UiO : Department of Informatics
University of Oslo

Problem Analysis:

Ensuring secure application development using the ASVS Standard

Vegard Fjogstad

12th of June, 2014

Contents

1 Introduction **3**

2 Background - State of the Art **3**
2.1 Methods . . . 3
2.2 Standards . . . 4
2.3 Tool Support for Security Analysis . . . 5
2.4 Gap . . . 5

3 Objective of the Thesis **6**
3.1 Artifacts . . . 7

4 Stakeholders **7**

5 Success Criteria **8**
5.1 Interests . . . 8
5.1.1 System Architect . . . 8
5.1.2 Maintenance Developer . . . 8
5.1.3 Operation . . . 9
5.2 Success Criteria . . . 9
5.2.1 Artifact 1 - SecuritySoftware . . . 9
5.2.2 Artifact 2 - End User Process Description . . . 9

6 Research Method **10**
6.1 Innovation . . . 10
6.1.1 Process Description . . . 10
6.1.2 The Software . . . 11
6.1.3 Artifacts Combined . . . 11
6.2 Evalution . . . 11
6.3 Select the Appropiate Research Method . . . 11

7 Research Plan **12**
7.1 Phase 1 - Process description . . . 12
7.2 Phase 2 - SecuritySoftware Design document . . . 12
7.3 Phase 3 - Implementing SecuritySoftware . . . 12
7.4 Phase 4 - Evaluate Process Description and Design Document . . . 12
7.5 Phase 5 - Second Implementation of SecuritySoftware . . . 12
7.6 Phase 6 - Re-evaluation . . . 12
7.7 Phase 7 - Finalizing Thesis . . . 13

8 Conclusion **13**

List of Figures

1 Research methods [33] 10

1 Introduction

Software development has evolved from something that is done in your garage after work, to the backbone of society. This growth has spawned numerous companies, and in turn increased the need for structure, planning and working methods that can provide results, regardless of project size. With the ever increasing market for computers and PCs in general, more and more people with malicious intents learned ways to steal and obtain information of high value. Hacking, creating computer worms and abusing loopholes became an ever increasing threat, in a market that seemed to grow endless.

Many companies started up businesses revolving solely around security. Norton [5], McAfee [4] & Avast [1] are some examples of companies that have created security related software.

Instead of repairing damages after an attack, why not prevent them from happening in the first place.

This paper will present an idea on how to design a software that would provide the means for security analysis at an early stage in software development planning, and what it would take to define it as a success.

The structure of this document is as follows. In section 2, we highlight what currently is out there and discuss their strength and weaknesses. Section 3 summarize needs and artifacts, and in section 4 we will list the stakeholders. The success criteria for these stakeholders are listed and explained in section 5, while section 6 explains the artifacts we listed in section 3 and highlights how to evaluate whether or not they could be called a success when the project is complete. Section 7 gives an overview to the research plan, showing milestones and their respective dates. Finally, in section 8, we show our conclusion.

2 Background - State of the Art

Security related practices and frameworks are split into different camps. Some focus on improving security during the implementation phase of software development, while others analyse already working applications. In this section we will mention some of the methods, standards and tools surrounding these subjects and highlight the gaps that this thesis would try to fill.

2.1 Methods

The Microsoft Security Development Lifecycle (SDL) Process is a multi-step process for creating secure applications using both risk analysis and thread modelling to map security related issues [10]. One of the steps in this seven step process is “Perform Dynamic Analysis”, which includes running a software that monitors an application and checks for critical security problems. This step is performed with an already running version of an application, meaning it is of no use to someone who are still in the planning phase of software development.

A different approach is the Information Security Management System. ISMS is a set of policies "concerned with information security management or IT related risks"[16]. ISMS main goal is to minimize risk and reduce the impact a security breach could have. Where the SDL process revolves around the development of software and creating design specifications covering security issues, ISMS focuses more on management. Security breaches could happen from inside the company, by their own employees and this could be prevented using policies defined in an ISMS.

As touched on in the introduction, a different method to prevent security issues is to design software to be secure from the ground up. This is known by several different names, such as "Secure by design" [19], "Secure programming" and "Defensive programming" [20]. The idea is using good programming practices that would prevent situations where bugs related to security could occur. This would require a programmer to be highly familiar with the programming language that is being used to develop with, in order to recognize situations where faulty programming techniques could lead to potentially vulnerable systems.

2.2 Standards

As mentioned in the section above, ISMS is an approach to security management. This practice was in 2005 implemented in the ISO/IEC 27001 standard [17], which is covering cyber security related issues. The standard was in 2013 canceled with the publication of the latest revision, ISO/IEC 27001:2013 [21][18]. The newest revision changed some of its controls 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 [23].

The real benefits of these standards are, with a proof of certification, customers can be sure that a company is using proper practices to make sure that their sensitive information is secure from external and internal threats.

One of the older guides for security related standards is the "Standard of Good Practice" [24], published by the Information Security Forum (ISF). While the name says standard, it is a guide providing information on identifying and managing security risks. In 2011, SoGP was updated to make it cover the requirements for an ISMS that was set out in the ISO 27001 standard.

More related to software development itself, is an organization known as "The Open Web Application Security Project" [28]. OWASP provides a platform for anyone to read and create security related articles, documentation and more. In 2008 they published their first standard, namely the Application Security Verification Standard (ASVS) that provide requirements for measuring what degree of trust that can be placed in a web application. Some of the requirements found in this standard vary from authentication, access control, error handling, malicious controls and more [30]. ASVS lets the user set a risk level, from one to three, depending on the level of security required for the application. Level one provides security against vulnerabilities that are easy to discover.

Level two is achieved if the application can defend against vulnerabilities that pose moderate-to-serious risk and level three requires an application that can "adequately defend against all advanced application vulnerabilities, as well as demonstrate principles of good security design" [34].

2.3 Tool Support for Security Analysis

In the introduction we briefly mentioned software companies like Norton and Avast. They provide both companies and private persons with security tools like firewall applications and anti-virus software. These are tools that provide protection against security breaches via for example applications with security holes, visiting websites with malicious intents, out of date firmware on network routers or the operating system itself.

Several software companies have been creating tools to monitor and manage risks, and one of these are Acuity Risk Management. They've created a software called STREAM [26]. By feeding the software various information regarding the user's company and its structure, it could output risks analysis relevant to the ISO standard mentioned in the section above. To help this process further, they also created an application that would monitor the information provided to STREAM and automatically report the compliance with specifically the ISO/IEC 27001 standard.

Vigilant Software have created a software named vsRisk [27] that lets the user input various information about their company such as asset owners, risk assessments, threats and more. This tool then use this data to create different reports that helps companies to comply with the requirements in the ISO/IEC 27001 standard.

One of the tools that support the ASVS Standard is a plugin for an application called Fiddler. This plugin, named Watcher, is developed by Casaba and can perform some OWASP ASVS activities, which you can then use to manually review it in comparison with the requirements found in ASVS.

2.4 Gap

These tools provide ways to ensure the correct policies are in place to comply with requirements found in the highlighted standards. The ISO 27001 standard already have a few tools that performs checks, and compare it to the requirements, but ASVS is lacking in this department. The plugin listed does not work with ASVS directly, but provide the means to let the user compare results manually. There is potential for a software that could take a software specification and compare it directly with the requirements found in the ASVS Standard. This software would then output the necessary measures that has to be taken, or the changes needed in the specification to comply with the requirements.

ASVS in particular provides a good foundation for such a software, as its focus is on web applications. Web applications working together with cloud are becoming increasingly popular, and they will continue to grow as the internet grows even larger [32]. This software would output the needed changes that has

to be done in order to comply with the standard. In the following sections, we will discuss what such a software could look like, and the criteria required to consider it a success.

3 Objective of the Thesis

How can we best design an application, hereby referred to as SecuritySoftware, to solve the needs highlighted in the previous section.

There is need for a process to handle these needs, and the software to implement that process. The goal for SecuritySoftware is to provide developers with a tool to help implementing features that are secure against external threats like attacks through security holes, viruses and poor programming solutions (an example would be using Statements [7] instead of PreparedStatement [6] when developing with Java). This software has to be usable by an end user and consist of the following main features:

- Comprehensible
- Usefulness
- Acceptable coverage related to ASVS
- Resource efficient
- Scalability in relation to software architecture size

Comprehensible SecuritySoftware will be designed to let a system architect use it without requiring assistance from a consultant already familiar with the application or other security related programs. Each step in the process needs to be understandable, and flow organically from the previous step. Meaning, should step 1 list information about the clients software, then step 2 would use this information to build on.

Usefulness Determining whether or not something is useful is often decided by the end user. The goal for SecuritySoftware is to provide a service that would analyse a software specification and provide a list of possible security risks and the measures that has to be taken. The output would ensure that the software complies with the ASVS standard, meaning a user has a level of assurance that the application is secure. Making sure that it stays updated with the latest issues and solutions is of utmost importance to make sure it stays useful.

Coverage Security issues are constantly being discovered, and the correct measures that have to be taken to prevent them. For SecuritySoftware to provide an acceptable coverage, the application has to be easily updated to stay relevant. The ASVS Standard might see new revisions being released and this

could include new measures. The database providing SecuritySoftware with information has to be maintained, making the application able to keep up with the changes.

Resource efficiency Slow and inefficient applications are one of the more annoying problems for software developers. Being held back by the software itself is a waste of time that might not "wasteable". SecuritySoftware aims to provide a service that will run efficient on modern computers, meaning the system requirements are low enough to allow multitasking with other applications simultaneously. The bottleneck, if any, should not be the software but the user.

Scalability SecuritySoftware will be usable for both large scale architectures and smaller sized projects.

3.1 Artifacts

SecuritySoftware will consist of two artifacts. The software itself, and the process description for an end user. This process description is the step-by-step process encountered when a software architect uses SecuritySoftware.

4 Stakeholders

There are three main stakeholders of interest in relation to SecuritySoftware:

- A system architect (the end user)
- Maintenance developer
- Management.

System Architect The end user of SecuritySoftware is called a system architect. When the software reaches a useable state, the system architect would be able to use the program to analysis a system specification. In its current iteration, there are no plans for requiring a consultant to provide assistance with the process.

Maintenance Developer For SecuritySoftware to stay relevant, the system needs to be updated regularly. A maintenance developer would update the database with the latest information regarding the ASVS standard. Other responsibilities such as keeping the software bug free and usable on newer hardware, also falls to the maintenance developer.

Operation SecuritySoftware gets all its information from a database, and this database has to stay online and be maintained from an operational point of view. Updating the content of the database is the maintenance developers responsibility.

5 Success Criteria

In light of the stakeholders, we will highlight what their needs are and define criteria that will be used when evaluating whether or not the highlighted artifacts were a success.

5.1 Interests

Each stakeholders interests are different. In other words, the criteria for success vary for each one. Below is an overview of the interests each stakeholders has, as well as a short description explaining each point.

5.1.1 System Architect

A System architect has the following interests:

- Comprehensible
- Usefulness
- Acceptable coverage related to ASVS
- Resource efficient
- Scalability in relation to software architecture size

For a more detailed explanation, please refer to section 3.

5.1.2 Maintenance Developer

A maintenance developer has the following interests:

- Easy database access
- Modularity

Database access One of the main responsibilities for a maintenance developer is to make sure the database is constantly being updated with new information regarding security issues and measures. Instead of updating the database manually via for example SQL, SecuritySoftware will support this natively. Adding information to the database is a separate feature of the application, alongside the security process itself.

Modularity New issues and measures might require new features in SecuritySoftware. To make implementing new features as easy as possible, SecuritySoftware will be developed with modules, meaning that you could update one part of the system without touching other features. This also makes it easier for maintenance developers to troubleshoot features that aren't working or have known bugs.

5.1.3 Operation

Operation has the following interests:

- Database based on modern solutions

Database Operations only responsibility is to make sure the database is up and running. Most of that lies within operation itself. Making sure that the database is running on a solid server is outside the bounds of SecuritySoftwares responsibility, but making sure that the database itself is using a modern solution like MySQL or Oracle [15] are part of them.

5.2 Success Criteria

Based on the interests of each stakeholder, a set of success criteria for each artifact (listed in section 3.1) is listed below:

5.2.1 Artifact 1 - SecuritySoftware

For SecuritySoftware to be considered a success, the following criteria has to be met:

1. the application is easy to use and require no prior knowledge to the ASVS standard
2. the application is providing the user with system specifications that comply with the requirements found in ASVS
3. the application is implemented in a way to allow updates and bug fixes without compromising the functionality of the software
4. the application's knowledge base is updated continuously to make sure SecuritySoftware stay relevant and useful
5. the application's knowledge base is easily updated to cover criteria four
6. the application runs natively on (at least) Windows 8 and 7, Ubuntu 14.04 & OSX 10.8

5.2.2 Artifact 2 - End User Process Description

For the end user process description to be considered a success...

1. the process has to be logical and easily understandable for new users
2. the process has to be precise
3. the process should take no more than 15 minutes for someone already familiar with it

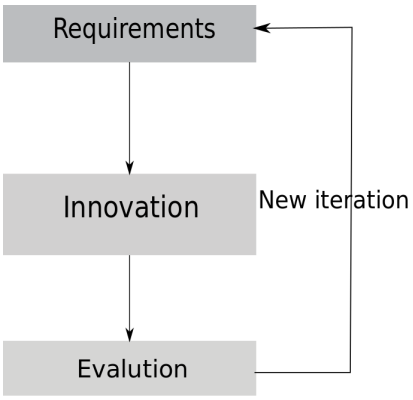

Figure 1: Research methods [33]

6 Research Method

Software development and theses use the same approach from inception to completion. See Figure 1 for an overview of this approach. When needs arise for something new, a set of requirements is presented. These requirements are then built on, either via scientific findings for a thesis or computer programming for an application. At some point, evaluation is needed. There are different methods for evaluation, such as field studies, surveys or even experiments in a laboratory. When evaluation has completed, the cycle repeats itself. In light of the results, the requirements might need modification, which is then built on with new findings or different programming methods.

SecuritySoftware and the process description will live through this cycle, and will see changes from its inception to finalization through several iterations.

6.1 Innovation

As presented in section 3.1, there are two artifacts for this thesis. SecuritySoftware, and the end user process description. While the software itself is important, the backbone of the project is the process description itself.

6.1.1 Process Description

The process description is a three step process. The first step is a risk evaluation of the application that the end user will be developing. The user then maps this to the security levels found in the ASVS Standard. After this step has been completed, the user lists all the features that the application will provide.

Once these two steps have been completed, the analysis steps in. By combining the information provided in the previous steps, SecuritySoftware uses this to query the database. The result is a list of requirements, that changes based on the ASVS risk level set in step one. This step is automated by the software

running the process, meaning the user will have no control over how the analysis is performed.

6.1.2 The Software

SecuritySoftware will provide the user with a graphical user interface (GUI) that will help guiding the system architect through each step in the process. Maintenance developers will download a seperate version of the application, to gain access to the special features relevant to updating the database. Along the way, helpful information will be provided to make the process as easy and smooth as possible.

6.1.3 Artifacts Combined

When a user works with SecuritySoftware, both artifacts are in play. Each step presented in the application is a step in the process. SecuritySoftware will allow the end user to go back and forth between steps if needed, making it easier to change up the specification should it be needed.

6.2 Evalution

Evalutating a software is important to ensure the success of SecuritySoftware. Unknown issues, weird graphical solutions and awkward use of language could go unnoticed by the development team. For SecuritySoftware and the process description, the best way to evaluate would be through a field study. This would provide the precision and realism to help fine tune both the process, and the software itself. A way to go about this, would be to provide a number of people with a copy of the software and giving them a task to complete. Because all subjects are working on the same problem, more issues would arise from the various mindsets each end user have when going through the step-by-step process.

6.3 Select the Appropiate Research Method

The reasoning behind choosing a field study, is because it provides the realism needed to properly evaluate the performance of the software and if the process is natural for a system architect to use. Another possibility would be to use a survey, asking test subjects whether or not they would be interested in using the services provided by SecuritySoftware. Problem with this is that each test subject have different ideas behind the best way of doing things, and might not agree with the theory that the software is built on. General evaluation methods like surveys are avoided, because SecuritySoftware is a tool that is very specific in its target audience, and a field study would provide better, more accurate results in a realistic environment.

7 Research Plan

Below is an overview of the planned milestones of this project, and their expected dates. The thesis itself is continuously worked on during each phase.

7.1 Phase 1 - Process description

Before any form of implementation or design can be done, the end user process description itself has to be completed. A document describing each step encountered in the software will be written.

Expected date of completion: August 15th, 2014.

7.2 Phase 2 - SecuritySoftware Design document

After the first version of the process description is done, a design document highlighting how SecuritySoftware will be implemented and what technologies will be used is needed.

Expected date of completion: September 15th, 2014.

7.3 Phase 3 - Implementing SecuritySoftware

Implementing the software itself is a long and drawn out process, that will change numerours times during the course of the thesis. A first prototype version of the software will be made.

Expected date of completion: December, 1st 2014.

7.4 Phase 4 - Evaluate Process Description and Design Document

There are bound to be problems with either the design or the process that will arise during the implementation phase. After phase 3 has completed, a proper evaluation is needed to prepare for the next implementation phase.

Expected date of completion: December 1st, 2014.

7.5 Phase 5 - Second Implementation of SecuritySoftware

The second implementation phase will begin right after the re-evulation has completed. This is the planned last phase of implementation, and therefore where SecuritySoftware will reach its final version.

Expected date of completion: January 15th, 2015

7.6 Phase 6 - Re-evaluation

After implementation has completed, a re-evaluation according to section 6.2 is needed to verify if the success criteria has been met.

Expected date of completion: March 1st 2015

7.7 Phase 7 - Finalizing Thesis

The last phase is finishing the thesis.

Expected date of delivery: May 1st, 2015.

8 Conclusion

As seen in the background information, there is an opportunity for a new kind of software to emerge on the development market. A lightweight, cheap and easy to use software for security related issues that are covered in the ASVS Standard. Where other solutions provide features to help during development or after, SecuritySoftware aims to target issues earlier in the planning phase. As a cost-benefit analysis is crucial during the making of software specifications, or creating use-case diagrams, the features provided by SecuritySoftware could be an asset to go along these. For a year long thesis, there is not much time to develop a fully functional software that would hit the market upon release. But, the discussion, results and evaluations found in the eventual thesis could provide the foundations for an idea and a prototype that could be developed further.

References

[1] Avast.com AVAST 2014 | URL: http://www.avast.com/no-no/ [Accessed 16 May. 2014].

[2] Website AsyncTask | URL: http://developer.android.com/reference/android/os/AsyncTask.html [Accessed 16 May. 2014].

[3] Website Service (Java EE 6) | URL: http://docs.oracle.com/javaee/6/api/javax/xml/ws/Service.html [Accessed 16 May. 2014].

[4] Website McAfee | URL: http://www.mcafee.com/no/ [Accessed 16 May. 2014].

[5] Website No.norton.com | URL: http://no.norton.com/ [Accessed 16 May. 2014].

[6] PreparedStatement (Java Platform SE 7) | URL: http://docs.oracle.com/javase/7/docs/api/java/sql/PreparedStatement.html [Accessed 21 May. 2014].

[7] Docs.oracle.com Statement (Java Platform SE 7) | URL: http://docs.oracle.com/javase/7/docs/api/java/sql/Statement.html [Accessed 21 May. 2014].

[8] Oss-watch.ac.uk What is version control? | URL: http://oss-watch.ac.uk/resources/versioncontrol [Accessed 22 May. 2014].

[9] Build software better, together In-text: (GitHub, 2014) | URL: https://github.com/about [Accessed 22 May. 2014].

[10] Microsoft.com Microsoft Security Development Lifecycle URL: https://www.microsoft.com/security/sdl/default.aspx [Accessed 23 May. 2014].

[11] AppSec Counsulting | URL: https://www.appsecconsulting.com/Application-Security/application-security-program-development/menu-id-62.html [Accessed 21 May. 2014]

[12] How To Set Your Consulting Fees | URL: http://www.forbes.com/2006/11/06/bostonconsulting-marsh-mckinsey-ent-fin-cx_mc_1106pricing.html [Accessed 23 May. 2014].

[13] Startmyconsultingbusiness.com How to set your hourly consulting rate | URL: http://startmyconsultingbusiness.com/how-to-set-your-rate/ [Accessed 23 May. 2014].

[14] Forbes Low Consulting Rates Leave Accenture High And Dry | URL: http://www.forbes.com/sites/greatspeculations/2011/02/14/low-consulting-rates-leave-accenture-high-and-dry/ [Accessed 23 May. 2014].

[15] Oracle.com Oracle Database 12c - Plug into the Cloud | URL: http://www.oracle.com/us/products/database/overview/index.html [Accessed 29 May. 2014].

[16] En.wikipedia.org Wikipedia Information Security Management System - Wikipedia, the free encyclopedia | URL: http://en.wikipedia.org/wiki/Information_security_management_system [Accessed 5 June. 2014].

[17] En.wikipedia.org ISO/IEC 27001:2005 - Wikipedia, the free encyclopedia | URL: http://en.wikipedia.org/wiki/ISO/IEC_27001:2005 [Accessed 5 June. 2014].

[18] Iso.org ISO/IEC 27001 - Information Security Management | URL: http://www.iso.org/iso/home/standards/management-standards/iso27001.htm [Accessed 5 June. 2014].

[19] En.wikipedia.org Secure by design - Wikipedia, the free encyclopedia | URL: http://en.wikipedia.org/wiki/Secure_by_design [Accessed 6 Jun. 2014].

[20] En.wikipedia.org Defensive programming - Wikipedia, the free encyclopedia | URL: http://en.wikipedia.org/wiki/Defensive_programming [Accessed 6 Jun. 2014].

[21] En.wikipedia.org ISO/IEC 27001:2013 - Wikipedia, the free encyclopedia | URL: http://en.wikipedia.org/wiki/ISO/IEC_27001:2013 [Accessed 6 Jun. 2014].

[22] DNV BA Information Center Security updates: The upcoming revision of ISO/IEC 27001 | URL: http://www.isocertificationuk.co.uk/blog/2013/03/14/security-updates-the-upcoming-revision-of-isoiec-27001/ [Accessed 6 Jun. 2014].

[23] Praxiom.com ISO IEC 27001 2013 Translated into Plain English Intext: (Praxiom.com, 2013) Bibliography: Praxiom.com, (2013). ISO IEC 27001 2013 Translated into Plain English. [online] Available at: http://www.praxiom.com/iso-27001.htm [Accessed 6 Jun. 2014].

[24] En.wikipedia.org Standard of Good Practice - Wikipedia, the free encyclopedia | URL: http://en.wikipedia.org/wiki/Standard_of_Good_Practice [Accessed 6 Jun. 2014].

[25] En.wikipedia.org Cyber security standards - Wikipedia, the free encyclopedia | URL: http://en.wikipedia.org/wiki/Cyber_security_standards [Accessed 6 Jun. 2014].

[26] Acuityrm.com Software for Risk Management | URL: http://www.acuityrm.com/products [Accessed 6 Jun. 2014].

[27] Vigilantsoftware.co.uk vsRisk Standalone – Basic URL: http://www.vigilantsoftware.co.uk/p-203-vsrisk-standalone-basic.aspx [Accessed 6 Jun. 2014].

[28] Owasp.org OWASP | URL: https://www.owasp.org/index.php/Main_Page [Accessed 6 Jun. 2014].

[29] Owasp.org Category:OWASP Application Security Verification Standard Project | URL: https://www.owasp.org/index.php/ASVS [Accessed 6 Jun. 2014].

[30] Clerkendweller.com OWASP ASVS for Web Applications 2013 Beta Release | URL: https://www.clerkendweller.com/2013/9/24/OWASP-ASVS-for-Web-Applications-2013-Beta-Release [Accessed 12 Jun. 2014].

[31] Watcher: Web security testing tool and passive vulnerability scanner | URL: http://websecuritytool.codeplex.com/

[32] Forbes Roundup Of Cloud Computing Forecasts And Market Estimates, 2014 | URL: http://www.forbes.com/sites/louiscolumbus/2014/03/14/roundup-of-cloud-computing-forecasts-and-market-estimates-2014/ [Accessed 12 Jun. 2014].

[33] Ida Solheim and Ketil Stølen. "Technology Research Explained", SINTEF REPORT 2007-03-20. Print.

[34] OWASP. "OWASP Application Security Verification Standard 2.0 (beta) 2013". Print. | Available at URL: https://www.owasp.org/index.php/ [Accessed 12 Jun. 2014].