System Description and Risk Analysis

Badoux Nicolas Chibotaru Victor Ilunga Marc

24.10.2016

Contents

1 System Characterization

1.1 System Overview

1.1.1 System mission

The system is a Certificate Authority (CA). A CA provides services such as issuing or revoking certificates. Certificates are electronic credentials that are used to certify the identities of individuals, computers, and other entities on a network [10]. By issuing certificates, a CA plays a key role in the chain of trust which is used in modern world wide web.

To be able to issue a certificate, the CA first has to verify the identity of the user requesting a certificate. This can be done through physical way (telephone call, visit in person, etc.) or by the use of identification methods established in the past (previous certificate, password, smartcard, etc.).

Once the user identity is verified the CA can share a certificate with him. The certificate itself is part of a Public Key Infrastructure (PKI). The user will receive a private key, with which he will be able to sign messages and decrypt messages sent to him. The CA then publishes the corresponding public key. This key will let other users, who trust our CA, verify that a message coming from the user has a valid signature. It will also let other users encrypt their messages before sending them to our user. These messages will only be decryptable by a user in possession of the private key.

It becomes obvious that, once the certificate is published and the user is in possession of his private key, it's his role of keeping this key secret. Since something can always go wrong and users are usually not great at keeping things secret, CAs provide a way to revoke a certificate.

To do that, the CA must be sure that the person trying to revoke a particular certificate is effectively the one that acquired it. Once a user decided to revoke a certificate, the CA should tell other users that this certificate is not valid anymore and should therefore not be trusted in the future. This is done with the help of Certificate Revocation List (CRL).

Our system will provide these two main functions to the clients. It will provide additional functions to the CA administrators and to the system administrators.

1.1.2 System boundaries

Our system will be accessible to the clients via a basic web interface. This interface will provide the above mentionned functions to the clients once logged in.

We will also offer any account creation or removal as this is most of the times linked to other account in the company we don't manage.

The system will not grant certificate to users who have generated their own key pair as it might be done with other Certificate Authorities.

The secrecy of the private keys hold by the users is not guaranteed by the system. As a consequence, a user whose private key and certificate have been

compromised can find himself knocked out of the system if the malicious user revokes all the certificate and changes the login credentials.

1.1.3 Overall system architecture

Our system is composed of four elements internally. First at the network entry point we have a firewall, behind it we have one server providing the web-server as well as running the CA functionality. We have one server in charge of the logs and the backups and a last one hosting the databases we use.

- Firewall: The goal of the firewall is to filter traffic. It is our first line of defense against outside attacker. It's dropping traffic we don't expect and redirect the expected traffic to the correct machine. By limiting the number of entry points we apply one core principle of security design: limiting the surface of attack.

- Webserver and CA server: On this machine, we host the core functions of our system. The web-server provides a client web interface for login and ordering a new certificate or revoking an old one. It also provides a way for the client to amend the personal information we have about them. The login process can be done either with a pair of identifier and password or with an issued certificate. It also provides an interface for the CA admin (accessible only with a valid CA admin certificate).

- Database server: The server hosting our database runs MySQL and hosts one database containing the user login and personal data.

- Log and backup server: This server manages the backup and information logging processes. It assures that backups are done regularly and that logs are tamper-proof.

The four servers communicate between themselves through the internal network.

We also provide SSH access for the system administrators based on administrator certificates.

The internal network consists of all servers located behind the firewall. The users, as drawn in Figure 1 can be in or out of the company network but will get through the same architecture and process to access the system.

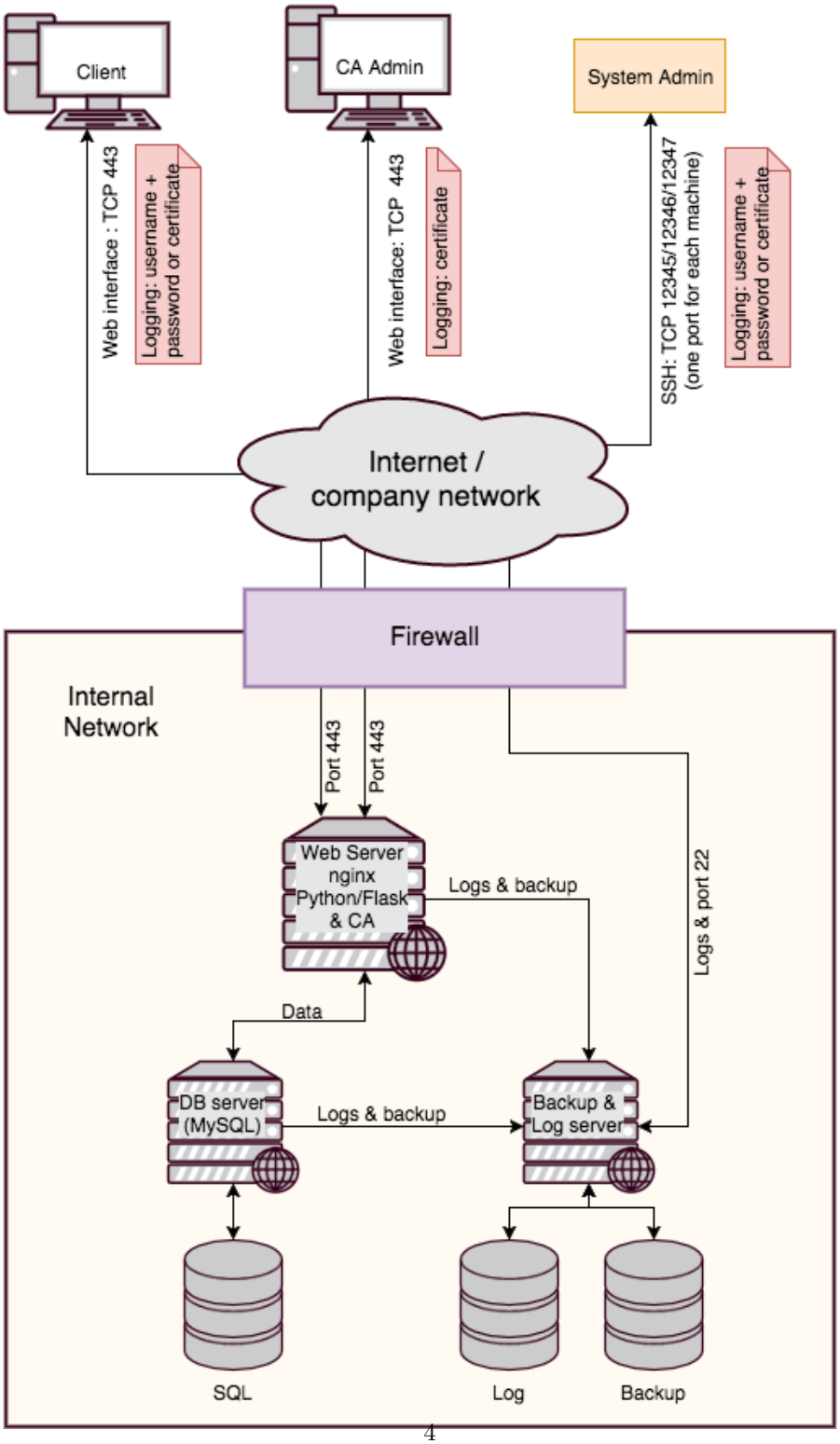

Figure 1: Overall system architecture

1.2 System Functionality

We can separate our system functions in five different categories:

Client web interface: The first webpage displayed to the client will be a login page. He will have the choice to login with his username and password or with a certificate. Once the login is successful, the user's information (name, email, etc.) will be displayed to him. The same page will also display valid certificates owned by the user. For each certificate there is a link to download it and the corresponding private key in PKCS12 format or to revoke it. There is also a link which offers the functionality of amending user's information. There is also a log out button to terminate the session.

CA administrator web interface: The landing page will offer to login with a certificate. After login, the administrator will have access to different statistics about the current state of the CA. Compared to regular users, the CA admin is not issued a session id. His login process implies checking the validity of his certificate and is done for each request.

System administrator: The system administrator will be able to login over SSH. It will be the only interface provided to him. From there, he will be able to manage the system with certain privileges and access backups and logs.

Logging: We will log different information we think could be useful to the system administrator in case of a failure or an attack. This data contains but is not limited to:

- Actions regarding certificates (time, user, type of action, certificate ID)
- Change in user information
- Logging sessions lifecycle (time, user, method (password/certtifcate), success/failure)
- Errors
- Network data (access to restricted ports, ...)

Backup: The critical data (keys, certificates and configuration files) is regularly backed up. The backups are stored on a dedicated server and designed to be tamper-proof. Backups will only be accessible by system administrators.

1.3 Security Design

For this project we try to follow the security principles outlined in *Applied Information Security* [4]. As apparent in the previous section, we will keep our system small and by the *Minimum Exposure* principle reduce the risks. We will

also use the different tools offered by the Linux operating system to compartmentalize the resources we offer to our users.

We outline below some security design of critical parts of our system.

Session Management: We need to track our identified users as well as provide some security for their sessions. For that we will implement a session management mechanism based on cookies. Cookies are data a website can ask the browser to store. The next time the browser connects to the website, he will automatically submit the cookies to the website. Based on these cookies, the website can infer which user it is and what it wants to display.

We followed the recommendations of OWASP on session management. [14] These are:

- Session ID: The session ID let us assign an ID to every logged in user. This way we can recognize him in the future. We need to randomize the session ID to avoid a brute-force attack. Assigning them in linear order for example, makes it obvious to know the ID of the person connected directly after you. To prevent brute-forcing or guessing attacks we will take a session ID length of 128 bits.

- Cookie attribute: We will set both the *secure* and *httpOnly* attribute to prevent eavesdropping the session ID as well as Cross Site Scripting (XSS) attacks stealing the cookies. We will not have persistent cookies to avoid storing them on our client disk.

- Session expiration: Our session management will also implement a timeout after which the user will be logged out. Basically, if the user doesn't interact with our website, it will wait for the timeout and if a new request comes after this timeout it will ask for a renewed login. This timeout provides a security in case the user doesn't log off by himself. By automatically logging him off, our system ensures that if somebody gets access to his laptop long after his session, he will not be logged in. We will have an idle timeout as described above but also an absolute timeout to prevent, in case of hijacking of the session, the attacker to keep the session on for an infinite amount of time.

- We will also implement measures against Cross Site Request Forgery (CSRF tokens to avoid CSRF vulnerabilities).

Key management: As for the session management, we will, again, follow the key management guideline from OWASP. [13]

- Key generation: We generate keys with a public and well known library; *OpenSSL*. We searched online for the last standard on how to use *OpenSSL* in the correct manner. We make sure to always use up-to-date algorithms in all our operations (for example, specifying SHA-256 as signature algorithm instead of SHA-1).

We built a *root* CA that issued a certificate to a second *intermediate* CA. All our further certificates are issued from the *intermediate* CA so that, if the private key of the intermediate is compromised we can revoke it with the help of the *root* CA elements that are keeps in a air gapped safe outside of the company. All our certificates are therefore part of the chain of trust that is composed by the *root* and *intermediate* CAs.

- Distribution: We will ensure that the private keys contained in the `.pk12` files transmitted are provided to our client over a secure channel. For that we will use TLS and SSL to provide data integrity and privacy. This is all implemented in the web-server.

- Storage: A copy of the certificates is kept by the web-server to let the user download again the certificates in case he wants too. The keys will be backed up regularly and the backup will be tamper-proof. Since the web-server needs to be able to use the private keys, encrypting them does not provide more security. The access to the folder containing them is strictly restricted however as the principle of Least Privilege. We apply the same strategy to the `.pk12` files as they also contain the private keys.

Network security The main line of defense of our private network is the firewall. The firewall controls the single entry point of the network and is strictly configured. Only the necessary ports are open and they redirect. We open ports for the web interface and for the SSH logins in for different users. All other ports are blocked. Inside the internal network, we limited the relations the different services can have between themselves. For example, the web-server can only upload backup to the backup server but not retrieve some. The security of the data over the network is guaranteed by creating a secure connection between the client and our system. For the web-server, this consists of SSL and TLS. For the remote management capabilities, it means SSH.

Security of data at rest To ensure data confidentiality on the disk, we have applied full disk encryption (FDE) to all the hard drives in our system. To do that, we enabled the encryption capabilities of our Debian systems. We also enforced a strict policy of least privilege. By applying the smallest set of required access to the users, we decrease the risk they generate if they are compromised. The legacy users database is not as strong as we would want it to be. By providing us with passwords that are not salted and the interdiction to change that, we are unable to provide the security standards that require handling accounts' information. Adding a salt to the passwords would protect our database, in case it gets compromised, from rapidly leaking all the users passwords.

1.4 Components

1.4.1 Platform

- Webserver: The web-server is a Debian 8.8.6 Jessie machine. Three users accounts are configured on the machine, root, admin and caweb. All three

users have passwords corresponding to the actual standards [9]. The web-server is run by the caweb user with reduced privilege (Least privilege principle). The admin account is used by the system administrator over SSH.

- Firewall The firewall is also a Debian 8.8.6 Jessie server. It provides a firewall service (powered by iptables [7]) to the system and is therefore the only point of entry to our internal network.

- Database server: The machine running the MySQL database is also a Debian 8.8.6 Jessie server. Three users exists on this machine, root, admin and mysql. The admin account is only used by the system administrator. As for the web-server, the root account should not be in used. The last account, mysql, is used to run the MySQL database server.

- Backup and log server: This machine provides the logs and backups for our whole system. As for other servers there are three main actors: admin, root and backups. User backups runs all the backup operations. User admin is used for administration. User root should not be used.

- Client Machine: The client is also a Ubuntu 12.04 machine. We have configured the client to trust the CA that issued our SSL certificate for our web-server. This way, we don't have a red lock in the URL bar of the client browser. We have also created a certificate for SSH sessions and different bash scripts to aid the administration process. The client's browser also has an CA admin certificate imported.

1.4.2 Applications

- Web Server: It provides user interfaces, as well as mechanisms for certificate creation, delivery and revocation. The web application is built with Flask [2]. The web-server is run with nginx 1.11.6 [11].

- Core CA: It's hosted on the web-server. It manages user certificates and CA configuration. The certificates are issued with the OpenSSL 1.1.0c library [8] and some automation script in bash [15]. CA configuration files are our own but built upon the work of Jamie Nguyen [12].

- MySQL Database: Legacy database with user's data. The database is run with MySQL [6] and is hosted on the database server.

- Backup: Keeps a backup of keys, certificates from the core CA and logging information. The backups only accessible to the system administrator.

Backups are done automatically at regular intervals to provide us with a recent state of the system in case of failure. This is done with the help of the rsyslog [3] and SSH commands. The backups are made tamper-proof by the usage of append-only files (created with linux command chattr [5]) and secure communication channel (offered by SSH). The logs' integrity can only be compromised if the attacker manages to obtain the root privileges on the backup server.

- Client: Sample system that allows to access the CA's functionality from outside the company's network. It is configured such that all functions can be tested. Besides it provides configuration of a special certificate to test the administrator interface.

1.4.3 Data records

- Users' data: Those are stored in the MySQL database and consist of:

  1. **user id**(uid)
  2. **username**(first and last name)
  3. **email**
  4. hashes of the **password**

  We store the passwords hashed. The rest is stored in plain text. The user id is not modifiable by the user.

- Logs: Contain useful information to monitor the system. The logs will be periodically cleaned up manually, however, relevant information of the logs will be backed up. Logs will be only accessible to the system administrator. We tried to make the logs as tamper-proof as possible by using append-only files. If we had unlimited resources, we would write them on a write-once media (DVD for example).

- Keys backup: When issuing a certificate, we have to generate a private key that is associated to it. This private key is then sent to the client with the certificate under the `.pk12` file format. The web-server keeps a copy of both to be able to offer the client to download the certificate and associated private key again. These keys are stored on the encrypted disk of the web-server and are also backed up by the backup server.

2 Risk Analysis and Security Measures

2.1 Assets

All the system's assets can be divided into the following groups:

1. Physical assets

(a) **Web server**: The machine on which the CA and the web server run. As this server contains all the private keys and is hosting the core service of our company, it's crucial to protect it against physical access by attackers. Another very important property, which should be respected for the web server, is availability.

(b) **Database server**: This machine hosts all the passwords of the user accounts. Compromising this database would be a severe hit to the security of our employees private life and our company if not discovered quickly enough.

(c) **Backup server**: Since it contains a backup of the web server and the database server this means that backups contain both private keys and user's passwords. The backups have also a huge value to us if we need them. Therefore, protecting the backup server should be high on our priority list. For example, we should prevent physical access to the backup server.

(d) **Firewall**: The firewall doesnt contain any critical information. However, since it plays a big role in our defense mechanisms we should make sure it's always available.

2. Logical assets

(a) **Private keys**: The certificates are used for secure email communications. The security of the certificates is guaranteed by the confidentiality of the private keys associated to them. Thus, the main security requirements for the private keys are confidentiality and integrity - nobody besides legit users of the system (listed below) should be able to read the private key's data. To keep our CA trustworthy, we also need to make sure that the CA private key stays in our control and that we are the only one capable of issuing certificates signed by our CA.

(b) **User data**: The users' data stored in the database contains information such as the first name, last name and email of the users. As for private keys, such data must be kept confidential and tamper-proof. Only the legit users should have access to it. These assets are however less critical than the private keys and certificates. Indeed, if somebody manages to steal a pair of private key/certificate, they can log on our system as the legit user and tamper with his private information. The reverse is however not possible.

(c) **System logs**: The logs are used to retain traces of different types of events: data manipulations, system activity, user logins and logouts, etc. To keep these logs trustworthy, we require integrity. As the logs may reveal some sensitive information about the system, they should also be confidential (only the system administrators should be able to read them).

(d) **Internet connection**: System's users should be able to access it from outside the company's network, so the Internet connection is also an important asset. All the communications with the system should be confidential to not allow Man-In-The-Middle (MITM) attacks.

(e) **Trust**: The role of a CA is proving the other parties the identity of a client. Therefore, the CA must be trustworthy from the other parties' point of view. As we are in a company, we can enforce some users to trust the CA but we still need to keep our CA trust intact.

3. Persons

(a) **Employees (aka clients)** are the largest group of system users. Unfortunately, they are also exposed to various risks (e.g. social engineering attacks, credentials theft, etc.). Thus, the employees should also be considered as assets and should therefore be protected.

(b) **CA admins** have access to different confidential data regarding the certificates. CA admins are exposed to same types of attacks as the employees. Hence, we need to protect them too. Because of their function, they also should have received training regarding the different risks and are therefore less vulnerable than a normal employee.

(c) **System admins** are the group of users with the most privileges. Anybody having the system administrator permissions can do almost whatever he/she wants with the system. Sadly, as all other users, the system administrators are prone to different attacks. If we take in account the likelihood and the impact of such attacks, then it becomes obvious that defense for this risk is an imperative. On the other hand, system administrators are trained people and have in most cases a stronger background in security than the average user.

2.2 Threat Sources

- **Script Kiddies**: The system's web server is accessible from Internet and, thus, is susceptible to script kiddies' attacks. Their main motivation is glory rather than profit. Hence, their goals would be to leak the confidential data or to crash the system.

- **Skilled Hackers**: Skilled hackers differ from script kiddies in their monetary interest. They may be hired by company's competitors to steal the sensitive data. Their attacks will probably be more stealthy and harder to discover than the script kiddies' ones.

- **Malware**: As said before the system uses Internet as main communication medium. This means that it is also exposed to malware. Malware can arrive on our system due to a delayed update that was correcting some vulnerabilities. A common example is ransomware that encrypts the database and requires some ransom for the decryption keys.

- **Employees**: As the employees may have different reasons to harm to the company (e.g. bribery from competitors, low salaries, discontent with current job, ...) they also should be considered a threat source.
- **National agencies/state sponsored attackers**: It doesn't seem likely that our business is of interest for an attacker of this level. Since their resources are almost unlimited, as a company we will not have the tools to defend against them. In case we detect such an attack, we would probably request help from our own country cyber security group.

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 assets 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 assets 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 assets 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.

Likelihood definition taken from Basin et al. [4].

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 organizations 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 organizations 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 organizations mission, reputation, or interest.

Impact definition taken from Basin et al. [4].

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 Asset "Web Server"*

No.	Security requirement	Consequences if not respected
1	Restricting physical access	Gaining physical access to the web server will allow the attackers to literally do whatever they want with the system. By having FDE [1], we make the life of the attacker a bit harder, but it's still possible to change the root password (coldboot attack, keylogger).
2	Availability	Compromising the main server's availability is equivalent to making the system useless

No.	Threat	Countermeasure(s)	L	I	Risk
1	Organisation with both skilled hackers and burglars get physical access to the web server	Placing the web server in a place accessible only by authorized personnel (system administrators). Implementing some additional physical security measures (e.g. guards, cameras, ...) to provide proper access control	*Low*	*High*	*Low*
2	Script kiddies or skilled hackers deliver a DDOS attack	(please see chapter risk acceptance)	*Medium*	*High*	*Medium*
3	Malware attacks the web server and makes it nonfunctional. This could be the case with a ransomware for example	Proper maintenance; regular security patches	*Low*	*High*	*Low*
4	Script kidies or skilled hackers get remote access to the webserver and render the web interface nonfunctional (e.g. defacing the website or stop the webserver completely)	Applying patches as they roll out; monitoring intrusion and attacks on our system	*Low*	*High*	*Low*
5	Employee harms the CA by revoking certificates or by issuing malicious certificates or deleting/publishing private keys	Monitoring logs regularly and taking countermeasure as fast as possible as well as contacting clients. Require two users credentials to access the CA private key.	*Low*	*High*	*Low*

2.4.2 *Evaluation Asset "Database server"*

No.	Security requirement	Consequences if not respected
1	Restricting physical access	Gaining physical access to the database server will allow the attackers to try to bypass the disk encryption and in case of success get access to everything that is stored on it.
2	Availability	Compromising the database server's availability is equivalent to making the system useless as the users will not be able to login with passwords or review their personal information before asking for a new certificate.

No.	Threat	Countermeasure(s)	L	I	Risk
1	Organisation with both skilled hackers and burglars get physical access to the server (e.g. intrusion in the datacenter)	Placing the database server in a place accessible only by authorized personnel (system administrators). Implementing some additional physical security measures (e.g. guards, cameras, ...) to provide proper access control	*Low*	*High*	*Low*
2	Script kidies or skilled hackers get remote access to the server and can remove proper access to the database. (e.g. ransomware with encryption)	Applying patches as they roll out; monitoring intrusion and attacks on our system	*Low*	*High*	*Low*
3	Employee remove or delete the database.	Doing backup regularly and protect the database against a single user misbehaving (see 2.4.3)	*Low*	*High*	*Low*

2.4.3 *Evaluation Asset "Backup and log server"*

No.	Security requirement	Consequences if not respected
1	Restricting physical access	Gaining physical access to the database server will allow the attackers to try to bypass the disk encryption and in case of success get access to everything that is stored on it.
2	Availability	Compromising the backup and logs server's availability is putting the system at risk in case of failure (physical/bugs) as we would not be able to restore it to a working state.

No.	Threat	Countermeasure(s)	L	I	Risk
1	Organisation with both skilled hackers and burglars get physical access to the server	Placing the backup server in a place accessible only by authorized personnel (system administrators). Implementing some additional physical security measures (e.g. guards, cameras, ...) to provide proper access control	*Low*	*High*	*Low*
2	Script kidies or skilled hackers get remote access to the server (via misconfiguration, 0d vulnerability or unpatched known vulnerabilities) and are able to remove proper access to the logs and backups.	Applying patches as they roll out; monitoring intrusions and attacks on our system	*Low*	*High*	*Low*
3	An employee destroy backups and logs.	Requesting two users credentials to remove backups.	*Low*	*High*	*Low*

2.4.4 *Evaluation Asset "Firewall"*

No.	Security requirement	Consequences if not respected
1	Restricting physical access	Gaining physical access to the firewall would let the attacker remove the firewall from our network architecture decreasing the overall security.
2	Availability	If we don't have an active firewall, the network security is way lower.

No.	Threat	Countermeasure(s)	L	I	Risk
1	Skilled hackers get physical access to the server running the firewall	Placing the backup server in a place accessible only by authorized personnel (system administrators). Implementing some additional physical security measures (e.g. guards, cameras, ...) to provide proper access control	*Low*	*Medium*	*Low*
2	An employee deactivate the firewall	Putting up an alert that run automatically if the firewall is deactivated.	*Low*	*Medium*	*Low*

2.4.5 *Evaluation Asset "Private keys"*

No.	Security requirement	Consequences if not respected
1	Confidentiality	Attackers will be able to read all email communications that were supposed to be confidential
2	Integrity of client keys	Attackers would make email received by the client unreadable and/or make the communication originating from the client vulnerable to a MITM attack
3	Integrity of CA private key	Attackers would be able to sign certificates and therefore impersonate clients or create fake clients.

No.	Threat	Countermeasure(s)	L	I	Risk
1	Script kiddies or skilled hackers get unauthorized access to the CA private key and issue new certificates via the web interface	Implementation of proper access control mechanisms and stric least privilege policy	*Low*	*High*	*Low*
2	Script kiddies or skilled hackers steal the user private keys by eavesdropping a user download. Eavesdropping can be done on the network or on the client machine	Use of encrypted communications (HTTPS) to prevent eavesdropping over the network (e.g. MITM). We cannot assure confidentiality of the key as soon as it enters the users machine	*Low*	*High*	*Low*
3	Skilled hackers get access to the keys backup by exploiting a vulnerability in one of the system components (0day vulnerability, unpatched vulnerabilities or misconfiguration)	1. Encryption of the backup to make these useless in case of data leaks 2. Control of backup integrity to avoid data tampering 3. Regular updates of all software to fix discovered vulnerabilities 4. System supervision by administrators with a notification channel (e.g. pager) to quickly react to all system irregularities	*Low*	*High*	*Low*

2.4.6 *Evaluation Asset "User data"*

No.	Security requirement	Consequences if not respected
1	Confidentiality	Attackers will find out the employees' data (first name, last name and email), which can be used for further social engineering attacks. Also, the attackers will get the hashes of users' passwords, which can then be used to recover the original passwords (these are not salted). It will also cause a reputation loss in case the leak become public
2	Integrity	Attackers will be able to modify employees' data. The most probable attack would be to modify the stored hash of a user's password and get access to the system on his behalf.

No.	Threat	Countermeasure(s)	L	I	Risk
1	Script kiddies or skilled hackers get unauthorized access to users' data via the web interface. Could occur in case of misconfiguration of the authorization mechanism	Implementation of proper access control mechanisms	*Low*	*High*	*Low*
2	Script kiddies or skilled hackers steal users' data by eavesdropping a user session	Use of encrypted communications (HTTPS)	*Low*	*Medium*	*Low*
3	Skilled hackers get access to the database by exploiting a vulnerability in one of the system components	1. Regular updates of all software to fix discovered vulnerabilities 2. System supervision by administrators with a notification channel (e.g. pager) to quickly react to all system irregularities	*Low*	*High*	*Low*

2.4.7 *Evaluation Asset "Logs"*

No.	Security requirement	Consequences if not respected
1	Confidentiality	Attackers will be able to discover some information about the system, which can simplify further attacks.
2	Integrity	Attackers will be able to modify the logs to hide the traces of their attacks.

No.	Threat	Countermeasure(s)	L	I	Risk
1	Skilled hackers or script kiddies eavesdrop a system administrator's session and get access to the logging system	Use of encrypted communication (SSH) as well as training our system administrator to identify and avoid risky situation (public unencrypted wifi network, shared internet caf compuer...)	*Low*	*Medium*	*Low*
2	Skilled hackers get access to the logs by exploiting a vulnerability in one of the system components	1. Encryption of all logs to keep confidentiality 2. Integrity control to avoid tampering 3. Regular updates of all software to fix discovered vulnerabilities 4. System supervision by administrators with a notification channel (e.g. pager) to quickly react to all system irregularities	*Low*	*Medium*	*Low*

2.4.8 *Evaluation Asset "Internet connection"*

No.	Security requirement	Consequences if not respected
1	Confidentiality	Attackers will be able to eavesdrop all the users' (employees, CA admins, system admins) communications with the system. This will obviously compromise all the CA infrastructure
2	Integrity	Attackers will be able to modify user requests to the server and server response. They could do a MITM attack during the certificates issuance.

No.	Threat	Countermeasure(s)	L	I	Risk
1	Skilled hackers eavesdrop or tamper user requests	Use of secure communications (HTTPS and SSH)	*Low*	*High*	*Low*

2.4.9 *Evaluation Asset "Trust"*

No.	Security requirement	Consequences if not respected
1	Integrity	Client and other parties would not use the services provided by the CA making it useless

No.	Threat	Countermeasure(s)	L	I	Risk
1	The CA admin or system administrator makes an error which result damages to the users and loss of trust	Train our employees properly and regularly. Ask for a confirmation before every critical operation (revoking a user certificate for example)	*Low*	*Medium*	*Low*

2.4.10 *Evaluation Asset "Persons"*

No.	Threat	Countermeasure(s)	L	I	Risk
1	Social engineering attacks against personnel may lead to unauthorized access to confidential data (in case if a user or a CA admin is attacked) or to more devastating consequences (in case of a system admin)	Proper training for all personnel against different types of social engineering attacks (e.g. phishing)	*Low*	*Medium*	*Low*
2	Unintended credentials loss (password and/or certificates) may have the same consequences as the social engineering attacks	Employees are trained to report cases of credentials loss/theft; "revoke certificate" function	*Low*	*Low*	*Low*
3	Competitors bribe employees to get confidential information	Non-Disclosure Agreement and other contractual commitments	*Low*	*Medium*	*Low*

2.4.11 Risk Acceptance

Threat	Proposed additional countermeasure including expected impact
DDOS attacks	Because of the simplicity of current system design there are no countermeasures implemented against DDOS attacks. But, to further reduce the risk the following steps can be taken: 1. Purchasing special systems for detecting and preventing DDOS attacks. 2. Using the help of Cloud Mitigation Providers to filter out traffic. Implementing one of these measures will reduce the likelihood, and thus the risk, from *Medium* to *Low*.

References

[1] Full disk encryption wikipedia article. URL https://en.wikipedia.org/wiki/Disk_encryption.

[2] Flask (a python microframework), 2016. URL http://flask.pocoo.org/.

[3] rsyslog official web-site, 2016. URL http://www.rsyslog.com/.

[4] David Basin, Patrick Schaller, and Michael Schläpfer. *Applied Information Security: A Hands-on Approach*. Springer, 2011.

[5] Remy Card. chattr(1) - linux man page, 2016. URL https://linux.die.net/man/1/chattr.

[6] Oracle Corporation. Mysql the world's most popular open source database, 2016. URL https://www.mysql.com/.

[7] Herve Eychenne. iptables(8) - linux man page, 2016. URL https://linux.die.net/man/8/iptables.

[8] OpenSSL Software Foundation. Openssl cryptography and ssl/tls toolkit, 2016. URL https://www.openssl.org/.

[9] Dropbox Inc. zxcvbn password strength estimator, 2016. URL https://github.com/dropbox/zxcvbn.

[10] Microsoft. What are certificates?, 2003. URL https://technet.microsoft.com/en-us/library/cc758348(v=ws.10).aspx.

[11] nginx.org. nginx, 2016. URL https://nginx.org/en/.

[12] Jamie Nguyen. Openssl certificate authority, 2015. URL https://jamielinux.com/docs/openssl-certificate-authority/.

[13] Brian Russell & Drew Van Duren & Vanessa Amador & Susanna Bezold & OWASP. Key management cheat sheet, 2016. URL https://www.owasp.org/index.php/Key_Management_Cheat_Sheet.

[14] Raul Siles & OWASP. Session management cheat sheet, 2016. URL https://www.owasp.org/index.php/Session_Management_Cheat_Sheet.

[15] GNU Project. Gnu bash, 2016. URL https://www.gnu.org/software/bash/bash.html.