ASL - System Description and Risk Analysis

Andrei Herasimau Bradley Mathez Mihhail Sokolov
Jean-Luc Stoupy

December 1, 2021

Page limit: 25 pages.

Contents

1 System Characterization 3
1.1 System Overview . . . 3
1.2 System Functionality . . . 4
1.2.1 User Authentication . . . 4
1.2.2 Certificate Issuing . . . 5
1.2.3 Certificate Revocation . . . 6
1.2.4 CA Administrator's Dashboard . . . 7
1.2.5 Backup . . . 7
1.2.6 System Administration . . . 9
1.3 Security Design . . . 10
1.3.1 Access Control . . . 10
1.3.2 Key Management . . . 10
1.3.3 Session Management . . . 10
1.3.4 Security of Data at Rest . . . 11
1.3.5 Security of Data in Transit . . . 11
1.4 Components . . . 12
1.4.1 Platforms . . . 12
1.4.2 Applications . . . 12
1.4.3 Data Records . . . 13
1.5 Backdoors . . . 14
1.5.1 Trivial Backdoor . . . 14
1.5.2 Non-Trivial Backdoor . . . 14

2 Risk Analysis and Security Measures 15
2.1 Assets . . . 15
2.1.1 Physical Assets . . . 15
2.1.2 Logical Assets . . . 16

2.1.3 Persons 18
2.1.4 Intangible Goods 18
2.2 Threat Sources 18
2.3 Risks Definitions 19
2.4 Risk Evaluation 21
2.4.1 *Evaluation of Physical Assets* 21
2.4.2 *Evaluation of Logical Assets* 21
2.4.3 *Evaluation of Person Assets* 24
2.4.4 *Evaluation of Intangible Goods Assets* 24
2.4.5 Risk Acceptance 25

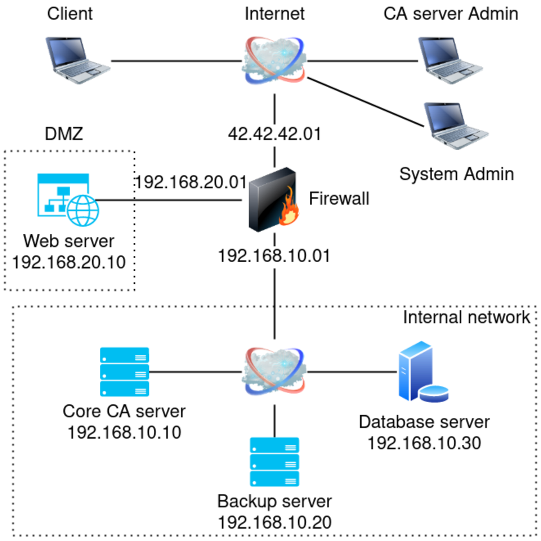


Figure 1: Architecture Overview

1 System Characterization

1.1 System Overview

iMovies is a company which produces independent movies with a focus on investigate reporting. During the production of their movies, the company exchanges information with informants. This is sensitive data and it must be handled confidentially. To do that, the company has chosen to use a CA to generate certificates to secure its employees' e-mail communications.

The CA must be able to issue digital certificates requested by iMovies' employees, as well as revoke said certificates. Another core idea of this project is to create a system with high usability, since the users of the system are not IT specialists.

The relevant machines and services are depicted in Figure 1. The system's network includes four different parts:

1. **Clients** are employees of iMovies (inside or outside the company's network), which use the web server to authenticate themselves and exchange e-mails among each other. They can also be CA administrators or systems administrators.

2. A **web server** acts as an intermediary between the clients and the internal network, which allows a user to access the database, to modify his information and to perform operations on the certificates (request a new one, revoke one or consult the statistics). CA administrators are also able to consult the CA's statistics.

3. An **internal private network** containing the *Core CA server*, the *database server* and the *backup server*. Their roles are explain in detail later in this document.

4. A **CA server** which consists of a **root CA** and **intermediate CAs** that are used as a certificate hierarchy to create and revoke certificates.

5. A **database server**, which will store user data.

6. A **firewall** will filter connections between the different components listed above. A demilitarized zone (DMZ) encapsulates the web server to protect the rest of the company if the security of the web server is compromised.

All communications within this system are secure (confidentiality, integrity and authenticity). Moreover, to protect highly sensitive data, the company holds a secure vault in their basement which is under video-surveillance and requires multiple keys and code to access it. Every attempted break-in is immediately reported to the security team that will neutralize the threat.

1.2 System Functionality

The system will have several functions, which are listed in the subsections below.

1.2.1 User Authentication

In order to access the CA services, users must first be authenticated. The authentication system is accessible through a web page hosted on the web server and the users can authenticate themselves in two ways, namely by using a password or a certificate.

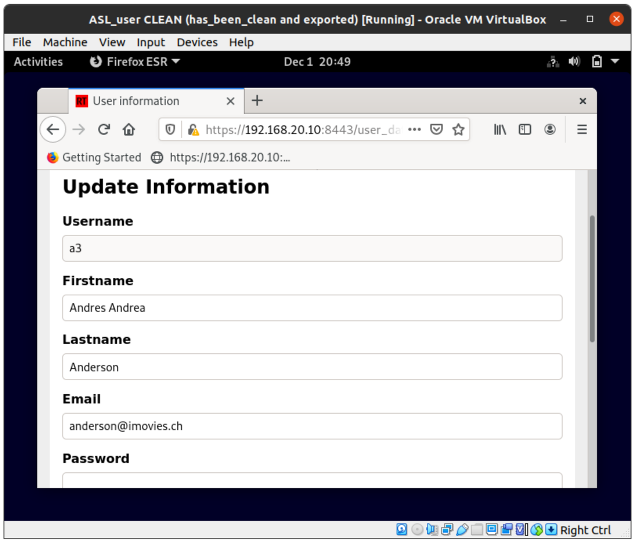
- The username and a SHA-1 hash of the password will be sent to the database (the hash will be computed on the client side) to compare them with the database entries.

- The certificate will be selected by a user in the PKCS#12 format (certificate and private key). In order for the client to not have to send his private key, the server sends a challenge nonce to the client. The client will then sign the nonce using his private key and send the signed challenge, along with the client certificate, to the web server. Thus, the client only sends his certificate and a signed nonce, not his private key.

  After that, the server will verify the clients signature and the certificate's time validity. The server then checks that the certificate is not in the revoked certificates list.

The web server provides a one-way authentication towards the client with a self-signed certificate architecture. The certificate hierarchy has been installed on the client. Another solution may be to sign the web server's TLS certificate by a trusted CA already present in the user's browser. An authenticated user can:

ht[]


ASL_user CLEAN (has_been_clean and exported) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Firefox ESR
Dec 1 20:49
User information
https://192.168.20.10:8443/user_da
Getting Started
https://192.168.20.10:...
Right Ctrl

Update Information

Username

a3

Firstname

Andres Andrea

Lastname

Anderson

Email

anderson@imovies.ch

Password

Figure 2: User data displayed on the web server

- Access their personal information currently stored in the database, except the password to avoid privacy leakages (see Figure 2).
- Modify their personal information currently stored in the database, except its user ID (since it is used as the primary key in the database), on the condition that they provide their password.
- Request a new certificate
- Revoke their current certificate

We will use a database to allow users to login using username and password (see Figure 3). The database will contain a single table with user data.

1.2.2 Certificate Issuing

At the moment a user is logged in, they can request a new certificate. When a user utilizes the web interface to request a new certificate, the request first arrives at the web server. After that, the web server sends its own request to the CA server along with the user information. The CA server will only accept requests from the web server and cannot be accessed outside the internal network. Certificate issuing is done by the Intermediate CA.

The Core CA generates a new RSA private key and a new certificate in the X.509 format, which contains user information (user id, firstname, lastname and email) and the public key derived from the private key. The new user certificate is signed by the intermediate CA. We will also generate a random serial number for each certificate issued by the Core CA.

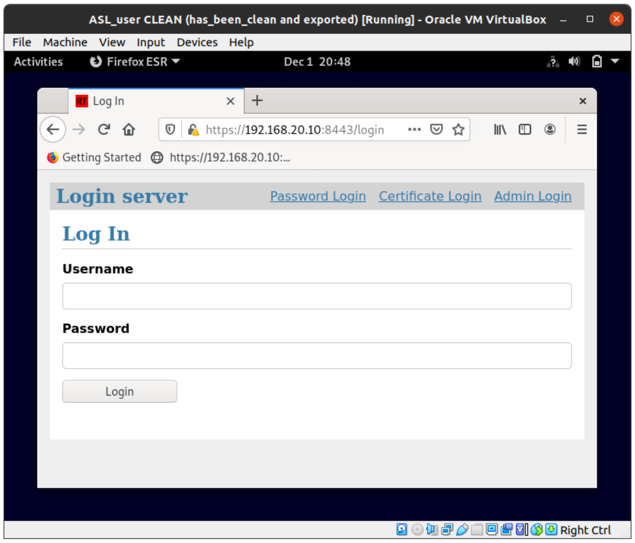

ASL_user CLEAN (has_been_clean and exported) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Firefox ESR
Dec 1 20:48
Log In
https://192.168.20.10:8443/login
Getting Started
https://192.168.20.10:...
Login server
Password Login
Certificate Login
Admin Login
Log In
Username
Password
Login
Right Ctrl

Figure 3: The login with password interface on the webserver

After its generation, the certificate and its private key (a PKCS#12 file) will be forwarded to the web server to allow the user to download it once. After the user has downloaded their certificate, the web server deletes its copy. Thus, the user will never be able to download this certificate ever again. The only possible to obtain a certificate will be to generate a new one. This process reduces the attack surface. Each user's certificate will have a validity period of 365 days. Even though a shorter time period would result in higher overall security, we opt for a longer one to increase usability.

The new certificates are also sent from the CA server to the backup server after being encrypted using AES256-CTR with the CA server's private key. Finally, the certificate is stored in the core CA without the private key, since we will not use it anymore in the core CA (we will access the backup system in case of a key loss). This design minimizes the exposure by reducing the attack surface presents to adversary.

1.2.3 Certificate Revocation

There are two ways to revoke a certificate. Either the user revokes his own certificate of his own volition, or the CA server revokes a certificate and adds it to the CRL.

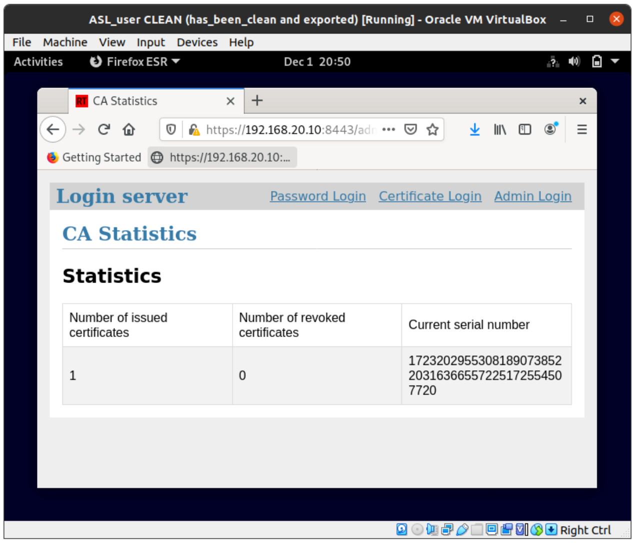
Any user can manually revoke their certificate from the web interface on the web server. Similarly to the certificate issuing process, the user request is forwarded from the web server to the CA server, where the intermediate CA can revoke the certificate by adding it to its certificate revocation list (CRL) in the X.509 format. After that, the Core CA will confirm the revocation to the web server. Future connections will be checked against the CRL first before checking their validity. Thus, connections attempts using invalid certificates must not be able to connect the web server anymore and these attempts will be logged into the CA server logs for further investigations. This system may be coupled with an intrusion detection system (IDS) reading logs to discover possible threat.

1.2.4 CA Administrator's Dashboard

The CA administrators will be able to see certain statistics about the CA on the web interface such that:

- number of issued certificates
- number of revoked certificates
- current serial number (last certificate's serial number that was issued)

CA administrators authenticate themselves with their certificate as explained before, but on the *administrators login page* (see Figure 4).


ASL_user CLEAN (has_been_clean and exported) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Firefox ESR
Dec 1 20:50
CA Statistics
https://192.168.20.10:8443/adı
Getting Started
https://192.168.20.10:...
Password Login
Certificate Login
Admin Login
Right Ctrl

Login server

CA Statistics

Statistics

Number of issued certificates	Number of revoked certificates	Current serial number
1	0	172320295530818907385220316366557225172554507720

Figure 4: Administration page for CA administrators

1.2.5 Backup

In order for the system to be resilient against system crashes, and for system administrators to be able to rapidly reconstruct a given component (or even the entire system), if needed, a backup system of the crucial systems components and logs of the different system parts is implemented.

The backup server's functionality is implemented in Python scripts. These scripts notice every modifications of the various system components for crucial data (i.e. database queries, logs, issued certificates and associated privates keys, etc.) in real time. If this data changes, it will be sent to the backup server with two different modes:

- Backing up the whole file at each modifications. Used for "non-log" elements.
- Append the modifications to the end of the previously backed up file's version (it creates the file if it does not exists). Principally setup for logs.

Furthermore, the different system components can encrypt the sensitive data with AES256-CTR using a key generated on their own. These encryption keys are only known by the system that issued them (e.g. the web server knows only its encryption key and does not know the database's one). To avoid data loss in the case where a server loses its encryption key, a printed copy is kept in the secure vault.

Developing our own backup system allows the company to be flexible in case of new requirements and to encrypt the data before sending them to the backup server if it is needed. In addition, we observed that `rsyslog` has some vulnerabilities, then we chose to implement our own backup system.

We want to make sure that the content of the communications between the system components and the backup server is confidential. Thus we use a TLS connection with mutual authentication (i.e. each side possesses a certificates and a private keys to established the secure TLS connection).

- **Keys and Certificates:** A copy of all issued certificates and private keys is stored in the backup server to ensure that an employee can always have access to their data, even if they lost their private keys. The keys are encrypted using AES256-CTR with a key generated by the CA server and known only by itself (as explained before). An encrypted copy of the root CA and its private key as well as an encrypted copy of every intermediate CAs and their respective private key are also stored on the backup server. The backup is updated each time a certificate is issued or revoked.

  This design allows any employees to ask system administrators to recover their key and certificates from the backup in case of a loss. During this process, the user's identity must be verified using an official ID document or his employee card). Thus, the employee must be present in the company's offices to recover it. Through this design, the employees are less willing to keep insecure copies of their private key due to the fear of losing it, which reduces the attack surface.

- **Logs:** All relevant log files, such as system logs, database queries logs, web server connection logs, etc. are stored on the backup server. This serves two purposes. Firstly, log files can be a great aid to system administrators in fixing technical problems on the various components. Thus, having backups for these files ensures that system administrators have access to them, even if all data on a given component is lost, the system administrator will still have access to log files to at least figure out what happened and/or retrieve crucial information for the restoration of a given component. Secondly, if a security breach is detected, log files can be used as an aid to pinpoint the location of a vulnerability that was exploited to create this breach.

In case of a threat detection, the further investigation must be performed with the logs stored on the backup system since the ones stored on the machine may have been tempered with.

The exact backup logs are:

  - Information related to SQL queries received by the database, such as the date, the query's type, the user that issued the query, etc.
  - Any interaction with the web server (e.g. connection, file execution, etc.) along with its timestamp.
  - Any interaction with the CA server (e.g. connection, file execution, etc.) along with its timestamp.
  - Any changes in the configuration or code files of every machine will be backed up. We store the date, and the modified files.
  - The `/var/log/wtmp`, `/var/log/lastlog` and `/var/log/auth.log` files are tracked by the backup system to trace the active users in the systems. This can be coupled with an IDS to alert system administrators in case of an unexpected intrusion.

- **Machine backup:** Each scripts and configuration files stored on the machines are also uploaded (once they have been modified) to the backup server to enable fast recovery in case of system failure.

- **Database:** The content of the database is sent encrypted to the backup server every time that it is modified.

- **Backup failure:** To keep the backup system resilient against a possible failure, we apply the 3-2-1 backup rule[1]. We thereby avoid having a single point of failure on the backup system.

1.2.6 System Administration

No system can run smoothly without system administration. Therefore, we offer our system administrators the possibility to perform remote administration via the `ssh` protocol, which provides a secure connection.

[1] https://www.acronis.com/en-eu/articles/backup-rule/

1.3 Security Design

1.3.1 Access Control

Each server in our system will have a root user and a normal user. The relevant services will be run by the normal user. However, to follow the Least Privilege principle, the normal user will not able to modify code, which is running on the server, as well as the keys and certificates stored on the server.

An exception to the above principle is the database server, where the mysql service is run by the mysql user. However, the mysql user only has access to files that he needs to run the mysql service. Thus, we preserve the Least Privilege principle established above.

The system administrators will be able to access the various machines via ssh. The password authentication access is disabled on every machine to reduce the attack surface (e.g. brute force attacks). Then, the access is only possible by using an ssh key-pair. The public keys are different on every machine to compartmentalize the system access by providing only the needed accesses to the system administrators. This process also respects the least privilege principle.

1.3.2 Key Management

The main function of our system is the issuing, revocation, and management of user certificates. Therefore, key management is a vital security concern.

To preserve the confidentiality of the users' private keys, all newly generated certificate and the corresponding private keys will be deleted from the web server after being sent to the client. This ensures that even if the server is compromised, the private keys and certificates of the clients will remain confidential.

Following the principle of minimum trust and maximum trustworthiness, it will not be possible to create a new account from the web service. To do this, an employee must first verify his identity in person, after which the CA administrator will create an account for this employee and issue a certificate to them.

In order to establish secure channels between the components, each server will use TLS 1.3. Therefore, each server will store its own private key, which also need to be managed properly. We manage these keys with the help of the least privilege principle. Only the root user will be able to modify these private keys and only users, which run services that communicate with other system components, will be able to read these keys.

1.3.3 Session Management

Regular users will establish sessions via TLS 1.3. In order to manage user sessions on the web server's REST API, cookies within the Flask framework will be used. These cookies will be signed to prevent tampering. This will also protect the client against attacks such as CSRF and session hijacking, since only verified cookies (which can only be produced by the web server) will be accepted by the client.

1.3.4 Security of Data at Rest

We will store data on different servers, in particular on the database server, the CA server and backup server.

The database will store sensitive data, i.e. user logins, passwords, and email addresses. The data will be encrypted with InnoDB, which uses AES in CBC mode. Furthermore, the passwords will not be stored in plaintext, but as SHA-1 hashes. To protect the database from SQL injections, all database queries will be formed with the help of prepared statements.

Following the principle of minimum exposure, the user can only possess one valid certificate at a time, the issuing of a new certificate is only possible if the user does not possess any valid certificate on the CA server. If a user already has a valid certificate issued, and try to get a new one, then it will results in an error displayed to the user that asks to revoke his certificate before.

The backup server stores data that must be kept at all cost. Our system also provides the possibility to encrypt the data which will be stored on the backup server. Some examples of encrypted data are the database entries and the certificates and keys that were issued by the CA server.

1.3.5 Security of Data in Transit

Secure communication between the different components of our system is vital to its functionality. Therefore, we want all data to be sent via secure channels. To achieve this, the system will only use TLS 1.3 connections, and refuse all connections that use an earlier version of TLS (or does not use TLS at all). Furthermore, to ensure that an attacker never sees the users' passwords in the clear, they are hashed on the client's machine.

The database server will only accept requests from the web server, the backup server, and system administrators. We will use iptables to ensure that only requests from the two above-mentioned servers will be accepted. We will use the

TLS_CHACHA20_POLY1305_SHA256

ciphersuite for all TLS connections. Since the answers to SQL queries can be quite large, we opt for using a stream cipher to ensure good performance. Furthermore, since the database will only communicate with the web and backup servers, it will offer the above-mentioned ciphersuite as the only valid option during the TLS handshake and will refuse any client that does not provide it as a supported ciphersuite.

We will also send a lot of data to the backup server. All data sent to the backup server will be encrypted with AES256-CTR with a key only known by the machine which is sending the data. Printed copies of these keys are stored in a secure physical vault to avoid data loss.

1.4 Components

1.4.1 Platforms

All machines in our system use Debian 11.1.0 as an operating system (OS) and are the systems administrators must check and install the updates regularly (once a day). We distributed the tasks among different machines to follow the compartmentalization principle. A summary of the open ports is available in Table 1.

1.4.2 Applications

Machine	Port	Protocol	Usage
Web Server	8443	TLS	Access the web site through HTTPS
Web Server	22	SSH	Maintenance access for system administrators
CA Server	8080	TLS	Access to the CA server API
CA Server	22	SSH	Maintenance access for system administrators
Database	3306	TLS	Interact with the database
Database	22	SSH	Maintenance access for system administrators
Backup	8888	TLS	Receive the backup data
Backup	22	SSH	Maintenance access for system administrators
Firewall	22	SSH	Maintenance access for system administrators

Table 1: Open ports

Firewall The firewall handles the network separation through routing and filtering the traffic between the internal network, the demilitarized zone (DMZ) and the external network (i.e. internet). It is maintained through `ssh` by the system administrators on port 22. The firewall uses a white list approach (deny every connection by default and accept only the ones specified) for the connections coming from the external network. This follows the secure, fail-safe defaults principle. In addition, the DMZ protects the internal network against any undesired connections in case the web server has been compromised, since the web server is located in a separate network.

Web server The web server is running a Flask server in Python which is started at the boot of the machine. The web server works on all platforms and browsers and provides both front-end and back-end functionalities. Below is the list of provided functionalities and their URLs:

- User login with username and password: https://192.168.20.10:8443/login
- User login with certificate: https://192.168.20.10:8443/login_certificate
- Admin login with certificate: https://192.168.20.10:8443/login_admin
- Log out and clear the session: https://192.168.20.10:8443/logout

- Get and update user data: https://192.168.20.10:8443/user_data
- Get new certificate for the current user: https://192.168.20.10:8443/issue_certificate
- Revoke currently used certificate: https://192.168.20.10:8443/revoke_certificate
- Get CA statistics for admin: https://192.168.20.10:8443/admin_stats

In order to function properly, the web server communicates with the CA server and the database server that are both located on separate machines, following the separation of concerns principle. CA server's API is used for authorization with a certificate, obtaining new certificate, revoking existing certificate and getting admin statistics about the CA. The database server's API is used for authorization with username and password, as well as retrieving and updating user information. The communication with both servers is secured with TLS 1.3 and the web server accepts only secure requests with TLS 1.3 from the client, thus providing confidentiality and integrity of the data in transit.

Backup Server This machine stores the backups of the other system components, such as their logs and their data. The exact data that is saved is the logs of the systems, the firewall configuration, the certificates and keys issued by the CA server (keys and certificates are encrypted before being sent to the backup server), the database (encrypted before being sent to the backup server) and the web server source code. This follows the traceability principle.

It is also maintained by system administrators via `ssh` through port 22. It only accepts TLS 1.3 connections on port 8888 that are coming from the system components with signed and valid TLS certificates. To perform the backup, a Python script is running on the system components and triggers a connection to the backup server at every tracked file modification.

CA Server This server manages the certificates. It listens to incoming TLS connections over the port 8080 and is maintained by system administrators via `ssh` through port 22. This server uses a certificate hierarchy with three levels (i.e. root, intermediates and end-users). The root level issues, signs and revoke the intermediate certificates, which in turn can issue, sign and revoke user certificates. This hierarchy allows to split different regions of the iMovies company and rapidly revoke an entire group of certificates, if needed (e.g. after the loss of an intermediate certificate). This hierarchy increases the attack surface, but avoids having a single point of failure (i.e. only the root certificate). In addition, the TLS certificates used for the communications between the components in the system are generated by the CA server.

1.4.3 Data Records

We use a MySQL database, run on a Debian machine. It supports basic database functionalities, as well as system administration. The database contains a single table (users), which consists of the columns username, firstname, lastname,

Column Name	Type
username	varchar(64): PK, NOT NULL
firstname	varchar(64): NOT NULL
lastname	varchar(64): NOT NULL
email	varchar(64): NOT NULL
pwd	varchar(64): NOT NULL

Figure 5: Structure of the sole database table.

email, and pwd (refer to Figure 5). The database provides two interfaces for connections, a TLS connection on port 3306 and a ssh connection on port 22.

1.5 Backdoors

1.5.1 Trivial Backdoor

The web server provides a favicon over the address https://192.168.20.10:8443/favicon.ico through a GET request. However, if we perform a POST request at the same address with the password *"abcdefghijklmnopqrstuvwxyz1234567890"* on the first line, followed by any command on the second line, then the web server will check the password against its SHA1 representation. If it matches, the web server will forward the input command to the CA server on port 6666 through a TLS 1.3 connection (where both sides are authenticated). After that, the CA server executes the command with root privileges and returns the result to the web server, which is then forwarded back to the client. This allows the attacker to have complete control over the certificates. The attacker will be able to issue, revoke and sign every certificate(s) she wants to. We considered this backdoor as trivial since the code is clearly visible in the implementation and the password is easily crackable due to its common use.

1.5.2 Non-Trivial Backdoor

For this backdoor, we have created a script that will be run every hour by the root crontab on the web server. However, the cron job is hidden through a new command in the root's `.bashrc` file[2]. This script will create a root access through `ssh` on port 22 if this access is non-existent by generating a new ssh key pair and installing the public key in the `/root/.ssh/authorized_keys` file. The private key will be hidden using steganography in the favicon provided by the web server at the address https://192.168.20.10:8443/favicon.ico through a GET request. The attacker will thus have root access on the web server. Since the web server is the only access point for the company's employees, the attacker will be able to observe every certificates transaction, store it and decrypt the employees' communications afterwards. In addition, every interaction with the system can be spied on. Furthermore, an attacker can steal the web server's

[2] As suggested in https://unix.stackexchange.com/a/564777

TLS certificate and send requests to directly interact with the other components while impersonating the web server (e.g. interact with the database to modify a user's password).

We considered this backdoor as hard since there are absolutely no hints that the key is extracted through steganography and we think that the most common machine where people may insert a backdoor will be on the CA server, not on the web server. In addition, it is not possible to observe the open ports or the cron jobs. The only way to notice it is to observe that the picture hash has been modified, or that there is an unknown public key in the root directory, to observe that the `.bashrc` file has been tampered or to find the script that creates the hidden access through an elaborated find command for example.

Hide this subsection in the version handed over to the reviewing team by setting the flag `showbackdoors` at the top of this document to `false`.

2 Risk Analysis and Security Measures

2.1 Assets

2.1.1 Physical Assets

Our physical assets is the hardware on which all the components of our system run, i.e. the core CA, the firewall, the web server, the database, and the backup server. All these machines should be located in different locked and well-cooled (to prevent malfunction due to overheating) rooms. Furthermore, only authorized personnel (i.e. system administrators and security personnel) should have access to these rooms. A second backup server will be installed in a separate location, which will also only be accessible to system administrators and security personnel, thereby avoiding having a single point of failure.

All the machines will be stored in lockable racks. They will also operate in normal mode, and will have no ports to insert portable media devices (e.g. CD-ROM, USB stick, etc.).

In distributing the different servers of our system among different rooms in the iMovies office building, our goal is to achieve maximum physical isolation between the different components. We therefore propose the following distribution.

- **Core CA:** Located in a heavily guarded room. Only the CA administrator and two designated system administrators have access to this room. They must also be let in by a security guard.

- **Firewall:** Is located in the same room as the Core CA, but in a different locked rack.

- **Web server:** Located in a room on the same floor as the iMovies IT department. It is stored in a locked room, where all employees can see it (to observe if someone tries to enter into the room).

- **Database:** Located in the same room as the web server, but in a different locked rack.
- **Backups:** Located in a different room as the web server and database, to increase physical separation. Another copy of the backup server is also kept at a separate location following the 3-2-1 backup rule.
- **Internet connectivity:** A router that connects to the provider's network is kept in the same room as the firewall. It has a stable connection with 1 Gb/s bandwidth.
- **IMovies internal network:** The internal network is an Ethernet local area network (LAN). The firewall plays a central role in this architecture since it connects the various networks (i.e. Internet, DMZ and internal network).

2.1.2 Logical Assets

Software:

- **Core CA:** Debian 11.1.0, TLS 1.3, Python3, Flask, ssh
- **Firewall:** Debian 11.1.0, TLS 1.3, Python3, iptables, ssh
- **Web server:** Debian 11.1.0, TLS 1.3, Python3, Flask, Javascript, ssh
- **Database:** Debian 11.1.0, TLS 1.3, MySQL Community Edition, with InnoDB for encryption, ssh.
- **Backups:** Debian 11.1.0, TLS 1.3, Python3, ssh

Information:

- **Logs:** Log files of the various components. They can be used to help fix technical difficulties, as well as help pinpoint an entry point used by an attacker, if an attack occurs.
- **Configuration files:** Configuration files are needed to rebuild components after an attack or system failure. A list of relevant configuration files for all components is given below. The network configuration file for every machine is located at */etc/network/interfaces*.
  - **Web server:** A general web server log is stored at */home/webserver/logs/webserver.log*. It logs any requests that the web server receives and any encountered errors.
  - **CA server:** The CA server configuration file is located at */home/ca-server/ca_config.py*. The logs are located at */home/ca-server/Documents/ca-server.log*.
  - **Firewall:** The firewall configuration file is located at */etc/firewall_iptables.sh*.

  - **Database:** The database configuration file is located at */etc/mysql/my.cnf*. The MySQL error log is located at */var/log/mysql/error.log*. The MySQL query log is located at */var/lib/mysql/query.log*.
  - **Backup server:** The backup server configuration files are located at */etc/backup_server.cfg* and */opt/ASL-backup/utils.py*.

- **TLS privates keys:** TLS private keys are used to establish secure connections between two different components, which ensures confidentiality, integrity and authenticity of both participants. Furthermore, the web server's private key is also used to establish a secure connection with the client.

- **TLS certificates:** Used to authenticate the different components to each other.

- **Root CA certificate and private key:** The private key is used to create intermediate certificates. The certificate identifies the root CA.

- **Intermediate CAs' certificates and private keys used to issue user certificates:** The private keys are used to create employee certificates. The certificates identify the intermediate CA that issued a given user's certificate.

- **Intermediate CA certificates and private keys used to issue component TLS certificates:** The private key is used to create TLS certificates for system components. The certificate identifies the intermediate CA which issued a given component's TLS certificate.

- **Certificate Revocation List (CRL):** Lists of all revoked certificates. They are updated and stored at the CA server. There is one for the root server and one per intermediate CA.

- **User private keys:** Used by users to sign a challenge sent by the web server in order to authenticate the user to the web server via certificate.

- **User certificates:** Used by users to authenticate themselves to the web server and potentially to other users.

- **User data:** User data which is stored in the database (refer to Figure 5).

- **Backup keys:** Keys used to encrypt data sent to and stored on the backup server. They are only stored on the corresponding machine (e.g. the web server's encryption key is stored on the web server) and a copy of each encryption key is printed and stored in a physical vault. These keys are stored at */opt/ASL-backup/<server name>_backup_key.key*.

- **ssh keys:** Keys used by system administrators to directly access various system components.

2.1.3 Persons

- **CA administrators:** Only manage the CA server. Have supervised physical access to the CA server. They can view the current CA status via a dedicated interface (i.e. number of certificates issued, list of revoked certificates, last issued certificate, etc.).
- **System administrators:** Have access to every machine via `ssh` using public/private key authentication, as well as supervised physical access to every machine.
- **Employees:** Regular employees of iMovies, mostly journalists, screenplay writers, video editors, etc. They can request certificates, revoke their own certificate, and view their own user data.
- **Security personnel:** iMovies security personnel consists of two divisions. The first are security guards, who ensure that no unauthorized person enters the office building, or a particular room with access restrictions. The second are information security engineers, who supervise physical access to the machines, and perform regular security inspections on the various components.
- **External users:** Mostly informants that give information to iMovies, but could be any third party that temporarily works/cooperates with iMovies. Their access to the web server must be handled with particular care, as they pose the biggest security risk out of all persons.

2.1.4 Intangible Goods

- **Employee confidence:** Employee confidence is required to build a good working environment and it is also necessary to avoid malicious behaviour from an internal employee.
- **Company's reputation:** The company's clients trust the company to not mishandle user data or disclose any confidential information.

2.2 Threat Sources

- **Nature:** Natural disasters, such as fires, floods, earthquakes, volcano eruptions, etc. This obviously depends on the location of the company, and impacts physical assets.
- **Government Agencies, Terrorist Organizations, Criminal Organizations:** iMovies is a company that focuses on investigative journalism. Its films can therefore touch on topics, which may be of interest for these threat agents. Their possible goals are acquiring information about iMovies's informants, modifying data about their inner workings, which were discovered by iMovies.

- **Employees:** Disgruntled employees may cause serious damage, especially if they have a high security clearance. Their main motives are selling information about iMovies. The technical and cleaning staff might even sabotage iMovies's servers.

- **Ex-Employees:** Ex-employees are less dangerous than employees in terms of impact. However, ex-employees might still cause damage by, e.g. disclosing confidential information about informants that they used to work with. This could hurt the company's reputation, as well as violate certain people's privacy. If an employee has parted with the company on bad terms, they might seek revenge against iMovies.

- **Hackers:** Various hackers (from script kiddies to hardened professionals) always pose a potential threat to any system. Their motives vary greatly, ranging from earning money via blackmail and/or extortion to simple boredom.

- **Malware:** Any system can be a target of a malware attack, both directed and undirected. It can be used to extort money, steal information, or generally compromise the system in a certain way.

2.3 Risks Definitions

Likelihood	
Likelihood	Description
High	The threat source is highly motivated and sufficiently capable of exploiting a given vulnerability in order to change the asset's state. The controls to prevent the vulnerability from being exploited are ineffective.
Medium	The threat source is motivated and capable of exploiting a given vulnerability in order to change the asset's state, but controls are in place that may impede a successful exploit of the vulnerability.
Low	The threat source lacks motivation or capabilities to exploit a given vulnerability in order to change the asset's state. Another possibility that results in a low likelihood is the case where controls are in place that prevent (or at least significantly impede) the vulnerability from being exercised.

Impact	
Impact	Description
High	The event (1) may result in a highly costly loss of major tangible assets or resources; (2) may significantly violate, harm, or impede an organization's mission, reputation, or interest; or (3) may result in human death or serious injury.
Medium	The event (1) may result in a costly loss of tangible assets or resources; (2) may violate, harm, or impede an organization's mission, reputation, or interest, or (3) may result in human injury.
Low	The event (1) may result in a loss of some tangible assets or resources or (2) may noticeably affect an organization's mission, reputation, or interest.

Risk Level			
Likelihood	**Impact**		
	Low	Medium	High
High	Low	Medium	High
Medium	Low	Medium	Medium
Low	Low	Low	Low

2.4 Risk Evaluation

2.4.1 *Evaluation of Physical Assets*

No.	Threat	Countermeasure(s)	L	I	Risk
1)	Natural disasters, i.e. floods, volcanoes, earthquakes, hurricanes, tsunamis, meteorites, fires, space radiation from the Sun, etc.	A backup server in a different part of the world that will not be affected by the disaster. The company buildings should be constructed in a location with a low probability of natural disasters.	*Low*	*High*	*Low*
2)	Physical access to the machines (i.e. inside threat or industrial spy operation)	Restrict the physical access to the machines (e.g. using bio-metric and/or RFID/NFC based locks, alarms, or even armed guards). Also, by placing the machines at a different location and requiring a special security clearance to access them.	*Low*	*High*	*Low*
3)	Material getting old: components break and need to be replaced	Build the systems in a way that swapping components is easy and that the machines do not require specific material to be built. Installation can be performed and data can be recovered using the configuration scripts and the backups.	*Medium*	*Low*	*Low*

2.4.2 *Evaluation of Logical Assets*

Software

No.	Threat	Countermeasure(s)	L	I	Risk
4)	Attacker gains access to the database and deletes the existing data	The database is password-protected and is backed-up periodically.	*Medium*	*Medium*	*Medium*
5)	Attacker requests the Core CA for a certificate or revokes existing certificate without being properly authenticated	Core CA accepts only requests that come from the web server and the requester must be authenticated	*Low*	*Medium*	*Low*
6)	Attacker circumvents the firewall and gains access to the internal network in order to eavesdrop or change messages in-transit	All communication in the internal network is secured using `TLS 1.3`	*Medium*	*Low*	*Low*
7)	Attacker gains access to the web server and is able to impersonate it when sending requests to Core CA	Use Host-based Intrusion Detection System on the web server to discover any suspicious activity	*Medium*	*High*	*Medium*
8)	Attacker gains access to backups and is able to modify backup data	Backup data can be modified only when receiving new data from legitimate systems (web server, core CA, database)	*Medium*	*High*	*Medium*

Information

No.	Threat	Countermeasure(s)	L	I	Risk
9)	Attacker is able to recover user password from the database	The database is encrypted and only password hashes are stored	*Low*	*High*	*Low*
10)	Attacker reads/modifies log files	Store log files with the append-only attribute enabled to prevent the attacker from modifying existing entries. Use an IDS (e.g. AIDE) to detect any suspicious changes. Store logs in directories that require root privileges to access	*Medium*	*Low*	*Low*
11)	Attacker modifies configuration files	Configuration files are stored with the **immutable** attribute. Only an administrator may create new configuration files.	*Low*	*High*	*Low*

12)	Attacker compromises TLS private keys	Regularly renew private keys of system components.	*Low*	*High*	*Low*
13)	Attacker impersonates a user by compromising his credentials/stealing his private key	Regular users only have access to their own data, and cannot access data of other users.	*High*	*Medium*	*Medium*
14)	Attacker compromises the Root CA	Shut down CA and rebuild it from scratch with new keys, certificates, etc. Store the root CA's private key in a directory that requires root privileges to access it	*Low*	*High*	*Low*
15)	Attacker removes a compromised certificate from the CRL	Use an IDS to detect changes to CRL, and restore deleted certificate(s) from backup server. Store the CRL in a directory that requires root privileges to access it	*Medium*	*Medium*	*Medium*
16)	Attacker retrieves user data from the database	All user data is stored encrypted.	*High*	*Low*	*Low*
17)	Attacker compromises ssh key(s)	Shut down entire system and re-build with different keys, certificates. Require a minimum of two administrators to confirm major changes.	*Low*	*High*	*Low*
18)	Attacker compromises a backup server's TLS keys	Replicate backup server multiple times. Shut down the compromised server and renew its keys.	*Low*	*High*	*Low*
19)	Attacker tries to intercept user data in transit	All data sent between different system components is secured by TLS 1.3.	*High*	*Low*	*Low*

2.4.3 *Evaluation of Person Assets*

No.	Threat	Countermeasure(s)	L	I	Risk
20)	A system administrator/member of the security staff gets sick and is not able to preform his/her functions.	Employ teams of system administrators and security staff, instead of individuals, s.t. another team member can step in for the sick one	*Medium*	*Medium*	*Medium*
21)	An employee receives a bribe to perform a certain malicious action	Closely monitor employee actions. Pay high salaries to disincentivize accepting bribes	*High*	*High*	*High*
22)	An employee is attacked by someone at work	Have a team of security guards in the office building	*Low*	*High*	*Low*

2.4.4 *Evaluation of Intangible Goods Assets*

No.	Threat	Countermeasure(s)	L	I	Risk
23)	Data theft and disclosure by an attacker.	Always encrypt all the sensitive data (keys, certificates and user's information) for transmission and storage.	*Low*	*High*	*Low*

2.4.5 Risk Acceptance

No. of threat	Proposed additional countermeasure including expected impact
4	Monitor logins to detect suspicious connections. An attacker's connection can then be manually closed by a system administrator
7	Limit the web server's power, i.e. limit the types of queries the web server is able to make to other components, send sensitive data encrypted, so that the web server cannot decrypt it, etc. This will reduce the attackers capabilities, even if he is able to gain control of the web server
8	Store backup data encrypted, and keep the key in a secure place. This will make it more difficult for the attacker to read the backup data, even if he somehow gains access to the backup server
13	Inform users about good security practices, i.e. always use client certificates, secure storage of secret keys, only use strong passwords, etc. This will reduce the likelihood of an attacker being able to compromise the user's credentials
15	Regularly perform consistency checks on the CRL, and restore lost data when needed, thereby reducing the impact of a deleted revoked certificate, as it will be shortly added anew to the CRL
20	Every year, require all employees to go through a full medical checkup. This will reduce the likelihood of an employee getting sick, as any health problem might get preemptively treated
21	Zero-tolerance policy for accepting bribes, i.e. immediate firing and arrest of anyone involved in bribery. This will result in a lower likelihood of someone accepting a bribe