System Description and Risk Analysis

Cyrill Krähenbühl Silvan Egli Lukas Bischofberger

December 8, 2016

Contents

1 System Characterization

1.1 System Overview

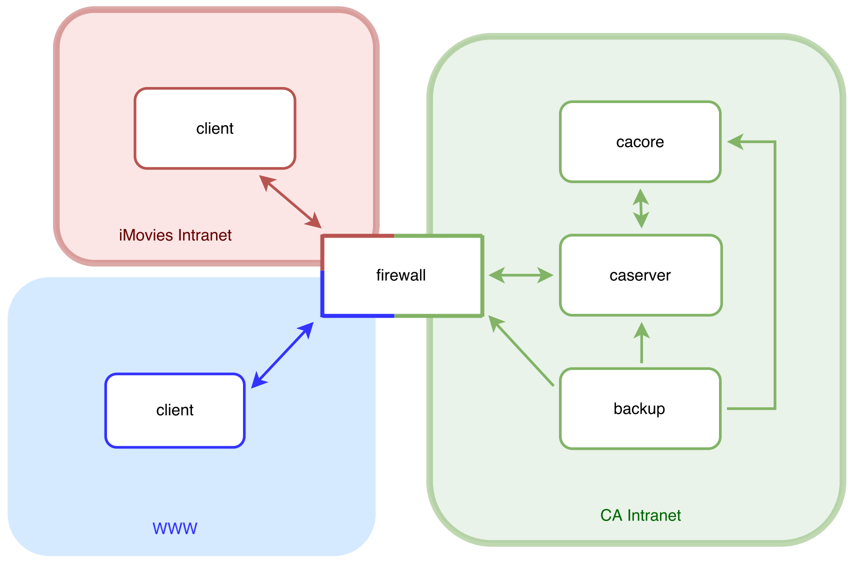

Figure 1: System overview

The mission of the system consist of providing a certificate authority (CA) service through which employees of the fictional company iMovies can request and revoke digital certificates. These certificates will be used for secure e-mail communication between employees. In the following these employees will be referred to as clients and users.
Figure 1 shows an overview of the proposed system. It consist of four subsystems, namely firewall, CAServer, CACore and backup and forms the CA Intranet. Clients either use the service from inside the company's network or from the Internet. They are interfaced with the system by the firewall which blocks unallowed connections and forwards the request to the CAServer. The CAServer subsystem provides a web interface for clients and administrators. It performs the authentication and authorization and handles the user requests as well as their sessions. In order to do so, the CAServer makes use of the CACore's service. The CACore's task is to control the process of issuing and revoking certificates as well as managing the user data.The backup is responsible for maintaining a copy of all system relevant data, including information on users, certificates, keys, logs and configurations. Therefore the backup has access to the other three subsystems.

1.2 System Functionality

In this section we describe the systems functions in terms of system design based on the use cases given in the project assignment. The focus lies on the implementation and the information flows. In subsection 1.4 the properties of the mentioned applications will be described in more detail.

1.2.1 Components

Client Clients use the system through a web interface which is built using the Angular2[1] framework written in TypeScript[2]. The web front end consumes the RESTful[3] service provided by the back end on the CAServer by performing http requests to the firewall.

Firewall In order to block connection attempts to unallowed ports the firewall makes use of the iptables[4] functionality. In addition to the packet filtering, iptables is configured to act as a NAT and thus forwarding requests from the internet and the company's intranet to the web server on the CAServer. In order to configure iptables the csf[5] application is used.

CAServer The nginx[6] webserver is responsible for handling the forwarded requests from the firewall. This includes delivering static content (front end and certificate revocation list) and forwarding the other requests to the back end. In addition it checks if the client sent a user certificate and verifies it using the root certificate.
The back end consist of a Django[7] application written in python which is served by a uwsgi[8] application server The following tasks are subject to the backend:

- user authentication
- authorization checks
- session management
- user input validation
- output sanitization
- database queries
- site calls to the CACore subsystem

[1]`https://angular.io/`
[2]`https://www.typescriptlang.org/`
[3]`https://en.wikipedia.org/wiki/Representational_state_transfer`
[4]`http://ipset.netfilter.org/iptables.man.html`
[5]`https://configserver.com/cp/csf.html`
[6]`https://nginx.org/`
[7]`https://www.djangoproject.com/`
[8]`https://uwsgi-docs.readthedocs.io/en/latest/`

- providing the CA administrator interface
- providing a RESTful[3] API to the front end

CACore On the CACore there is a MySQL[9] database containing the legacy user data table and all tables needed by the Django application. The CAServer accesses the database over the network. In addition, the CACore provides the functionality of a CA by using the openSSL[10] library. There is an intermediate CA and a root CA on the CACore. The intermediate CA is used for issuing the user certificates. It's private key is not encrypted in order to sign the user certificates non interactively. The root CA's only task was to issue and sign the intermediate CA's certificate. It's private key is encrypted and kept offline. From now on, when speaking of the CA, the intermediate CA is referenced if not stated otherwise. The CA supports the creation of new X509[11] user certificates as well as their revocation by using the "ca" command of openSSL[10]. The certificate creation and revocation are implemented as a Bash[12] script which are called by the Django[7] application using ssh[13].

Backup The backup periodically (once every hour) backs up all system relevant data of the firewall, CACore and CAServer. This includes application and system logs, configurations on all machines and the key/certifcates archive and database on the CACore. To perform a backup, the backup mounts the filesystem of each server that should be backed up using SSH file system (sshfs)[14]. All of the servers have the public key of the backup stored in their roots authorized_keys file, such that the backup can connect to the machine without password prompt and access any file. The relevant folders are incrementally backed up using duplicity[15] which encrypts the backed up data on the system. Duplicity is based on rsync[16] and GnuPG[17] and allows encrypted and signed incremental backups. The backup is scheduled using cron and executed by the root user. The backup scripts and the mounted file system are only modifiable by root.

1.2.2 User Authentication

The first thing a user has to do before using the system is to authenticate. This can be done in two different ways. The term Django will be used interchangeably for the Django[7] application from now on.

- **credentials**

[9] https://www.mysql.com/
[10] https://www.openssl.org/
[11] https://en.wikipedia.org/wiki/X.509
[12] https://www.gnu.org/software/bash/bash.html
[13] labelsshhttps://en.wikipedia.org/wiki/Secure_Shell
[14] https://github.com/libfuse/sshfs
[15] http://duplicity.nongnu.org/
[16] https://sourceforge.net/projects/librsync/
[17] https://www.gnupg.org/

   1. Client calls `https://web.seclab/` and enters his username and password
   2. Information is forwarded to the Django application which compares the credentials with the existing ones in the database.
   3. If they match a json web token (JWT)[18] containing all user data together with an expiration date is issued. The JWT is signed using Django's secret key.
   4. For further authentication, the front end sends the obtained JWT with every subsequent request.

- **user certificate**
   1. Client calls `https://web.seclab/` and selects "Login with certificate". The prompt for a user certificate should have been performed by the browser.
   2. The nginx web server notices that a certificate is attached. It validates the certificate using the the key contained in CA's certificate and checks if the certificate is valid using the certificate revocation list. If the verification is okay, nginx extracts the user data out of the subject field of the user certificate. If not, "a verification failed" header field is added to the request. The request and the information are forwarded to Django.
   3. The following steps are identical to the case with credentials.

1.2.3 Profile Changes

The user can update his information by submitting the new values in the Profile form. The values are validated by Django before being written to the database.

1.2.4 Logout

A click on "Logout" makes the front end delete the current jwt.

1.2.5 Certificate Issuing

1. The user logs in according to section 1.2.2.
2. The user can correct his information as described in 1.2.3.
3. After having chosen a certificate name (must be alphanumeric only) the request is submitted.
4. Django calls the certificate create script on the CACore via ssh.

[18] `https://jwt.io/`

5. A new X509 certificate containing the user's id and email is created. The certificate is signed with the CA's private key. Additionally, the certificate is bundled together with the corresponding rsa private key into a file in PKCS#12 format (pkcs12 file). The key, certificate and pkcs12 files are stored on the file system.

1.2.6 Certificate Download

1. In his certificate list, the user selects a valid certificate to download.
2. Django uses the python sftp client pysftp[19] to read the pkcs12 file of the user on the CACore.
3. The file is attached to the response which is sent back to the client.

1.2.7 Certificate Revocation

1. In his certificate list, the user selects a non revoked certificate to download.
2. Django calls the certificate revoke script on the CACore via ssh.
3. The CA on the CACore revokes the certificate and build the new Certificate Revocation List (CRL)
4. The new CRL from the CACore is fetched to the CAServer via scp and nginx is reloaded in order to make the changes visible. The CRL is published under `https://web.seclab/crl.txt` (human readable) and `https://web.seclab/crl.pem` (binary).

1.2.8 CA Administrator interface

CA administrators have access to the same web interface with the same functionality like a normal user. By providing a valid admin certificate, a CA administrator can additionally consult the CA's current state under `https://web.seclab/api/certificates-info`. The dedicated web interface provides a list with all issued certificates including their information and revocation status. The current serial number is displayed too.

1.2.9 Backup

The functionality of the automated backup is described in section 1.2.1

[19] `https://pypi.python.org/pypi/pysftp`

1.2.10 System Administration

In order to administrate the system remotely, all subsystem except from the backup run an openSSH[20] server which is accessible from the firewall. The backup is only accessible through a reverse ssh tunnel [21]. The tunnel from the backup to the firewall is established using autossh[22].

1.3 Security Design

1.3.1 Data in transit

All communication channels used are encrypted. This includes the communication from the client to the web server over https, Django to MySQL server using a server certificate, Django to CACore using sftp and ssh, backup to all other subsystems over sshfs.

1.3.2 Data at rest

Cacore All keys and certificates are read only by the main user (cadmin). The MySQL database is restricted to local root access as well remote access for a dedicated user from the CAServer only.

Caserver The nginx web server is run as www-data user which is limited to read the static files, the CRL and CA certificate. The Django settings file containing the database credentials and the applications secret key is only modifiable by the main user (cadmin).

Backup All backups from the subsystems are encrypted. The backup scripts and the mounted file systems are only modifiable by root.

1.3.3 Key management

The private key of the root ca is encrypted and not stored on the system. In case that the intermediate CA's key gets compromised, the intermediate CA certificate can be revoked by the root CA. The encryption of user keys is left to the user.

1.3.4 Session management

In this system user sessions are explicitly created when a user signs in. When sending his credentials to the server he receives a token (JWT) as return. This token can then be added by the user (e.g. the front end application) to every request in the HTTP header. In the browser the JWT is saved in the local session, thus it is deleted once the tab is closed. Also, when the user logs out,

[20] `https://help.ubuntu.com/lts/serverguide/openssh-server.html`
[21] `http://unix.stackexchange.com/questions/46235/how-does-reverse-ssh-tunneling-work`
[22] `https://linux.die.net/man/1/autossh`

the token is deleted. There is no possibility for session stealing if the user does not disclose his token to anyone, otherwise its trivial. The token alone is responsible to authenticate the user, it contains the necessary infos as username and privileges. To counter token forgery, every token is signed by the issuer with its private key. So as long as the private key is kept secret and the signing algorithm is chosen from a secure set, an adversary is not able to forge a token.

1.4 Components

1.4.1 Platforms

Firewall The firewall runs a Ubuntu server 16.10[23] and is connected to the CA intranet with the IP `10.0.0.1`. Its purpose is the separate the CA network from the internal network of the company and the Internet. For that iptables are used with the CSF interface.

CAServer The CAServer runs on a Ubuntu server operating system 16.10 and is connected to the CA intranet with the ip `10.0.0.2`. It runs the webserver (nginx) and the Django application described later, uwsgi connects these services. The server is protected by Uncomplicated Firewall (UWF)[24] and only necessary ports as SSH, HTTP and HTTPS are open.

CACore The CACore runs on a Ubuntu server operating system 16.10 and is connected to the CA intranet with the ip `10.0.0.3`. It hosts the CA scripts, the mandatory certificates and keys and has the openSSL package installed. This server is also protected by UFW, having only ports for MySQL and SSH open.

Backup The backup runs on a Ubuntu server 16.10 and is connected to the CA intranet with the IP `10.0.0.4`. The backup is connected to the firewall using an SSH tunnel, which allows users on the firewall to open an SSH connection the local port 10022 which is forwarded to the backup. The firewall of the backup is as always UFW, which is configured to allow only outgoing SSH connections. As already described above, duplicity is used to backup every server once an hour securely.

1.4.2 Applications

User web interface The user web interface is build on Angular2 as a front end application. It does not offer security measures, as they could easily be bypassed by looking at the code. Therefore the app sends requests to the API which performs the appropriate data validation. The requests are secured in transit to the API using an SSL connection, thus providing confidentiality. The API checks the authenticity of the requests using the JWT which the front end app appends to every request in the header.

[23]https://www.ubuntu.com/server
[24]https://launchpad.net/ufw

Admin interface The CA admin interface is provided by Django directly. Meaning it can only be reached by one specific URL, thereon Django secures the site access by relying on the certificate validation done by nginx. The admin interface exposes the lists of all issued and revoked certificates with their respective details. It also shows the current serial.

API The API which is part of the Django application (Version 1.10) is the interface provided to the user web interface. It receives the requests, authenticates them based on the JWT and issues further actions. Theses actions include accessing the database directly to modify data or calling scripts on the CACore server to create and revoke certificates. The API also sends back the downloadable certificates to the front end.

CACore scripts The CACore scripts are called by the Django application. They then use the openSSL package to perform the necessary actions. The certificates which get created like this, are stored on the filesystem.

User database The database consists of an mysql server running on the CACore server. It receives connections from Django to read and write data. The user that can access it remotely (dbuser) has restricted rights and is only allowed to setup a connection from the CAServer.

1.4.3 Data records

All relevant data records are regularly backed up over an SSH secured channel to the backup and stored encrypted.

User information The user information consists of the legacy MySQL database which is stored as a table on the new database. This includes username, first name and last name as well as the users email and password.

Certificates All the certificates created for the clients, they are signed using the CA intermediate key and are stored on the CACore server and eventually backed up. The certificates as well as the private keys are read only.

Private keys The private keys corresponding to the certificates. They are also stored next to the certificates.

Configuration files Configuration files include all files needed to configure a service our system uses. They usually lie in the /etc folder of every server, they are backed up regularly.

Log files Log files in the /var/log folder of every server contain valuable information produced by the running services. They can help the system administrator find error sources and irregular behaviour.

User credentials User credentials are user data with which the users need to authenticate themselves to the application.

1.5 Additional Material

Firewall seclab: e3sk15mlp

CAServer cadmin: g68?M_tk

CACore coreadmin: pF.ooa5,
mysql:
dbuser: fp..X,i3
root: _Zr.M2gW

Backup backupuser: xsitdn4783
duplicity: : ocneitns37

Client client: 123456
Admin User:
admin: gz_51Mq*

2 Risk Analysis and Security Measures

2.1 Assets

2.1.1 Physical assets

Firewall: The firewall is located in a locked and air conditioned room. There is redundant power supply for its server rack. The states of the firewall are running, compromised and down. Running means everything works as expected, compromised means an unauthorized user has had physical access to the machine and down means the firewall is not running.

Application server: The application server is located in the same server room with redundant power supply, but in a different rack than the firewall. The same states as in the firewall apply here.

CaServer: The CAServer is located in the same server room with redundant power supply, but in a different rack than the firewall. The same states as for the firewall apply here.

CACore server: The CACore server is located next to the CAServer. The same states as for the firewall apply here.

Backup server: The backup server is located in the same rack as the CAServer also equipped with redundant power supply. The same states as in the firewall apply here.

Backup server: The backup server is located in the same rack as the application server also equipped with redundant power supply. The same states as in the firewall apply here.

Internal network: The internal network is an Ethernet local area network connecting the above mentioned components. The components are connected using layer 2 switches located in the server room. The states are running, compromised and down. A running state indicates that only authorized devices are connected to the network. A compromised state may indicate that an unauthorized user has added his own device to the network and is sniffing connections or injecting and blocking messages. A down state indicates that the network is shut down.

External network: The external network connects the firewall to the internet by Ethernet cable using a router that is also located in the server room. The same states as in the internal network apply here.

2.1.2 Logical assets

Connectivity: Connections between each components and connection to the ISP. For the system to work properly, all components need to be properly connected. The states are connected and not connected.

2.1.3 Logical software assets

Firewall operating system: The operating system of the firewall is the latest Ubuntu server edition. It is managed by the system administrator who installs all relevant updates and patches within few hours after their release. The states are running, vulnerable, compromised and down. A vulnerable state indicates that the system is not up-to-date and vulnerable to known exploits. A compromised state means the system was already exploited by an attacker.

Firewall service: The firewall that separates the internal and external network is the latest edition of the Config Server Firewall (csf). The states are the same as for the Firewall operating system.

Appserver operating system: The operating system of the appserver is the same as for the firewall and the same states apply.

Appserver webserver: The appserver runs a nginx webserver which handles all http and https requests. It is updated by the system adminstrator. Its states are running, compromised and down. A compromised webserver allows an attacker for example to perform a man-in-the-middle attack.

Appserver application: The application is written in python and uses the Django framework. It manages the database and creates, revokes and provides certificates to the user. Both python and the Django framework are regularly updated by the system administrator. The states are similar to the webserver, but in a compromised state, an attacker might change the behaviour of the application.

Appserver certificate authority scripts: The functionality as a certificate authority is provided by a set of scripts that rely on the openssl library. The behaviour of the scripts is monitored by the system administrators. The states are similar to the webserver states, but in a compromised state an attacker also has access to certificate related functionality.

Appserver database The database is running MySQL and is updated and monitored for misbehaviour by the system administrator. The states are similar to the webserver, but in a compromised state an attacker has altered the database.

Backupserver operating system: The operating system of the backupserver is the same as for the firewall and the same states apply.

Backupserver duplicity: Duplicity periodically runs on the backupserver and backs up and encrypts valuable data from both the firewall and the appserver such as configurations, logs, certificates, private keys and the database.

2.1.4 Logical information assets

User database: The database contains user ids, email addresses and hashed passwords. The states are confidential and leaked. A confidential state means that only authorized system administrators and corresponding users have access to these information. In a leaked state, an attacker was able to read the whole or part of the database.

Certificates: The certificates of each user, the certificate of the webserver and the root certificate. If a certificate is used by someone other than its owner or a certificate is used even though it was revoked, its state is invalid. Otherwise its state is valid. The severity of an invalid certificate depends on which certificate it is and if the usage of such an invalid certificate was detected, since user certificates can easily be revoked.

Appserver configuration: Configuration files of different services such as webserver, database, Django, certificate authority or ssh can give insight into how the system behaves and might help detect misconfigured and thus exploitable services. The states are the same as for the user database.

Private keys: The private keys for certificates or for ssh connections within the system. Similar states to user database, but the private key is either private or leaked.

Crl: The certificate revocation list has to be up-to-date and available to any user. The states are available if any user can get the list and unavailable if this is not the case.

Backupserver configuration: Configuration files for services such as duplicity. The states are the same as for the appserver configuration.

Logs: Logging information about various services. The states are the same as for certificates.

Login credentials: Login credentials for ssh connections to different machines that may be leaked by a system administrators and login credentials from users that log into the CAServer. The states are the same as for the private keys, but for ssh login credential the security concern is much higher.

JWT: A JSON web token (JWT) describes an active connection of a user to the webserver. If an attacker manages to compromise the system in a way that he is also part of this connection, the state is compromised. For an active confidential connection the state is confidential and after the connection is closed the state is closed.

Archive key: The key that is used to encrypt all backed up data on the backupserver. The states are similar to the private keys.

Intermediate & Root key: The intermediate key to sign the webserver certificate and user certificates and the root key which signs the intermediate certificate. The states are similar to private keys.

2.1.5 Persons

User/Employee: The users of the authenticated mail server, which are employees of iMovie. The state of a user is either loyal or unloyal depending on which relation he has to the company.

CA Administrator: The CA administrators can query the certificate authority for additional information about its state but cannot modify, revoke or create any certificates (except for his own). The states are the same as for user/employee.

System Administrator/Insider: The system administrators manages the system. The states are the same as for user/employee.

Private key holder: The CA administrator holds the private key of the root certificate. The states are the same as for user/employee.

2.1.6 Intangible assets

User confidence: The trust a user has in the system. This is influenced by security breaches, usability of the webserver and other factors. The user either has confidence in the system or not, which means there are two states, confident and not confident.

2.2 Threat Sources

Nature: Environmental factors can hinder the execution of the system. There could be water leaks that would cause damage to servers and where data can be lost.

User: The employees of iMovie can intentionally misbehave and manipulate the system or unknowingly help an attacker compromise the system.

System administrator/Insider: A system administrator is a more impactful threat source to the system than a user, since a compromised system administrator leads to much bigger security concerns than a compromised user.

Script kiddies: Script kiddies most likely do not have iMovie as their primary target, but might still try for example to infect the servers with malware to use them in a botnet. They do not have the skills to infiltrate a well protected system and so the usual security measurements and regular updates should be enough to sufficiently protect against them.

Skilled hacker: A skilled hacker is a big threat source and the usual security measurements most likely do not give enough protection against such an attacker. He might try to infiltrate the CA server and extract private keys to be able to imitate the webserver itself, issue arbitrary certificates or use the keys to perform man-in-the-middle attacks between employees and extract valuable information. He is most likely to be hired by a competitor or a criminal.

Malware: There is always the possibility of either directed or undirected malware infection if users with infected systems interact with the system.

Organized crime: Criminals that try to extract information from the system to blackmail people or steal valuable login credentials that are used across multiple systems.

Competitors: Competitors that want to undermine the reputation of iMovie, gain knowledge about company secrets or simply cause them damage.

2.3 Risks Definitions

The risk and the likelyhood are both defined qualitatively with the values: **Low**, **Medium** and **High**. The definition of each value is taken from "Applied Information Security - A Hands-on Approach" [1].

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.

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.

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

2.4 Risk Evaluation

In the following section we will give a risk evaluation for all possible threats and their impact on each of our assets described above.

2.4.1 *Evaluation physical asset: Hardware*

We can evaluate the risk for our servers and the firewall jointly as the same physical threats apply to them.

No.	Threat	Countermeasure(s)	L	I	Risk
1	Nature: Component failure	Standard configuration, configuration backups, spare machines / components	*Medium*	*Medium*	*Medium*
2	Insider: Accidental or intentional destruction of components	Restrictive room access policies, spare machines / components	*Low*	*Medium*	*Low*
3	Nature: Flooding, fire etc.	Place fire alarm and sprinkler in server room, server room is located in a building on elevated level	*Low*	*High*	*Low*
4	Competitors / Organized crime: Get physical access to server room	Location of server room not public, restrictive access policy	*Low*	*High*	*Low*

2.4.2 *Evaluation physical asset: Internal network*

The networking assets include the network cables and the switches/routers used in the server room.

No.	Threat	Countermeasure(s)	L	I	Risk
5	Nature: Component failure	Commodity switch/router, spare cables	*Low*	*Medium*	*Low*
6	Insider: Accidental or intentional destruction of components	Restrictive room access policies, spare cables, backup switch	*Low*	*Medium*	*Low*
7	Insider: Network misconfiguration	Standard configuration, clear documentation	*Medium*	*Medium*	*Medium*
8	Nature: Flooding, fire etc.	Place fire alarm and sprinkler in server room, server room is located in a building on elevated level	*Low*	*Medium*	*Low*
9	Competitors / Organized crime: Get physical access to server room	Location of server room not public, restrictive access policy	*Low*	*Medium*	*Low*

2.4.3 *Evaluation physical asset: External network*

No.	Threat	Countermeasure(s)	L	I	Risk
10	Nature: ISP failure	Redundant connection to ISP	*Low*	*Medium*	*Low*

2.4.4 *Evaluation logical asset: Firewall software*

No.	Threat	Countermeasure(s)	L	I	Risk
11	System administrator: Misconfigure firewall, purposely include backdoor	System administrators check for misbehaviour of other system administrators	*Low*	*High*	*Low*
12	Skilled hacker: Bypass firewall	Use restrictive access rules, regularly update system, keep access logs	*Medium*	*Medium*	*Medium*
13	Espionage / Organized crime: Bypass firewall, use zero day exploits	As above	*Medium*	*Medium*	*Medium*

2.4.5 *Evaluation logical asset: CA server software*

No.	Threat	Countermeasure(s)	L	I	Risk
14	System Administrator: Install bad software (e.g. sniffer), do not correctly update/configure system	Use skilled employees for the task, review system by second party	*Low*	*High*	*Low*
15	Script kiddies: DDoS	Limit incoming connections from same IP in firewall	*Medium*	*Medium*	*Medium*
16	Skilled hacker / Organized Crime: Get system access	Stop all unused services, close all unnecessary ports	*Low*	*High*	*Low*
17	Malware: Use server for sending spam or distribute itself on webpages	Same as above	*High*	*Medium*	*Medium*

2.4.6 *Evaluation logical asset: CAServer app. / CACore scripts*

No.	Threat	Countermeasure(s)	L	I	Risk
18	System Administrator: Create certificate for some user	Log all certificate creation procedures	*Low*	*High*	*Low*
19	Script kiddies / Skilled hacker / Organized Crime: XSS	Validate and sanitize all input	*Low*	*High*	*Low*
20	Script kiddies / Skilled hacker / Organized Crime: Eavesdrop on communication	Only use HTTPS for communication	*Low*	*High*	*Low*

2.4.7 *Evaluation logical asset: CA server database*

No.	Threat	Countermeasure(s)	L	I	Risk
21	Script kiddies / Skilled hacker / Organized Crime: SQL injection	Sanitize all inputs	*Medium*	*High*	*Medium*

2.4.8 *Evaluation logical asset: Backup server software*

No.	Threat	Countermeasure(s)	L	I	Risk
22	System administrator: Turn off backup, misconfigure backup (encryption)	Monitor backup service	*Low*	*Medium*	*Low*
23	Skilled hacker: Get access to system	Restrict access, turn off unused services, log activities	*Low*	*High*	*Low*

2.4.9 *Evaluation information asset: User data*

No.	Threat	Countermeasure(s)	L	I	Risk
24	User: Lose their username and password	Allow them to login using a certificate	*Low*	*Low*	*Low*
25	System Administrator: Intentionally or accidentally modify user data	Don't allow data access to administrators	*Low*	*Medium*	*Low*
26	Script kiddies / Skilled hacker: Steal data	Always use encrypted communication, store data encrypted on backup, restrict access on user data	*Medium*	*Medium*	*Medium*

2.4.10 *Evaluation information asset: Certificates*

No.	Threat	Countermeasure(s)	L	I	Risk
27	User: Lose the certificate	Ability to revoke certificates	*Medium*	*Low*	*Low*
28	System Administrator: Modify data linked to certificate	Restrict data access	*Low*	*Medium*	*Low*
29	Skilled hacker: Issue bogus certificate	Don't allow user registration, log certificate creations	*Low*	*High*	*Low*

2.4.11 *Evaluation information asset: Private keys*

No.	Threat	Countermeasure(s)	L	I	Risk
30	System Administrator: Leak to external party	Only root is allowed to access private keys	*Low*	*High*	*Low*
31	Script kiddies / Skilled hacker: Steal private keys	Private keys are only accessible for root users, keys are encrypted in transfer	*Low*	*High*	*Low*

2.4.12 *Evaluation information asset: CRL*

No.	Threat	Countermeasure(s)	L	I	Risk
32	System Administrator / script kiddies / skilled hacker: Insert fake or remove real entries	Restrict write access to crl file to root	*Low*	*High*	*Low*

2.4.13 *Evaluation information asset: Server configuration*

No.	Threat	Countermeasure(s)	L	I	Risk
33	System Administrator: Leak configuration	Place configuration in standard place (secured by access policies)	*Low*	*Medium*	*Low*
34	Script kiddies / Skilled hacker: Alter configuration (e.g. weaken preferred security algorithms)	As above, additionally backup config incrementally (spot alterations)	*Low*	*High*	*Low*
35	Malware: Delete or alter configuration randomly	Backup configuration (incremental), access logs, restrictive access policies	*Medium*	*High*	*Medium*
36	Competitors / Espionage: Access configuration and use for own system	Hide internal configurations of the system from the outside	*Low*	*Medium*	*Low*

2.4.14 *Evaluation information asset: Logs*

No.	Threat	Countermeasure(s)	L	I	Risk
37	System Administrator: Accidentally or intentionally delete logs	Policy to not delete logs before they are backed up	*Low*	*Medium*	*Low*
38	Script kiddies / Skilled hacker: Insert or delete messages from the logs	Restrict access to logs to application and root	*Medium*	*Medium*	*Medium*
39	Malware: Insert random logs	Restrict access to logs to application and root	*Low*	*Medium*	*Low*

2.4.15 *Evaluation information asset: Login credentials*

No.	Threat	Countermeasure(s)	L	I	Risk
40	System Administrator: Forget login credentials	Backup offline, allow login with ssh key	*Low*	*Low*	*Low*
41	Script kiddies / Skilled hacker: Brute force password guessing	Restrict amount of connections from same IP, enforce strong passwords	*Low*	*High*	*Low*

2.4.16 *Evaluation information asset: JWT*

No.	Threat	Countermeasure(s)	L	I	Risk
42	User: Lose JWT	Saved in browser session	*Low*	*Low*	*Low*
43	Script kiddies: Steal JWT from a not closed browser window	Short lifetime of token	*Low*	*High*	*Low*
44	Skilled hacker: Steal JWT (e.g. by malicious browser plugin)	Only store JWT in local session, short lifetime of token, enforce PW/certificate login afterwards	*Low*	*High*	*Low*

2.4.17 *Evaluation information asset: Archive key*

No.	Threat	Countermeasure(s)	L	I	Risk
45	System Administrator: Lose key (stored offline)	Store at different locations (e.g. in several safes)	*Low*	*High*	*Low*

2.4.18 *Evaluation information asset: Root key*

No.	Threat	Countermeasure(s)	L	I	Risk
46	System Administrator: Lose key (stored offline)	Store at different locations (e.g. in several safes)	*Low*	*High*	*Low*

2.4.19 *Evaluation person asset: User/Employee*

No.	Threat	Countermeasure(s)	L	I	Risk
47	Competitor / Skilled hacker: Steal private key to be able to read email communication	Instruct users how to safely store confidential information, regularly renew certificate, use passphrases or keys to encrypt private key on harddisk	*Medium*	*Medium*	*Medium*
48	Competitor / organized crime: Blackmail or bribe users to hand over their private key	Have users sign non-disclosure agreements, prosecute them legally	*Medium*	*Low*	*Low*

2.4.20 *Evaluation person asset: CA administrator / Insider*

No.	Threat	Countermeasure(s)	L	I	Risk
49	Competitor: Leak system implementation	Use standard implementations if possible	*Low*	*Low*	*Low*
50	Skilled hacker: Steal private key to have access to CA information	Encrypt private key on harddisk, sign agreement to follow the company guidelines to keep key secret	*Low*	*Low*	*Low*
51	Competitor / organized crime: Blackmail or bribe CA administrators to hand over their private key	Have CA administrators sign non-disclosure agreements, prosecute them legally	*Medium*	*Low*	*Low*

2.4.21 *Evaluation person asset: System administrator*

No.	Threat	Countermeasure(s)	L	I	Risk
52	Skilled hacker: Steal private key to have full access to the system	Encrypt private key on harddisk, unmodifiable log files to detect intrusion into & modification of the system	*Low*	*High*	*Low*
53	Competitor / organized crime: Blackmail or bribe system administrators to hand over their private key	Have system administrators sign non-disclosure agreements, prosecute them legally	*Medium*	*High*	*Medium*

2.4.22 *Evaluation person asset: Private key holder*

No.	Threat	Countermeasure(s)	L	I	Risk
54	Skilled hacker: Steal private key to be able to imitate the CA and sign any certificate	Only store the private key encrypted on an isolated machine. Once the key is stolen, there is no possible countermeasure since most security guarantees are based on the secrecy of this key	*Low*	*High*	*Low*
55	Competitor / organized crime: Blackmail or bribe the private key holder to hand over their private key	Have the private key holder sign non-disclosure agreements, prosecute them legally	*Medium*	*High*	*Medium*

2.4.23 *Evaluation intangible asset: User confidence*

No.	Threat	Countermeasure(s)	L	I	Risk
56	Competitor: Hires skilled hacker to breach the security of the system to leak customer data, which reduces the confidence of the user in the system	Insure against the risk of leaked customer data and the resulting damage	*Low*	*Low*	*Low*

2.4.24 Detailed Description of Selected Countermeasures

System design The system is designed to be as simple as possible while still retaining the necessary functionality, which means we used as few different applications as possible, because every application adds a potential vulnerability to the system. Any unnecessary port is closed and every application is executed with the smallest sufficient set of permissions.

Instruct users The biggest security concern is the behaviour of the users of the system. We have to train users to be sensitive to security concerns. An attacker can try to gain information about the system or access to the system through social engineering, extract keys from private unprotected machines, install malware on these machines that will infect the system or extract unencrypted mail correspondence from them.

Logging All relevant log files are periodically incrementally backed up. They should only be readable (and not writable) by the system administrator. If this is the case, an attack can be detected even if the attacker attains the same access rights to the system as the administrator.

Standard configurations Whenever possible we use standard configurations for our services. The first reason is that one is less likely to misconfigure a service (because of misunderstanding the exact functionality) by using the standard configuration. The second reason is that even if one perfectly understands the service, it is still possible to e.g. have a typo in a config file and introduce a vulnerability.

Redundancy Whenever there is a hardware component that is critical for the functionality and security of the system there has to be a replacement available at all time or the component has to be redundant itself. Since every configuration file and all applications are backed up, a replacement machine can quickly be installed.

2.4.25 Risk Acceptance

We list all risks that are classified as **Medium** or **High** and describe further countermeasures that could be implemented against them.

No. of threat	Proposed additional countermeasure including expected impact
1	Replicated hardware to ensure redundancy in any case or sign a service-level-agreement with a third party to provide a backup system, which can quickly be deployed.
7	Have the system administrators regularly review each others changes to the system.
12	Insure against the risk of a security breach and the resulting damage.
13	Same as above.
15	Install additional hardware and use a load balancer to be able to handle a bigger amount of requests.
17	Pay an external entity (Security operations center) to monitor the system for patterns that would indicate a malware infection. To protect against malware, it is important to install security updates as soon as possible.
21	Log all SQL requests and do a pattern matching on these logs that detects unusual requests (e.g. `or ""=""`).
26	Insure against the risk of leaked user data and the damage that might arise for the company & the user.
35	The same countermeasures as for risk 17 apply here. A modification of the server configuration further impacts the security of the system by possibly introducing new vulnerabilities.
38	An attack that modifies the logs is especially problematic because it becomes almost impossible to detect the attack. An undetected attack is much more problematic, because the vulnerability causing it cannot be removed.
47	Allow users to only use company certified machines for work which are generally better secured than personal machines.
53	Insure against the risk of a system administrator losing his private key and the resulting damage to the company.
55	Same as above. Additionally the company needs to instruct all users to stop using the (most likely compromised) system immediately, delete the now invalid certificates and create new certificates.

References

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