ETH zürich

Applied Security Laboratory
Prof. David Basin AS 2023

Certificate Authority Project

1 Background

The (fictional) company iMovies produces independent movies of various kind but with a focus on investigative reporting. Therefore, information exchanged within the company and with informants must be handled confidentially.

To do so, iMovies wants to take its first steps towards PKI-based services. For this reason, a simple certificate authority (CA) should be implemented, with which employees can be provided with digital certificates. These certificates will be used for secure e-mail communication.

2 Assignment

In groups of four students, you are expected to design and implement a CA according to the requirements given below. In a second step, the implementations will be exchanged among the groups and each group should then review another group's CA.

2.1 Functional Requirements

We describe the functional requirement of the CA. Note that the entire functionality must be accessible to a remote client (outside the company's network).

2.1.1 Certificate Issuing Process

The company already maintains a MySQL database in which all employees are listed, along with their personal data as well as a user ID and a password. This database is a legacy system, which cannot be migrated. The CA should verify authorized certificate requests on the basis of this database.

The process of granting certificates should be as follows:

1. The user logs in via a web form by entering his user ID and his password. The user ID and password are verified by consulting the information stored in the database. Alternatively, certificate-based login using a valid certificate is also possible;

2. The user is shown the user information stored in the database. If required, the user may correct this information and any changes will be applied to the database;

3. A certificate is issued based on the (possibly corrected) user information from Step 2;

4. The user is offered the possibility to download the new certificate, including the corresponding private key, in PKCS#12 format.

2.1.2 Certificate Revocation Process

Employees need the possibility to revoke certificates, for example, when their private key is compromised or lost.

The process of revoking a certificate should be as follows:

1. The affected user authenticates himself to a web application. Authentication can either be certificate-based client authentication over SSL/TLS (if the user still holds the certificate and the corresponding private key) or the user uses his user name and password stored in the database (if the user has lost the certificate or the corresponding private key);

2. After successful authentication, the certificate in question (or all affected certificates of the affected user) will be revoked. Additionally, a new certificate revocation list will be generated and published on the web server. Make sure that login with revoked certificates is not possible.

2.1.3 CA Administrator Interface

Using a dedicated web interface, CA administrators (not necessarily system administrators!) can consult the CA's current state. The interface provides at least the following information:

1. Number of issued certificates;

2. Number of revoked certificates;

3. Current serial number.

CA administrators use certificate-based authentication.

2.1.4 Key Backup

A copy of all keys and certificates issued must be stored in an archive. The archive is intended to ensure that encrypted data is still accessible even in the case of loss of an employee's certificate or private key, or even the employee himself.

2.1.5 System Administration and Maintenance

The system should provide appropriate and secure interfaces for remote administration from the internet. In addition, an automated back-up solution must be implemented, which includes configuration and logging information.

Note that these interfaces do not need to be especially comfortable or user friendly. It is sufficient to provide suitable and simple interfaces with the help of standard protocols such as SSH and SFTP.

2.1.6 Components to Be Provided

Web Server: User interfaces, certificate requests, certificate delivery, revocation requests, etc.

Core CA: Management of user certificates, CA configuration, CA certificates and keys, functionality to issue new certificates, etc.

MySQL Database: Legacy database with user data. The database specification can be found in Section 2.4;

Backup: Backup of keys and certificates from the Core CA, databases, and configuration and logging information.

Client: Sample client system that allows one to test the CA's functionality from outside the company's network. The client system should be configured such that all functions can be tested. This includes the configuration of a special certificate to test the administrator interfaces.

Describe exactly how these components are distributed on various computers and exactly what additional components are required to enforce the security measures. The implementation is left up to the project team.

All systems must be built using VirtualBox, which defines the "hardware". The software is freely choosable. However, the operating systems must be Linux variants, and the legacy database requires MySQL.

2.2 Security Requirements

The most important security requirements are:

- Access control with regard to the CA functionality and data, in particular configuration and keys;
- Secrecy and integrity with respect to the private keys in the key backup. Note that the protection of the private keys on users' computers is the responsibility of the individual users;
- Secrecy and integrity with respect to user data;
- Access control on all components.

Derive the necessary security measures from a risk analysis. Use the methodology provided in the book, and *justify your design choices using the security principles*. Hint: Avoid monolithic designs where many functionalities are clumped together on a single machine (why?).

2.3 Backdoors

You must build two backdoors into your system. Both backdoors should allow remote access to the system(s) and compromise its purpose. The reviewers of your system will later have to search for these backdoors.

Design and implement a first backdoor so that it will be nontrivial but likely for the reviewers to find it. Give your best effort when it comes to the second backdoor! Try to hide it so well that the reviewers will not find it. Do not forget to hide your traces in the end (e.g., shell history).

2.4 Database Specification

The database contains the following information:

```
mysql> select * from users;
+-----+-------------+-------------+----------------+----[...]---+
| uid | lastname    | firstname   | email          | pwd        |
+-----+-------------+-------------+----------------+----[...]---+
| lb  | Bruegger    | Lukas       | lb@imovies.ch  | 19c[...]31 |
| ps  | Schaller    | Patrick     | ps@imovies.ch  | 836[...]e1 |
| ms  | Schlaepfer  | Michael     | ms@imovies.ch  | 99b[...]80 |
| a3  | Anderson    | Andres Alan | and@imovies.ch | 86c[...]d7 |
+-----+-------------+-------------+----------------+----[...]---+
4 rows in set (0.03 sec)
```

The password entries in column "pwd" correspond to the SHA256 hashes of the original passwords. The plaintext passwords are:

UserID	Password
lb	`D15Licz6`
ps	`KramBamBuli`
ms	`MidbSvlJ`
a3	`Astrid`

The checksums are generated, for example, as follows:

```
bob@bob:~$ echo -n KramBamBuli | openssl sha256
```

Your system's database must contain the information included in this initial database, except for the password hashes, which may be changed. The database scheme may be extended with additional fields and tables. Additional accounts

might be created, but the accounts in the initial table must still be functional in your system (using the given user names and passwords).

A dump of the initial database above can be found on the Moodle (imovies_users.dump). The database is then created as follows:

```
echo 'create database imovies' | mysql -uroot -p<password>
```

and the table `users` is imported using the command

```
mysql -uroot -p<password> imovies < imovies_users.dump
```

2.5 Recommendations for Building your System

Here are some recommendations for building your system:

- Do not reinvent the wheel: build your system from existing components as much as possible.
- Keep the the overall complexity of your system within reasonable limits by avoiding the use of overly complex software frameworks such as client-side web frameworks. The system we build is relatively simple and does not need a super-fancy web interface. Taming its complexity will help you all with the subsequent reviewing task.
- Make sure that the size and number of your VMs remains reasonable. All you appliances should not occupy more than 12 GB of disk space, so that your colleagues do not need a super-sized laptop to review your system. We strongly suggest that you *avoid using disk encryption at the level of VirtualBox*, as this substantially increases the VM size.

2.6 Note about Virtualization

Working with virtual machines in this project just enables a simpler simulation of the real world situation described in this project. However, for the project work, *do consider the involved machines as real.* This is particularly relevant for the risk analysis where you should avoid reasoning like "We do not need to physically secure the backup, because it is all virtual anyway."

3 Review

After the implementation of the infrastructure described above, access to it will be given to another team that will review it. Simultaneously, your team will review another team's project.

3.1 Preparing the Review

In order to enable a smooth reviewing process, make sure your submission includes the following information besides the VMs themselves:

- the system description and risk analysis report *without description of backdoors*,
- a `README.txt` file including sections with
  1. concise instructions how to get your system up and running (please avoid lengthy procedures with a dozen of steps),
  2. all passwords used in the system (users, CA admin, system admin).
- the source code of the different system components (except backdoors): either include the code separately (e.g., as a zip archive) or include a section in the `README.txt` with the precise locations where it can be found.

3.2 Reviewing Procedure

The review should be conducted first as a "black-box" and then as a "white-box" review.

Black-box review For this review, you should examine the system from outside through appropriate scans and tests. You should decide on an appropriate "review computer" to conduct this analysis from. The choice is yours. You could, for example, use the virtual machine mallet, develop tools yourself, or use freely available collections such as Kali Linux (`www.kali.org`).

Whitebox review For this review, the other team will provide you with all necessary information for access to all components that you must review, so that you can analyze the entire system from the inside (see Section 3.1).

Clearly document the approach that you have used to find each backdoor or other security flaw (e.g., blackbox or whitebox, steps and tools used).

4 Written Reports and Final Presentation

4.1 Structure

You will write reports both on your own system and on the system that you review. Templates specifying the structure of these reports are provided on Moodle. Follow the instructions in the templates.

4.2 Writing Guidelines

Please observe the following points when writing your reports:

- Adhere to the given templates.
- Refer to the security principles in the book for justification.
- Use clear terminology:
  - secure = confidential + authentic. Be clear about which properties you are writing.
  - Are pairwise distinct: certificate, private key, public key, archive to of certificate with private key. Please avoid mixing these up.
- Refer to the source document of your risk definitions if appropriate.
- For the risk evaluation, number the threats consecutively and formulate them in active, not passive, voice: who (threat source) does what (bad things)?
- Use a spell checker before hand-in!

4.3 Evaluation Criteria

Both the form and the content of your two reports will be evaluated. The content will be more strongly weighted than the form. We attach special importance to the following points:

- The reports fulfill the instructions given in the templates, are structured according to the templates and are complete.
- The reports are written in a precise, correct, and clear way. *Brevity is preferable to verbosity.* Extracts from configuration files and lists of executed commands, for example, are not desired.
- The risk analysis corresponds to the project description and provides an adequate catalog of countermeasures. All decisions are justified.
- All discussions are objective.
- The system built reflects what is described in the report, and the functionality described in the requirements must be entirely present. However interfaces must only be functional, and their graphical attractiveness is irrelevant.
- The review of the other team's system is careful and systematic.

We provide detailed evaluation criteria on the course's webpage.

4.4 Final Presentation

Keep the following in mind when preparing your final presentation:

- Time is short: 10 min for your system and 10 min for the review.
- Focus on the essentials. Omit software version numbers and long discussions of the frameworks you have used.
- Make sure each group member contributes about equally much.

5 Important Dates

Note that we will use Moodle for group registration as well as all report hand-ins and feedback. For the VM hand-ins and exchanges, we will use (provided) USB sticks and/or Polybox.

Sep 28, 2023	Register teams, project start.
Oct 6, 2023	Last date to deregister from the course. Dropping the course after this date will count as "fail".
Oct 23, 2023	Submit the first rough system description and risk analysis. This should include a network diagram with all machines and services, a rough overview of the system with the most important components and information flows, the most important assets and threat source, and the worst threats with proposed countermeasures. This initial draft is not expected to be complete. Also, try to be concise.
Oct 27, 2023	You get feedback to your rough system description and risk analysis .
Nov 30, 2023	Hand-in of complete system description and risk analysis report (max. 25 pages). Hand-in of the VMs composing your system (max. 12 GB). Hand-over of your system to reviewing group. Note that later VM updates between groups are disallowed.
Dec 21, 2023	Submit the system review report (max. 15 pages). Every group is expected to give a short (20 minutes) presentation about their system and their review highlights today.