

UiO : **Department of Informatics**
University of Oslo

Ease implementation of security standards related to web application development

Vegard Fjogstad
Master's Thesis Autumn 2015



Ease implementation of security standards related to web application development

Vegard Fjogstad

19th July 2015

Abstract

In this thesis, the author attempt to design a process that will help web application development companies produce more secure services. This is achieved by using a security standard relevant to the services these applications provide. Throughout the course of this thesis, the author attempts to highlight why this is needed, and how it will be achieved.

Acknowledgements

At the time of writing this, the author of this thesis has officially been a student for 7 years. It is with mixed feelings that such a large chapter of my life is coming to an end. 7 years of work culminating into a single document. It has been quite a journey, filled with all the emotions a journey usually bring. As I look out the window, I see the every day life of Oslo slowly passing by. Cars driving on the highway, people walking on the sidewalk. In less than a month I'll be part of that life, no longer a student, but as an employed member of this society, crippled with student debt and frustration over the cost of a 1 bed-room apartment in downtown Oslo.

I would like to thank my parents and my siblings for their support through out my life, as well as their support while writing this thesis. Thanks to my supervisors, Aida Omerovic and Ingrid Chieh Yu. Thanks to Martin Andreas Juell and Thomas Torp for their help and continuous support during my time here at the University of Oslo. Thank you Ingunn Skjevdal Hansen for entertaining me over Facebook with nonsense through instant messaging. Last, I'd like to thank Finn.no for allowing me to evaluate my application at their office, and especially Stig Kleppe-Jørgensen, Per-Jørgen Walstrøm, Alexander Sagen & Henrik Brautaset Aronsen for their involvement in this evaluation.

Contents

| | | |
|-----------|---|-----------|
| 1 | Introduction | 1 |
| 1.1 | Structure of this thesis | 3 |
| I | Foundations | 5 |
| 2 | Problem statement & research method | 7 |
| 2.1 | Scope of this thesis | 7 |
| 2.2 | Goals of this thesis | 7 |
| 2.3 | Research method & evaluation strategy | 9 |
| 2.3.1 | Reasoning behind chosen evaluation strategy | 10 |
| 2.3.2 | Suitable alternatives to chosen evaluation strategy | 10 |
| 3 | Background & state-of-the-art | 11 |
| 3.1 | Development model | 12 |
| 3.2 | Development processes | 13 |
| 3.3 | Security standards & code of practice | 15 |
| 3.3.1 | ISO 27002 | 16 |
| 3.3.2 | Application Security Verification Standard | 17 |
| 3.4 | Conclusion | 19 |
| 4 | Stakeholders | 23 |
| 4.1 | Company | 23 |
| 4.2 | End user | 24 |
| 4.3 | Designer | 25 |
| II | Artifacts | 27 |
| 5 | The process | 29 |
| 5.1 | Success criteria of the process | 29 |
| 5.2 | Process description | 31 |
| 5.3 | Security standard requirements | 33 |
| 6 | The application | 35 |
| 6.1 | Success criteria of the application | 35 |
| 6.2 | Process description in the application | 36 |
| 6.3 | Features that did not make it into the prototype | 38 |

| | | |
|------------|---|-----------|
| 7 | Architecture for the application | 41 |
| 7.1 | Architecture and technology | 41 |
| 7.1.1 | CSS & Twitter Bootstrap | 41 |
| 7.1.2 | PHP | 42 |
| 7.1.3 | JavaScript | 42 |
| 7.1.4 | Alternatives | 42 |
| 8 | Demonstration | 45 |
| 8.1 | Project description | 45 |
| 8.2 | Run-through | 46 |
| 8.3 | Conclusion | 53 |
| III | Evaluation | 55 |
| 9 | Evaluation plan | 57 |
| 9.1 | Who is the target audience | 57 |
| 9.2 | What evaluation strategy is used | 57 |
| 9.3 | Evaluation description | 58 |
| 9.4 | Evaluation procedure overview | 59 |
| 10 | Evaluation environment | 61 |
| 10.1 | Respondents | 61 |
| 10.2 | Case description | 62 |
| 11 | Evaluation results | 63 |
| 11.1 | Case 1 | 63 |
| 11.2 | Case 2 | 63 |
| 11.3 | Case 3 | 64 |
| 11.4 | Case 4 | 64 |
| 11.5 | Case 5 | 64 |
| 11.6 | Case 6 | 64 |
| 11.7 | Case 7 | 64 |
| 11.8 | Case 8 | 65 |
| 12 | Reliability & validity of evaluation | 67 |
| 12.1 | Random elements & non-random elements | 67 |
| 12.2 | Evaluation environment & respondents | 68 |
| 12.3 | Questionnaire & case correlation | 68 |
| 12.4 | Would a re-evaluation provide similar results | 69 |
| 12.5 | Strength of ASVS implementation | 69 |
| 12.6 | Shortcomings of ASVS | 69 |
| 12.7 | Conclusion | 69 |
| IV | Conclusion | 71 |
| 13 | Summary & conclusion | 73 |
| 13.1 | Discussion of the evaluation results | 73 |

| | |
|--|-----------|
| 13.2 Conclusion of this thesis | 75 |
| 13.3 Future work | 76 |
| A Evaluation templates | 79 |
| A.1 Questionnaire for the respondents | 79 |
| B Prototype application availability | 81 |
| C Measures included in the report for the demonstration | 83 |
| C.1 Authentication | 83 |
| C.2 Malicious input handling | 83 |
| C.3 Cryptography at REST | 84 |
| C.4 Error handling and logging | 84 |
| C.5 Mobile | 85 |
| C.6 Malicious controls | 85 |

List of Figures

| | | |
|------|---|----|
| 2.1 | Overview of the iteration process for each artifact in this thesis | 9 |
| 3.1 | General overview of the waterfall method | 12 |
| 5.1 | Where the process can be used in a waterfall method | 30 |
| 5.2 | An end user could hold multiple roles within a project, making more standards relevant. | 31 |
| 6.1 | Screen capture showing ISO 27002:2013 in an unreleased prototype version of the application | 37 |
| 6.2 | Screen capture showing the login and restore session data feature | 39 |
| 8.1 | Screen capture of the web application question. | 47 |
| 8.2 | Screen capture of the list of technologies. | 48 |
| 8.3 | Screen capture of a relevant standard, and why it was selected. | 48 |
| 8.4 | Screen capture of the project descriptions. | 49 |
| 8.5 | Screen capture of the chosen level. | 49 |
| 8.6 | Screen capture of some of the ASVS areas in the confirmation step. | 50 |
| 8.7 | Screen capture of the “include in report” feature of step 5. | 51 |
| 8.8 | Screen capture of the “no measures” needed message. | 52 |
| 8.9 | Screen capture showing how the uncertain area was empty. | 52 |
| 8.10 | Screen capture showing that some measures were missing. | 53 |
| 8.11 | Screen capture showing a measure that should not be included in the report. | 53 |

Chapter 1

Introduction

The result of this thesis is an application that uses a process designed to help development companies implement more secure web applications. One of the features of this process is providing information about security and potential measures that can be taken from a relevant security standard. Relevant in this case referring to what the process deems best suited to solve any potential security threats a user might face, based on the answers provided by this user. Throughout this thesis, the foundations for this result will be introduced and explained. Questions such as "What problem is this application trying to solve?", "How will this process help my developers creating more secure applications?" and "Why use a standard to help solve security related issues?" will be answered, and the necessary background information on why these questions were made in the first place will be revealed. The problem that this application tries to solve, is the key problem and what this thesis is based on; "How can software development companies create more secure applications?". A more in depth problem statement and defined goals for this thesis will be given in chapter 3.

Security standards provides a company with some benefits over developing in-house measures against security related issues. One of these benefits are that for certain standards, a certification by an accredited body can be obtained. This means that a company could then inform their users that all or some of the services they provide have been certified, meaning that a user knows that any information they provide to these services are being stored securely in some way. A different benefit is that a standard is developed by a company or organisation with expertise specifically for the area this standard is trying to cover, compared to a development company that might not have any experience at all with handling the security issues they might face after deployment. Using a standard basically allows a development company to "borrow" the experience and expertise in a field they might have none, from someone that does. However, standards can be quite complicated and intimidating to use. Some standards spans over hundreds of pages, and would take a significant amount of time for each developer to learn. Certain ISO standards are so complicated that web sites have been developed, dedicated solely to inform companies on how

to implement them in their projects. One of these is the IsecT Ltd owned web-site iso27001security.com.

The necessity of handling security properly is obvious when you look at the amount of major security breaches in the last five years. One of the companies that have experienced multiple breaches over the last years is Sony. In November 2014, a major security breach was announced to the public[3]. A group calling itself Guardians of Peace, or #GOP, attacked Sony Pictures' corporate systems. According to messages left behind by this group, they had gained access to all of Sony Pictures' internal data, including personal data of Sony's employees, and threatened to release it to the public. What followed was a leak of several unreleased films, one being the blockbuster movie called Fury, starring Brad Pitt, and the cancellation of an, at the time, unreleased film named "The Interview". This film would eventually be released a month later. How such an attack was possible have been the source of rumours since it was announced to the public. What is known is that the attackers managed to install a malware on Sony's servers, known as the Destover[25], allowing them to completely erase any content found on the hard drive they are installed on.

This breach was not the first security breach Sony have experienced. In 2011, the Sony PlayStation branch suffered a security breach. While the breach at Sony Pictures in 2014 affected only their internal structure and employees, the PlayStation Network breach resulted in a total of 77 million compromised user accounts[4]. These accounts contained real names, addresses, birth dates, user names, passwords and even possibly credit card data. The research director of SANS Institute, Alan Paller, said that Sony most likely did not spend enough time and resources on security when they developed the software that the PlayStation Network runs on. SANS Institute is a major actor in security related work and according to their website is "the most trusted and by far the largest source for information security training and security certification in the world"[26]. Paller suspected that the perpetrators gained access to the network using malware, similar to what happened in the 2012 breach. This malware were installed on a system administrator's computer via an email.

These examples show that even a major company, whose budget reside in the billions (US Dollars), struggle to protect their services, networks and the data found on these networks. This thesis aims to provide assistance to help solve this problem by using security standards relevant to software development.

1.1 Structure of this thesis

The structure of this thesis is as follows:

Chapter 2 highlights the problem statement, scope for this thesis and the chosen research method, while chapter 3 contains background information that the author deems relevant to the goals of this thesis. The stakeholders of this thesis are described in chapter 4.

Chapter 5 and 6 describes the artifacts of this thesis. The process, followed by the application. 7 is a short summary over the technology used to implement the application, and chapter 8 shows a demonstration of the application that was eventually used in the evaluation.

The following 4 chapters are the evaluation chapters, starting with a description of the evaluation plan, followed by the evaluation environment and chapter 11 shows the results of the evaluation. Chapter 10 describes what possible threats to reliability might affect the outcome of the evaluation.

The last chapter of this thesis is the conclusion. This chapter also contains a section describing any potential future work.

Part I

Foundations

Chapter 2

Problem statement & research method

The information provided in the introduction gives some insight into some of the security related problems companies face today. Some of these problems are related to management; employees not being careful enough when opening emails, or disgruntled former employees that still have access to systems they should not have access to. Other security related issues happen due to poor programming when developing both private and public services.

Problem statement This thesis aims to design a process that can help development companies reduce the risk of security breaches or other security related issues, by providing security measures taken from security standards to help prevent these breaches or minimise the damage they can inflict.

2.1 Scope of this thesis

For evaluation purposes, the scope of this thesis is narrowed down to web application development companies, and the security related issues they face during development. This means that the certain parts of the process will be written specifically with web applications in mind. Details about how the process is tailored towards web application will be given in the chapter 5. This scope helps define the goals of this thesis.

2.2 Goals of this thesis

Listed below is an overview of the goals of this thesis.

1. Evaluate whether or not development companies spend enough time on security before deploying a web application to the public.
2. Design a process that can be used early in the planning stages of a development project.

3. Provide security measures from a standard deemed relevant by the process.
4. Allow the process to be expanded upon, providing more coverage.
5. Develop an application that implements this process in an efficient manner.
6. Reduce the complexity of using security standards in development.

First goal The first goal is important to this thesis, as the idea of introducing security at an early stage is what the process and application is built upon.

Second goal The second goal is the suggested solution to achieve the first goal; creating a process that can be used as early as possible, before any form of implementation has begun, to make it easier for development companies to implement more secure web applications. This would help companies spend more time on security before deployment happens, and in turn possibly reduce the risk of a security breach.

Third goal The third goal is mentioned in the problem statement. Throughout the process, the user will encounter various questions that help the process determine what security standard are relevant to this user. This standard would then present any security measures that could help secure a user's project. Relevant in this context means that the security standard that were chosen, contains measures that can be taken to make the project more secure. One important part of this goal is that the process does **not** aim to provide any form of certifications of these standards. A more detailed description of standard certification will be given in the chapter 3, background.

Fourth goal The fourth goal is important to make sure the application and process stay relevant, as new security leaks are discovered. Coverage refers to adding more security standards or newer revisions to the process, giving users a wider array of potential projects that can use the application.

Fifth goal The fifth goal is the application that implements this process. Efficient refers to providing a user with a clean graphical user interface that provide enough help to allow an in-experienced user of this application to complete the process, without having to ask an already experienced user for help.

Sixth goal The sixth and final goal aims to enable more companies access to security standards by reducing the complexity of them. Reducing the amount of information presented to a user, by stripping away any information not relevant to a specific project should lower the difficulty of implementing a security standard into a company's development structure.

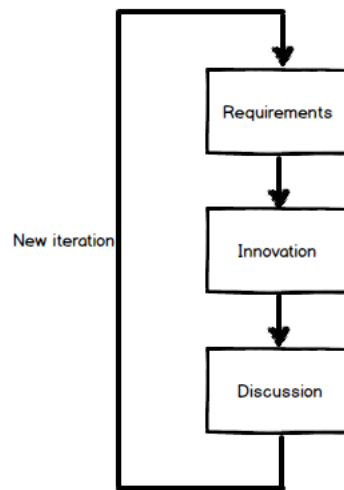


Figure 2.1: Overview of the iteration process for each artifact in this thesis

2.3 Research method & evaluation strategy

Before any work were done on the thesis itself, a problem analysis were written. This problem analysis contained information such as the objective of the thesis, as well as the stakeholders and success criteria for the suggested artifacts found in this analysis. This information served as a foundation for most of the work done in this thesis.

The artifacts that were made for this thesis is the result of several iterations performed on the original artifacts found in the problem analysis. Most of these iterations happened after discussion that occurred during meetings between the author of this thesis and the author's supervisors. For each iteration, these meetings served as an evaluation of the current artifact, which helped shape the artifacts. Figure 2.1 shows a simple overview over this process. This figure is based on figure 5 on page 15 in the 2007 report written by Ida Solheim and Ketil Stølen for SINTEF, named "Technology Research Explained"[27].

Ideally, there would be a proper evaluation with a suited evaluation strategy for each iteration, and the respondents would have not have any connection to the author. When these meetings were held, the artifacts had not reached a stage where an evaluation like this would provide enough results to warrant the time investment. The final evaluation is therefore the only evaluation where the chosen evaluation strategy have been used.

As this thesis aims to design an application to help solve the problem statement, a practical evaluation strategy would be the best fit to help determine whether or not the goals of this thesis have been met. The evaluation of the application presented in this thesis will consist of a field study, combined with a qualitative interview at a major software development company in Oslo, that specialises in web applications.

2.3.1 Reasoning behind chosen evaluation strategy

A field study and a corresponding qualitative interview is a good fit for the thesis, as well as using non-empirical evidence. This is because the environment in which the field study will be conducted provides a good foundation for evaluating all the goals of this thesis. Combining the observation with a interview allows for a direct evaluation of how well the goals of this thesis are met. This environment will be described in the evaluation chapter.

2.3.2 Suitable alternatives to chosen evaluation strategy

There were other alternatives to a field study that could be used for the evaluation. Some of these potential strategies were described in J.E. McGrath's "Groups: interaction and performance", written in 1984. 8 different strategies were divided into four groups, depending on the evaluation environment and if it is a theoretical or a practical evaluation strategy. Of these 8, the most relevant to this thesis apart from the two already chosen, would be to a field experiment or with non-empirical evidence.

A field experiment is similar to a field study, major difference being a field study interferes as little as possible with the evaluation during observation. A field experiment uses a natural environment, but in stead of simply observing, the observer can intervene or manipulate certain elements of the evaluation to affect its outcome. By doing this, certain situations that might not occur naturally with the given respondents, could happen. A possible example would be to offer a security standard that might not be relevant, to observe how a respondent would react, and how much he or she would "trust" the application to present the user with the best suited standard for his or her project.

Non-empirical evidence refers to presenting strength and weaknesses of the artifacts, based on logical reasoning. Important factors would be the observers experience with the services the artifacts provide, such as the given security standard or how an average user reacts to a step-by-step process encountered in the prototype application.

Chapter 3

Background & state-of-the-art

Mentioned in the first chapter were two separate security breaches that occurred at two different departments at Sony. This chapter will introduce some of the potential measures that Sony could have taken to maybe prevent these incidents from happening in the first place. "Maybe" being a keyword here, because making sure something is 100% secure is close to impossible. What these measures are designed to do is to reduce the risk of a breach happening, through the use of specific development processes, or applications designed to analyse a project to discover any possible security related issues.

Both of the security breaches mentioned in the introduction happened due to multiple reasons. Both attacks were so successful because the attackers managed to install a malicious piece of software on a server or multiple servers located inside Sony's network. How the Trojan Destover malware got installed during the 2014 breach is not entirely clear, but the malware used for the 2011 attack got installed by accident via an email sent to a system administrator. The amount of data that got stolen during the 2011 attack were so high, that this breach was reported as being one of the largest ever security breaches. Alan Paller, mentioned in the introduction, said that part of why this attack was so devastating were due to sloppy development practices that were allowed to take place because Sony rushed their products. This reduced the amount of attention given to security during development, allowing products with error-filled code to be deployed. This code were then exploited to gain access to information not intended for the general public.

This chapter looks at the current processes and security standards that have been developed to help prevent situations like these from occurring, and how they can be related to the goals of this thesis. The conclusion of this chapter contains a summary of how the listed processes and security standards match up with the goals of this thesis, and the gaps that the artifacts of this thesis tries to fill.

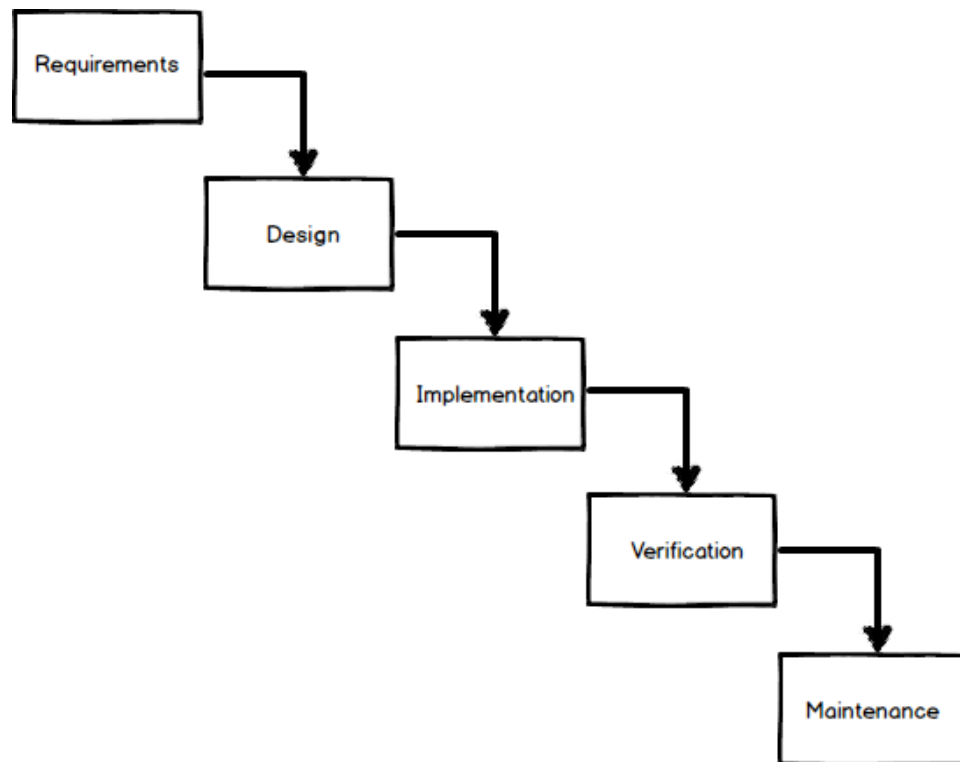


Figure 3.1: General overview of the waterfall method

3.1 Development model

A development model is a process that describes the life-cycle of a development project, from inception to after deployment[32]. By using a development model, companies can more easily define what work has to be done during which phases. The "waterfall model", one of the oldest development models, is based on fully completing a step before continuing to the next step. A waterfall model is often divided into five different steps, first step being the "Requirements" step and the last being "Maintenance". Figure 3.1 show an overview over all 5 steps encountered in the waterfall model. The V-model is a slightly more complicated version of the waterfall-model, where a test life-cycle is performed alongside the regular development life-cycle.

The way any waterfall-based model work is that once a step is completed, developers can not go back to a previous step, making it impossible to commit changes or fix any errors. This means that should any of the initial requirements made during the first step be wrong, the project have to start from scratch. Due to this strict nature, handling security issues related to programming have to be done before the verification phase, which requires a great amount of knowledge from the developers, as they have to identify all possible threats related to the project.

At the opposite side of the spectrum is an "Agile model". These models are based on completing projects in iterations. The life-cycle itself can be

very similar to a waterfall-model, the difference being that this life-cycle is repeated several times during the course of a project. These repetitions are often referred to as iterations. Agile allows developers to, as the name might suggest, make changes to previous steps after they have been initially completed. Combining an agile work flow with a minimum viable product would allow developers to iron out security issues as they surface, and then implement more features that would lead to additional possible security related issues.

These development models are used in one of the development processes that will be described in the following section.

3.2 Development processes

A method to prevent security issues is to design software to be secure from the ground up. These theories are known by several different names, such as "Secure by design"[9], "Secure programming" and "Defensive programming"[10]. The idea is using good programming practices that would prevent situations where bugs related to security could occur. This approach put a lot of responsibility on each developer, as they would have to be very experienced with the programming language that is being used to develop with, in order to recognise situations where faulty programming techniques could lead to potentially vulnerable systems, much like the waterfall model. The problem with these theories is that while the idea behind them are great, applying them in practice is difficult because developers can not be expected to recognise every single possible situation that can lead to a potential security issue. However, these theories can serve as a good foundation for more in-depth processes that tries to help a developer spot issues while programming.

Some development companies have implemented their own processes specifically with security in mind, in an attempt to develop more secure applications. Microsoft released in 2004 their first version of the Security Development Lifecycle (SDL)[7]. SDL's main goals are to reduce the amount of security issues that arise due to design and programming, and to limit the ramifications of any security issues that might still exist after development. The Microsoft Security Development Lifecycle process is a multi-step process for creating secure applications using both risk analysis and thread modelling to map security related issues. The process consists of seven step, referred to as phases. Each of these phases have several sub-steps that have to be completed before moving on to the next phase. From start to end, the seven phases are "Training", "Requirements", "Design", "Implementation", "Verification", "Release", "Response". These names are similar to some of the stages encountered in both a waterfall model, as well as an agile development model. One step of the "verification" phase, is the "SDL practice #11: Perform Dynamic Analysis". Microsoft themselves have developed two different applications for this step; Application Verifier and BinScope. The latter were built in compliance with SDL's requirements and recommendations. The BinScope application lets a user

choose a file containing code from their project to analyse; the application then scans this file and concludes with a report of which of the SDL requirements that have been met and not met. What is not completely clear is what programming language this software supports. There is a version that is implemented in Microsoft's Visual Studio, an Integrated development environment (IDE)[28] used to help developers program more efficiently. This suggests that any language supported by Visual Studio can be analyzed, such as JavaScript and ASP .NET, which are both used to create web applications.

Microsoft developed a guide for SDL that would help companies integrate the security practices found in SDL into an agile development environment. Each step in the different phases belong to one of three categories; "every-sprint practices" are performed for every iteration of a project, while "one-time practices" happen only once during the start of a project. The third and final category is the "bucket practices"; this step contains all practices that have to be completed on a regular basis, but does not have to be done in each iteration. Should a company want to use SDL alongside a waterfall model, then the process is a simple start-to-end process that need no further explanation.

Microsoft's Security Development Lifecycle is a process that, by using the theories of "secure by design" intent to prevent attacks from happening. A different approach revolve around reducing the ramifications of a breach if it happens, instead of outright trying to prevent them from happening in the first place. The Information Security Management System, or ISMS, is a set of policies for "systematically managing an organisation's sensitive data"[8]. ISMS main goal is to minimise risk and reduce the impact of a security breach should it occur. SDL is a process that focuses on what a developer can do while writing code to improve security; ISMS focuses more on the management side of a company. It recognises that not all security breaches occur due to an external source such as a hacker, but sometimes comes from inside the company's structure. Examples include former employees that felt they had been treated poorly, or even current employees. Internal security breaches can also occur by accident, with no malicious intent. The Utah State University reported on February the 6th, 2015 that a staff member had accidentally sent an email message containing 347 individual names and social security numbers to a group of USU student Veterans[29].

Some development companies choose to instead of adapting these policies or processes to their own development and management practices, they hire external advisers or companies that work with security related issues in software development companies. The company that were used to evaluate the artifacts of this thesis is a development company that, for the project that was used in the evaluation, hired a security company to come in after development had completed and analyse the application for any potential security issues. This approach frees up the developers time to focus purely on implementing the application according to their requirements, and letting the advisers solve any security related issues after development has reached a certain stage.

3.3 Security standards & code of practice

A "high level" description of a standard is that it is a good way of doing "something". Standards cover a vast amount of areas, ranging from wireless Internet solutions, to electrical toothbrush recharge stations. One of the major benefits a standard can provide is a certification. For an ISO standard, this involves meeting the requirements set by a specific standard, and then having a certification body (a third party company) come in to verify these requirements[36].

A disadvantage with using a standard is the sheer size of some of the standards that might be relevant to whatever a company is trying to do. Some ISO standards span over a hundred pages. The language used in some standards can also be quite complicated, which means that any employees involved might have to read it two times or more to fully understand the material.

This section will give an overview over some of the security standards related to software development, as well as software development companies in general; namely ISO 27001, ISO/IEC 27002:2005 and OWASP.org's ASVS standard.

Mentioned in the previous section, ISMS is an approach to security management. This practice were in 2005 implemented in the ISO/IEC 27001 standard[13], and subsequently deprecated in 2013, with the publication of the latest revision, ISO/IEC 27001:2013. The newest revision changed some of its security controls, a term that will be explained later, to better support newer technologies like "Cloud Computing". The 27001 standard focuses on management, providing a set of requirements to properly assess a company's risks and how to prevent them in a step-by-step process[15]. These requirements have to be first planned, then implemented and later on reviewed to improve them. A related standard is the ISO/IEC 27002 standard. In the annex of ISO 27001, a number of security controls are listed. These security controls refer to different areas in a company's structure that needs various policies related to security, and are taken directly from the ISO/IEC 27002 standard. Examples include "Information security policies", "Asset management" & "Supplier relationships". ISO 27002 is a code of practice, rather than a full fledged certification standard, and is often used in conjunction with ISO 27001[35]. Section 3.3.2 explain more about ISO 27002.

In short, the ISO 27001 standard tries to help an organisation define an ISMS. One of the parts of the process in this standard is to select any security controls and security control mechanisms. The ISO 27002 provides the organisation with a number of potential security controls and security control mechanisms that can be used to complete the ISMS. These controls are what is listed in the annex of ISO 27001.

The "Standard of Good Practice" (SoGP)[16], published by the Information Security Forum (ISF) is a guide providing information on identifying and managing security risks. In 2011, SoGP was updated to cover the requirements for an ISMS that were set out in the ISO 27001 standard.

In 2009, an organisation called Open Web Application Security Project

(OWASP) released their first revision of a security standard called Application Security Verification Standard (ASVS)[30]. This standard is the product of a collaboration between the members of OWASP. The latest revision of ASVS saw release in 2014, its first since the original release. The goal of ASVS is to provide a basis for verifying a web application. Verify and verification is a word often used in relation to standards. In this thesis, the term verified refers to if a development project meets the requirement found in a security standard.

3.3.1 ISO 27002

ISO 27002 is a code of practice that is often used in correlation with ISO 27001. The "security controls" found in ISO 27002 help user meet the requirements that is set in ISO 27001. Defined in section 2 of the ISO 27002:2005 standard, control is used as a synonym for safeguard or countermeasure[37]. The 2005 edition, the one the author of this thesis have access to, have a total of 11 security controls clauses. These clauses contain a total of 39 main security categories. Each security category contain a control objective, highlighting what needs to be achieved and one or several controls that can be used to achieve this objective. The following list shows these 11 clauses, and the number of security categories in each clause. They are taken directly from the 2005 edition of ISO 27002, section 3.1:

- Security Policy (1)
- Organising Information Security (2)
- Asset Management (2)
- Human Resources Security (3)
- Physical and Environmental Security (2)
- Communications and Operations Management (10)
- Access Control (7)
- Information Systems Acquisitions, Development and Maintenance (6)
- Information Security Incident Management (2)
- Business Continuity Management (3)
- Compliance (3)

As an example, some of the security categories found in the Access Control security clause are "User Access Management" and "Network Access Control." Some of these clauses bear little relevance to web application development, such as the Security Policy clause. The control objective states that it aims to "provide management direction and

support for information security in accordance with business requirements and relevant laws and regulations." (Section 5.1 of ISO 27002:2005). However, other clauses contain categories that could be used in relation to development, examples include the "User registration" category, where some of the controls involve maintaining user IDs, granting users a written statement of their access rights and requiring users to sign statements that show they understand the terms of their access. (such as many of the Terms & Conditions found in various applications and websites)

3.3.2 Application Security Verification Standard

The following sub-sections is an in-depth description of how the Application Security Verification Standard is used in practice. This standard is written in a more detail compared to the other standards mentioned in this chapter, as it was chosen as the standard to be implemented for the evaluation of this thesis.

The objective of the ASVS standard is to provide a basis for testing is "to normalise the range in the coverage and level of rigor available in the market when it comes to performing Web application security verification using a commercially-workable open standard"[30]. What this means in practice is that the standard tries to provide measures that are relevant to the size of a project. In the ASVS standard, features and services a project have implemented is referred to as security requirement areas, similar to the security controls in the ISO 27001/27002 standard. Coverage refers to how well a project manage to meet the security requirements found in these areas. A smaller project should require fewer measures to achieve an acceptable coverage, and a larger project require more measures. The standard provides a user with help on how to determine the size of their project. This is achieved by dividing the measures into different levels of verification.

Level of verification

The level of verification is a term defined in the ASVS standard that help describe the size of a project, and how much effort is needed to achieve verification.

If an application meet all the requirements for a certain verification level, then it would be considered an OWASP ASVS Level X application, X referring to the level of verification for the project. ASVS operate with 4 different levels. The first, and easiest level to comply with is level 0, known as "Cursory". The purpose of this level is to define that some form of security review have been performed for the project. The ASVS standard does not provide any requirements for this step, it is instead suggested that a company define these themselves. Level 0 is not a prerequisite for the other levels. What this means is that a project can skip level 0 and instead go directly to level 1.

Level 1, named "Opportunistic", is the first level where ASVS provide requirements that has to be met in order to achieve verification. The ASVS

standard reads that "An application achieves Level 1 (or Opportunistic) certification if it adequately defends against application security vulnerabilities that are easy to discover".

The second level is level 2, "Standard". This is aimed at applications that require security against vulnerabilities that poses moderate-to-serious risk. This can include services that handle money transactions such as web shops, but also applications that process sensitive information like health care or personal information about the companies employees.

The highest level of verification that can be achieved in this standard is level 3, "Advanced". Projects that should strive for this level of verification can include applications that in some form handle life and safety, such as a control system for the traffic lights in a city, where any errors could lead to vehicles crashing into each other or into pedestrians.

Once the verification level have been established, an overview of the functionality the project will offer is required to proceed. This information is used to define security requirements based on the verification level. The security requirements are then divided into what the standard refers to as security requirements areas.

Security requirement areas

The security requirement areas defined in the ASVS standard mostly revolve around the functionality a project offers, or how this functionality is implemented. Some of these areas are similar to the security controls found in ISO 27002. Each of these areas have a different amount of requirements that has to be met in order to achieve verification. The amount of requirements depends on the chosen security level. A project with a security level 1 would have to meet eight specific requirements in order to achieve verification for the "Authentication" security requirement area. A project with a security level of 2 would have to meet the same eight requirements, and an additional eleven. In order to make it easier to transition from older versions of ASVS, the original numbering scheme have been kept from the previous versions. As of the 2014 revision, there are 13 different security requirement areas. Below is a short description of four of these areas.

V2. Authentication Authentication contain the requirements related to log in/out services. Measures include ensuring that passwords field do not echo the user's password when it is entered, or that all credential and identity information is properly encrypted.

V3. Session Management Sessions handle data that is temporarily stored on the server. The information that is stored in a session object is subject to the developer, but examples include passwords and answers given by users to questions answered with radio buttons. During the implementation of the application artifact in this thesis, sessions were used to allow users the ability go back a step to change the answer to a question.

An example measure for this area is to verify that sessions timeout after a specified period of inactivity, a feature often implemented in Internet banks.

V4. Access Control Access control contains requirements that are related to authentication, and preventing users from accessing functions or services which they do not have the require authentication to access. Situations where this could occur is for services like email, where a user should only have to access to his or her own emails.

V5. Malicious Input Handling Measures that are designed to prevent users from inputting symbols, or even pieces of code into the web application is found here. Unprotected forms can for example be used by people with malicious intent to access a database, or even wipe out all data found on a database. This is known as SQL-injection.

Remainder The remaining nine security requirement areas are

- V7. Cryptography at Rest
- V8. Error Handling and Logging
- V9. Data Protection
- V10. Communications,
- V11. HTTP
- V13. Malicious Controls
- V15. Business Logic
- V16. File and Resource
- V17. Mobile.

Notice how the numbers do not progress naturally, indicating how some areas were either removed completely or merged with a different area at some point during a revision of the standard.

3.4 Conclusion

The information found this chapter highlighted some of the current processes out there that tries to make applications more secure. The Microsoft Security Development Lifecycle is particularly interesting because it meets two of the five goals of this thesis, namely the second and third goal. The SDL process is used before any form of implementation have been done, as well as during development and after deployment. It is a process that stretches from inception to evaluation.

The main issue with this process is that it is so extensive that a company has to change their own development practices to accompany the requirements made by SDL. The first step includes giving employees training in what they refer to as "core security training", which basically mean that a company have to send their employees on seminars to learn about security. The process assumes that a company that wants to implement it into their practices have employees that are already experienced with security, such as threat modeling, secure coding, security testing and secure design[31]. The second issue with SDL is that in order to achieve the third goal of this thesis, the company has to determine what security standard is relevant to their project and then find the relevant measures, and this only partially achieves it as the goal states that the process should provide the standard, and not the user.

Mentioned in section 3.2, ISMS tries to reduce the damage caused by a breach by designing policies that can be implemented into a company's practices. Due to the sheer size of the standards that implements this practice, such as ISO 27001, the inclusion of such a standard in the eventual process was put on hold, as it simply was too complicated for a thesis of this size. It is relevant to this thesis however, as the fifth goal aims to create artifacts that in the future can be expanded upon to include more standards. This would allow the process to not only offer measures related to development, but also for structuring a web application development company according to the requirements found in a ISMS standard. The ISO 27001 standard is not aimed at development project, but company policies and structure in general. The second goal of this thesis specifically aims at introducing a process early in a development project, which means that a standard such as the ISO 27001 standard would have to be used in a way that only included the project members. Just how well such a standard could be implemented into this standard would have to be tried and tested however. A more relevant standard would be the ISO 27002, as it provides direct measures, or controls, to safeguard information and a users interest. In order to best make use of it, several of the security clauses would have to specifically filtered out to provide only the relevant results for a user that is interested in web application development. In short, 27001 at this point do not offer enough relevancy to what this thesis aims to accomplish related to web application development. ISO 27002 will be used at various points during this thesis as a demonstration of how a standard (or in this case a code of practice) can be implemented into the artifacts of this thesis.

A standard that were well suited to this thesis however, is the ASVS standard described in detail in section 3.2.1. This is the standard that was eventually implemented into the process that this thesis produced. The reason for this is that, as mentioned, it allows both larger and smaller projects to achieve verification, as the measures complexity scale in comparison to the size. Should the project grow, the level of verification can simply be increased if needed. The standard itself is also easy enough to understand, making it possible to implement into a rather simple application.

What this chapter highlighted was that while there was a development

process (SDL) that fit nicely with some of the goals of this thesis, it does not cover all of them. The lack of a clear connection between the security process and a security standard, means that in order to achieve the goals of this thesis, a new process has to be made. Some of the information found in this chapter helped define the success criteria for the artifacts presented in part II.

Chapter 4

Stakeholders

Using the information from the previous part of this thesis, the following stakeholders have been created. These stakeholders represent the relevant parties related to the current state of this thesis. Notable differences in this chapter would be the description of the end user. The last goal of this thesis notes that the artifacts should be expandable to include more standards. Should the process implement a standard related to management such as the ISO 27001 standard, then the end user stakeholder would also include employees working in management, and a company's needs might change.

This chapter will describe what each stakeholder represents and their needs. Each stakeholder's needs are defined in relation to the goals of this thesis. Unless stated otherwise, process in this chapter refers to the artifact process this thesis produced.

4.1 Company

The company stakeholder represents a development company that have employees spending time developing web applications. The company is interested in implementing the services this thesis provides into their current development practices.

Needs

1. The process should be usable without having to re-structure a company's current development practices in a drastic manner.
2. The application that implements the process should be available on Linux, Windows and OSX.
3. A company do not have to train their employees in how to use the artifacts produced by this thesis.

Indications of needs being satisfied A company that wants to implement the process into their current development practices should only need to allocate time to complete it, some time before implementation has begun. Any company that do not spend time before implementing would need to

do so in order to use this process, but are not required to make any changes outside of that to their development practices.

The application should be able to run on any operating software that can run a modern web browser e.g. Internet Explorer, Firefox, Chrome, Safari. This allows their employees to use the application on their operating system of choice.

If a company can begin using the application immediately after acquiring it, then the last need is satisfied.

4.2 End user

The end user is a person using the process. This is in most cases a developer that will work on implementing a web application, but can also be someone working in management e.g. a project leader. A project leader would use this application to establish an overview over what measures are needed for the project, much like a developer, but would not implement them.

Needs

1. The process is comprehensible.
2. The process suggests relevant security standard for the end user, and any relevant measures from this standard.
3. The user do not require former knowledge about these security standards to use the process.
4. The application allows an end user to see a previous run-through of the process.
5. The application allows an end user the ability to jump back and forth between each step in the process.

Indications of needs being satisfied Comprehensible means the process should present the information in a an understandable way. An in-experienced end user should not spend significantly more time than an experienced end user.

As indicated in the conclusion of the previous chapter, some standards are quite complex, meaning it can be difficult to evaluate whether or not it is relevant to a user's project. Making the process give suggestions should save some time for the end user. This also include any potential measures that this standard suggest an end user to implement. This also related to the third need of an end user, which is taken from the last goal of this thesis. Prior knowledge to a security standard should not be needed to complete the process.

The application should allow an end user to either log in if they want to store the results for the future, or simply use it without logging in to make it as fast as possible. Allowing an end user access to previous run-throughs means that if they for some reason have lost the results or are interested in

them because they might be relevant to a current project, it should be easily available if they were logged in while using the application. A potential situation where this can occur is if a project that uses the ASVS standard were deemed as a security level 1 sized project before implementation begun, but during development grew to level 2, a user can simply increase the security level on the previous run-through to retrieve all the additional measures they need for the new security level.

At any point during the process, an end user should be able to jump to a previous step if needed. A possible scenario include if a mistake was made at some point that needs to be changed before progressing further. This change should then be reflected in the results of the succeeding steps.

4.3 Designer

The designer is the creator of the process, and the developer of the application that implements the process.

Needs

1. The designer needs an application that is easy to maintain, which also makes it possible to further develop the application once a prototype has been released.
2. The process has to be designed in such a way that it can be expanded to include more standards

Indications of needs being satisfied The application project has to be maintained with a version control application such as Git[2]. Documentation also has to be in place to allow a new possible developer to read up on the structure of the program if needed. Each step implemented in the process has to be separate to each other, allowing a developer to add new standards and any relevant information or questions surrounding this standard.

Part II

Artifacts

Chapter 5

The process

The process is what defines this entire thesis. Most of the background information have been collected to build a foundation to design the best possible process that meets the goals of this thesis.

This artifact is based on the idea that asking project related questions at an early stage, before implementation, allows developer to more easily produce secure web application. These questions are asked during a step-by-step process, where the end user is asked several questions related to a project, that takes little to no effort to answer. Most of these questions are simple questions that can be answered with "yes", "no" or "uncertain". As the process is aimed to be used early on during the life-cycle of a project, these questions reflect that by not going into specific details, but rather aims to highlight the general features of a project.

One of the key features of the process, which is described as the third goal of this thesis, is that any security related measures or requirements offered by this process, are taken from a security standard. The fourth goal states that this process should also be expandable, by making it possible to add support for more security standards. In order for any security standard to be implemented into the process, they have to meet certain requirements.

5.1 Success criteria of the process

The success criteria of the process are based on the goals of this thesis. These criteria, alongside the needs of the three stakeholders, will help evaluate the success of the artifact.

1. The process provides an end user with standards related to their role within a project.
2. An end user do not require any prior knowledge to any of these security standards.
3. The process is precise enough to provide two different users from the same project, sharing the same role, with close to identical suggestions.
4. The process can be used before any implementation have begun.

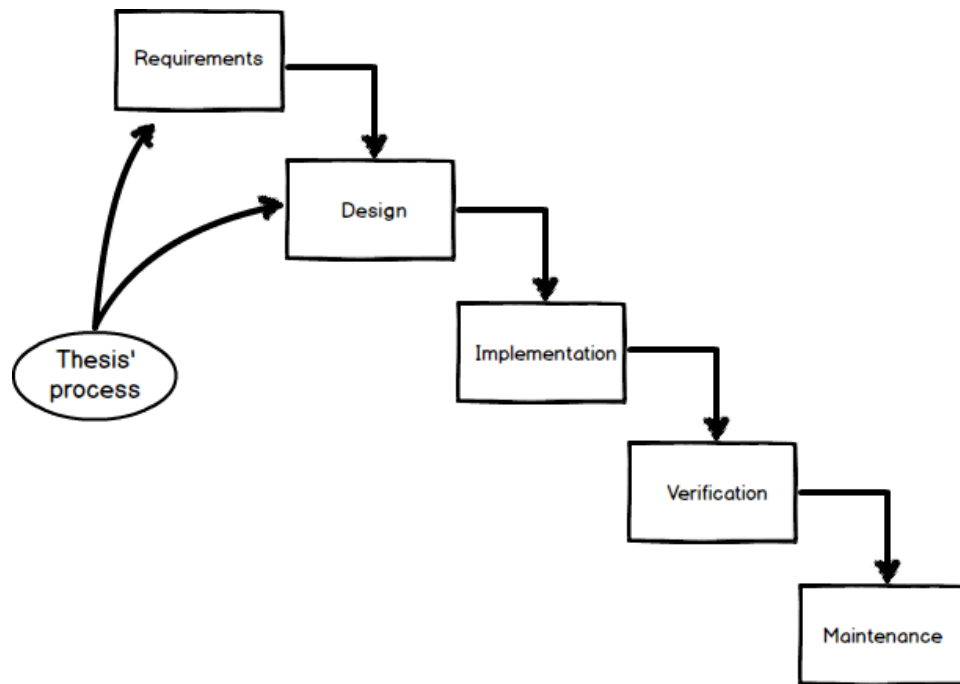


Figure 5.1: Where the process can be used in a waterfall method

The first criteria is similar to the third goal of this thesis, difference being that the process should offer not only project related standards, but standards related to an end user's specific role within a project. These standards should be presented in such a way that the message they convey should be understandable for someone with no previous knowledge or experience with them. This criteria is important because it allows the process to be implemented into a company's practice without requiring their employees to undergo training related to these security standards. Mentioned in section 3.4, Microsoft SDL's first step in the first phase is the core security training which means that all team members have to "attend at least one security training class each year"[34].

In order to assure any users that the suggestions being offered are relevant to their interests, the process should be designed in such a way that two separate team members working on the same project, sharing the same role, is offered the same suggestions. The way in which information is gathered, and the formulation of the questions will play a big part in achieving this. Any confusion from an end user can lead to less relevant suggestions, due to misunderstanding the information, or questions, given by the process.

Section 3.1 mentioned development models, and the figure showed the example of a waterfall model. Using this example we can highlight where the process should be used in a development project. Figure 5.1 contains the same example, with the process included. At some point during the requirements or design, the process of this thesis should be usable.



Figure 5.2: An end user could hold multiple roles within a project, making more standards relevant.

However, this does not mean that the process is useless outside of this scope. If a company decides to use this process at a later stage, that is up to them. This criteria simply defines that it must be usable at an early stage, without excluding possible use later.

5.2 Process description

The following paragraphs will describe what the thought-process and intention behind each step is.

Step 1 - User role step The overall goal of the general step is to establish what kind of security standard might fit the current end user. This is achieved by first asking what type of role this end user holds within the project, allowing the process to more easily determine what type of security standard might fit this end user's needs. A single person could hold multiple roles in a project, and this is taken into consideration during this step. The team leader might work actively as a developer in this project, which would make any development related standards relevant to this person's interest. This interaction is represented in figure 5.2.

Once a role has been established, the process would then ask questions related to that role. A developer's interest might not be the same as a team leader or the project owner. If the end user is a developer, the process would ask if the project is a web application project to help establish if the ASVS standard is relevant to their interests. A team leader might be asked if the company have implemented any form of security related policies. If no, this would make any ISMS related standard relevant, such as the ISO 27001 standard.

Step 2 - Security standard step The second step builds on what the user role step established, and presents the end user with a list of standards that might be of relevance to the project. The second goal of the thesis states that the process should present security measures from a standard that the process itself deemed relevant, and this also matches the second need of an end user. In order to keep the process as comprehensible as possible, only one security standard can be selected during each run-through. This is to prevent each step from being overly complicated, and contain too much

information. Should an end user feel that more than one standard might be of relevance, then the process would have to be completed more than once.

Once the user have either agreed with the suggested standard, or chosen a better alternative, the process begins to map some of the relevant information that is required to continue to the next step. This information would be related to the scale of the project or how critical the data that is being handled in the project is. This helps define the degree of security that is needed, which some standards use to determine the amount of measures that are needed. In relation to ASVS, this information would be used to determine the level of verification described in section 3.3.1.1

ISO 27002 do not directly separate large projects from smaller ones, so in order to best filter out security controls, the designer of the process would have to fully understand the standard, and then add a custom designed level of verification to each of these security controls, much like how ASVS levels work. Due to time constraints of this thesis, the author were not able to implement this in time. A potential solution would be to simply map the security clauses in this step.

For a management related standard, this information might include the number of employees that will be working on this project, any relationships with external sources, or what privileges the employees have related to security e.g. if they have access to any sensitive information.

Step 3 - Project step The third step's goal is to establish all the necessary information related to the project. The process first use the information provided in the security standard step to suggest the degree of security needed for this project, if required by the standard. Once an end user have chosen this, the process asks a series of questions designed to map the necessary information about a project. These questions are specifically designed to fit the chosen security standard, and the amount of questions would change accordingly.

For a project that chose ASVS, this step would be to map the security requirement areas described in section 3.3.1.2. ISO 27002 would use this step to map the security categories found in each security clause that was mapped in the previous step.

Step 4 - Confirmation step The fourth step in the process is the easiest step. Using the answers given in the project step, the process suggest which security controls are relevant to the project, out of all possible security controls in a given standard. An end user can then change this list accordingly, if needed. Each of these controls are described in a way that allows a user with no experience with the given standard, to understand what each security control mean and what they try to cover.

Step 5 - Checklist step The fifth step is where the chosen security standard lists all the measures or requirements for each security control that the end user selected from the confirmation step. These requirements would be of a different nature depending on the standard. Both ASVS and

ISO 27002 contain multiple suggestions for each security control that could be listed in the checklist step. A management standard such as ISO 27001 might not have any direct requirements that could be listed in such a way.

The philosophy behind this step is to provide an end user with the possibility of choosing the most relevant measures for the project, instead of simply completing the process and here and list them all as required. The checklist step is part of the reason why this thesis do not aim to provide certification, as this would be in direct conflict with the intentions of this step. Most standards, like ASVS, requires all measures for a given security control to be implemented to be certified.

Step 6 - Print-screen step The final step is called the print-screen step. This is because the only intention of this step is to provide an overview over the measures or requirements chosen in the checklist step. This overview can then be printed out or print screened if wanted, hence the name.

5.3 Security standard requirements

For the process to meet the goals of this thesis, the success criteria for the process and the needs of the stakeholders, the security standards implemented in this process have to meet a set of requirements.

The first requirement is that the standard cannot be so complex that it would affect the efficiency of the process. One of the needs of an end user is that the process has to be comprehensible, and too much required information might make this difficult. Defining what makes a standard too complex is difficult, and this is something that would have to be decided by the designer of the process. This requirement is what might exclude the ISO 27001 process, as it will probably prove difficult to implement it without having to remove too much information.

The second requirement is that a standard has to provide measures or requirements during the checklist step. For the ISO 27001 standard, this could be the list of potential security clauses/categories that have to be considered, while the 27002 standards could refer to the potential security controls within each of these security clauses/categories.

Outside of these two requirements, any standard that is of relevance to web applications can be implemented. The reason for this is that the process does not necessarily aim to implement a standard word for word, but rather use them as guidelines. The ISO/IEC 27001 is a standard that is aimed at an entire organisation. However, small organisations can also implement the practices found in this standard[33], which means that it can be used for a small development team as well. This development team would then represent a small organisation. Modifications like these only apply to standards which require them, others, like ASVS can be implemented as is, and allows the process to be used to achieve verification through a third party.

Chapter 6

The application

The fourth goal of this thesis states that an application should be developed to implement the process described in the previous chapter. The application is the attempt to accomplish this goal, and serves no other purpose than to present the process in a manner which allows an end user to interact with it. This application implemented the process with support for one standard, namely the ASVS standard. ASVS were chosen due to the relevancy of its contents to the evaluation environment the author had access to. A version of the application that did not make it before the evaluation had partially implemented the ISO 27002 code of practice as a second option available to users.

This chapter will describe the success criteria for the application and thought-process behind the presentation of each step in the application.

6.1 Success criteria of the application

1. The application is available to all major operating systems; Windows, OSX and Linux.
2. The application do not require any formal knowledge of the application itself to be usable.

In order for the application to target a wide audience, it needs to be available on Windows, OSX and most Linux distributions. This reduces the chance that a company might not be able to provide their employees with the application, because it does not support the operating system they use in their development environment.

As the application aims to guide the user through each step in the process, the questions gradually become more detailed. Each of these questions only require information about the project itself, and not the security standard. This means that any member of a project should be able to use the application, provided he or she has the necessary information to answer the questions. If a user is unsure of a certain step or question, there is a button that will inform the user why this specific option was highlighted, or why this question is relevant to the process.

6.2 Process description in the application

The following section described how the application implemented the process, and the thought-process behind each step. The information found here is different from section 5.2 in that it highlights the specific choices made to implement the ASVS standard, and the help provided on each step from the application. Each step in the application were named differently than the six listed in the process description from section 5.2, as the idea to give them a descriptive name were thought of after the evaluation had been completed. The actual names used in the application are given in the parenthesis after the descriptive names.

The descriptive names will be used in the description below, to avoid any possible confusion.

Step 1 - User role step (Initial step) The end user encounters three different panels, each containing a question. The key question in this step is if the project is a web application step, as this notifies the process if the ASVS standard is relevant to this user or not. This also tells the process that ISO 27002 is relevant.

The final panel presents a list of technologies that can be used to develop a web application. The intention behind this step were to offer direct technical solutions on how to best implement the suggested measures the user eventually encounters in the checklist step. This feature did not make into the prototype application. Reasoning for this is found in section 6.3

Each of the panels contain a button that tells the user why the question asked in this panel is relevant to the process.

Step 2 - Security standard step (Step 1) The first panel that greets the end user contain a check box with the ASVS standard already checked if the key question from the user role step was answered with yes. Beneath this option is the ISO/IEC 27002:2013 Standard. However, no further implementation of this standard made it into the application due to time constraints. The following steps therefore only contain information regarding ASVS.

This panel also contains a button that explains why this standard was selected.

The second panel in the security standard step contains a list of various statements, accompanied by a check box. The statements describes various scenarios or information that might be related to the project. The end user can then select as many of these check boxes, and then press the submit button to be taken to the next step. Each of these check boxes have a hidden value of either 1, 2 or 3. These values are then counted, and the value that is most represented is then chosen as the appropriate security level for this project.

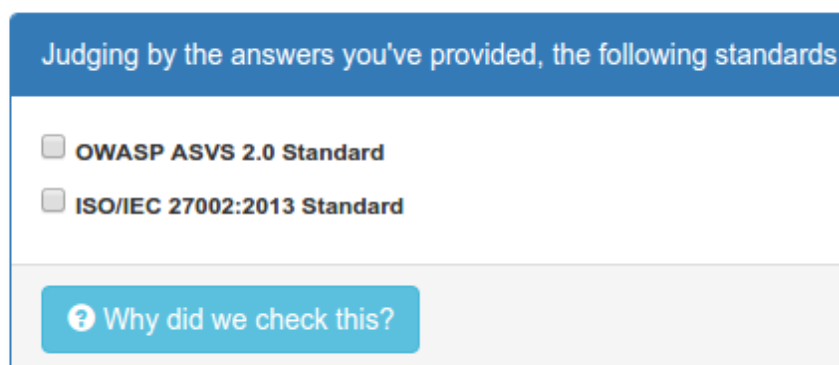


Figure 6.1: Screen capture showing ISO 27002:2013 in an unreleased prototype version of the application

Step 3 - Project step (Step 2) Judging by the statements that were chosen from the security standard step, the application have selected a level of verification that best suits these statements. There is also a button that explains why this level were chosen for this project.

What follows are nine different panels, each containing a single question that relates to a specific security requirement area. These questions can be answered either “yes”, “no” or “uncertain”. “Yes” and “uncertain” are both treated as a yes, difference being how they are represented in the checklist step. Should an end user forget to answer one of these questions, a warning will appear that prevents progression to the next step.

Step 4 - Confirmation step (Step 3) The confirmation step presents the end user with a panel that list all the security requirement areas taken from the ASVS standard. Each of these areas have a related check box. Some of these check boxes have already been selected if the end user answered yes to any of the questions in the project step. The end user can then change the values of these check boxes to checked or unchecked if necessary.

The security requirement areas found in the confirmation step are described in a general way, allowing an end user with little to no experience with the ASVS standard to understand what security controls they cover.

The panel containing this list also contains a button that explains why some of the check boxes were already checked, and a button to submit the data to the next step.

Step 5 - Checklist step (Step 4) The checklist step contains a single panel that informs the end user of the chosen verification level, as well as a list of each security requirement area that were selected in the confirmation step. Each of the elements in this list can be clicked, which expands a drop-down panel containing a new list. This new list contains all the measures

for that specific security requirement area, with the given verification level. Each measure can then be selected for inclusion in the report that will be presented in the final step.

Above these elements is a text that briefly explains how the interactions in this step works.

Step 6 - Print-screen step (Step 5) The print-screen step contains a single panel that allows no interaction outside of reading the contents. This panel simply lists the measures that were chosen in the checklist step, in a simple numbered list. The end user can then use this list in whichever way wanted.

6.3 Features that did not make it into the prototype

Several planned features were either scrapped during development or not finished in time due to various reasons, such as time constraints or technical difficulties. The most notable omission is the log-in functionality that would allow an end user to store any previous run-through. This was tried implemented, and a functional login system was implemented, but the session data took too much time to implement for it to make it into the application before the evaluation. Figure 6.2 shows a screen capture of the state of this feature. A second major feature were an interactive "flow-chart" that would show an end user the progress made, and how the choices made during each step influenced the results, and how a different answer could given a different result. The current prototype simply list why certain elements were selected.

Of the features that did not make it into the application before the evaluation, were some of the help buttons encountered during the process. The "why is this relevant?" buttons encountered during the user role step contains no information at this point. The description of each security requirement area was also not written in time for the evaluation. Both of these omissions had minor implications to the result of the evaluation, which will be described at a later point.

Been here before? Sign in if you have an account!

login

....

Remember me

Sign in

Register

All your previous session data is here.

Previous session data 1.

Restore

Figure 6.2: Screen capture showing the login and restore session data feature

Chapter 7

Architecture for the application

Choosing the right architecture for an application is not an easy decision. Issues like performance, portability, availability, can it be reduced to modules, how easy is it to maintain after release all vary based on the technology used. In order to cover the first success criteria of the application, the choice ended up with developing the application as a web application. This made it easy to distribute, and meant that it would run on any platform that could run a modern web browser, because the technologies used to implement web applications supports availability and portability. Web applications also require no installation or upgrades, which allows an end user to simply focus on using the application. The following sections will highlight the architectural choices made while developing this prototype, why they were made and what alternatives could be used instead, as well as a description of the process that was implemented in the prototype application.

7.1 Architecture and technology

The web application was implemented using PHP, JavaScript, CSS and regular HTML. Each step in the application is designed with a minimalistic look. Buttons, forms, information panels all use a consistent theme to make it easier to understand. If the submit button was blue in step 1, it's blue in step 2. Buttons that provide help or information have their own separate color. The design also appears similar in nature on smaller devices like cell-phones and tablets.

7.1.1 CSS & Twitter Bootstrap

Cascading Style Sheets, better known as CSS is a programming language designed to change the look and formatting of an XML document[17]. It is mostly used in relation to web pages written in HTML. CSS makes it easy to maintain a consistent theme across the web application, which in turn follows the design philosophies described in the previous chapter.

The front-end framework Twitter Bootstrap[5] was chosen to cut down on the time spent on design. Twitter Bootstrap is a framework that consists

of a large amount of CSS files and images. These make up various components like buttons, panels, menus, etc. It is designed to make it easier for web developers to create good looking websites that scale automatically depending on the size of the user's screen. This means that you can use the prototype application on devices such as tablets and cell-phones.

7.1.2 PHP

PHP is a scripting language that is mostly used for web development. One of the reasons PHP was chosen, was due to how it handles HTTP POST and GET methods[18], and the ability to save sessions. These methods are used several times during the course of the web application, for instance when a user submits one of the forms encountered during the process.

This information is used throughout the application to determine what is relevant for this user, and has to be stored. The application first retrieves the data using the GET method, and then save this information in a PHP session object[19]. This session object allows a user to go back to a previous step in the process and make changes if necessary.

The primary reason PHP was chosen, was because it was bundled in the application called WampServer[20]. WampServer is a bundled web development environment that installs Apache2, the server that allows you to test your applications offline, the PHP framework and a MySQL database for you. This allows for effortless testing locally, instead of having to upload to a remote server each time somethings change.

7.1.3 JavaScript

Certain parts of the web application required features that would prove to be overly complicated to solve in PHP. This was easily solved by instead implementing them in JavaScript. Due to the nature of HTML, combining JavaScript code with PHP proved to be easy. The main issue that was solved in JavaScript was to allow the user to click on a security area in the checklist step, which would in turn reveal a panel containing all the measures. At first, it was partially solved using PHP, but the implementation was buggy and half working at best. Manipulating CSS code is an easy matter in JavaScript, which made the solution both work better for the user, and easier to read for the developer.

7.1.4 Alternatives

At first, the plan was to use the same implementation process and technology used at Finn.no. Reasoning being the author worked there during the summer of 2014, and figured it would prove useful for an eventual job application after completing the thesis. The back-end would be written in Java, using Spring to handle web related features such as deployment, dependency injection and RESTful services[21]. Combining this with a tool known as Gradle, meant it would be easy to add any external frameworks if needed[22]. Front-end would consists of mainly

the same technologies as the current prototype, exception being PHP as JavaScript would be enough to handle any front-end related scripting issues. Twitter Bootstrap would also be used to handle design.

The main reason for not choosing this environment was due to several problems that occurred while setting up the development environment. Some of the frameworks did not work properly, and some did not work at all. It was made clear that the effort to fix these problems simply were not worth it, as it felt overly complicated for a rather simple prototype application. The WampServer solution took 2 minutes to install, and could be used immediately.

One of the advantages of using this alternative solution would be for future development. Separating front-end and back-end into two completely different modules means that it is easier for development teams to work on them independently. Module based development also leads to code that is easier to understand, because each module does exactly one job. A random page of the current prototype application contains both back-end and front-end related code, meaning a front-end developer have to find the relevant bits of code manually. Using Gradle also lets each developer add any framework they need, and the other developers could simply update their project to download the requiring files automatically to use this framework.

Chapter 8

Demonstration

This chapter will present an example case that will be used to complete the process, as it is presented in the application. The case that will be used for this demonstration is: the application itself! During this chapter, "Project" will refer this case. The end user in this case will be the author of this thesis, who is also the developer of the application. Some of the features that were eventually scrapped or not finished are included in this demonstration, to provide some additional depth. The demonstration is written in a way that makes fun of some of the bugs that the author did not manage to remove in time for the evaluation.

The version of the application that is used for this demonstration is located at http://fjoggs.com/old_site/master/. The author made a newer version of the application that fixed quite a log of the bugs encountered during this demonstration. The reasoning behind this was to provide the same version of the application as the evaluation respondents were using during the evaluation.

8.1 Project description

This section contains all the required information about the project that is required to complete the process.

Project role The end user of this demonstration is a developer.

Description The development project is a web application project that plan to implement the following features:

- Storing data in session objects.
- Allowing users to log in and out of the application.
- Usable on mobile devices.
- Implement a REST service that provides the application with information from the security standards.

The technologies that will be used to implement this is:

- Java 8
- Hibernate
- Spring
- Javascript
- Groovy
- Gradle

The project contains no sensitive data of any sort, apart from usernames and passwords that are created by the user when they want to register, allowing them to store their run-throughs.

8.2 Run-through

We enter the URL given early, and are greeted by some text and a big blue "improve my project". Our aim here is to indeed improve our project, so we step right in. This takes us to the first step. For the sake of this thesis, the names used in this demonstration corresponds with the names given in chapter 5, and not the ones used in the prototype application.

Step 1 - User role step The first question that's asked of us is what role we have in our project. According to the description, we are a young and up-and-coming developer, so we naturally select this option. We try to click the "why is this relevant?" button, but nothing happens.

The next question is if the project we are working on is a web application project. The given description does say that this is a web application, so we say yes. Much to our disappointment, the button that should tell us why this is relevant is not working.

The following panel lists a number of technologies, and asks if any of these will be used to implement the project. Out of the technologies listed in the description, we select the ones that are supported, and press submit.

Step 2 - Security standard step The first things that greets us in this step is a checkbox with the label OWASP ASVS 2.0 Standard. This checkbox is already checked off, and we press the "why did we check this button?" to see why.

We un-check the checkbox to see what happens, which hides a panel that contains a number of various statements that tries to describe our project. We re-check the checkbox to make it appear again.

Out of all the various descriptions given, only the first two sounds relevant to this project. There are no data outside of log in information that could be categorized as sensitive. The examples given does sounds more critical though, which makes the second description slightly confusing. Onto the next step!

The image shows a user interface for a survey question. At the top, a blue header bar contains the text "Is your project a Web Application?". Below this, there are three radio button options: "Yes" (which is selected), "No", and "Uncertain". At the bottom of the form, there is a light blue button with a question mark icon and the text "Why is this relevant?".

Figure 8.1: Screen capture of the web application question.

Step 3 - Project step The next step, we're greeted by a message that tells us that the application have already selected an option for us. However, no choice has been made. Undiscovered bug in the programming, perhaps? Either way, we choose the option that sounds the closest to the description of our project.

What follows are a series of questions that asks for details about the project. Listed below are each of these questions and the answer given:

- Will your project use some form of authentication?
 - Yes! The description states that users will be allowed to log in.
- Will your project store cookies at some point during the application's life cycle?
 - No. The project will use session objects instead of cookies to save data.
- Will some features only be available to certain users?
 - No.
- Will your project use any kind of forms or user input fields?
 - Yes! The project will asks their users a series of questions, which needs forms to be stored in sessions.
- Will your project use RESTful services?
 - Yes! The description states that all standard related information will be fetched via a REST service.

Will any of the following technologies be used for your project?

- Javascript
- jQuery
- PHP
- Java 6
- Java 7
- Java 8
- JSP
- Python 2.X
- Python 3.X
- MySQL
- PostgreSQL
- C#
- .NET

[Submit](#) [Reset](#) [? Why is this relevant?](#)

Figure 8.2: Screen capture of the list of technologies.

Judging by the answers you've provided, the following standards have been deemed relevant:

- OWASP ASVS 2.0 Standard

[? Why did we check this?](#)

Because you chose YES to our question if the application is a web-application.

Figure 8.3: Screen capture of a relevant standard, and why it was selected.

Which of the following descriptions match your project the most?

- A simple internet-accessible application.**
- Contains sensitive information like Credit Card numbers or personal information.**

Figure 8.4: Screen capture of the project descriptions.

Judging by the answers you've provided, the follow

- Level 1 - A simple internet-accessible application.**
- Level 2 - Web application that will store various pers**
- Level 3 - Project containing large amounts of sensiti**

[? Why did we check this?](#)

Figure 8.5: Screen capture of the chosen level.

The screenshot shows a confirmation step with three checkboxes and three buttons. The checkboxes are: **Malicious Controls**, **Business Logic**, and **File and Resource**. Below the checkboxes are three buttons: a blue **Submit** button, a white **Reset** button, and a light blue button with a question mark icon and the text **Why did we check this?**

Figure 8.6: Screen capture of some of the ASVS areas in the confirmation step.

- Should users be sent to custom error messages if something goes wrong?
 - Uncertain. The project description states no such thing, but implementing error messages is good practice to help informing users what went wrong.
- Will your project allow users to communicate with each other internally?
 - No.
- Will users perform any forms of transactions like payments on your site?
 - No.
- Are you planning to support mobile devices?
 - Yes!

We press the submit button after we're done answering the questions.

Step 4 - Confirmation step The confirmation steps reveals a panel that contains a number of different checkboxes, that are either checked or not checked. These checkboxes represents areas, according to text provided. The five checkboxes that are already checked read "Authentication", "Malicious Input Handling", "Cryptography at Rest", "Error Handling and Logging" and "Mobile". These all match the description of our project, but we feel that maybe the "Malicious Control" area should be included as well, so we check it.



Figure 8.7: Screen capture of the “include in report” feature of step 5.

Step 5 - Checklist step The chosen measures are represented in full in Appendix C.

The application tell us that the following measures can be taken for our chosen security level, which was level 1. We click on authentication to see the measures for this area. The project description does not contain any specific information on how the various features will be implemented, so we choose all the measures as they seem relevant to this project.

There are no information about a database in the project description, which means that the measures that handle SQL injection in the malicious input handling area are not needed. Some of the various injections presented handle topics that we are not familiar with. We choose therefore to include them as we do not have the knowledge required to potentially exclude them.

For the Cryptography at Rest area, nothing happens when we click the area. We assume this means that no measures have to be taken for this level. Error handling and logging have one measures that we include in the final report.

The only measure we do not include from the mobile area is related to storing data in an SQLite database.

The final area is the malicious controls. This only show one measure which reads that “No measures needed for your chosen security level”. This message did not show up for the cryptography at rest, which means that maybe there was a second undiscovered bug in the application. The

| Malicious Controls | |
|--------------------|--|
| # | Measure |
| | Include in report? |
| 1 | No measures needed for your chosen security level. <input checked="" type="checkbox"/> |

Figure 8.8: Screen capture of the “no measures” needed message.

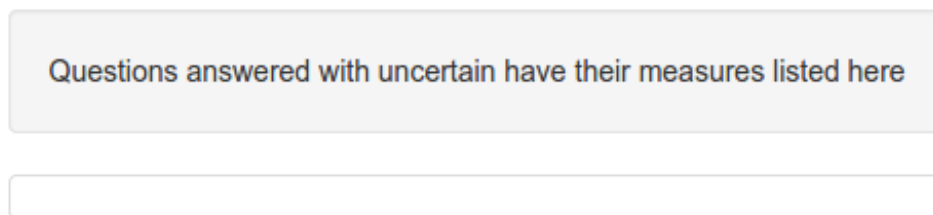


Figure 8.9: Screen capture showing how the uncertain area was empty.

malicious control measure can still be included in the report, even though it is not a measure. We chose to do so as we are rebels like that.

We note that even though we chose uncertain to whether or not we wanted error handling, this area did not show up in the uncertain measures area, but using the button that allows us to read the stored session data, we see that it is stored in the uncertain array. A possible slip up in the coding included it in the wrong area.

The amount of measures we selected totals to 15, if we exclude the joke include of the no measures available.

Step 6 - Print-screen step We counted the amount of measures, because the print-screen step shows four elements that contains no values, and that the total actual measures presented to us counts 13, which means that two are missing. We notice that the ones that are missing are the first measure of each area, namely:

1. Verify all pages and resources require authentication except those specifically intended to be public (Principle of complete mediation).
 - (a) From authentication.
2. Verify that the runtime environment is not susceptible to buffer overflows, or that security controls prevent buffer overflows.
 - (a) From malicious input handling.
3. Verify that the application does not output error messages or stack traces containing sensitive data that could assist an attacker,

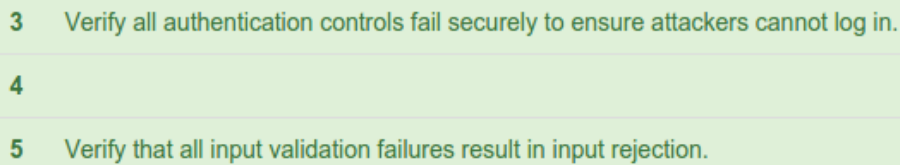
- 
- 3 Verify all authentication controls fail securely to ensure attackers cannot log in.
 - 4
 - 5 Verify that all input validation failures result in input rejection.

Figure 8.10: Screen capture showing that some measures were missing.

- 
- 16 Verify that sensitive data is not stored in SQLite database on the device.

Figure 8.11: Screen capture showing a measure that should not be included in the report.

including session id and personal information.

- (a) From error handling and logging.
4. Verify that the client validates SSL certificates.
 - (a) From mobile.

What is also interesting is that both measures that were not selected to be included in the report showed up anyway, which means that the actual number of missing measures is four, which matches the four given above.

It's clear that the logic that handled the inclusion is slightly off, and that there is a good chance that the loop that handles the printing starts at the wrong index. This is because the first element of each area were not included. Why the measures that were not selected showed up anyway is a different matter.

8.3 Conclusion

Throughout the application, several bugs were present that had some influence on the outcome of this run-through. However, they did not prevent the user from completing the process itself. The ramifications of these bugs will be explained in further detail in chapter 12, and the influence they might have had on the results of the evaluation.

What this demonstration did show was that the some of the statements presented in the security standard step (step 2) related to the scale and size of the project were obviously written for more complex projects than this, which made it harder to choose the correct statements.

Part III

Evaluation

Chapter 9

Evaluation plan

The goal for the evaluation is to provide information that will help answer the following questions:

- Have the goal of the problem statement presented in chapter 2 been met?
- Have the success criteria for each artifact been achieved?
- Have the needs of each stakeholder been met?
- Have the goals of this thesis been accomplished?

This chapter will describe how the evaluation will try to answer these questions.

Both artifacts are tested at the same time, using a prototype of the application described in chapter 6. Once a respondent have completed the run-through of the prototype application, a questionnaire is given out to the respondent that has to be filled out. This is then followed by an oral discussion among all the respondents and the observer.

The questionnaire used for the evaluation is provided in Appendix A, Evaluation templates.

9.1 Who is the target audience

The target audience for both artifacts is a developer, or a team leader of developers. They share the same interests in the development of the web application. The developer is interested in a time efficient process, and a security standard that have easily understandable measures. The team leader uses the application as if he was a developer, to know what his team is working with and being able to assist them if needed. A team leader can also be a developer, as noted in the process description.

9.2 What evaluation strategy is used

The evaluation consists of a field study, held at a local development company. More information about the environment for the field study is

listed in chapter 10.

The field study will, with the help of a questionnaire, provide information about the success of the process and the application. This section will split the information into their own sub-tests, defined by a question. Each of these tests were all conducted during the same run-through, and are used simply to separate the results into relevant categories.

9.3 Evaluation description

Each of these cases will be referred to as their own separate sections in the evaluation results. The question each case asks do not refer to a specific question in the questionnaire, but rather an overall question that can be answered by combining some of the questions from the questionnaire.

Case 1 - Did the respondents spend enough time on security before deployment? Related to the first goal of this thesis. The goal is to determine, as said, if the company used for this evaluation spent enough time on security before they deployed their project. "Enough time" will be defined based on what the respondent answers.

Case 2 - Could this process be used before development begun to prevent some of the issues they might have faced? Related to the second goal of this thesis.

Case 3 - Did the process provide any security standards relevant to the project? Related to the first success criteria for the process.

Case 4 - Did the security standard provide any measures relevant to the project? Related to the third goal of this thesis, and the second need of the end user.

Case 5 - Did this process make it easier to use security standards? Related to the sixth goal of this thesis.

Case 6 - Could this process be used in your company? Related to the first and third need of the company.

Case 7 - Did the lack of experience with the application hinder the respondent in any way? Related to the second success criteria of the application, the third need of the company and the fifth goal of this thesis.

Case 8 - Did the lack of experience with the security standard hinder the respondent in any way? Related to the first and third need of the end user, and the second success criteria of the process.

9.4 Evaluation procedure overview

What follows is a short description of how the evaluation was done, written in the perspective of the author, hereby referred to as the observer, as the one doing the evaluation.

After a short round of introductions, the observer starts off with a presentation. This presentation talks about the master thesis, followed by the ASVS standard and how it relates to the security issues the company of the respondents might face. Lastly, the presentation talks about how the evaluation will be completed, and what happens in the weeks that follow.

After the presentation is completed, the respondents, with the help of the observer, agree on a project they can use for this evaluation. Once an agreement have been met, each respondent is presented with the URL that the prototype application resides on, and they start the process. Once each respondent have completed this process, they receive a questionnaire from the observer that they fill out with no communication between the respondents or the observer.

When they are finished filling out the questionnaire, a short oral discussion follows, related to their experience with the process and the application.

Chapter 10

Evaluation environment

The evaluation were conducted at a major software development company called Finn.no, residing in Oslo, Norway. According to their web-pages, every Norwegian spends on average 21 hours each year using their services[23]. FINN.no's main service is what they call "torget", which translates to "the marketplace". This is a service where users can, amongst other things, buy and sell items they no longer have need for. The marketplace is in short a localised version of eBay[24], where both private customers and companies can sell their merchandise at a fixed price, or to the highest bidder.

FINN.no's services operate on the web, through various web applications that handle all the logistics that is needed to run a web site of this magnitude. Due to sheer size of their user base and the amount of sensitive and critical data that a marketplace contain, FINN.no is a good place to conduct an evaluation of the artifacts presented in this thesis.

10.1 Respondents

The 4 respondents that all work for the same development team at FINN.no, with different roles within this team.

Respondent 1 Respondent number 1 is a current lead developer at FINN.no, and holds a degree of bachelor in informatics. Respondent 1 has 19 years of relevant experience, but have no prior knowledge of security standards, including ASVS, and seldom works with security related issues in general.

Respondent 2 Number 2 is a senior developer at FINN.no with 15 years of relevant experience, and graduated with a master of science. Unlike the first respondent, respondent 2 works with security related issues, and have some experience with security standards. Even though he has experience with OWASP, respondent number 2 have no prior knowledge of the ASVS security standard.

Respondent 3 The third respondent is the current team leader, who previously worked as a developer for the same team. Graduated with a master of science in systems engineering, and have 15 years of relevant experience. Respondent number 3 do not work with security related issues, and have no experience with either ASVS or any security standard in general.

Respondent 4 The last respondent is a developer, who has built up 10 years of relevant experience after graduating with a master of science in IT. Number 4 rarely spends time working on security related issues, and have no prior knowledge of any security standard, including ASVS.

10.2 Case description

The project that is used for evaluation of the process and the prototype application, is a project all four respondents worked on at a previous time. This project was a web application development project that implemented a search engine for airline travels. An end user of this service would submit information about the travel, such as where and when. This data would then be forwarded to the travel agencies in real time, which would then send relevant information to the user. This was achieved by collecting flight data from almost 30 different suppliers, and then represent this data to an end user in a manner which made it understandable. This information could then be filtered, based on a number of different parameters, such as location or price. The user can then decide which of the offers represented is the most desirable, and click a link that leads to the third party supplier where the user could buy this ticket.

Chapter 11

Evaluation results

The following chapter highlights the results of evaluation in relation to each artifact. The sections in this chapter attempts to provide the information, related to the goals of this thesis. These results are taken from the observations made during the evaluation, and the answers from the questionnaire given by the respondents. Most of the data provided is taken directly from the answers of the respondents.

11.1 Case 1

Did the respondents spend enough time on security before deployment?

A common theme among the respondents was that they either spent no time, or close to no time on security related issues before development begun on the project. This was also the case during development. The main reason for this is that Finn.no usually dealt with security related issues by hiring external advisers that would test the application for any threats, and then potentially solve them. This means that the developers had no responsibility in relation to security. However, each respondent have years of experience with development in general, and they have acquired knowledge regarding how to handle certain situations. Most noticeably is any database related work that handle SQL statements would use some form of prepared statement to prevent SQL injection.

All four respondents mentioned in the questionnaire that the process suggested measures that they did not think about, and they did experience security related issues after development.

11.2 Case 2

Could this process be used before development begun to prevent some of the issues they might have faced? All four respondents reported that the process suggested measures that they did not previously think about during development. One respondent specifically mentioned that they had problem related to cross-site scripting (XSS)[12], and that the process provided a suggested measure to take to solve this problem. The same

respondent also noted that the security measures related to mobile phones would be useful during development of the project.

The process did not however, present the one of the respondents with any measures related to a DOS-attack or SQL injection, which they experienced after deployment. A second respondent however, noted that the process did recommend measures against SQL injection.

This is interesting, as shown during the demonstration, the security requirement area related to Malicious input handling contains a measure that is related to SQL injection, as well as the Mobile security requirement area.

11.3 Case 3

Did the process provide any security standards relevant to the project?

The prototype application, as mentioned earlier did only include one standard. The author knew that this standard would suit some of the projects the respondents have completed, as he spent a summer as an intern working with web application related development. Neither of the respondents had any experience with ASVS.

11.4 Case 4

Did the security standard provide any measures relevant to the project?

The respondents wrote that the process recommended measures that they already had implemented, and a few noted that it also recommended some measures that they did not yet implement or think about.

11.5 Case 5

Did this process make it easier to use security standards? All four respondents had no previous experience with security standards, but managed to complete the process mostly without problems.

11.6 Case 6

Could this process be used in your company? Most of the respondents noted that the process and application presented to them during the evaluation could most probably be used in the company, but one of the respondents suggested that if not the entire company then maybe allow each development team decide on whether or not to use them.

11.7 Case 7

Did the lack of experience with the application hinder the respondent in any way? The application did not provide enough help for in-

experienced users for them to fully understand each question, but they did complete the process without help. This would mean that some of the security requirement areas presented to the user might not be relevant, or that the security level is wrong because they did not completely understand the question or the statements given. During the oral discussion, it was noted that the list of security requirement areas during the confirmation step were too technical in nature, and should instead be a general description of what each covered. This was a problem noticed even before evaluation, but it was sadly not implemented in time.

The prototype application did not have any help buttons implemented, and it is clear that this is a feature that would be required if the application is to be used in a development company. This is something that was noted during the demonstration as well.

11.8 Case 8

Did the lack of experience with the security standard hinder the respondent in any way? Two of the respondents had some problems understanding all of the questions. Reasoning was that they had little experience with security and security standards, and that there was a lack of explanation. One respondent reported that some information about why these measures will be effective and against what, could help explain them more easily.

Chapter 12

Reliability & validity of evaluation

The following sub-sections are some of the most important sections in this thesis. Recognizing any potential factors that could compromise the evaluation is the key to an accurate conclusion. Random events could interfere with the evaluation in a way that would lead to a different result, had there been a re-evaluation. One of the main factors in potential threats to the evaluation is the fact that the evaluation is only performed once, several months before the completion of the thesis. The problems that this cause will be highlighted in the following sections

12.1 Random elements & non-random elements

Determining what factors were coincidences, and which were not makes it easier to confirm or deny any of the points made in the previous 2 sections. Too many random elements weakens the reliability of the evaluation, as it means there is a good chance a re-evaluation would provide a completely different result.

The following factors during this evaluation could be deemed random/coincidental:

- Performance related to prototype application.
- Respondents, see section 12.2 for further elaboration.

When developing an application on any level, unforeseen elements and bugs always happens. There has been no form of QA (quality assurance) work done on the prototype application or process before this evaluation. This means that bugs might occur during the evaluation, which could affect the outcome of this evaluation. The best way to minimize the outcome of these situations are simply to conduct more evaluations. Sadly there were no time for a second evaluation.

The following factors are deemed not random/not coincidental:

- How easy to use the prototype application is.

- Relevance of the ASVS security standard.
- Evaluation environment, see section 12.2 for further elaboration.
- Relevance of questionnaire, see section 12.3 for further elaboration.

12.2 Evaluation environment & respondents

Biggest factor to the results of an evaluation is the environment in which it was performed. The previous section deemed this a non coincidental factor to the evaluation. This is mainly due to the fact that the observer is the one engaging a company for a possible evaluation, meaning there is a degree of control. What is coincidental, is the respondents that a company allows to participate in the evaluation. These respondents may have previous experience in the relevant fields, or no experience at all. This can have ramifications on the answers given in the questionnaire.

In this specific evaluation, the observer contacted one of the eventual respondents directly, and not the company itself. This meant that at least one of the respondents would be decided beforehand, and the rest would most likely be co-workers who had worked with this respondent. The random factor in this specific case would be what role each respondent had in the project that was used for the evaluation.

The company that participated in the evaluation were a good target for this thesis, because in this specific evaluation case they did not spend any time on security before implementing. If the evaluation had been performed at a company where planning and designing secure web applications from the ground up were common practice, then the immediate success the artifacts presented in this thesis would seem doubtful at best. It is important to recognise these factors before concluding anything, and use them to highlight that any form of evaluation is subject to variance.

12.3 Questionnaire & case correlation

One of the issues that arose after evaluation was related to the questionnaire and their correlation with the success criteria for each artifact. One of the artifacts, namely the first artifact, were conceived after evaluation already had taken place, meaning there were no direct questions related to this artifact in the evaluation. The success of this artifact therefore had to be interpreted from the questions that were aimed to answer the other artifacts. As mentioned in section 7.7, the evaluation were performed several months before the completion of this thesis. The success criteria for the artifacts saw some slight re-iterations during this time, meaning that some of the questions from the questionnaire might have lost some of its accuracy related to these criteria. These changes were kept to a minimum though, as the evaluation would prove pointless, had the criteria changed drastically.

12.4 Would a re-evaluation provide similar results

Due to the simplicity of the prototype application, a re-evaluation would most likely end up with close to similar results. This assumes that the evaluation case and environment would remain the same. Much of the success related to the standard were the revelation that all four respondents noted that the application presented them with measures they had not previously thought about. Had they used a different project where this had not happened, it would definitely affect the outcome of the evaluation results. This indicates that ideally there should have been more than one evaluation to strengthen the conclusion of this thesis. This section have to be taken with a grain of salt, as it is merely speculation. Judging whether or not a re-evaluation would prove the same results is next to impossible, as there are things you do not think about that could happen. Murphy's law springs to mind.

12.5 Strength of ASVS implementation

The implementation of ASVS in the process can affect the outcome of the evaluation. Some of the descriptions listed in the second step to determine the level of verification, are poorly written, which could lead to users either using a level higher than they should, or worse, a level below. Combine this with the questions asked in the step after, and you could end up missing some measures. The only way to prevent this would be to conduct more evaluations that could help define these questions and descriptions better.

12.6 Shortcomings of ASVS

ASVS in itself might also not be best suited to demonstrate the power of the approach the process in this thesis attempts. The best way to determine this would be to conduct more evaluations, with different respondents and a different environment.

12.7 Conclusion

The author deems the most significant factor in the outcome of this thesis is the low number of respondents that participated in the evaluation. Four is an incredibly small amount if you relate it to empirical evidence that uses statistics to prove or disprove a statement. Even with a large sample size, an evaluation such as this will always be subject to the quality of the respondents, and when the number is as low as four, the outcome of the evaluation depends largely on chance. Any conclusions made after these results are therefore closer to a guideline or as a map of the general response, and not an absolute conclusion.

Part IV
Conclusion

Chapter 13

Summary & conclusion

This thesis has tried to supply some background information on why security is important, and how to use what already exists to improve security. The artifacts presented in part II tried to achieve this by creating a process that would provide an end user with measures taken from a relevant security standard.

The problem statement presented in chapter 2 helped shaped the goals of this thesis. The following section will attempt to deconstruct these goals into smaller goals, and verify to what degree they have been accomplished,.

13.1 Discussion of the evaluation results

Evaluate whether or not development companies spend enough time on security before deploying a web application to the public The evaluation results revealed that the company and respondents used in the evaluation did not spend any time on security before they begun developing the project. As noted, this was partially due to how that company uses external advisers to help with security. However, they did experience security related issues after deployment, which might indicate that these advisers either did not do a sufficient job, or the issues were outside of their domain.

The background information also revealed how Sony would in some cases rush out products to the public, which in turn was one of the reasons why the 2011 Sony security breach caused so much damage.

Drawing a conclusion for an entire industry based on two different scenarios is foolish, but what these scenarios show is that some development companies do not spend enough time to prevent their services from being affected by security related issues. For the company used in the evaluation, they experienced DOS attacks, while the attack on Sony resulted in 77 million compromised user accounts.

Design a process that can be used early in the planning stages of a development project In order to determine the success of this goal, we list the success criteria for the process taken from section 5.1:

1. The process provides an end user with standards related to their role within a project.

Mentioned in the evaluation result was that the prototype application used for the evaluation did only offer one security standard, and that this standard was chosen due to the relevancy with the evaluation environment. The outcome of this criteria can therefore not be determined.

2. An end user do not require any prior knowledge to any of these security standards.

The current process implemented the measures and security requirement areas of the ASVS standard as they were written in the standard. The questions that lead to the process suggestions were designed by the author to make it as easy as possible to end up with the “correct” results. What occurred in evaluation case 2 was that one of the respondents did not get any SQL injection related measures, while a different respondent did. This might suggest that some of the questions asked during the process did not describe the related security requirement area good enough

3. The process is precise enough to provide two different users from a project, sharing the same role, with close to identical suggestions.

The previous criteria noted how this did in fact not happen, and that process were not precise enough to avoid the situation that happened during the evaluation.

4. The process can be used before any implementation has begun.

The project that were used for the evaluation, were an already completed project. Drawing any conclusions based off this related to how early the process can be used would therefore be pointless. However, what the evaluation showed was that the standard presented the respondents with measures that they could have used during implementation. This indicates that the process can be used at a later stage as well.

Chapter 8 showed a demonstration of the process, using the project description of the application this thesis produced itself. This demonstration showed how you could complete the process without having to know specifics about how the project will be implemented. The user role step allowed an end user to list the technologies that might be used to implement the project, but it was not mandatory to progress to the next step.

Provide security measures from a standard deemed relevant by the process This goal was met during the evaluation, as all four respondents experienced that the process provided measures that were relevant to their project. As long as a project that attempt to use the process and prototype application is a web application, the ASVS standard should provide some form of measures that would relate to that project.

Allow the process to be expanded upon, providing more coverage

The process chapter attempted to highlight the possibilities to allow for expansion of other security standards, and not just development standards. How successful this implementation would be is hard to judge without actually doing it.

Develop an application that implements this process in an efficient manner

The time spent completing the process for the project described in section 10.2 summed up to around 30 minutes. While this does not necessarily mean that the application is efficient, it shows that for a project of moderate size, the process can be completed in a reasonable amount of time. Noted in the evaluation case 7 was that the lack of help on certain steps lead to some confusion.

In addition to this goal, are the success criteria related to the application:

1. The application is available to all major operating systems; Windows, OSX and Linux.

This was achieved by implementing the application as a web application, which allowed it to be used on most web browsers.

2. The application do not require any formal knowledge of the application itself to be usable.

Mentioned earlier was that implementing the help buttons should improve the user experience for users that have not used the application before.

Reduce the complexity of using security standards in development

As mentioned earlier, drawing a conclusion to a goal like this from a single evaluation is unwise. However, the evaluation revealed that not a single respondent had any experience with security standards, but they were provided with some measures that they agreed on and understood.

What this process does is that it tries to exclude all the non relevant information from a security standard, providing only the information that is related to this end users interests and the projects interests. What the questionnaire from the evaluation showed was that the idea behind the process had promise, and if expanded upon could be used in a development company today.

13.2 Conclusion of this thesis

The general consensus of the previous section is that both the process and the application show promise, but are in need of improvement. Many of the goals and criteria were met either fully, or to some degree, but under circumstances that does not allow to use that as a conclusion, but rather as an indication. Chapter 12 attempt to highlight what factors can affect the outcome of this conclusion.

The goals the author consider to be met fully are:

- Goal 2: Design a process that can be used early in the planning stages of a development project.
- Goal 3: Provide security measures from a standard deemed relevant by the process.
- Goal 4: Allow the process to be expanded upon, providing more coverage.

Goals met to some degree:

- Goal 1: Evaluate whether or not development companies spend enough time on security before deploying a web application to the public.
- Goal 5: Develop an application that implements this process in an efficient manner.
- Goal 6: Reduce the complexity of using security standards in development.

The standard used for the evaluation, ASVS, were not complicated enough to reveal if this type of process could simplify it. A standard such as ISO 27001 would be a better fit to draw any conclusions for this.

One of the possible additions to improve the results of the evaluation would be to expanding the amount of standards supported in the process would better highlight how well the step-by-step process simplify the use of a security standard in web application development.

The problem statement said that the process created by this thesis should help to prevent security breaches or minimise the damage they could inflict. The author of this thesis recognise that any conclusion based on the small amount of empirical data this thesis produced should be taken with a grain of salt. With this in mind, the discussion and results presented in previous section indicates that this statement has been fulfilled to some degree, **IF** you assume that the security standards implemented in this process can accomplish these goals.

All in all, the author of this thesis is pleased with the direction and results this thesis have achieved. He is not pleased with the low amount of evaluation that the application and process received, but things do not always go as planned. The idea of simplifying a security standard to make it easier for people to use them is an idea that have been received by others as highly interesting. The author himself believe that this idea could be very successful if it was developed by a serious development company.

13.3 Future work

Mentioned several times through the course of this thesis is the need for more than one evaluation. This would make it easier to evaluate whether or not the three goals that were not fully met, actually were met or not.

A second standard would have to be implemented into the process before a re-evaluation, such as the ISO 27002 standard. Work on this had already begun before the conclusion of this thesis, however, there were simply not enough time to complete it.

In order to meet all the needs of the stakeholders, the application would have to be further developed. The login feature that were not completed in time, and the restore session data are the most pressing.

Appendix A

Evaluation templates

A.1 Questionnaire for the respondents

1. Where do you work?
2. What is your current role within the company?
3. What is your academic degree?
4. Years of relevant experience (Software development, security, standards)?
5. Do you work with security related issues?
6. Do you have any experience with security standards?
7. If yes, which ones?
8. Do you have any experience with the ASVS Security Standard?
9. Write a short description of the project used in this evaluation:
10. How much time was spent on security related issues?
11. Any security issues after deployment?
12. If yes, did the application present you with any measures that you could have taken to prevent it?
13. What was/is your role for this project?
14. Based on your experience with this particular project:
 - (a) Did this prototype recommend security measures that you previously did not think about?
 - (b) Did you miss any recommendations of any security measures that you find relevant?
 - (c) Which of the recommended security measures did you identify in your project at a later stage?

(d) Were the questions you encountered in step 2 relevant to this project?

15. Is implementing web applications according to a security standard something you would consider for your future development projects?
16. Is this application something you think the company you work for would use?
17. What are the main benefits of this approach?
18. What were the main challenges you faced while using this approach?
19. Any further comments?

Appendix B

Prototype application availability

The prototype application used for both the demonstration and the evaluation is found at http://fjoggs.com/old_site/master.

Appendix C

Measures included in the report for the demonstration

C.1 Authentication

1. Verify all pages and resources require authentication except those specifically intended to be public (Principle of complete mediation).
 - (a) **Included** in report.
2. Verify all password fields do not echo the user's password when it is entered.
 - (a) **Included** in report.
3. Verify all authentication controls are enforced on the server side.
 - (a) **Included** in report.
4. Verify all authentication controls fail securely to ensure attackers cannot log in.
 - (a) **Included** in report.

C.2 Malicious input handling

1. Verify that the runtime environment is not susceptible to buffer overflows, or that security controls prevent buffer overflows.
 - (a) **Included** in report.
2. Verify that all input validation failures result in input rejection.
 - (a) **Included** in report.
3. Verify that all input validation or encoding routines are performed and enforced on the server side.
 - (a) **Included** in report.

4. Verify that the runtime environment is not susceptible to SQL Injection, or that security controls prevent SQL Injection.
 - (a) Excluded from report.
5. Verify that the runtime environment is not susceptible to LDAP Injection, or that security controls prevent LDAP Injection.
 - (a) **Included** in report.
6. Verify that the runtime environment is not susceptible to OS Command Injection, or that security controls prevent OS Command Injection.
 - (a) **Included** in report.
7. Verify that the runtime environment is not susceptible to XML External Entity attacks or that security controls prevents XML External Entity attacks.
 - (a) **Included** in report.
8. Verify that the runtime environment is not susceptible to XML Injections or that security controls prevents XML Injections.
 - (a) **Included** in report.
9. Verify that all untrusted data that are output to HTML (including HTML elements, HTML attributes, JavaScript data values, CSS blocks, and URI attributes) are properly escaped for the applicable context.
 - (a) **Included** in report.

C.3 Cryptography at REST

No measures for this security level.

C.4 Error handling and logging

1. Verify that the application does not output error messages or stack traces containing sensitive data that could assist an attacker, including session id and personal information.
 - (a) **Included** in report.

C.5 Mobile

1. Verify that the client validates SSL certificates.
 - (a) **Included** in report.
2. Verify that unique device ID (UDID) values are not used as security controls.
 - (a) **Included** in report.
3. Verify that the mobile app does not store sensitive data onto shared resources on the device (e.g. SD card or shared folders).
 - (a) **Included** in report.
4. Verify that sensitive data is not stored in SQLite database on the device.
 - (a) Excluded from the report.

C.6 Malicious controls

No measures for this security level.

Bibliography

- [1] SearchSoftwareQuality,. (2015). What is Web application (Web app)? - Definition from WhatIs.com. Retrieved 11 May 2015, from <http://searchsoftwarequality.techtarget.com/definition/Web-application-Web-app>
- [2] Git-scm.com,. (2015). Git. Retrieved 7 May 2015, from <http://git-scm.com/>
- [3] Steinberg, J. (2014). Massive Security Breach At Sony – Here’s What You Need To Know. Forbes. Retrieved 14 April 2015, from <http://www.forbes.com/sites/josephsteinberg/2014/12/11/massive-security-breach-at-sony-heres-what-you-need-to-know/>
- [4] Reuters,. (2015). Sony PlayStation suffers massive data breach. Retrieved 5 May 2015, from <http://www.reuters.com/article/2011/04/26/us-sony-stoldendata-idUSTRE73P6WB20110426>
- [5] Getbootstrap.com,. (2015). Bootstrap. Retrieved 16 April 2015, from <http://getbootstrap.com/2.3.2/>
- [6] Application Security Verification Standard (2014). (2014) (2nd ed., p. 5). Retrieved from https://www.owasp.org/images/5/58/OWASP_ASVS_Version_2.pdf
- [7] Microsoft.com,. (2015). Microsoft Security Development Lifecycle. Retrieved 22 April 2015, from <https://www.microsoft.com/en-us/sdl/default.aspx>
- [8] searchSecurity.in,. (2015). What is information security management system (ISMS)? - Definition from WhatIs.com. Retrieved 7 May 2015, from <http://searchsecurity.techtarget.in/definition/information-security-management-system-ISMS>
- [9] Wikipedia,. (2015). Secure by design. Retrieved 22 April 2015, from http://en.wikipedia.org/wiki/Secure_by_design
- [10] Wikipedia,. (2015). Defensive programming. Retrieved 22 April 2015, from http://en.wikipedia.org/wiki/Defensive_programming
- [11] Wikipedia,. (2015). ISO/IEC 27001:2013. Retrieved 22 April 2015, from http://en.wikipedia.org/wiki/ISO/IEC_27001:2013

- [12] Wikipedia,. (2015). Cross-site scripting. Retrieved 26 April 2015, from http://en.wikipedia.org/wiki/Cross-site_scripting
- [13] Iso.org,. (2014). ISO 27001 - Information security management. Retrieved 10 May 2015, from <http://www.iso.org/iso/home/standards/management-standards/iso27001.htm>
- [14] Iso.org,. (2014). ISO 27001 - Information security management. Retrieved 22 April 2015, from <http://www.iso.org/iso/home/standards/managementstandards/iso27001.htm>
- [15] Praxiom.com,. (2015). ISO IEC 27001 2013 Translated into Plain English. Retrieved 22 April 2015, from <http://www.praxiom.com/iso-27001.htm>
- [16] www.securityforum.org, I. (2015). Information Security Forum : The Standard of Good Practice for Information Security. Securityforum.org. Retrieved 7 May 2015, from <https://www.securityforum.org/tools/sogp/>
- [17] W3.org,. (2015). Cascading Style Sheets. Retrieved 11 May 2015, from <http://www.w3.org/Style/CSS/>
- [18] W3schools.com,. (2015). HTTP Methods GET vs POST. Retrieved 24 April 2015, from http://www.w3schools.com/tags/ref_httpmethods.asp
- [19] W3schools.com,. (2015). PHP 5 Sessions. Retrieved 24 April 2015, from http://www.w3schools.com/php/php_sessions.asp
- [20] WampServer,. (2015). WampServer. Retrieved 24 April 2015, from <http://www.wampserver.com/en/>
- [21] Projects.spring.io,. (2015). Spring Framework. Retrieved 24 April 2015, from <http://projects.spring.io/spring-framework/>
- [22] Pillgram-Larsen, J., Ryan, J., Cron, M., & Revilla, A. (2015). Gradle: Build Automation for the JVM, Android, and C/C++. Gradle. Retrieved 24 April 2015, from <https://gradle.org/>
- [23] Hjemmehos.finn.no,. (2015). Litt fakta - Hjemmehos FINN. Retrieved 26 April 2015, from http://hjemmehos.finn.no/no/om_oss/litt_fakta/
- [24] Ebay.com,. (2015). Electronics, Cars, Fashion, Collectibles, Coupons and More | eBay. Retrieved 26 April 2015, from <http://www.ebay.com/>
- [25] Securelist.com,. (2015). Mystery North Korean Actor's Destructive - Securelist. Retrieved 5 May 2015, from <https://securelist.com/blog/research/67985/destover/>

- [26] Sans.org,. (2015). SANS Institute: About. Retrieved 5 May 2015, from <https://www.sans.org/about/>
- [27] Solheim, I., & Stølen, K. (2007). Technology Research Explained. SINTEF.
- [28] SearchSoftwareQuality,. (2015). What is integrated development environment (IDE)? - Definition from WhatIs.com. Retrieved 7 May 2015, from <http://searchsoftwarequality.techtarget.com/definition/integrated-development-environment>
- [29] Vitale, T. (2015). Accidental security breach affects 347 students. Utah State Today Online News. Retrieved from <http://www.usu.edu/ust/print.cfm?article=54493>
- [30] Owasp.org,. (2015). Category:OWASP Application Security Verification Standard Project - OWASP. Retrieved 14 April 2015, from https://www.owasp.org/index.php/Category:OWASP_Application_Security_Verification_Standard
- [31] Microsoft.com,. (2015). Microsoft Security Development Lifecycle. Retrieved 7 May 2015, from <https://www.microsoft.com/en-us/SDL/process/training.aspx>
- [32] develstqbexamcertification.com,. (2015). What are the Software Development Models?. Retrieved 8 May 2015, from <http://istqbexamcertification.com/what-are-the-software-development-models/>
- [33] Bsigroup.com,. (2015). ISO 27001 for SMEs | BSI Group. Retrieved 10 May 2015, from <http://www.bsigroup.com/en-GB/iso-27001-information-security/ISO-27001-for-SMEs/>
- [34] Microsoft.com,. (2015). Microsoft Security Development Lifecycle. Retrieved 11 May 2015, from <http://www.microsoft.com/en-us/SDL/process/training.aspx>
- [35] Iso27001security.com,. (2015). ISO/IEC 27002 code of practice. Retrieved 16 July 2015, from <http://www.iso27001security.com/html/27002.html>
- [36] Iso.org,. (2015). Certification - ISO. Retrieved 17 July 2015, from <http://www.iso.org/iso/home/standards/certification.htm>
- [37] ISO/IEC,. ISO/IEC 27002: Information Technology - Security Techniques - Code Of Practice For Information Security Management. ISO/IEC, 2005. Print.