Front Page

Contents

List of Figures

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 various softwares for security. Mostly virus scanning programs and firewall applications. But security related issues still haven't reached a significant part of the planning process of software development. Instead of repairing damages after an attack, why not simply 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'll list the stakeholders. The sucess criteria for these stakeholders are listed and explained in section 5, while section 6 explains the artifacts we listed in section 3 and highlight how to evalute 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

As is, there are no applications out there that can analyse a software specification automatically. Most services are designed around practices while developing, like risk analysis or thread modelling, or analysing the software after it has been developed.

2.1 Processes

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 7 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's still in the planning phase of software development.

2.2 Software

As mentioned earlier, the application used in step 7 of the Microsoft SDL is designed to take an already existing software and analyse it.

2.3 Consultants

Other possibilities for security assessment are hiring consultants that would either help with developing, or install their customized applications after development has been completed. [11] This obviously comes with a price, and hiring several would be out of reach for smaller companies or for sole proprietorship. A single consultant would cost around $200 and upwards. [12, 13, 14]

There's a clear need for a solution that is designed to handle security issues earlier in the development process, performs analysis automatically and comes with a price that would make it available to home developers as well as large companies.

3 Objective of the Thesis

How can we best design an application, hereby referred to as SecuritySoftware, to solve the needs we've highlighted so far? There's a 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 plan for 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]). This software has to be usable by an end user and consist of the following main features:

- Comprehensible
- Usefulness
- Acceptable coverage on security issues
- 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. 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 has to be taken to prevent them. For SecuritySoftware to provide an acceptable coverage, the application has to be easily updated to stay relevant. There will always be issues that have not been discovered yet, but as long as the database providing information is being maintained, SecuritySoftware should be able to keep up with the discovery of new security issues and the measures that has to be taken.

Resource efficiency Slow and inefficient applications are one of the more annoying problems for software developmers. 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, and let the user be the possible bottleneck, and not the system itself.

Scalability As highlighted in the background information, there are services provided that are unreachable for smaller companies because of cost. SecuritySoftware will aim to be usable for both large scale architectures and smaller sized projects. This will make it available for both large and small software development teams.

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's 3 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 it's current iteration, there are no plans for requiring a consultant to provide assistance.

Maintenance Developer For SecuritySoftware to stay relevant in an ever changing software development world, the system needs to be updated regulary. A maintenance developer would update the database with the latest information regarding security leaks, measures and other relevant information. Other responsibilities such as keeping the software bug free and usable on newer hardware, also falls to the maintenance developer.

Management SecuritySoftware gets all its information from a database, and this database has to stay online and be maintained regulary. This is the managements responsibility.

5 Success Criteria

5.1 Interests

Each stakeholders interest are different, and therefore the criteria for sucess may be different 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 on security issues
- 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 seperate 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 should be developed with modules, meaning that you could update one part of the system without touching other features. This would also make it easier for maintenance developers to troubleshoot features that aren't working or have known bugs.

5.1.3 Management

Management has the following interests:

- Database based on modern solutions

Database Managements only responsibility is to make sure the database is up and running. Most of that lies within management 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 is. [15]

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...

1. the application has to be easy to use and require no prior knowledge surrounding security related application
2. the application has to provide security related coverage for both large and small architectures
3. the application has to be implemented in a way to allow updates and bug fixes without compromising the functionality of the software
4. the application's knowledgebase has to updated continously to make sure Securitysoftware stay relevant and useful
5. the application's knowledgebase has to be easily updated to cover criteria number 4

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
2. the process has to be precise
3. the process has to be time efficient

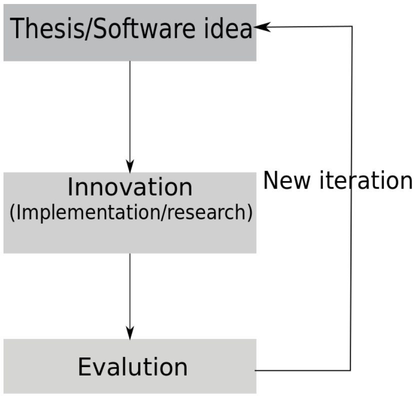

Figure 1: Research methods

6 Research Method

Software development and theses use the same approach from inception to completion. When needs arise for something new, a theory or an idea is presented. This idea is then built on, either via scientific findings for a thesis or computer programming for an application. At some point, evalution is needed. There are different methods for evalution, such as field studies, surveys or even experiments in a laboratory. When evulation has completed, the cycle repeats itself. In light of the results, the idea or theory 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 real backbone of the thesis is the process itself.

6.1.1 Process Description

The process description is a 4 step process where the user starts with a software specification and end up with a list of security issues and the measures that has to be taken to prevent them. The first step is where the end user lists all relevant clients the specificed system is going to be used on. Examples include iOS, Android, web browsers etc. After this step has been completed, the user is asked to specify how the server side will look like. Third step involve listing the features that this system will provide, and how these features communicate from the client to the server.

Once these 3 steps have been completed, the analysis steps in. By taking the information provided from the 3 previous steps, SQL querys will be performed to fetch information from the knowledge database.

6.1.2 The Software

SecuritySoftware will provide with a GUI to guide the user through this process. Along the way, helpful information will be provided to make the process as easy and smooth as possible. It will be available on Windows, Linux and iOS.

6.1.3 Working Together

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 one of the major aspects of development. Unknown issues, weird graphical solutions and awkward language could slip unnoticed for the development team that are focused on the development itself. 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 tuning 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 evalute the performance of the software and if the process is natural for a system architect to use. Another possiblity 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 the results would be subjective. 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 evalution methods like surveys are avoided, because SecuritySoftware is a tool that is very specific in its target audience, and a study would provide better, more accurate results in a realistic environment. A field study could change the opinions of these, by letting them try it and experience the results it provides.

7 Research Plan

Below is an overview of the planned milestones of this project, and their expected dates. The thesis itself is continously 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: September 1st, 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 14th, 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 - Re-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 re-evalution is needed to prepare for the next implementation phase.

Expected date of completion: December 7th, 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: March 1st, 2015

7.6 Phase 6 - Evaluation

After implementation has completed, an evalution according to section 6.2 is needed to verify if the success criteria has been met.

Expected date of completion: April 1st 2015

7.7 Phase 7 - Finalizing Thesis

The last phase is finishing the thesis. Deadline has not been set, so this date will most likely change.

Expected date of completion: June 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. 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's not much time to develop a fully functional software that would hit the market upon release. But, the discussion, results and evalutions 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].