing a SSL proxy server.

Using keytool, we first created a keystore.

## 2 Steps to run

1. Create a private key in a keystore by running

```
keytool -genkeypair -dname "CN=Very Secure Software, OU=Very Secure Software Group,
    O=Very Secure Inc, L=Cupertino, S=California, C=US" -alias mykey -keypass foobar
    -keystore ./keystore -storepass foobar -validity 365 -keyalg RSA
```

   The password for keystore is `foobar`, which is specified in `password.txt` as requested by the project assignment.

2. Run the SSL proxy server with the command

```
java -classpath ${CLASSPATH}:.:iaik_jce.jar mitm.MITMProxyServer
    -keyStore keystore -keyStorePassword foobar -pwdFile pwdFile -output
```

3. Set the SSL proxy of your browser to localhost:8001.

4. Visit a https site with the browser.

## 3 Short Answers

(1) Question: Suppose an attacker controls the network hardware and can intercept or redirect messages. Show how such an attacker can control the admin server just as well as a legitimate admin client elsewhere on the network. Give a complete and specific description of the changes you would make to fix this vulnerability.

Answer:

An attacker can set a MiTM server between MiTMAdminServer and MiTMAdminCilent. If MiTMAdminCilent sends a request to MiTMAdminServer, the attacker will intercept this packet and obtain the user password, and then authenticate as admin client with his credentials, and then play as a MiTMAdminCilent to send to MiTMAdminServer the forged request with the correct password.

To fix it, we can adopt sig-based challenge response verfication, MiTMAdminServer will issue a challenge to the client, which they must then answer with a response that contained the signature of the message and the command signed by the cilent's secret signing key and then the server verifies their identity. The MitM can not modify request packets as the client because he does not know the cilent's secret signing key.

(2) Question: Suppose an attacker is trying to gain unauthorized access to your MITM server by making its own queries to the admin interface. Consider the security of your implementation against an attacker who (a) can read the admin server's password file, but cannot write to it; (b) can read and/or write to the password file between invocations of the admin server. For each threat model, either show that your implementation is secure, or give an attack. (N.B.: For full credit, your implementation should at least be secure under (a). What, if anything, would you need to change in order to make it secure under (b)? If your answer requires any additional cryptographic tools, you should fully specify them (including the names of any algorithms, cryptosystems, and/or modes of operation that you would use.)

Answer:

If an attacker can read the password hash file of the admin server, he know salt and hashed password, and so he can use dictionary attacks or bruteforce to get the password hash. But as long as we use a secure and long password, the attack has to take a lot of time (say years) to recover the password.

If an attacker can write the password hash file of the admin server, he could simply replace the salt and hashed password with his own password hash and salt. Our implementation will be easily broken by this attack. To prevent this attack, we could sign the password file with a private key from the server that the attacker does not know. We could generate a new signing certificate by keytool.

(3) Question: How would you change a web browser to make it less likely that an end user would be fooled by a MITM attack like the one you have implemented? (This is an important question to ask because when dealing with security, we never just build attacks: we also need to think of ways to prevent them.)

Answer:

A simple solution is to never accept certificate with an invalid certificate chains.

For high security websites, we could adopt public key pinning technology in the browser to prevent MitM. This means that certificate chains for, say, `https://www.google.com`, must include a whitelisted public key. It's a fatal error otherwise.