Questions

1. Currently, commands sent to the adminServer are sent in the clear. Somebody monitoring the traffic can see the password being sent. To fix this, the client can encrypt the request using the server's public key before sending it over to the server, so that only the server with the private key can read the password contained inside. The server's public key can possibly be pinned on the client side.

2. a) If the attacker can read the admin server's password file, they still should not be able to gain access to the MITM server. Seeing the salted hash, they cannot deduce the password used to generate that salted hash, because the hash is a one-way function and collision resistant.

b) If the attacker can write to the file, however, then he can choose an arbitrary password, generate a salt and corresponding salted hash, and replace the existing salted hash in the password file with his. To prevent this, we can generate and save a MAC signature for the password file using a key derived from the admin's keystore password. When we load the password file and signature from disk, we recalculate the signature and verify that it matches the saved signature. If it doesn't, then we've detected a write to the password file. An attacker cannot generate the correct signature if he makes changes to the password file, because he does not know the correct key.

3. The warning dialog could be more informative, notifying the user that a potential MITM attack could be taking place. It could specifically mention that a connection through a proxy could be a red flag. For well-known sites such as banking sites or gmail, the browser can prevent the user from continuing, since those sites should be certified by trusted CAs.

4. Not sure. Why?

Design choices

1. Admin password file
We wrote a simple python script to generate 'pwdFile' that contains only single salted hash string. To simplify, we put 'pwdFile' file in the source folder 'src' so that we can load the file without specifying path.
2. Admin Authentication
If the admin password specified is wrong, the admin server won't send any warning message back since it's unnecessary to notify the potential attacker. From our test, if the admin client specifies wrong password, it will keep waiting for the return message.
3. We use RSA to sign the certificate. If we use a non-deterministic algorithm, then we will run into problems where multiple certificates have the same serial number but different signature.

Instructions (specifically on the Myth machine)

0. Go to src/ directory. Run make.
1. Start Proxy Server using command:

```
java -cp iaik_jce.jar:. mitm.MITMProxyServer -keyStore keystore -keyStorePassword
```

```
keystorepwd -outputFile log.txt -localPort <localPort> -adminPort <adminPort>
```

Note that it’s unlikely that the default port 8001 and 8002 will be available so we have to specify other ports for <localPort> and <adminPort>.

2. Change browser settings as suggested in the handout. To connect to the proxy server, we have to log in the same myth machine. In our experiment, we log in using ‘ssh -X’ for graphics mode and invoke ‘firefox’. Try https://www.google.com.

3. To test the admin client, log in the same Myth machine. Go to src/ directory and type the following command:

```
java mitm.MITMAdminClient -password adminPassword -cmd stats -remotePort <adminPort>
```

```
[sample output:
        Receiving input from MITM proxy:

        stats: total requests = 13
        Admin Client exited
]
```

or

```
java mitm.MITMAdminClient -password adminPassword -cmd shutdown -remotePort
<adminPort>
```

- Use the same <adminPort> as specified for the proxy server.
- We intentionally selected ‘adminPassword’ as the admin password for demonstration only.