Olli Kortelainen

Automating Certificate Enrollment

For Helsinki and Uusimaa Hospital District

Metropolia University of Applied Sciences

Bachelor of Engineering

Information Technology

Thesis

19 August 2018

Helsinki Metropolia University of Applied Sciences

PREFACE

Thesis project is done, all courses completed at Metropolia, there is still some summer left, and my summer vacation will start in a few weeks. Life is good! This project was fun and challenging to make. I do not really have any real experience in front-/back-programming which forced me to learn 3 new languages (HTML,JS and CSS), but I learned enough to finish the project, and I'm happy with the end result.

This thesis has gained some interest from a few companies that are struggling with the same problem as we do at HUS, so I hope that my findings and the product made during the project will at least give an idea on how to solve their problems.

I would like to, thank Timo Sainio from 2M-IT and Jari Pirinen from the Finnish Population Register Centre (VRK) for answering my queries, which gave me some ideas on how to design the product described in the thesis. I would also like to thank Helsinki and Uusimaa Hospital District (HUS) for allowing me to do my thesis project for them and use work hours to do so. Finally I would like to thank my fiancée for the support I got from home to finish school and thesis.

Helsinki, 19.8.2018
Olli Kortelainen

Author(s) Title Number of Pages Date	Olli Kortelainen Automating Certificate Enrolment For Helsinki and Uusimaa Hospital District 29 pages + 2 appendices 11 August 2018
Degree	Bachelor of Engineering
Degree Programme	Information Technology
Specialisation option	Software Engineering
Instructor(s)	Kimmo Sauren, Senior Lecturer, Project Supervisor

The objective of the final year project was to automate certificate enrolment and guiding HUS users with creating the certificate. To reach this objective, different programming languages for the back-end technology were explored, and modern JavaScript libraries were used in the front-end. Also past gained experience gained by the author of the thesis when working for HUS played a major role in inventing, designing and implementing the project.

It was important to solve this problem now, as increasing demands of certificates in HUS environment require more man-power to handle the certification request. By automating this process as much as possible and still following HUS security requirements, frees time of the certificate administrators to do tasks that are more demanding and speeds up the process from which customers ultimately gain a SSL-certificate.

The final outcome of the project was a web-based system that has multiple guides and FAQ web-pages that help user in creating a certification request. The webpage has a submit form that goes through the certification request and informs the customer if the request is not according to HUS standards. Once the request is correct, the form is forwarded to HUS IT administrators that create the certificate for the customer.

The project successfully reached its primary objective, but further improvements and updates to the website and back-end technology is required in order to fulfil the changing needs in HUS environment. This can be achieved by following the feedback provided by customers, but also following the new standards that are created for worldwide use.

Keywords	Certificate, certification request, SSL, PKI, Automation

Författare Rubrik Sidor Datum	Olli Kortelainen Automating Certificate Enrolment For Helsinki and Uusimaa Hospital District 29 sidor + 2 bilagor 11 Augusti 2018
Examen	Bachelor of Engineering
Examensarbete	Information Technology
Specialiseringsalternativ	Software Engineering
Instruktör	Kimmo Sauren, Universitetslektor, Projektledare

Målett för sista årets projekt, var att automatisera certifikat registrering och väglesa HUS personal med att skapa certifikat. För att nå detta mål undersöktes olika programmeringsspråk för back-end-tekniken, och moderna JavaScript-bibliotek användes i fronten. Tidigare erfarenhet av förättarens avhandling då han arbetade för HUS spelade en viktig roll för att uppfinna, designa och genomförå projektet.

Det var viktigt att lösa detta problem nu, eftersom ökande krav på certifikat i HUS, kräver mer manstyrka att hantera certifieringsbegäran. Genom att automatisera denna process så mycket som möjligt, och fortfarande följa HUS-säkerhetskrav, frigör tid av certifikatadministratörena, till att görå mer krävande uppgifter och påskyndar processen var från kunderna slutligen får ett SSL-certifikat.

Det slutliga resultatet av projektet var ett web-baserat system, som har flera guider och en websida med de vanligaste frågorna, som hjälpper användaren att skapa ett certifikatsbegäran. Websidan har en form som går igenom certifikatsbegäran automatiskt och informerar kunden om begäran inte är enligt HUS-standarder. När begäran är korrekt, skickas formuläret till HUS IT-administratorn som skapar certifikatet för kunden.

Projektet lyckades nå sitt främsta mål, men ytterligare förbättringar och uppdateringar till websidan och back-end-tekniken krävs för att uppfylla de förändrade behoven i HUS-miljön. Detta kan uppnås genom att följa återkoppling från kunderna, men också efter de nya standarder som skapas för världsomspännande användning.

Keywords	Certifikat, Certifikatförfrågan, SSL, PKI, Automation

Table of Contents

References

Appendices

Appendix 1. PKI.JS modified by author of the thesis

Appendix 2. scripts.js – Client side validation. Written by author

List of Abbreviations

AD	Active Directory
AD CS	Active Directory Certificate Service
BYOD	Bring Your Own Device
CA	Certificate Authority
CRL	Certificate Revocation List
CSR	Certification Request
CSS	Cascading Style Sheets
CTL	Certificate Trust List
DES	Data Encryption Standard
ESM	Enterprise Service Management
EU	European Union
FTP	File Transfer Protocol
FTPS	FTP Secure
GDPR	General Data Protection Regulation
GPO	Group Policy Object
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	HTTP Secure
HUS	Hospital District of Helsinki and Uusimaa
IIS	Internet Information Services
IM	Information Management
IT	Information Technology
ITS	Issue Tracking System
NPM	Numeric Production Mechanism
NSA	National Security Agency
PKI	Private Key Infrastructure
PO	Product Owner
RSA	Rivest, Shamir, Adleman
SATSHP	Satakunnan sairaanhoitopiiri (Hospital District of Satakunta)
SFTP	SSH File Transfer Protocol
SM	Scrum Master
SSH	Secure Shell
SSL	Secure Socket Layer
TLS	Transport Layer Security
VRK	Väestörekisterikeskus (Population Register Centre)

VSHP	Vaasan sairaanhoitopiiri (Hospital District of Vaasa)
VSSHP	Varsinais-Suomen sairaanhoitopiiri (Hospital District of Varsinais-Suomi)

1 Introduction

This thesis explores the topic of automating certificate enrollment to encounter the increasing need of securing network traffic. Transport Layer Security (TLS) and Secure Socket Layer (SSL) certificates are being used daily in many situations like banking transactions and authenticating users or devices. Credit-cards, software and most websites use certificates to secure network-traffic between a client computer and server providing the service. By using certificates, the network-traffic is encrypted and is therefore almost impossible to be misused by a malicious hacker. For a service to keep being secure, the certificates have an expiration date and need to be renewed.

The case organization Helsinki and Uusimaa Hospital District (HUS) has recognized a need for a website that provides customers with guides on how to create a certification request (CSR) file, and gives immediate feedback regarding the validity of the file. The objective of the project is therefore, to design and implement a web based system that is used to automate the enrollment process of a certificate. The website will provide customers with guidance on how to create a CSR file with different attributes, how the file is created on different operating systems, and automatically informs the user if the CSR is missing information or contains invalid information. Once the CSR has been done correctly, the system needs to automatically forward the request to the Certificate Authority (CA) for enrollment. The system also needs to store information on who has requested a certificate and when the certificate will expire and it needs to send an alert email to the customer informing about the upcoming expiry so that the process can be restarted.

HUS Information Management (IM) provides Information Technology services for healthcare, universities and citizens in the form of, but not limited to, servers, websites, datacenters, artificial intelligence and different devices. According to HUS information security requirements, software and services used in the HUS Intranet must be end-to-end secured. Securing is done by using certificates that can be either bought from commercial vendors, which can be costly, or as in the case of HUS, manage an own Private Key Infrastructure (PKI). As explained by Margaret Rouse and contributors [1], PKI is composed of hardware, software, guidelines and principles used when creating, administering, distributing and voiding keys and digital certificates. Certificates are used to prove the identity of a certificate user and tie that identity to the certificates public key, and are thus in the center of PKI.

This thesis intends to find a solution to the business challenge, proposed by the author also working for HUS-IM. Much time goes into answering the following questions made by the customers: “What is a certificate?”, “How does one acquire a certificate?” and “What is wrong with the CSR?”. This research strives to specify the requirements needed to develop a user-friendly website that automates the enrollment of SSL/TLS Certificates, allowing the IT-personnel to focus on more demanding tasks. This thesis does not try to study how certificates are installed onto software and devices as that information should be provided by the individual software/service provider, nor does it study how to deploy a PKI environment.

The thesis is divided into eight sections. The first section is the introduction, that covers the basic use of digital certificates, case company background, business challenge, and scope of the project. The second section describes the history of cryptography, and further explains the current and future needs for automating the digital certificate delivery process in HUS environment. The third section describes the history of the Hospital district of Helsinki and Uusimaa. The fourth section explores the tools and solutions used in other hospital districts to counter the problem regarding digital certificate delivery. The fourth section explains the requirements of the project, handling security, server- and client-side specifications. The sixth section covers the implementation and testing of the project. The seventh section presents the result of the project. Finally, the eight section concludes the project and discusses on how to further improve the system.

2 Research Backgrounds

This section covers the basics of cryptography and its history, from the beginning of hiding messages, to the present of securing network-traffic in healthcare. An overview of the history of cryptography is important to explain to understand how cryptography has evolved to the stage where it is today and where it is being developed.

2.1 History of Cryptography

Research into Cryptography has a long history. According to Fred Cohen (1990) [2], the history of cryptography reaches back 4000 years when Egyptians decorated the tombs of the rulers using hieroglyphs. Hieroglyphs told the life story of the rules and kings, and were intentionally cryptic, but not so that the text was supposed to be hidden. Cohen

explains that the hieroglyphs were intended to make the text seem more rich and important. As time went by, the hieroglyphs became harder to understand, and eventually people stopped deciphering them.

According to Cohen, in the Bible, a Hebrew ciphering method was sometimes used where the last letter of the alphabet was replaced by the first, and vice versa, the method is called “atbash”. An example of the cipher would be the English word “HELLO” becoming the word “SVOOL”.

Cohen also explains how the Spartans used thin sheets of papyrus wrapped around a rod, as shown in figure 1. Messages were written down the length of the rod and the papyrus was unwrapped. To read the message a person had to have a rod of equal diameter, wrap the papyrus around the staff and read the text. This cipher was called “Scytale” and the technique was used in 5th century B.C.

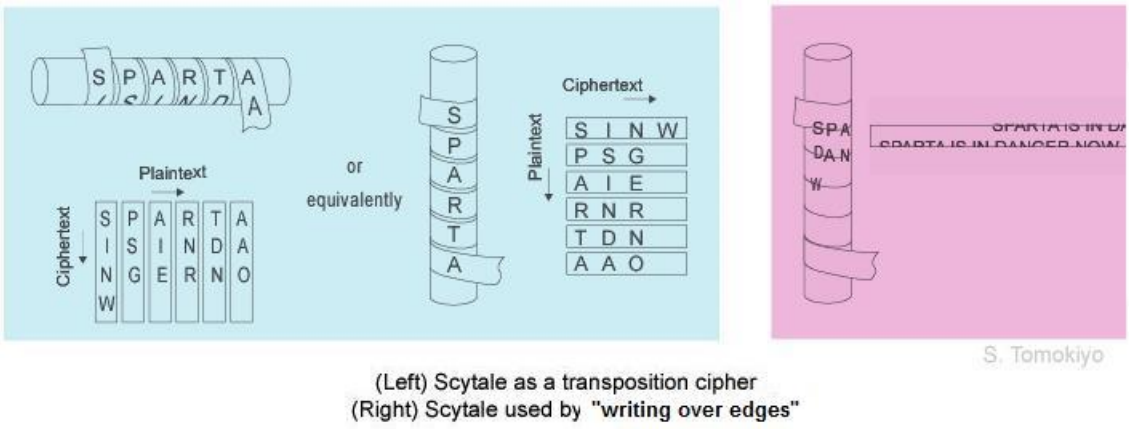

Figure 1. Picture of Scytale with two variations. [3]

Another Greek method according to Cohen, was developed by Polybius and named the “Polybius Square”, displayed in figure 2. The idea was to lay the alphabet in a 5 by 5 square and have I and J occupied the same square. Rows and columns were numbered from 1 to 5 so that each letter had a corresponding pair. The pairs could easily be signaled using torches or hand signals and decrypting the code consisted on mapping the

digit back to their corresponding characters. Cohen says that, this system might be considered the forerunner of modern binary representation of characters.

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	I/J	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

Figure 2. Picture of Polybius square [4]

According to Fred Cohen, modern encryption standards used today came to existence in the mid 1970's when the company IBM offered the American National Security Agency (NSA) an algorithm that could be tested and certified. The encryption is called Data Encryption Standard (DES). DES is a symmetric block cypher that is based on a 64 bit block of plain text that is run through an algorithm that returns 64 bit block of cyphertext. To encrypt and decrypt the text, the same algorithm and key is used. According to Dan Froomkin and Amy Branson [5], when using a *symmetric-key* encryption, both the sender and receiver uses the same key that must somehow be transported to both parties causing a problem where someone might capture the key, and gaining the ability to decrypt messages.

Cohen continues to explain that to counter the key exchange problem, in the year 1975, Diffie Whitfield, Martin Hellman and Ralph Merkle developed the concept of the *asymmetric cryptography (a.k.a Public Key cryptography)*. The concept never made it in to development in the hands of Whitfield, Hellman and Merkle, but got picked up by Ron Rivest, Adi Shamir and Leonard Adleman (RSA). When encrypting using an asymmetric cypher, a *private-key* is created, from which a *public-key* is derived. Cohen tells that, the keys are calculated from large 300-400 digit prime numbers, and are thus believed to be beyond the capability of even modern computers to crack the encrypted message without the required keys. Securing a message works as follows. The public-key is given by the

owner of the private-key to another person, and that person then uses the key to encrypt a message. Only the owner of the private-key can decrypt the message, thus solving the problem of unsecure key exchange. By using asymmetric encryption, authentication of user can also be made. If a message is signed using the private-key, everyone possessing the public-key can verify that the message has been sent by private key owner. Today this method is called *Digital signature*, and is used almost everywhere to authenticate the owners of credit-cards, identity-cards, web-sites, devices and much more.

Even though modern computers are not capable of braking the RSA encryption, the fear of future computers processing capabilities is something that should not be ignored. According to Tom Simonites [6], the NSA has expressed their concern that quantum computers could break the modern encryption methods. Work on new algorithms that can be used in the post-quantum era is still in the beginning, thus meaning that no solution has yet been discovered on how to secure messages between devices and people alike.

2.2 Problem with enrolling certificates

Helsinki and Uusimaa hospital district is Finland's largest hospital district. This also means that the IT infrastructure is one of the biggest, and maintaining its functionality is a challenge. One of the challenges is making sure that information transferred between customers and devices are managed in a secure way.
HUS has tens of thousands different devices in the internal network, including hand held medical scanners, mobile devices, computers and servers.

As previously mentioned on way of securing network traffic is done by using digital certificates to encrypt the data between devices. The challenge when enrolling certificates in an organization such as HUS are old devices. Some of the old devices cannot be updated and can only use certificates with less secure encryption. This is also true for many software in the environment, as updating software is either very risky for the production or the software companies simply do not support older devices anymore. This challenge with legacy systems and devices is also confirmed by Stuart Sumner in his column: "Research: Legacy systems the biggest challenge in digital transformation" [7] where Sumner explains the result of *Computing's* company's research where 100 senior IT leaders where asked about the digital transformation initiatives.

Another great challenge is the increasing number of devices in the network. As explained by Jennifer Lonoff Schiff in her article: “The 4 biggest healthcare IT headaches” [8], Bring Your Own Device (BYOD) is a growing trend in Healthcare as it enables personnel to use the devices they are most comfortable using. There is also the benefit of substantial computing power, portability and modern user interfaces. The problem is that there are a lot of mobile devices from various vendors (i.e. Apple, Samsung, Nokia) that have mobile devices that are of different operating systems (i.e. iOS, Android) with different versions. Mary Shacklett writes that a centralized network management software should be used to automatically download updates to devices when users join the organizations network. [9]

In the HUS environment the increase in devices is not limited to just BYOD, as new computers and servers are also added to the network daily. Laptops and desktop computers certificate enrollment is not an issue since these devices get their certificates automatically from the active-directory using Group Policy Objects (GPO) on the computers, and the computers do not usually provide services visible in the network. The servers are however another matter. As servers are used to provide various services, and use different software to do so, a fully automated system that installs the certificates in the software is not possible and thus the certificates have to be manually created and installed.

When something must be done manually, people are involved and, when people are involved that do not understand how a digital certificate works, a lot of explanation must be done on what has to be done in order to get a working certificate for the server.
An inquiry was done and sent to the Population Register Centre and other hospital districts operating in Finland to which three representative responded. The inquiry will be gone through in more detail in section four of this thesis report. Timo Sainio from the company 2M-IT that manages certificate enrolment for Varsinais-Suomen Hospital District (VSSHP), Satakunnan Hospital District (SATSHP), Vaasan Hospital District (VSHP) and Pori Basic Security, responded that is take approximately 30 minutes to 10 hours to explain to the customer how to create a certification request depending on the complexity of the software, when he was asked how much time goes to help the customer in creating a certification request.

By multiplying the time that goes to explain the creating of certificate request by number of servers in the network, we can easily see that a lot of time goes wasted on a usually simple task that is difficult to explain.

3 Brief History of Helsinki and Uusimaa Hospital District

This section explains shortly the history of Helsinki and Uusimaa to give the reader a mental picture of the size and complexity of the organization.

3.1 Helsinki and Uusimaa Hostpital District

The HUS Joint Undertaking, set up by the Uusimaa Municipalities as shown in figure 3, started operations on 1 January 2000. The hospital district, which was established at that time, was built on solid expertise, which includes, on the one hand, the legacy of the university hospital since the 1830s and the care of patients in specialized 19th-century hospitals and rural diversified hospitals.

The largest hospital district in Finland was born when the Uusimaa Hospital District and Helsinki Hospital District, which was established based on the Special Health Care Act of 1991, and the Helsinki University Central Hospital, were established in Uusimaa.

The hospitals of the former Uusimaa Hospital District were transferred to the new organization as such. The Hospital District's central administration was merged with the management of the new group.

As a legacy of history, HUS has a total of 21 hospitals, as well as two hospitals in Helsinki and many smaller units serving local residents. The oldest hospital buildings, have been treating the sick continuously from the end of the 18th century. Most of the hospital network was built in the 1960s, and the latest in recent years. There is both ongoing renovation work and new hospitals under construction.

Figure 3. Map of municipals covered by Helsinki and Uusimaa Hospital District. [10]

In the course of HUS's operations, the services supporting healthcare have been organized into municipal enterprises. At the same time, the organization of support services has been centralized and the scope of each commercial enterprise has expanded in scope to cover the entire HUS. Some HUS enterprises also sell their services to member municipalities.

In 2004, HUS-Röntgen (since 2012 HUS-Kuvantaminen), the HUSLAB business organization and Ravioli, which produces nutrition services, started as business enterprises. In 2008, the subscriber-producer model was further sharpened, when HUS-Logistikka responsible for logistics and ambulance services, HUS-Apteekki responsible for pharmacy and HUS-Desiko, which produces services for facilities and equipment, were organized into enterprises.

From the beginning of 2009, HUS-Servis, which produces the company’s internal documentation, business, personnel and financial services, HUS-Lääkintatekniikka which produces medical engineering, and HUS-Tietotekniikka (later HUS-Tietohallinto) which produces information technology services started as enterprises. [11]

HUS-Tietohallinto is a division of HUS and it provides information and IT services to HUS, to the joint venture partners and subsidiaries and associates. Other customers and

stakeholders in HUS's IT administration are municipalities, universities, private healthcare providers and public authorities.

HUS-Tietohallinto employs approximately 350 service experts with solid IT skills and the ability to look for solutions to meet customer needs. Strengths also include knowing the core business of our customers and utilizing a broad partnership and partner network.

The aim is to support the core business of customers by providing expertise, use, maintenance and support services for IT including: expert and project manager services, the development of modern online services and operating environments, basic information technology services, information system and datacentre services, IT training and support services

Key information systems include patient information systems, financial and human resource management systems, and support service unit systems.

Basic IT information management services such as telecommunication network, communication technology (voice solutions), workstations and servers enable the use of information systems, office applications, e-mail, the Internet and other information technology services.

In the organization of training and support services for information technology, agreements have been signed with long-time partners in the field [12]. HUS turnover is approximately 128,4 million euros [13].

4 Existing Solutions

This section covers how the challenge with enrolling digital certificates has been solved in other Finnish hospital districts and what available tools there are to combat the issue at hand. An overview of the existing solutions and tools are important for the continuation of this thesis, as this information defines if the challenge of manually enrolling digital certificates at Helsinki and Uusimaa Hospital District should be tackled with available tools, self-made tools or a combination of both.

4.1 Solutions in other hospital districts

To find out how the challenge of enrolling certificates has been solved in other Finnish hospital districts, an inquiry was sent via HUS IT Security Manager. The short inquiry was produced by the author of this thesis, in the hope that at least some representatives of the hospital districts would answer. As this thesis is publicly available for anyone to read, and the Private Key Infrastructure of an organization is one of the highest protected services, organizations do not care to reveal information about PKI in case of misuse by a malicious entity. In the end, only the previously mentioned Timo Sainio from the company 2M-IT, Jari Pirinen from VRK responded to the inquiry.

The inquiry consisted of eight questions: How does the organization handle enrolling of digital certificates (automatically / manually / delegated to a third-party)? Does the organization use one or more tools for enrolling digital certificates? The previously mentioned: How much time does it take to explain to customers how to create a certification request or does the organizations Certificate Authority do the whole enrolment proses from start to finish? How does the organization control the expiration of digital certificates? Does the organization use one or more tools to control the expiration of digital certificate? What processes doe the organization have when a digital certificate is about to expire? What operating systems are used on the computers handling the control of certificate expiry? And finally, the question: How important does the organization see automatisation of the digital certificate enrolment, control and informing customer of expiring digital certificates?

When asked the question: How does the organization handle enrolling of digital certificates (automatically / manually / delegated to a third-party)? Timo Sainio replied, that the enrolling of digital certificates has not been out-sourced to a third-party, but instead delegated to a set group of professionals, handling the procurement is mainly done manually, but the task has been eased with custom scripts. Method of delivering the CSR to the CA is done by using graphical tools. Pirinen said that VRK issues the certificates, but in some cases a third-party produces the certificates on VRK behalf. Enrolment of some VRK certificate types has been automated and others are created manually.

To the second question: Does the organization use one or more tools for enrolling digital certificates? Sainio relied that all standard digital certificates are generated with custom tools with OpenSSL doing most tasks. Sainio also said that some CSR’s are generated

on the servers, with software specific tools and that internal certificates are also enrolled using group policy objects, as enrolling and renewing digital certificates can easily be done using tools provided by Microsoft. VRK also has both custom solutions for generating certificates, but ready-made software are mainly used.

When asked the previously mentioned question: How much time does it take to explain to customers how to create a certification request or does the organizations Certificate Authority do the whole enrolment proses from start to finish? Sainio explained that it takes 30 min to 10 hours depending on the complexity of the software but by using a custom form the objective is to try to plan beforehand with project- and certificate specialists what the certificate request need to contain so that the enrolment would go through the first time, and therefore save time. Pirinen explains that it depends on the certificate. Guides of acquiring a Personal Certificate is provided by the license administration that is managed by the Finnish Police. For the organization cards (i.e Healthcare personnel identification cards) guides can be found in VRK intranet and card registration software. VRK web portal offers guides for service and server certificates. A majority of the certificate requests is done by self-service and VRK trains the staff of registration centres to do applications for ID cards.

When the group was asked the question: How does the organization control the expiration of digital certificates and does the organization use one or more tools to control the expiration of digital certificate? Sainio replied that monitoring the expiry of certificates is done with a public vendor tool for certificates provided by the public vendor, but the internal certificates are monitored using a custom script. Sainio says that the challenge of completely using the public vendor tool to monitor both private and public digital certificates is that to have the certificates included in the tool, certificates have to be imported manually, and this will easily be forgotten. Sainio says that by using their own custom script the alarms of expiring certificates can be easily produced as the process generates a copy of the certificate to a centralized place. Sainio tells that in the future, the alarming of expiring certificates will be moved to configuration management so that the alarms will in the first instance go to the owner of the certificate rather than the instance providing the digital certificate. VRK has a more strict and clear policy. According to Jari Pirinen, VRK only manages the Certification Authorities life cycle. Personal and Service Certificates life cycle is monitored by the customer. Pirinen explains that the certification expiry date can be checked by different means depending of the device. ID-cards have the

expiry date printed on the card and service certificates expiry date can found on the certificate itself, but a better way to monitor the expiry is needed.

To the question: What processes does the organization have when a digital certificate is about to expire? Sainio said that, when an alert of the expiring certificate pops up, the certificate specialist searches for notes about the certificate when it was created, and hopes that the notes contain the information about who ordered the certificate and where it was installed and after that the certificate owner is asked if the certificate is still needed. In the future the process is supposed to change so that the alarm goes to the owner of the certificate who then starts the renewal process, and the certificate specialist only receives an order in the ticket system. As previously mentioned by Pirinen, monitoring the expiry on VRK issued certificates belongs to the customer. Once the certificate has expired, it will be automatically removed from the system. The CA certificates expiry is monitored internally by VRK, and will be renewed well before the expiry date.

What operating systems are used on the computers handling the control of certificate expiry, was asked because it is important to know what technology is used at the very core of the organizations PKI as this is essential on how the thesis will use the information provided by Sanio. Sainio replied that the operating system used is Windows. VRK did not want to answer this question, because of security reasons.

To the final question: How important does the organization see automatization of the digital certificate enrolment, control and informing customer of expiring digital certificates? Sainio replied that, first of all the manual processes have to work as well as possible. It is essential for the owner of the business application to understand that the digital certificates are components of their applications, just like the servers, software and integration. This way it becomes clearer on what certificates need to be renewed and which do not. Also ordering of certificates would be good if handled by real processes instead of confusing emails and phone calls, Sainio says. In the further future when the number of digital certificates eventually rises, a better way of automating definitely needed. Once the manual process and responsibility questions are clear, certain processes can be automated without bigger drama for example by using Acme-protocol, Sainio explains. Sainio also makes clear that a fully automated certificate enrolment system is idiotic, as someone has to be making sure that a certificate is still needed so that the system does not order certificates when there is no need to do so. A system that automatically monitors and alerts of expiring certificates is most definitely important now, in the past and in

the future, Sainio says. This need is also recognized by VRK, Pirinen explains that there is a need for automation, and that VRK already has this under planning. Pirinen says that in the future it would be good to automate informing citizens of expiring ID cards as a part of a larger authoritative entity.

4.2 Project tools

By looking at the answers got from Timo Sainio and Jari Pirinen, it is clear that a fully automated digital certificate enrolment system is not wise or even wanted, and that there is no one specific tool to use on all devices that would make it easier for application owners to enrol for a certificate. It is therefore best to look at the tools available at the present time and try to use them in combination with each other. The areas and tools of certificate management this section covers is generating a certification request, generating a certificate, monitoring certificate expiry and generating an alert of expiring certificates and website technologies. Also tools used in the infrastructure is important to introduce as they are relevant to create the project specifically at HUS.

The first tool introduced is the Active Directory (AD) that works alongside the Private Key Infrastructure. According to Steve Clines and Marcia Loughry [14], AD was introduced in Microsoft's Windows Server 2000, and is one of the most popular directory service products in the world. Clines and Loughry describe AD as an umbrella that today includes many different technologies. AD is a store of information that organizes data into objects where each object has different attributes. This object could for example be a user account including information about a user or a group object that holds multiple users and devices.

One technology in the AD is the Active Directory Certificate Service (ADCS). Steve Clines and Marcia Loughry explain that ADCS is a software used to deploy Certification Authorities that can issue, revoke or renew certificates. One could therefore say that a Private Key Infrastructure is the ADCS, although the PKI used in an organization does not have to be a Microsoft product. The product used in the thesis project is ADCS.

The second tool used is Microsoft Internet Information Services (IIS). According to Margaret Rouse [15], IIS is a general-purpose web-server running on Windows operating systems that show HTML pages or files to the remote client computer visiting the web-server. IIS can be used with different standard languages and protocols, of which the

most used are the Hypertext Transfer Protocol (HTTP), the SSL protocol HTTP Secure (HTTPS), File Transfer Protocol (FTP) and its SSL version FTP Secure (FTPS). FTPS should not be confused with the SSH File Transfer Protocol (SFTP), as that protocol uses SSH connection to transfer files securely [16]. In the thesis project HTTPS will be used to encrypt the traffic between customers computer and the webserver.

The third and fourth tool are both for generating a certificate request, although they are not limited to that. These tools, will also only be used by the customers, but a guide to use these tools will be added on the project website, and it is therefore mentioned in the thesis report. The third tool is Microsoft Management Console snap-in Certmgr.msc. It is it its simplicity a tool to manage user or computer certificates, certificate trust lists (CTL) and certificate revocation lists (CRL) [17].According to Wikipedia, where as certmgr.msc is only available for windows, the fourth tool, OpenSSL is available for most Unix and Unix-like operating systems and Windows OS [18]. According to OpenSSL webpage [19], OpenSSL is a toolkit for the TLS and SSL protocols, and is an all-around cryptography library. OpenSSL open sourced, maintained by a team of committers and the OpenSSL Management Committee runs the project [20]. The tool runs in the command-line and can be used to manage certificates.

Entrust Cloud Certificate Management Portal [21] is the fifth tool in this section. It is used for obtaining Entrust issued certificates, and to monitor both Entrust and HUS CA certificates. It is a handy tool that can be used in combination with Entrust Discovery Agent that finds certificates based on the IP and Port set by an administrator. Once the agent finds a certificate, it send the information to the management portal that keeps track of certificate expiry and the location of the certificate. The tool can also be used to send automatic notifications when a certificate is about to expire and provides general information about what kind of certificates has been found. Entrust Cloud Certificate Management Portal will be used as the main tool to monitor certificate expiry in this thesis.

The sixth tool used in this thesis is Efecte [22]. Efecte Enterprise Service Management (ESM) is cloud-based service used by HUS as an Issue Tracking System (ITS). In this thesis, the webserver generates a ticket in Efecte for keeping track of certificate orders and informing administrators of new certificate requests.

The seventh tool, Node.js [23] is an asynchronous event driven runtime allowing to use Javascript programming language in the back-end of the web-server by handling multiple connections concurrently. The opposite of this technology is synchronous runtime, that only handles one connection at a time (i.e Microsofts .NET). The reason to use Node.js as the back-end technology was mainly because it allowed the author to code using Javascript, but also because by using Node.js there is not a need to increase the amount of back-end servers as often as when using a synchronous back-end technologies. To be able to use the best of both Node.js and IIS, a IIS module called IISnode was used. IISnode is developed by Tomasz Janczuk [24] and allows hosting of Node.js applications in IIS.

The eight tool Scrum is not really a tool, but rather a framework for an agile work method, which was used during the thesis. Scrum [25] consists of different events to handle work tasks, by different roles in the scrum team, in a time window called Sprint. A sprint can be one month or less in its length, and every sprint starts with an event called “Sprint planning”. Sprint planning is a meeting managed by the Product Owner (PO), who picks work tasks to a backlog, from where the team decides on what work will be taken under work. The team decides by itself how much work they think can be done during the length of one sprint. Once the sprint planning event is done the team starts to work on the picked tasks, and once every day during the length of the sprint, the team does an event called “Daily” managed by a Scrum Master (SM). A daily meeting is 15 min long where every team member explains what they did after the previous daily meeting and ask for help if they encounter an issue that they cannot handle themselves. At the end of the sprint, there are two more events. In the “Sprint Review” the team members presents the tasks that have been completed to the PO, who manages the meeting, closes the work tasks and informs the customer that the work has been done. After the review, the SM has a meeting with the team called “Sprint Retrospective” where all issues risen during the sprint will be presented. The objective of the retrospective is to find the obstacles that slowed down or hindered the work during the sprint, and find a way to overcome those obstacles in the future sprints. By this mentality the team would develop itself into a more effective team in every sprint.

The ninth tool used in the project was Bitbucket, which allowed to keep track of different versions of the website when programming both on a work computer and on a computer at home. Bitbucket is a version control repository owned by Atlassian [26].

5 Requirements of Certificate Enrolment System

This section covers the requirements for the software and hardware used in the thesis project. This section does not cover what the tools do in general, as that was described in the previous section, but the section covers in more detail how the tools will be used in the project. The project uses a refined waterfall model, known as the V-model (figure 4.) during the whole development process. The V-model includes different phases of testing of software. Testing of software will be discussed in the fifth section.

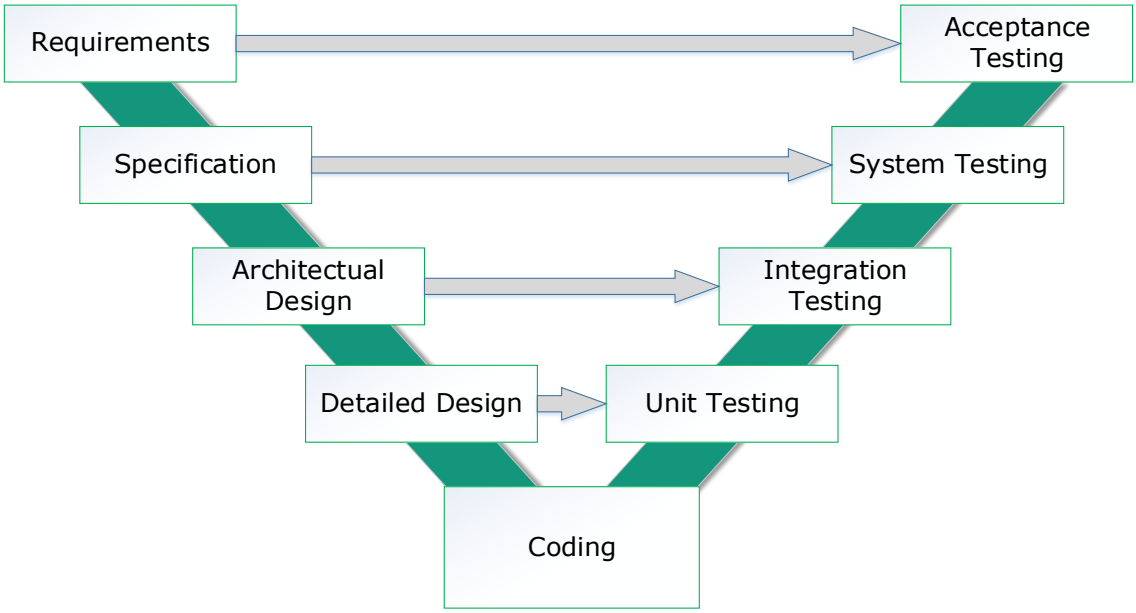

Figure 4. V-Model

To make the designing of the infrastructure easier, the author produced a mind map (Figure 5). Idea was to visualize as many parts of the infrastructure as possible before starting to split each area into smaller areas, and later on into even smaller tasks that could be included in a one-week sprint (see Scrum in section 4).

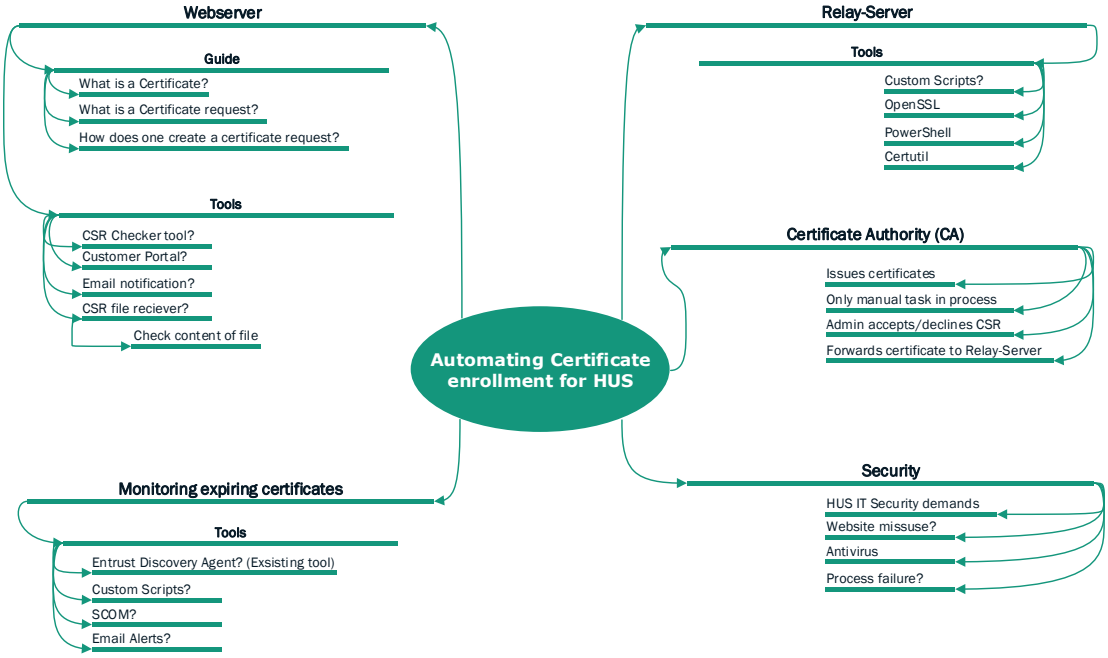

Figure 5. Mind map of project

5.1 Security

One of the key aspects of the thesis project is security. The intent is to create system that is as “hack-proof” as possible, and work in a way that the software does not malfunction and cause problems in the IT system. While working on the project, different scenarios where considered how a hacker might misuse the website. The new European Union’s (EU) General Data Protection Regulation (GDPR) will also be discussed.

The original project idea included a feature that would have forwarded the CSR to the CA server via a relay server doing the actual file forwarding. This would be easily implemented by storing the CSR to a temporary folder on the webservers hard-drive, have the relay server check the temporary folder frequently for new files and send them to the CA to await administrators approval. The reason to leave this feature out was to limit possible attack vectors on the CA, which is a critical part of HUS IT infrastructure. If the feature would have been added, and the website works as intended, the time saved by using the feature is a less than a minute. It is therefore not useful enough to implement. If the

number of certificates needed would suddenly increase, the forwarding feature could be reconsidered.

Data validation is handled on both client computer and backed server. Client-side validation concentrates more on informing the customer of missing or invalid data in the form, whereas the back-end validation concentrates tampered form data. If data has been tampered with or there is an error, the administrator of the website is informed.

Even though no information provided in the websites form is confidential, HUS IT security regulations require that all systems are end-to-end encrypted. This requirement is fulfilled by using HTTPS protocol on the website and of course a SSL certificate.

The EU's new GDPR [27][28] regulates how an individual, company or organization handles an individual's personal data within the EU. Personal data is any information that can be used to identify a particular person, be that by having the complete information at hand or pieces of information that collected together identify a person. This also includes information that has been encrypted or pseudonymised that can be reversed to re-identify the data. Information that cannot be decrypted does not fall under the scope of the regulation. Examples of personal data include [29]: name, surname, home address (that includes names), ID-card number, location data, IP-address, cookie ID, advertising ID on phone and data held be hospitals or healthcare personnel.

What is considered as data processing? Processing of data covers a wide range of operations, including manual or automated means, that are performed on personal information. Operations include: collection, recording, organisation, structuring, storage, adaptation or alteration, retrieval, consultation, use, disclosure by transmission, dissemination or otherwise making available, alignment or combination, restriction, erasure or destruction of personal data. [30]

The thesis project does not store any data on the back-end server and does not check if the email, name or surname exists. Personal information asked by the project website can be therefore be falsified. In the case the information is falsified the requester does not receive a certificate, as all requests are in the end verified by the administrator and all new and unknown requesters are validated manually before certificates are provided.

The information gathered by the website is sent in a html form directly to HUS work ticketing system where it is stored. After the form data has been emailed, the data does not exist on the back-end server anymore.

Emails cannot be sent outside of HUS-domain and website cannot be accessed anywhere else but the HUS internal network. A malicious hacker would therefore first need to get access to HUS internal network to be able to request a certificate.

5.2 Server-side

In this project, one server running Windows Server 2016 was used as the preferred operating system. The decision to use Windows servers was made on the basis that the Active Directory is used in the HUS IT environment to handle user, group and machine accounts and Group Policy Objects, which makes it easier to integrate other Microsoft products to the environment.

Server-side technologies also include the previously mentioned IIS and Node.js combination to handle the webpage publishing from the server.

According to HUS regulations, all websites and services visible to the internet must be published trough the existing F5 load balancer. Even though the project website will not be visible to the internet, it is still published via the F5 and is ready for possible future needs. By publishing the website via F5, HUS also gains the ability to add more back-end servers if one server is not enough, and balance the traffic between servers. A HTTP to HTTPS rewrite was also added on the F5 in case the user tries to access the website with an URL containing HTTP. By using this the user does not encounter an error if using HTTP.

5.3 Client-side

As previously mentioned, project will include a webpage explaining to customers what a digital certificate is, how one is made on Windows and Linux servers, explaining the difference between a public-, private- and Väestörekisteri Keskus (VRK) Certificate Authority issued certificates and also general HUS requirements for customers regarding certificate and monitoring of expiring digital certificates. Webpage will also provide a portal for customers to forward their certification requests files to the Certificate Authority.

Because the main aspect of the webpage is to function as a guide for the customer, the interface must to be simple and easy to use, guides must be clear with little text and many pictures on how to accomplish specific tasks. The reason for this is that most customers using the website will be people with little to no technical knowledge. Language used on the website must be in Finnish and English.

Importing of a CSR will be done alongside a form that has to be filled by the customer. The form will ask information who is ordering a certificate and who to contact when the certificate is about to expire. HUS IT provides customers with private, public and VRK signed certificates, all of which are acquired by different processes. Website will therefore have to explain the difference and requirements for each certificate and provide a tool to verify that the certificate request has been created in the right way. There will come times when a certificate has to be created that does not fit into the standard method of creating a certificate, and therefore the website will have to include the possibility to forward a certificate without normal verification. That does not mean that the customer shall be able to forward any kind of file, rather the information on the CSR can be outside the standard. When the user will forward a certificate for issuing that is outside the normal, users have to be alerted of what they are doing.

6 Implementation and Testing

This section describes how the front-end and back-end technologies were implemented, and how testing was done. This section is important as it demonstrates the actual website, HUS required website styles, client side validation and server side configurations. This section does not explain how a server operating system and roles are installed, as there are multiple guides about the subject on the internet.

6.1 Back-end

After the requirements had been made clear and the projects workflow was visualised (figure 6), the actual practical part of the project started with setting up the server and required roles (IIS), publishing a simple text file, giving a DNS name for the website (varmenne.hus.fi) and creating a SSL certificate for the website using the old method.

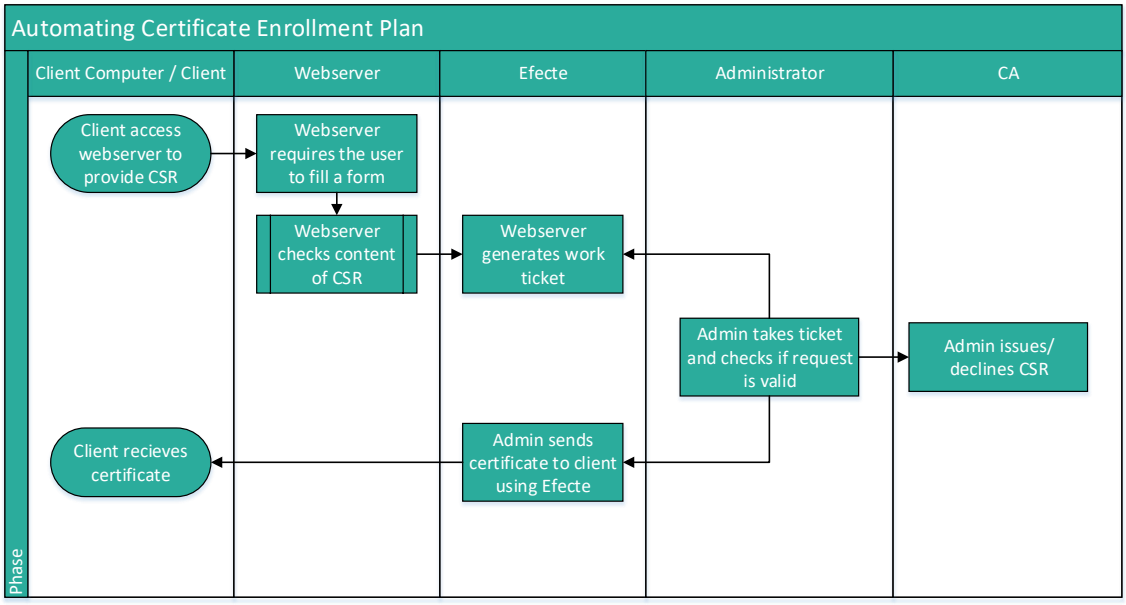

Figure 6. Visualization of the back-end process.

After the platform has been set up, the project concentrated on creating the front-end (described in section 6.2). Once the website was in the phase where back-end technologies had to be implemented, Node.js and IISnode was installed. Node.js is installed by running an executable [31] and following the instructions given in the installation program. When installing Node.js, Numeric Production Mechanism (NPM) is also installed. NPM is a critical part of Node.js, as NPM is used to download different packages created by the community. The packages used in this project are Express [32], Body-Parser [33], Express Handlebars [34], Nodemailer [35] and Path.

Express is a web application framework that renders the actual website content. Body-Parser returns the HTML body and only looks at the request where the content-type header matches the type value. In the project body-parser is used to direct the user

between websites when the use clicks on links and it is used to handle the form input that the user fills when applying for a certificate. Express handlebars is a template engine that dynamically generates a HTML page. In this the project handlebars are used to hold the actual HTML code and it also used to show a success / failure message to the user when the application form is submitted. Nodemailer is a package that enables Node.js to send emails directly from the application. In the project, Nodemailer is used to send the form data to HUS work ticket system, or if the back-end validation function encounters an error, the mail will be sent to the website administrator for further investigation. A Youtube video [36] made by Traversy Media was used to get the core of the Node.js application to work (figure 7) and the email configurations with a few modifications by the author (figure 8).

```
1   const express = require('express');
2   const bodyParser = require('body-parser');
3   const exphbs = require('express-handlebars');
4   const nodemailer = require('nodemailer');
5   const path = require('path');
6
7   const app = express();
8
9   // View engine setup
10  app.engine('handlebars', exphbs());
11  app.set('view engine', 'handlebars');
12
13  // Static folder
14  app.use('/public', express.static(path.join(__dirname, 'public')));
15
16  // Body Parser Middleware
17  app.use(bodyParser.urlencoded({ extended: false }));
18  app.use(bodyParser.json());
19
20
21  // Render websites
22  app.get('/', (req, res) => {
23      res.render('index');
24  });
25
```

Figure 7. Back-end app.js set up sourcecode.

Figure 8 shows the email configuration when sending the feedback form to the administrator. The ***msg*** variable on line 28 contains the string that will be shown to the user on the website after the user send the form and it has gone through validation successfully. A corresponding variable is added to the HTML code to where the message is shown in a H3 HTML tag once the customer submits to Certificate request form.

```
    // Set email configuration
    let transporter = nodemailer.createTransport({
        host: 'internalmail.hus.fi',
        port: 25,
        secure: false, // true for 465, false for other ports
        tls:{
            rejectUnauthorized:false
        }
    });

    // setup email data with unicode symbols
    let mailOptions = {
        from: '"Varmenne Sivusto" <noreply_varmenne.sivusto@hus.fi>', // sender address
        to: 'olli.kortelainen@hus.fi', // list of receivers
        subject: 'Varmenne sivusto / Palaute', // Subject line
        text: 'Hello world?', // plain text body
        html: output // html body
    };

    // send mail with defined transport object
    transporter.sendMail(mailOptions, (error, info) => {
        if (error) {
            return console.log(error);
        }
        console.log('Message sent: %s', info.messageId);
        console.log('Preview URL: %s', nodemailer.getTestMessageUrl(info));

        res.render('palaute', {msg:"Palaute lähetetty. Kiitos!"}); // return to "palute" website and show message on website
    });
});

app.listen(process.env.PORT, () => console.log("Server started..."));
```

Figure 8. Back-end app.js email source code.

Validation of the input data from the forms is validated in a function created completely by the author of this thesis, but because of security reasons and because this thesis will be publically available, the validation function will not be shown in the thesis report.

After the form has been validated, all the information in the form is forwarded to HUS IT work ticket management system, Efecte. A HUS CA administrator does a final check of the information, and if the data is valid, creates a certificate and emails it to the customer and technical contact provided in the form.

HUS is using a third party software to scan IP-addresses and ports for certificates. The software informs the CA administrators of expiring certificates when a predetermined time trigger is activated. With the combination of using the system created in the thesis project and certificate monitoring software a full certificate renewal cycle can be achieved.

6.2 Front-end

Development of the front-end began after the platform described in the beginning of section 5 had been created. The website is written in Finnish and includes a welcome-, guide-, certificate request- and feedback-page. Technologies used on the site is basic HTML, CSS and JavaScipt.
The style of the site uses the logo of HUS and follows HUS color and font standards, with minimal animations. The HTML document includes a head and footer that are identical on each webpage on the site. Body section of the HTML document changes depending on what webpage the user is viewing. Site also has the HUS logo as a favicon on the tab (figure 9).

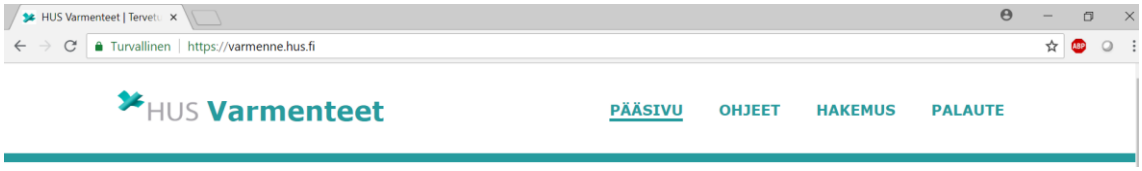

Figure 9. Header showing use of HUS standard colors and favicon logo

The colors standards at HUS are turquoise, gray and white. Alla colors are 100% without transparency. Text color is white on a turquoise or gray background or black text on white backgorund, HUS cross-logo is either turquoise on a white background or white on turquoise background. In addition to the previously mentioned main colors, green, blue, pink and violet colors are used as support. The main font style is Helvetica Neue or Verdana if Helvetica is not available.

The main webpage welcomes the user to the website and explains the sites purpose. Main page will also inform users of relevant updates to the site. The updates will be posted once the site has moved to the production phase.

Guides-webpage contains four sub-webpages. A F.A.Q. page which purpose is to teach users about certificates, certification request, what the user has to do and what to expect during the process of obtaining a certificate from HUS IT Management. At the time of

writing this report the F.A.Q tries to answer eleven questions that the author has encountered during a two year period.

Figure 10. HUS color codes (internally available pdf)

In addition to the F.A.Q page, the Guide-webpage includes step-by-step guides on how to create a certification request with subject alternative names on a Microsoft Windows Server 2016 and using the OpenSSL tool available on most Linux distributions. The guide for Windows Server 2016 is applicable on older versions of the operating system to at least Windows Server 2008. Because the guides have multiple steps, the webpages include collapsable sections to ease the reading of the guide. The users can choose the parts of the guide that they need to accomplish their objective.

The final webpage on the Guides-page is the up-to-date requirements for the certification requests. When the requirements for the CSR changes, the users will be informed of the change on the welcome-page and the information will be changed on the Guides-page.

After the Guides-webpage, is the certificate application and feedback form pages. The feedback form is a simple form where the users can send feedback to the sites administrator either using their name or anonymously. The certificate application form is the most

advanced webpage on the site. The webpage asks the users to fill in the contact information of the applicant and technical contact, project code to link the certificate to specific projects for billing purposes, the type of needed certificate and the certification request itself. The webpage asks the user to paste the content of the CSR in a text field and depending on the type of certificate chosen, the webpage displays the content of the CSR and highlights the missing required data with a red border.

When the applicant pastes the CSR content into the text field, the CSR is sent to a JavaScript function that parsers the CSR and returns the content in a readable form. The JavaScript function is from the example called “PKCS#10 complex example” and is written by Peculiar Ventures Yury Strozhevsky [37] with modifications (Appendix 1) made by the author of this thesis. The modifications include, display the data parsed in the correct HTML element, enabling CSR input to include a string that begins and end with “-----BEGIN NEW CERTIFICATE REQUEST-----“ and “-----END NEW CERTIFICATE REQUEST-----“ instead of Strozhevskys “-----BEGIN CERTIFICATE REQUEST-----“ and “

-----END CERTIFICATE REQUEST-----“ and also enabling the function to parse the Subject Alternative Names and displaying them to the HTML element.

Client-side data validation is handled in a JavaScript file (Appendix 2) written by the author of this thesis. The file checks the applicants input for special characters and missing information and CSR for duplicate input, missing data. Upon encountering unwanted or missing data, the applicant is informed of the error when trying to submit the form. The error message is shown as a pop-up alert displaying the type and place of error.

6.3 Testing

Testing of the website and back-end functions was done during the development of different functions to get the functions working. Once both the website and back-end were considered completed, further testing was done where different types of certificates was applied for using CSR’s with missing data, invalid data and valid data. Also testing was done from a hackers perspective where the forms input was modified directly in order to try and break the system. Once a fault was discovered, a fix was developed.

7 Project Results

This section summarises the results of the project and compares them to the requirements. The problems encountered in the implementation will be reviewed in the discussion section in addition to the sought solutions.

7.1 Project Outcome Summary

The result of the project is a working web-based certificate application system, that lets HUS employees and sub-contractors that have access to HUS internal network to apply for a certificate. The system adapts to the different CSR requirements based on the need of the customer, and automatically guides the user in applying a correct certificate.

The website has a F.A.Q webpage answering questions related to certificates and certification requests and a webpage with sub-webpages guiding the creation of a functional certification request based on HUS requirements.

Back-end scripts enable the webserver to send the application to HUS existing work ticket system for the CA administrators to take under consideration for issuance.

7.2 Development challenges

The project development progressed between the everyday workload, when the author had time off work and resulted in a presumed outcome. A few challenges was encountered during the project and thus this section explains the challenges and how they were overcome.

One challenge was the limited knowledge the author had about HTML and JavaScript, and no available people at work to ask for guidance when a problem was encountered. The result was for the author to learn more about HTML and JavaScript from various sources found on the internet, and this ultimately decided that Node.js was to be used as the back-end technology for the website.

The second challenge encountered was modifying the CSR parser function to parse the Subject Alternative Names from the CSR. First the author had to understand the ready

made code developed by Peculiar Ventures and modify the function to include the new parameters to parse. The challenge was overcome by studying the forums for bit's and pieces of information and trial and error.

The third challenge encountered was the freedom to develop. As the project was invented, designed and implemented by the author of this thesis, the project started to grow out of the range of the initial idea. The solution was to review the initial idea and to draw a line on how far the system would be developed until it could be considered done.

7.3 Sought Solutions

All problems were finally solved and most problems encountered were small typos in the code that were solved by reading the error message from the console output. Designing the look of the website was simple when following the style standards of HUS. Problems about the look was mainly about the placing of different HTML elements.

Most tasks were solved by first identifying the step that had to be done, figuring out how the task could be completed, solving the problems encountered and testing the solution, after which the process started from the beginning. The agile method SCRUM helped with keeping track of what was done and what still had to be done.

8 Conclusion

This section finalizes the thesis report by summarising the reason for the project and research question. In addition the section ends with discussion about the future needs of the new system produced by the project.

8.1 Demand and research

The reason for the project was to find a solution to free up the time HUS CA administrators use to explain to customers what a certificate and a certification request is, and how the customers can acquire one. In addition, there was also a need to automate the enrolment of certificates.

When researching the existing solutions, it became clear that the existing environment at HUS was similar to that of the Finnish Population Register Centre and other Finnish Hospital Districts. Thus, a new custom setup had to be developed. The research concentrated on the problems VRK and other Hospital Districts had in addition to the challenges at HUS. When the problems had been found out, the project concentrated on solving them. The final product being the website described in this report.

8.2 Follow-up questions

This last section discusses the future need of the final product. This section is important because it is gives the reader an idea on what to expect if the product is to be taken in to production.

In short, the world and technology changes and for this product there is a need to have a support for the product. The technology does not necessarily change in a month or a year, but every now and then, whether it is a worldwide or within an organization, a new standard is born that has not been accounted for in the product created during the thesis project. This requires that someone maintains the source-code and updates it accordingly.

The needs of the customers also change, and thus collecting feedback and responding to the needs of the customers is important for continued use of the product. This also requires that someone updates the site and its features.

Without having tested the product in full-scale with actual customers, it is difficult to predict what kind of response the product gains with the customers. There is a risk, where the product spreads into “production” when testing it with a smaller user base. If the product gains popularity, the word-to-mouth advertising of the product might gain the majorities attention before the product has actually been allowed to go into production. In that case, it can be difficult to remove or alter the product without causing confusion among the customers.

References

1 Rouse, M., Cobb, M., Brayton, J., Finneman, A., Turajski, N., & Wiltsey, S. “PKI (public key infrastructure).” Accessed 10 February 2018. http://searchsecurity.techtarget.com/definition/PKI

2 Cohen, F: A Short History of Cryptography. Accessed 10. February 2018.http://web.itu.edu.tr/~orssi/dersler/cryptography/Chap2-1.pdf

3 Picture of Scytale. http://cryptiana.web.fc2.com/code/scytale.htm

4 Picture of Polybius square. https://en.wikipedia.org/wiki/Polybius_square

5 Froomkin, D., Branson, A., Deciphering Encryption (Updated May 8, 1998). Assessed 12 February 2018. https://www.washingtonpost.com/wp-srv/politics/special/encryption/encryption.htm

6 Simonite, T. NSA Says It “Must Act Now” Against the Quantum Computing Threat. Accessed 12 February 2018. https://www.technologyreview.com/s/600715/nsa-says-it-must-act-now-against-the-quantum-computing-threat/

7 Sumner, S. Research: Legacy systems the biggest challenge in digital transformation. Accessed 15. March 2018. https://www.computing.co.uk/ctg/news/3011792/research-legacy-systems-the-biggest-challenge-in-digital-transformation

8 Lonoff Schiff, J. The 4 biggest healthcare IT headaches. Accessed 15. March 2018. https://www.cio.com/article/3197698/healthcare/the-4-biggest-healthcare-it-headaches.html

9 Shacklett, M. 10 BYOD concerns that go beyond security issues (Updated 20. August 2012). Accessed 15. March 2018. https://www.techrepublic.com/blog/10-things/10-byod-concerns-that-go-beyond-security-issues/

10 Map of HUS area. http://www.hus.fi/en/about-hus/organisation-and-representative-bodies/Pages/default.aspx

11 Historia. Accessed 15. March 2018. http://www.hus.fi/hus-tietoa/historia/Sivut/default.aspx

12 HUS-Tietohallinto. Accessed 15. March 2018. http://www.hus.fi/hus-tietoa/liikelaitokset-ja-tukipalvelut/tietohallinto/Sivut/default.aspx

13 Organisaatio. Accessed 15. March 2018. http://www.hus.fi/hus-tietoa/liikelaitokset-ja-tukipalvelut/tietohallinto/Sivut/Organisaatio.aspx

14 Clines, S. Loughry, M. Active Directory for Dummies, Second Edition. Pages 1,8. Wiley. 2008.

15 Rouse, M. Internet Information Services (IIS). Accessed 29. March 2018. http://searchwindowsserver.techtarget.com/definition/IIS

16 SFTP – SSH SECURE FILE TRANSFER PROTOCOL. Accessed 23. March 2018. https://www.ssh.com/ssh/sftp/

17 Microsoft, Accessed 4. May 2018. https://docs.microsoft.com/en-us/dotnet/framework/tools/certmgr-exe-certificate-manager-tool

18 Wikipedia, Accessed 4. May 2018. https://en.wikipedia.org/wiki/OpenSSL

19 OpenSSL, Accessed 4. May 2018. https://www.openssl.org/

20 OpenSSL, Accessed 4. May 2018. https://www.openssl.org/community/

21 Entrust. Accessed 4. May 2018. https://www.entrust.com/Certificate-Management/

22 Efecte. Accessed 4. May 2018. https://www.efecte.com/

23 About Node.js, Accessed 4. July 2018. https://nodejs.org/en/about/

24 Janczuk, T. Iisnode wiki, Accessed 4 July 2018. https://github.com/tjanczuk/iisnode/wiki

25 What is Scrum? 4. May 2018. https://www.scrum.org/resources/what-is-scrum

26 Bitbucket, Accessed 4. July 2018. https://www.atlassian.com/software/bitbucket

27 Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation), Accessed 12. August 2018. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679

28 What does the General Data Protection Regulation (GDPR) govern? Accessed 12. August 2018. https://ec.europa.eu/info/law/law-topic/data-protection/reform/what-does-general-data-protection-regulation-gdpr-govern_en

29 What is personal data? Accessed 12. August 2018. https://ec.europa.eu/info/law/law-topic/data-protection/reform/what-personal-data_en

30 What constitutes data processing? Accessed 12. August 2018. https://ec.europa.eu/info/law/law-topic/data-protection/reform/what-constitutes-data-processing_en

31 Node.js Download, Accessed 5. July 2018. https://nodejs.org/en/

32 Express. Accessed 5. July 2018. https://expressjs.com/

33 Express Body-Parser. Accessed 5. July 2018. https://www.npmjs.com/package/body-parser

34 Express Handlebars. Accessed 5. July 2018. https://www.npmjs.com/package/express-handlebars

35 Nodemailer. Accessed 5. July 2018. https://nodemailer.com/about/

36 Nodemailer – Send Emails From your Node.js App. Accessed 5. July 2018. https://www.youtube.com/watch?v=nF9g1825mwk

37 PKI.js, Accessed 6. July 2018. https://pkijs.org/

PKI.JS modified by author of the thesis

```
                //region Parse existing PKCS#10
                //*********************************************************************************
    function parsePKCS10() {
        //region Initial activities
        document.getElementById("pkcs10-subject").innerHTML = "";
        //document.getElementById("pkcs10-exten").innerHTML = "";

        document.getElementById("pkcs10-data-block").style.display = "none";
        //document.getElementById("pkcs10-attributes").style.display = "none";
        //endregion

        //region Decode existing PKCS#10
        if (document.getElementById("pem-text-block").value.includes("NEW"))
        {
            var stringPEM = document.getElementById("pem-text-block").value.replace(/(---
--(BEGIN|END) NEW CERTIFICATE REQUEST-----|\n)/g, "");
        }

        else
        {
            var stringPEM = document.getElementById("pem-text-block").value.replace(/(---
--(BEGIN|END) CERTIFICATE REQUEST-----|\n)/g, "");
        }

        var asn1 = fromBER(stringToArrayBuffer(fromBase64(stringPEM)));
        var pkcs10 = new CertificationRequest({ schema: asn1.result });
        //endregion

        //region Parse and display information about "subject"
        var typemap = {
            "2.5.4.6": "C",
            "2.5.4.11": "OU",
            "2.5.4.10": "O",
```

```
        "2.5.4.3": "CN",
        "2.5.4.7": "L",
        "2.5.4.8": "S",
        "2.5.4.12": "T",
        "2.5.4.42": "GN",
        "2.5.4.43": "I",
        "2.5.4.4": "SN",
        "1.2.840.113549.1.9.1": "Email",
        "2.5.29.17.2": "SAN",
        "1.2.840.113549.1.9.2": "SAN",
        "2.5.4.5": "SerialNumber"
    };

    // Check which certificate type the user has chosen
    var certificateType = document.getElementById("cert_type").value;

    // Store csr info into array
    var csrInfo = [];

    for (var i = 0; i < pkcs10.subject.typesAndValues.length; i++)
    {
        var typeval = typemap[pkcs10.subject.typesAndValues[i].type];
        if (typeof typeval === "undefined") typeval = pkcs10.subject.type-
sAndValues[i].type;

        var subjval = pkcs10.subject.typesAndValues[i].value.valueBlock.value;


        csrInfo.push([typeval, subjval]);
    }

    //region Put information about PKCS#10 attributes
    function getSANarray()
    {
```

```
var sanArray = [];
var count = 0;

                    if("attributes" in pkcs10)
                    {
                        for (var i=0; i<pkcs10.attributes.length; i++)
                        {
                            if(pkcs10.attributes[i].type === "1.2.840.113549.1.9.14")
                            {
                                var extensionsSAN = new Extensions({ schema: pkcs10.attributes[i].values[0] });

                                for (var j=0;j<extensionsSAN.extensions.length;j++)
                                {

if(extensionsSAN.extensions[j].extnID === "2.5.29.17")

{

    for (var k=0;k<extensionsSAN.extensions[j].parsedValue.altNames.length;k++)

    {

        sanArray.push(extensionsSAN.extensions[j].parsedValue.altNames[k].value);

    }

}
                                }
                            }
```

```
                                                    }

                                                    return sanArray;
                                                }

                                                return 0;
}


// Check to see if CSR includes required data
if (certificateType == "HUS-CA" || certificateType == "Public-CA")
{
    var valueCN = '';
    var valueO = '';
    var valueOU = '';
    var valueL = '';
    var valueC = '';
    var valueEmail = '';
    var sanArray = [];
    var sanCount = 0;



    for (var i=0; i < csrInfo.length; i++)
    {
        if (csrInfo[i][0] == "CN")
        {
            valueCN = csrInfo[i][1];
        }
        else if (csrInfo[i][0] == "O")
        {
            valueO = csrInfo[i][1];
        }
        else if (csrInfo[i][0] == "OU")
        {
```

```
            valueOU = csrInfo[i][1];
        }
        else if (csrInfo[i][0] == "L")
        {
            valueL = csrInfo[i][1];
        }
        else if (csrInfo[i][0] == "C")
        {
            valueC = csrInfo[i][1];
        }
        else if (csrInfo[i][0] == "Email")
        {
            valueEmail = csrInfo[i][1];
        }
        else
        {

        }
    }

    sanArray = getSANarray();

    if (valueCN == '') {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>CN</label><input type='text' class='missing' name='certCN' readonly value=''></li>";}
    else {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>CN</label><input type='text' name='certCN' readonly value=" + valueCN + "></li>";}

    if (valueO == '') {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>O</label><input type='text' class='missing' name='certO' readonly value=''></li>";}
    else {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>O</label><input type='text' name='certO' readonly value=" + valueO + "></li>";}
```

```
if (valueOU == '') {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>OU</label><input type='text' class='missing' name='certOU' readonly value=''></li>";}
else {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>OU</label><input type='text' name='certOU' readonly value=" + valueOU + "></li>";}

if (valueL == '') {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>L</label><input type='text' class='missing' name='certL' readonly value=''></li>";}
else {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>L</label><input type='text' name='certL' readonly value=" + valueL + "></li>";}

if (valueC == '') {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>C</label><input type='text' class='missing' name='certC' readonly value=''></li>";}
else {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>C</label><input type='text' name='certC' readonly value=" + valueC + "></li>";}

if (valueEmail == '') {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>Email</label><input type='text' class='missing' name='certEmail' readonly value=''></li>";}
else {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>Email</label><input type='text' name='certEmail' readonly value=" + valueEmail + "></li>";}


if (sanArray.length == 0 || sanArray == 0)
{
    console.log(0);
```

```
        document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>SAN</label><input type='text' class='certSAN missing' name='certSAN"+ (sanCount+1) +"' readonly value=''></li>";
    }
    else
    {
        for (var i = 0; i < sanArray.length; i++)
        {
            document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>SAN " + (sanCount+1) + "</label><input type='text' class='certSAN' name='certSAN"+ (sanCount+1) +"' readonly value=" + sanArray[i] + "></li>";
            sanCount++;
        }
    }


                var publicKeySize = "< unknown >";

                if (pkcs10.subjectPublicKeyInfo.algorithm.algorithmId.indexOf("1.2.840.113549") !== -1) {
                    var asn1PublicKey = fromBER(pkcs10.subjectPublicKeyInfo.subjectPublicKey.valueBlock.valueHex);
                    var rsaPublicKeySimple = new RSAPublicKey({ schema: asn1PublicKey.result });
                    var modulusView = new Uint8Array(rsaPublicKeySimple.modulus.valueBlock.valueHex);
                    var modulusBitLength = 0;

                    if (modulusView[0] === 0x00) modulusBitLength = (rsaPublicKeySimple.modulus.valueBlock.valueHex.byteLength - 1) * 8;else modulusBitLength = rsaPublicKeySimple.modulus.valueBlock.valueHex.byteLength * 8;

                    publicKeySize = modulusBitLength.toString();
```

```
                    var publicKeyInt = parseInt(publicKeySize);

                    if (publicKeyInt < 2048){var keyrow = "<li><label>Keysize</label><input type='text' class='missing' name='keysize' value=" + publicKeySize + " readonly></li>";}
                    else {var keyrow = "<li><label>Keysize</label><input type='text' name='keysize' value=" + publicKeySize + " readonly></li>";}
                    //var keyrow = "<li><p>Keysize<input type='text' name='keysize' value=" + publicKeySize + " readonly></p></li>";


                    //var keyrow = "<li><p><input type='text' name='keysize' value=" + publicKeySize + " readonly></p></li>";
                    }


                    document.getElementById("keysize").innerHTML = keyrow;

    }

    else if (certificateType == "VRK-CA") {
        var valueCN = '';
        var valueOID = '';
        var valueO = '';
        var valueL = '';
        var valueC = '';
        var valueS = '';
        var sanArray = [];
        var sanCount = 0;


        for (var i=0; i < csrInfo.length; i++)
        {
            if (csrInfo[i][0] == "CN")
```

```
    {
        valueCN = csrInfo[i][1];
    }
    else if (csrInfo[i][0] == "SerialNumber")
    {
        valueOID = csrInfo[i][1];
    }
    else if (csrInfo[i][0] == "O")
    {
        valueO = csrInfo[i][1];
    }
    else if (csrInfo[i][0] == "L")
    {
        valueL = csrInfo[i][1];
    }
    else if (csrInfo[i][0] == "C")
    {
        valueC = csrInfo[i][1];
    }
    else if (csrInfo[i][0] == "S")
    {
        valueS = csrInfo[i][1];
    }
    else
    {

    }
}

sanArray = getSANarray();

console.log(csrInfo[0][0]);
```

```
if (valueCN == ''){document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>CN</label><input type='text' class='missing' name='certCN' readonly value=" + valueCN + "></li>";}
else {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>CN</label><input type='text' name='certCN' readonly value=" + valueCN + "></li>";}

if (valueOID == ''){document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>OID</label><input type='text' class='missing' name='certOID' readonly value=" + valueOID + "></li>";}
else {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>OID</label><input type='text' name='certOID' readonly value=" + valueOID + "></li>";}

if (valueO == ''){document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>O</label><input type='text' class='missing' name='certO' readonly value=" + valueO + "></li>";}
else {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>O</label><input type='text' name='certO' readonly value=" + valueO + "></li>";}

if (valueL == ''){document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>L</label><input type='text' class='missing' name='certL' readonly value=" + valueL + "></li>";}
else {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>L</label><input type='text' name='certL' readonly value=" + valueL + "></li>";}

if (valueC == ''){document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>C</label><input type='text' class='missing' name='certC' readonly value=" + valueC + "></li>";}
else {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>C</label><input type='text' name='certC' readonly value=" + valueC + "></li>";}
```

```
if (valueS == ''){document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>S</label><input type='text' class='missing' name='certS' readonly value='' + valueS + ''></li>";}
else {document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>S</label><input type='text' name='certS' readonly value='' + valueS + ''></li>";}

if (sanArray.length == 0)
{
    // Uncomment if SAN is required in VRK Certificate
   //document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>SAN</label><input type='text' class='certSAN missing' name='certSAN"+ sanCount +"' readonly value=''></li>";
}
else
{
   for (var i = 0; i < sanArray.length; i++)
   {
      document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + "<li><label>SAN " + (sanCount+1) + "</label><input type='text' class='certSAN' name='certSAN" + (sanCount+1) + "' readonly value='" + sanArray[i] + "'></li>";
       sanCount++;

   }
}

        var publicKeySize = "< unknown >";
```

```
if (pkcs10.subjectPublicKeyInfo.algorithm.algorithmId.indexOf("1.2.840.113549") !== -1) {
var asn1PublicKey = fromBER(pkcs10.subjectPublicKeyInfo.subjectPublicKey.valueBlock.valueHex);
var rsaPublicKeySimple = new RSAPublicKey({ schema: asn1PublicKey.result });
var modulusView = new Uint8Array(rsaPublicKeySimple.modulus.valueBlock.valueHex);
var modulusBitLength = 0;

if (modulusView[0] === 0x00) modulusBitLength = (rsaPublicKeySimple.modulus.valueBlock.valueHex.byteLength - 1) * 8;else modulusBitLength = rsaPublicKeySimple.modulus.valueBlock.valueHex.byteLength * 8;

publicKeySize = modulusBitLength.toString();

var publicKeyInt = parseInt(publicKeySize);

if (publicKeyInt < 4096){var keyrow = "<li><label>Keysize</label><input type='text' class='missing' name='keysize' value=" + publicKeySize + " readonly></li>";}
else {var keyrow = "<li><label>Keysize</label><input type='text' name='keysize' value=" + publicKeySize + " readonly></li>";}
//var keyrow = "<li><p>Keysize<input type='text' name='keysize' value=" + publicKeySize + " readonly></p></li>";


//var keyrow = "<li><p><input type='text' name='keysize' value=" + publicKeySize + " readonly></p></li>";
}


document.getElementById("keysize").innerHTML = keyrow;
```

```
    }
    /*
    for (var i = 0; i < pkcs10.subject.typesAndValues.length; i++) {
      var typeval = typemap[pkcs10.subject.typesAndValues[i].type];
      if (typeof typeval === "undefined") typeval = pkcs10.subject.typesAndValues[i].type;

      var subjval = pkcs10.subject.typesAndValues[i].value.valueBlock.value;
      //var ulrow = "<li><p><span>" + typeval + "</span> " + subjval + "</p></li>";
      //var ulrow = "<li><p><span>" + typeval + "</span><input type='text' name=" + typeval + " value=" + subjval + " readonly></p></li>";
      //var ulrow = "<li><p>" + typeval + "<input type='text' name=" + typeval + " value=" + subjval + " readonly></p></li>";
      var ulrow = "<li><label>" + typeval + "</label><input type='text' name=" + typeval + " value=" + subjval + " readonly></li>";

      document.getElementById("pkcs10-subject").innerHTML = document.getElementById("pkcs10-subject").innerHTML + ulrow;
      //if (typeval === "CN") document.getElementById("pkcs10-subject-cn").innerHTML = subjval;
    }
    */
    //endregion

    //region Put information about public key size

    //document.getElementById("keysize").innerHTML = publicKeySize;
    //endregion
    /*
    //region Put information about signature algorithm
    var algomap = {
      "1.2.840.113549.1.1.2": "MD2 with RSA",
      "1.2.840.113549.1.1.4": "MD5 with RSA",
      "1.2.840.10040.4.3": "SHA1 with DSA",
      "1.2.840.10045.4.1": "SHA1 with ECDSA",
```

```
            "1.2.840.10045.4.3.2": "SHA256 with ECDSA",
            "1.2.840.10045.4.3.3": "SHA384 with ECDSA",
            "1.2.840.10045.4.3.4": "SHA512 with ECDSA",
            "1.2.840.113549.1.1.10": "RSA-PSS",
            "1.2.840.113549.1.1.5": "SHA1 with RSA",
            "1.2.840.113549.1.1.14": "SHA224 with RSA",
            "1.2.840.113549.1.1.11": "SHA256 with RSA",
            "1.2.840.113549.1.1.12": "SHA384 with RSA",
            "1.2.840.113549.1.1.13": "SHA512 with RSA"
        };
        var signatureAlgorithm = algomap[pkcs10.signatureAlgorithm.algorithmId];
        if (typeof signatureAlgorithm === "undefined") signatureAlgorithm = pkcs10.signa-
tureAlgorithm.algorithmId;else signatureAlgorithm = signatureAlgorithm + " (" +
pkcs10.signatureAlgorithm.algorithmId + ")";

        document.getElementById("sig-algo").innerHTML = signatureAlgorithm;
        //endregion
    */

        //endregion

        document.getElementById("pkcs10-data-block").style.display = "block";
    }
                //*****************************************************************************
                //endregion
```

scripts.js – Client side validation. Written by author

```
// function to verify form content on client-side
function validateForm() {

		function checkMatchingCAandSAN()
		{
			var valueCN = document.forms["csr-form"]["certCN"].value;
			var amountOfSAN = $('#pkcs10-subject .certSAN').length;
			for (var i=1; i<=amountOfSAN; i++)
			{
				var sanValue = document.getElementsBy-
Name('certSAN'+i)[0].value;
				if ( valueCN == sanValue)
				{
					return true;
				}

			}

			return false;
		}

		// Begin: Check that element exists
		var pkcsValue = document.getElementById("pkcs10-subject").innerHTML;
		var larr = pkcsValue.split(' ');
		var len = 0;

		// For each iterates over the index of arrays
		for(var i in larr) {
		    len += larr[ i ].length // Acummulate the length of all the strings
		}

		if (len == 0)
		{
```

```
                alert("Virheellinen varmennepyyntö.");
        return false;
        }
        // End: Check that element exists

        // Begin: check content of user input

        var firstname = document.forms["csr-form"]["firstname"].value;
    var lastname = document.forms["csr-form"]["lastname"].value;
    var email = document.forms["csr-form"]["email"].value;
    var tech_fname = document.forms["csr-form"]["tech_fname"].value;
    var tech_lname = document.forms["csr-form"]["tech_lname"].value;
    var tech_email = document.forms["csr-form"]["tech_email"].value;
    var P_code = document.forms["csr-form"]["P_code"].value;

    var userInputArray = [firstname, lastname, email, tech_fname, tech_lname,
tech_email, P_code];

        var format = /[!#$%^&*()_+\=\[\]{};:"\\|,<>\/?]+/;
        var invalidValuesArray = [];

        for (var i=0; i<userInputArray.length;i++)
        {
                if(format.test(userInputArray[i]))
                {
                        invalidValuesArray.push(userInputArray[i]);
                }

        }

        var invalidValuesMessage = "";
        if (invalidValuesArray.length >= 1)
        {
                for (var i=0; i<invalidValuesArray.length;i++)
                {
```

```
                                        invalidValuesMessage += "Syöte " +
invalidValuesArray[i] + " sisältää kiellettyjä merkkejä.\n" ;
                                }

                                alert(invalidValuesMessage);
                                return false;

        }
        // End: check content of user input

        // Begin: check duplicates in CSR Text field
    var csrInput = document.forms["csr-form"]["csrInput"].value;
    var pattern = /BEGIN/g;
    var result = csrInput.match(pattern);

        if (result.length > 1)
        {
                alert("Tiedosto kentässä virhe. Tarkista että tekstikenttään on
kopioitu vain yksi varmennepyyntö.");
                return false;
        }
        // End: check duplicates in CSR Text field

        // Begin: Check content of CSR
        var certificateType = document.getElementById("cert_type").value;

        // Begin: Check content of HUS-CA and Public-CA CSR
        if (certificateType == "HUS-CA" || certificateType == "Public-CA")
        {
                // Store values in variables
            var valueCN = document.forms["csr-form"]["certCN"].value;
            var valueO = document.forms["csr-form"]["certO"].value;
            var valueOU = document.forms["csr-form"]["certOU"].value;
            var valueL = document.forms["csr-form"]["certL"].value;
            var valueC = document.forms["csr-form"]["certC"].value;
```

```
	var valueEmail = document.forms["csr-form"]["certEmail"].value;
	var san = document.forms["csr-form"]["certSAN1"].value;
	var keysize = parseInt(document.forms["csr-form"]["keysize"].value);

	//Check matching CN and SAN
	if(!checkMatchingCAandSAN())
{
		alert("Varmennepyynnöstä puuttuu SAN joka on sama kuin CN");
		return false;
}



	// Check keysize
	if (keysize < 4096)
		{
			alert("Avaimen koko on liian pieni.\nHUS ja Julkisen varmentajan varmenteessa pienin sallittu avaimen koko on 4096.");
			return false;
		}

		// Check CSR content for empty string
	if (valueCN == "" || valueO == "" || valueOU == "" || valueL == "" || valueC == "" || valueEmail == "")
	{
		alert("Varmennepyynnöstä puuttuu tietoja. Tee uusi varmennepyyntö");
		return false;
	}
	else
	{
		return true;
	}
}
```

```
// End: Check content of HUS-CA and Public-CA CSR

// Begin: Check content of VRK-CA CSR
else if (certificateType == "VRK-CA")
{
        // Store values in variables
        var valueCN = document.forms["csr-form"]["certCN"].value;
        var valueOID = document.forms["csr-form"]["certOID"].value;
        var valueO = document.forms["csr-form"]["certO"].value;
        var valueL = document.forms["csr-form"]["certL"].value;
        var valueC = document.forms["csr-form"]["certC"].value;
        var valueS = document.forms["csr-form"]["certS"].value;
        var keysize = parseInt(document.forms["csr-form"]["keysize"].value);

        // Check value of VRK certificates sub-category
        var VRK_type = document.forms["csr-form"]["VRK_type"].value;
        if (VRK_type == "None")
        {
                alert("Valitse VRK varmenteen tyyppi.")
                return false;
        }

        // Check keysize
        if (keysize < 4096)
        {
                alert("Avaimen koko on liian pieni.\nVRK varmenteessa pienin sallittu avaimen koko on 4096.");
                return false;
        }

        // Check CSR content for empty string
        if (valueCN == "" || valueOID == "" || valueO == "" || valueL == "" || valueC == "" || valueS == "")
```

```
                {
                        alert("Varmennepyynnöstä puuttuu tietoja. Tee uusi
varmennepyyntö");
                        return false;
                }
                else
                {
                        return true;
                }
        }
        // End: Check content of VRK-CA CSR

        else
        {
                alert("Jotain meni pieleen.");
        return false;
        }
        // End: Check content of CSR

}
```