FACULTY OF SCIENCE & TECHNOLOGY

BSc (Hons) Software Engineering
May 2018

Developing a Software System to Improve Analysis and Communication of Website Domain Security

by

Ryan Howell

Faculty of Science & Technology

Department of Computing and Informatics

Final Year Project

Abstract

This project explores existing browser security indicators, and how these indicators can be improved in terms of their analysis and communication of website security. The security indicator used by most modern browsers has its roots from a twenty year old release of the Netscape web browser. These indicators only cover the existence of valid TLS encryption, which is the minimum configuration for a secure website. To complicate matters further, browsers from differing vendors fail to analyse or communicate the result of this analysis in a standardised way, which can be confusing for users.

The background study detailed in this project highlights the numerous best practices and standards which have been developed over the past two decades. These greatly improve website security when configured correctly. Notable standards include “HTTP Strict Transport Security” which enforces use of encryption for a domain, and “Content Security Policy” to restrict the type and location of content the browser can use, such as Flash objects or Java applets.

The artefact shows the feasibility of improving existing browser security indicators, by expanding the factors used for security analysis. The outcome of this analysis is communicated through a browser extension icon, using a simple colour coded A - F scoring system. User feedback shows that the artefact proved useful for normal and technical users, developers and system administrators. The performed user testing indicates fulfilment of user stories, and an improvement of awareness surrounding domain security, which was achieved through a consistent platform-independent security indicator.

Heimdall allows users to more effectively evaluate the security aspect of software quality in the context of websites, allowing them to better compare, contrast and determine trustworthiness, based on the security configuration of a domain.

Dissertation Declaration

I agree that, should the University wish to retain it for reference purposes, a copy of my dissertation may be held by Bournemouth University normally for a period of 3 academic years. I understand that once the retention period has expired my dissertation will be destroyed.

Confidentiality

I confirm that this dissertation does not contain information of a commercial or confidential nature or include personal information other than that which would normally be in the public domain unless the relevant permissions have been obtained. In particular any information which identifies a particular individual's religious or political beliefs, information relating to their health, ethnicity, criminal history or sex life has been anonymised unless permission has been granted for its publication from the person to whom it relates.

Copyright

The copyright for this dissertation remains with me.

Requests for Information

I agree that this dissertation may be made available as the result of a request for information under the Freedom of Information Act.

Signed: Ryan Howell .

Name: Ryan Howell

Date: 08/05/18

Programme: BSc (Hons) Software Engineering

Original Work Declaration

This dissertation and the project that it is based on are my own work, except where stated, in accordance with University regulations.

Signed: Ryan Howell .

Acknowledgments

I wish to thank my supervisor, Gernot, for his support and feedback throughout the project. Our weekly meetings were intrinsic to me achieving the success criteria and learning outcomes of the project. His insight, in conjunction with feedback from Shamal, was especially helpful when managing legal requirements.

I would also like to thank the friends, family and companies who were so receptive to my request of providing feedback and test results. They really helped to make the artefact more stable and feature complete, as well as keep the project on track. Special thanks to Anna and Dan for their unwavering support.

Lastly, I would like to thank everyone at Bournemouth University for the opportunities afforded to me.

TABLE OF CONTENTS

LIST OF FIGURES

1 INTRODUCTION

1.1 PROBLEM BACKGROUND

Between 2008 and 2017, the United Kingdom saw a 28% rise in online banking and a 24% increase in online shopping (ONS 2017). Users need to trust such services with sensitive information, therefore these services need to be secure and an average user needs to be able to determine how secure they are. In software security, industry and user incentives are not inherently aligned. Unlike more obvious software quality aspects, such as usability or performance, users have very few ways to determine that software is secure. Legal and societal expectations put pressure on industry to better align their interests with users, however, security is still often an afterthought (C. Steward et al. 2012).

As shown in Figure 1.1, modern web browsers add a padlock and often prefix 'https://' to the start of the address bar to communicate to users that a domain is secure. This security indicator is an important factor in how most users determine website security.

Figure 1.1 Padlock and https:// prefix in Mozilla Firefox Web Browser

The padlock icon can be clicked to display certificate information; however, eye tracking data has shown that while the padlock is commonly viewed, its interactive capability is often ignored, and that certificate information is seldom used and rarely understood (Whalen and Inkpen 2005). Furthermore, this indicator only pertains to one aspect of domain security; Transport Layer Security (TLS) (Dierks 2008). The security indicators are inconsistent in location, size, behaviour and colour across browsers. In the authors opinion, current web browser security indicators are insufficient and can give users a false sense of security. The validity of these statements will be evaluated during the project background study and critical evaluation.

1.2 PROBLEM CONTEXT

In 1995 the company Netscape released Secure Sockets Layer (SSL) v1.1, which later became modern Transport Layer Security (TLS) through work done by the Internet Engineering Task Force (Comodo 2018). As seen in Figure 1.2, the security indicator used by most modern browsers has its roots from a twenty year old release of the Netscape web browser. Note the 'security' button on the toolbar, the padlock icon in the bottom left of the status bar and the 'https' prefix in the address bar. This security indicator is persistently displayed, showing an open/closed padlock to show whether valid SSL/TLS was used.

Figure 1.2 Netscape Web Browser

More recent browsers display the security padlock indicator at the top of the browser window near to the address bar only when a secure connection is present. Apart from these alterations, security indicators have changed little since their introduction with Netscape. Following the introduction of SSL 1.1 in 1995, newer security standards such as HTTP Secure Transport Security (HSTS), HTTP Public Key Pinning (HPKP) and Content Security Policy (CSP) have been developed. However, modern security indicators do not factor adherence of such standards when evaluating website domain security. This called attention to a need for re-evaluation of browser security indicators.

Users have an expectation that websites with a padlock and https prefix are private and secure. When viewing these indicators, users can only discern if valid TLS is used. They cannot easily determine if a domain adheres to other standards and therefore has strong enough security given the use case. Users are more invested in the security of an online banking service than that of a personal blog, therefore the user should have a way of determining whether the bank implements newer security standards such as HSTS, HPKP or CSP.

Configuring domain security is complex and industry lacks a granular way of communicating to end users how secure their domains are. A minimal TLS configuration will produce a green padlock and thus give the same impression of security to an end user, as a configuration more closely following best practices, such as which TLS ciphers to use. As a consequence, server administrators may have less incentive to harden servers.

1.3 Problem Definition and Rationale

Current web browsers only take in to account the presence of a valid TLS configuration when displaying a green padlock to end users. This does not fully encompass the many issues a server administrator must address in order to follow best practices and make use of modern web security standards.

Modern security indicators fail to communicate security information in a sufficiently granular way. Because of this, users can fail to properly understand security differences and are thus hampered when attempting to compare the security of websites or determine if a website is secure enough to use given their use case.

1.4 Proposed Solution

The proposed solution is called “Heimdall”. The goal is to develop a system that analyses various important aspects of website security. Figure 1.3 shows how Heimdall will provide a simple, easy to understand score which is colour coded ‘*A+*’ to ‘*F*’.

Figure 1.3 Proposed Heimdall Scores

As Figure 1.4 shows, summarised results will be presented to the end user through a browser extension by setting an icon on the toolbar to show the score. Further information will be made available to the user when the icon is clicked.

Figure 1.4 Proposed Heimdall Web Browser Extension Icon in Mozilla Firefox

1.5 Aims and Objectives

The aims of this project are to provide an artefact that more accurately represents how secure a website domain is. An additional aim is to successfully manage the project using software engineering best practices, such as design, build and testing phases.

The main objective is to explore the feasibility of analysing the many aspects of domain security and reducing these aspects to a simple score which accurately conveys actual domain security in a simple and understandable way.

Additional objectives are to:

- Implement an artefact that solves end-user needs
- Design and implement a system which is modular and extensible
- Implement support for multiple operating systems and web browsers
- Use modern programming practices
- Manage the scope of the project

1.6 Success Criteria

The project will have been successful if all system components have been developed and can be used to solve the user need of more accurately representing how secure a domain is in a simple and understandable way. This representation should allow different domains that serve similar purposes to be comparable through the Heimdall scoring system.

End-user testing shall provide evidence as to the usability of the software and whether it fulfils the user need of being simple to install and use. This testing will be completed per user story, with normal users, technical users, developers and system administrators being fulfilled through friends, family and interested companies respectively. The aim of managing the project using software engineering practices will be evaluated by assessing how well the project progressed when compared to the project plan. Any difficulties that arose during development and how well they were dealt with will be a good indicator of this.

1.7 Summary

The complete security characteristics of website domains are not accurately conveyed by the current security indicators in web browsers. This project will build a web extension artefact which more accurately analyses domain security and conveys the associated score to the end user in a way which is easy to understand and enables the user to better understand how secure a website is. Using good project management and software development best practices, the project will be well managed to help mitigate risk and complete the artefact in accordance with aims and objectives as part of a successful overall project.

2 BACKGROUND STUDY

2.1 INTRODUCTION

A background study was performed as part of the requirements elicitation phase, with the goal of analysing relevant past work, standards, best practices and Human-Computer Interaction (HCI).

This was performed to answer a number of questions:

- How are security indicators implemented in popular browsers today?
- What is the HCI surrounding security indicators?
- Which server configuration options are relevant to the aspect of security?
- What are the existing solutions for testing website domain security and how are their scores visualised?

Results from the background study were then critically evaluated.

2.2 SECURITY INDICATORS IN WEB BROWSERS

2.2.1 Security Indicator Implementation Across Popular Web Browsers

Figure 2.1 shows the top five desktop web browsers in use as of March 2018 are Google Chrome, Microsoft Internet Explorer, Mozilla Firefox, Apple Safari and Microsoft Edge (Wikimedia 2018).

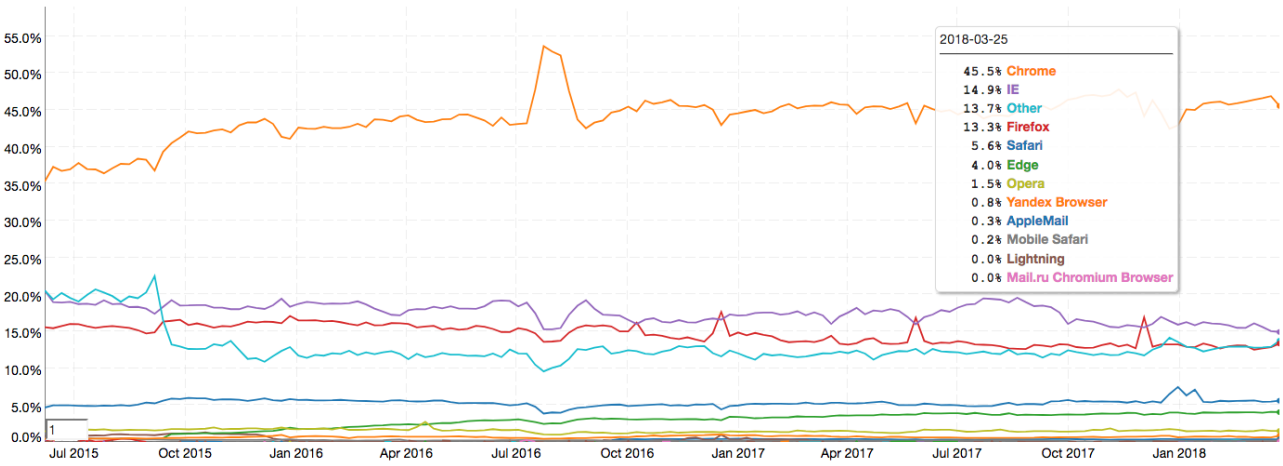
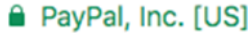
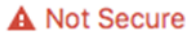


Figure 2.1 Desktop Browser Usage July 2015 - March 2018 (Wikimedia 2018)

The following is an overview of the implementation of security indicators in each browser.

Browser	HTTPS	Extended Validation	HTTPS Minor Error	HTTPS Major Error	HTTP	Indicator Location	Notes
Google Chrome (Google 2018a)						Left of Address Bar	The major error icon is also used for sites that are blacklisted by Google Safe Browsing for malware or phishing (Google 2018b). Blocks 1024-bit Diffie Hellman key.
Mozilla Firefox (Mozilla 2018b)						Left of Address Bar	Does not block 1024-bit Diffie Hellman key.
Apple Safari (Apple 2018)				None	None	Centre of Address Bar	Prefers full page errors instead of toolbar changes, shows toolbar as secure when adding an exception. Blocks 1024-bit Diffie Hellman key.
Microsoft (MS) Edge in Windows 10 (Microsoft 2017)						Left of Address Bar	Drops highlighting address bar. Blocks 1024-bit Diffie Hellman key. Displays pop-up to enable flash.

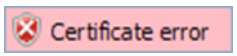
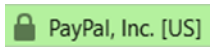
MS Internet Explorer 11 in Windows 7					None	Right of Address Bar, Highlights bar red on error and green on Extended Validation	Prefers full page errors instead of toolbar changes, only shows toolbar change on adding an exception. No documentation found. Blocks 1024-bit Diffie Hellman key. No documentation found.
MS Internet Explorer 11 in Windows 10					None	Right of Address Bar, Highlights bar red on error and green on Extended Validation	Prefers full page errors instead of toolbar changes, only shows toolbar change on adding an exception. Blocks 1024-bit Diffie Hellman key. No documentation found.

Figure 2.2 Brower Security Indicator Comparison

Figure 2.2 shows that the location, size, behaviour and colour of browser security indicators are inconsistent. Browser security indicators can vary across operating systems even when the operating system and browser are from the same vendor. An example of this is Microsoft Internet Explorer 11 in Microsoft Windows 7 and Windows 10. The majority of browsers put the security indicator on the left, whereas outliers such as Internet Explorer put the security indicator on the right and website controlled favicon on the left. As Figure 2.3 shows, this can make spoof attacks more effectively trick the end-user in to believing the connection is secure.

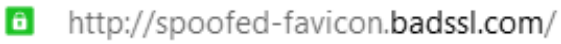


Figure 2.3 Spoofed Security Indicator Using Favicon in Microsoft Internet Explorer 11

Inconsistencies between security indicators across various browsers can lead to confusion for end users. This highlights the need for this project to further evaluate what these indicators should analyse, and how that analysis can be effectively communicated to end users.

2.3 Human-Computer Interaction and Web Browser Security Indicators

Eye tracking has been used to study visual security cues in web browsers. One notable study found that the lock icon has become a standard symbol for a secure connection and as such, it is the security cue that is most often looked at (Whalen and Inkpen 2005). However, because it did not clearly indicate that it is interactive, few users interacted with it and of the few users who did interact with the padlock icon, the information contained in the resulting popup was poorly understood.

The lock icons small size and often peripheral location can cause it to be misidentified, confused or overlooked by the casual observer, especially given the non-standard layouts among browsers. Making major modifications to the padlock symbol, such as using a different object may be disorientating, although the existing icon could be made to be more prominent (Whalen and Inkpen 2005).

A study by Kelly and Bertenthal found that the likelihood to enter user credentials and login to a website is modulated not only by security indicators but also by security domain knowledge and website familiarity. Mouse tracking has revealed that spoofed websites with security indicators and especially those with extended validation certificates, resulted in less rigorous interrogation of the website to determine its authenticity than websites without security TLS based security indicators. The study goes on to say that spoof websites are much more deceptive than a non-encrypted website, because many of the security indicators are often present, despite the website being insecure. It found that for either encrypted or non-encrypted websites, the presence of a lock icon was the primary influence on decision making. Furthermore, the study challenges the view that

security indicators are ignored, instead, it indicates that the problem for users is the inconsistency of certificates between websites and how the security indicator of the browser conveys that to the end user (Kelley and Bertenthal 2016).

These studies reinforce the need for the Heimdall browser extension, which analyses website domain security and visualises the score in a consistent manner across different web browsers. Furthermore, unlike the security padlock icon it requires more effort to get a high Heimdall score if you are a spoof website author, making it easier for users to distinguish spoof from non-spoofed.

2.4 Existing Website Domain Security Indicators

Chrome, Firefox, Internet Explorer, Safari and Edge each offer their own add-on/extension stores where end users may install tools that add functionality to their respective web browsers. Extensive searching was conducted in an attempt to find existing tools which sought to fulfil the same user needs as Heimdall. Various security add-ons exist, however unlike Heimdall, these are designed to block browser pop-ups, detect and block phishing sites, manage passwords, protect privacy, disable JavaScript, stop malware, etc.

Attention was therefore turned to websites which offer domain scanning services. There are three popular websites which fall into this category. As seen in Figure 2.8 below, SSL Labs (Labs 2018) focuses mostly on the security of the SSL/TLS configuration, securityheaders.io (Helme 2018) focuses entirely on the configuration of HTTP headers and High-Tech Bridge (Bridge 2018) offers tools which cover SSL/TLS configuration, header configuration and compliance checking.

Test	SSL Labs	securityheaders.io	High-Tech Bridge
HTTPS	Yes	No	Yes
Powered By Header	No	No*	No*
Referrer Policy Header	No	Yes	Yes
Server Header	No*	No*	Yes
HTTP Strict Transport Security Header	Yes	Yes	Yes
HTTP Public Key Pinning Header	Yes	No	Yes
Content Security Policy Header	No	Yes	Yes
Content Type Options Header	No	Yes	Yes

Frame Options Header	No	Yes	Yes
XSS Protection Header	No	Yes	Yes
Certification Authority Authorisation (CAA)	Yes	No	Yes
HTTP Redirect	Yes	No	Yes
TLS Certificate	Yes	No	Yes
TLS Ciphers	Yes	No	Yes
TLS OCSP	Yes	No	Yes
TLS PFS	Yes	No	Yes
Heartbleed	Yes	No	Yes
OpenSSL CCS flaw	Yes	No	Yes
OpenSSL Padding-oracle flaw	Yes	No	Yes
POODLE	Yes	No	Yes
BEAST	Yes	No	No
Compliance	No	No	Yes

Figure 2.4 Existing Solutions Functionality Comparison

Figure 2.4 shows the functionality of three top web domain security scanners.
* Indicates test conducted but not accounted for in the final score.

In order to give the most relevant and realistic security analysis, and effectively utilise the time during the project, this information was a factor in the decision of which library modules to write for Heimdall.

2.4.1 Existing Security Scanner Score Visualisation

Figure 2.5 shows that the SSL Labs test produces a score and a chart with an individual technical breakdown. Figure 2.6 shows that the Security Headers test produces a score and individual breakdown. Figure 2.7 shows that High-Tech Bridge produces a score, standards compliance checks (PCI-DSS, NIST, HIPAA) and individual technical breakdown.

This information informed the design decisions of how to convey the security of a domain through a colour-coded score.

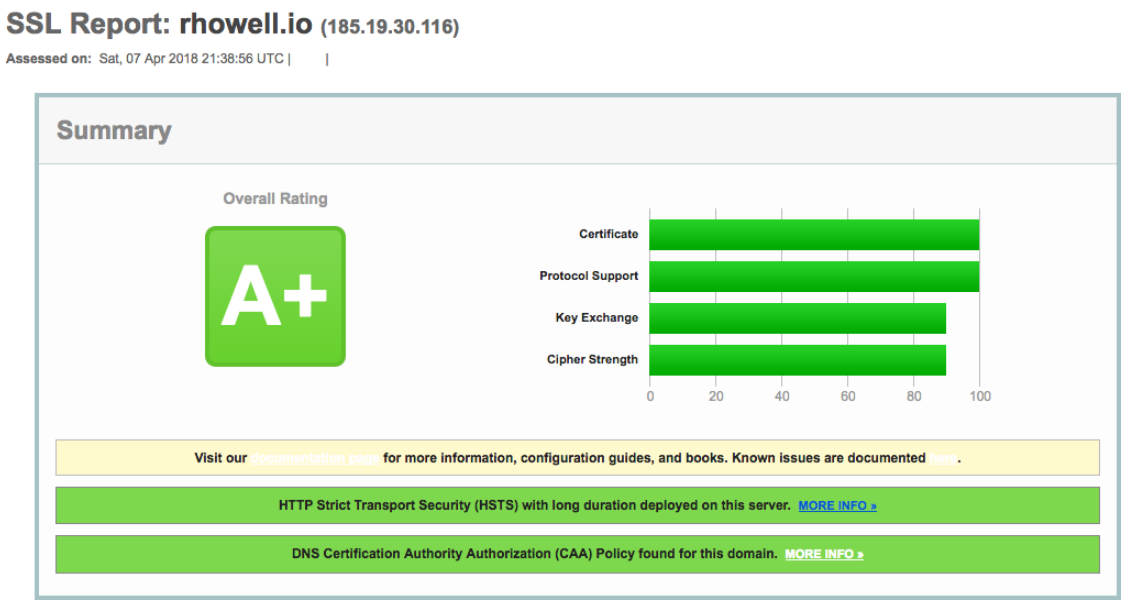


Figure 2.5 SSL Labs Score for rhowell.io

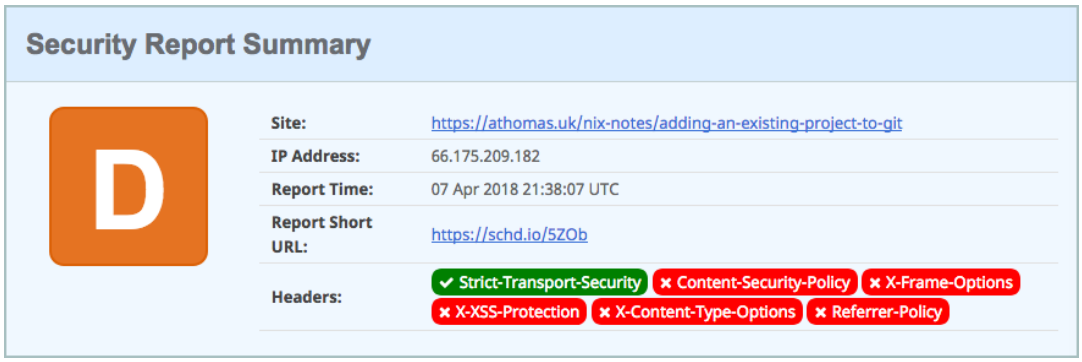


Figure 2.6 Security Headers Score for athomas.uk

Figure 2.7 High-Tech Bridge Web Server Security score for rhowell.io

2.5 Security Options

2.5.1 Compliance

There are two main sets of compliance standards followed by industry, PCI DSS (Council 2018) and NIST (Polk et al. 2014). PCI DSS is a global standard, NIST is the relevant standards body in the United States.

2.5.2 Standards

The following non-exhaustive technical standards were used to inform which key areas Heimdall modules should cover. Critical evaluation of these and other key standards is detailed in Chapter 2.6.3.

- Hypertext Transfer Protocol Secure (HTTPS) (Rescorla 2000) is the standard of using HTTP over TLS. This is the minimal requirement to have any security, as otherwise all communication between server and client are sent as unencrypted plain-text.
- HTTP Strict Transport Security (HSTS) (Jackson et al. 2012) is the standard by which a server can communicate to clients that it wishes them to only connect to the server using HTTPS for the specified period of time (max-age). This standard also allows ”pre-loading”, a method of telling browser vendors to include this statement statically with the browser upon compilation.
- Content Security Policy (CSP) (West 2018) is the standard by which a server can communicate to clients that it wishes to constrain what resources are accepted by the browser. For example, sources of JavaScript or which plugins are allowed to load.
- Domain Name System Security Extensions (DNSSEC) (Eastlake 1999) is the standard by which entries in the Domain Name System (DNS) can be cryptographically authenticated from the root name-servers to the domain-specific name-servers.
- DNS Certification Authority Authorization (Hallam-Baker and Stradling 2013) is the standard by which a domain can declare via DNS which Certificate Authorities (CAs) are allowed to sign certificate requests for that domain; thus preventing rogue CAs.
- HTTP Public Key Pinning (Sleevi et al. 2015) is a standard by which a domain can declare what certificates should appear within the certificate chain for a given period of time. This allows the domain owner fine-grained control that helps mitigate the risk of forged certificate chains.

2.5.3 Best Practices

The following sources of best practices were used to inform decisions on how modules should score different results.

- OWASP has an ongoing project of collecting best practices in server configuration (OWASP 2018a). This guide includes recommendations such as not leaking information about the server version number for Apache (OWASP 2018b) and Nginx (OWASP 2018c).
- The Mozilla “OpSec” team maintains a set of recommendations for Server Side TLS (Mozilla 2018c). These recommendations cover many aspects of TLS configuration, such as Certificate Algorithm, Certificate Key, Key Size, Ciphers etc. These recommendations are roughly categorized by browser compatibility into ‘Modern’, ‘Intermediate’ and ‘Old’.
- Qualys SSL Labs has similar best practices to Mozilla OpSec, explicitly covering TLS configuration in detail (Qualys 2018).
- Google lists some recommendations on HTTPS deployment, covering 301 redirects and HSTS (Google 2018c).

2.6 Summary and Critical Evaluation

2.6.1 Security Indicators in Web Browsers and Human-Computer Interaction (HCI)

A strength of the padlock security indicator is that end users commonly view it when checking the security of a website however, a weakness is that few realise the padlock is interactive and of those that are aware, the information displayed is rarely understood (Whalen and Inkpen 2005). Further weaknesses are the non-standardised placement and design of the icon, coupled with varying representations of different Transport Layer Security (TLS) server configurations between browsers. This combined with the padlock icons small size can often lead to confusion in the user (Whalen and Inkpen 2005) and the potential to spoof the security indicator through the favicon, to aid in an attack going unnoticed.

It is significant that the current security indicators only convey information relating to basic TLS, because configuration of a modern server that follows best security practices includes numerous other options, especially HSTS, CSP and Certification Authority Authorisation (CAA), which dramatically impact how secure a domain is. Either of which are important when preventing Flash content or a single certificate authority breach from making all websites vulnerable to a man-in-the-middle (MITM) attack (Korde 2016). Just using TLS is a suboptimal security measure if no HSTS header is provided, as a MITM attack simply has to catch the first normal non-encrypted request made and present the same website unencrypted. This allows the attacker to record all interaction with the website, including sensitive information such as a username and password during login. Additionally, it is less technically challenging to spoof a website with a basic security configuration and still present a positive security indicator, compared to spoofing a site with a more complex

security configuration. Heimdall would differentiate between these two configurations through a different score, allowing the user to more easily distinguish the real from spoofed website.

The background material was both useful and relevant because it utilised eye tracking and mouse tracking technologies to better understand the HCI involved with security indicators. This highlighted the importance of Heimdall covering more than TLS configuration, and that the communication of the security analysis needed to be simple and understandable. The studies fail to properly invest the users in the information they are meant to protect from malicious websites, which is of significance, as people are inclined to take less risks when invested in securing their own data. This information is of use to the Testing phase of the project, where emphasis on getting users invested is required.

2.6.2 Existing Website Domain Security Scanners and Score Visualisation

Of greatest significance when researching existing security scanners, was the lack of web browser extensions that checked the security of the domain as a whole. This is a poor state of affairs, as browser extensions are a good fit for this type of tool. Such an extension would allow users to browse as normal, whilst gaining additional information about the site they are visiting through an extension icon, along with further information via a popup when the icon is clicked.

Of the three website based domain scanners studied (SSL Labs, Security Headers and High-Tech Bridge), none covered the entirety of the security landscape. High-Tech Bridge was the most feature complete, but failed to include some vulnerability tests or not take the outcome of header tests into account when calculating a final score. A weakness of SSL Labs was that it largely failed to test for HTTP headers but it excelled at checking TLS configuration. Conversely, Security Headers strength laid with checking HTTP headers, but it only did basic checking of TLS configuration. Because the aforementioned scanners were all website based, it requires a change of user browsing habits to gather security information, rather than a cohesive browser extension based workflow with a passively displayed score icon. For these reasons, no one security scanner met all the needs of user stories laid out in the project requirements.

Visualisation of the score a website domain achieved was consistent throughout each of the three security scanners studied, with a colour-coded green '*A*' to red '*F*' representation being utilised. This is a strength, as continuity allows end users to easily understand the score visualisation of another scanner, e.g. a '*C*' is equal to '*C*'. Although a weakness is that users may fail to realise that different tools have separate test suites that give different weightings to each test, resulting in the same domain getting dissimilar security scores on different scanners. This may lead to confusion when aforementioned security scanners yield different scores. This highlights the complexity and difference of opinion regarding what is required to secure a website domain, and what is sufficient given the context of the website.

Research has shown that “color is superior to size, brightness and shape in identifying items that vary in only one aspect (e.g., only in color or in size), but that color cannot be identified as accurately as text” (Christ 1975). Another study asserts that in a user interface, colour can be used to call attention in order to signal the user with green for acceptable, yellow for caution and red for error or stop (Meier 1988). Food labels use a traffic light system to indicate high, medium or low amounts of sugars, salt, fat and saturated fat. This allows users to quickly understand that the greener the label, the healthier the choice (Temple and Fraser 2014). These studies align closely with my personal observations making the colour-coded A to F scoring system well matched with these type of security scanners.

2.6.3 Security option compliance, standards and best practices

2.6.3.1 Compliance

PCI DSS (Council 2018) in the authors opinion it is most comprehensive compliance standard, with the least consideration given to backward compatibility.

NIST (Polk et al. 2014) covers many of the same topics as PCI DSS, however, it still has some legacy recommendations which could be considered outdated, such as TLS 1.1. The author is more distrusting of NIST standards since the Snowden leaks cast doubt on their credibility as a standards body (Menn 2013).

2.6.3.2 Standards

Figure 2.8 shows the standards that were discovered and evaluated as part of the background study.

Standard	Significance / Relevance	Strengths	Weaknesses	Reasoning
TLS	Minimum requirement for secure communication.	The simplest way to gain security.	Complex configuration to meet best practices. Underlying cryptography requires expert knowledge.	Without TLS all traffic is sent in plain text.
HSTS	Allows server to tell client only to connect using TLS.	Easy to use, makes MITM attacks significantly harder.	Requires trust in the certificate chain.	Without HSTS MITM attacks are significantly more trivial.
CSP	Allows server to control what content is available to the client.	Allows detailed configuration to limit attack surface of the browser.	Easily stripped if successful MITM attack.	Without CSP the client will accept any content, e.g. Flash.
DNSSEC	Allows cryptographic chain of trust to verify DNS records.	DNS MITM made ineffective.	Centralised trust model. Poor client support.	Without DNSSEC, DNS is unauthenticated, unverifiable, and easy to MITM/spoof.
DNA CAA	Allows narrowing of attack surface by limiting what certificate authorities can issue certificates for a domain.	Breach of a single certificate authority does not break entire certificate trust model.	Only relevant for highly funded attackers. Not widely deployed.	Without CAA, a breach of a single certificate authority means a valid certificate can be issued for any site without it and perform an undetectable MITM attack.
OCSP	Allows verification whether a certificate is valid after revocation.	Prevents a leaked certificate/key pair from being used to perform an undetectable MITM attack.	Ineffective client implementation; does not fail-safe.	Without OCSP, you cannot in a simple efficient way check whether a certificate was revoked.
Referrer Policy Header	The referrer policy header allows the server to tell the	Prevents the domain from gathering information of	Pertains more to privacy than security.	Without a properly configured referrer policy

	client whether to send or how much information, the "referer"[1] header should contain.	where users came from.		header, the user can unknowingly leak browsing history.
HTTP Public Key Pinning Header	HPKP allows a server to pin certificates within its trust chain.	Prevents forging certificates within a chain that contains a pin above the forged certificate.	Complex configuration. Can be a target of ransomware. Largely obsoleted by CAA/Certificate Transparency.	Without HPKP/CAA a rogue certificate authority could forge a certificate allowing an undetectable MITM attack.
Content Type Options Header	Allows a server operator to tell the client what mine type a piece of content is.	Prevents loading executable content via mime type sniffing.	Mime type sniffing algorithms are less aggressive than they used to be. Requires ability to modify server content.	Without the content type options header if a carefully crafted file was successfully uploaded to the target server, clients could end up running code without the knowledge of the server administrator.
Frame Options Header	Allows server operator to tell the client what types of embedded content is allowed.	Prevents loading of embedded content that could be used for clickjacking attacks.	Not fine-grained control. Modern websites heavily rely on certain embedded content, such as Facebook like buttons.	Without the frame options header, malicious embedded content could be used to perform a clickjacking attack.
XSS Protection Header	A way of the server operator telling the client that they do not use cross-site scripting.	Prevents a cross-site scripting attack by detecting the use of XSS and blocking it.	Obsoleted by CSP.	Without the proper configuration of the XSS protection header or CSP, cross-site scripting attacks are easier to perform.

Figure 2.8 Table of Security Standards Comparison and Critical Evaluation

[1] The HTTP referer header is a misspelling of "Referrer" from the original proposal.

2.6.3.3 Best Practices

Figure 2.9 shows the best practices that were discovered and evaluated as part of the background study. These best practices were used to write modules for Heimdall that determine if best practices were followed.

Best Practice	Significance / Relevance	Strengths	Weaknesses	Reasoning
Sanitize Powered By / Server Header	A way of hiding information that would be valuable to attackers: what version of application software you are running.	Makes the analysis/data gathering phase of an attack harder.	Security by obscurity. Does not directly provide a security benefit.	Without sanitization of these headers, an attacker can search for known vulnerabilities within the version of a specific software you are running.
HTTP Redirect to HTTPS	A way of forcing users to use secure communication.	Makes communication encrypted between client and server.	Easily stripped out of response headers if no HSTS seen/preloaded.	Without an attempt to redirect to HTTPS, all communication is insecure plaintext.
TLS Ciphers and Order	Determines which ciphers are preferred and how strong the encryption used within TLS are.	Allows the server to prefer secure ciphers, and fall-back to legacy ciphers when required.	Supporting legacy ciphers can give users a false impression of security, given the possibility of downgrade attacks and use of insecure legacy software.	Without the server configuration declaring the preferred ciphers it is determined by the browser which may not be up-to date or fit for purpose.

Figure 2.9 Table of Security Best Practices Comparison and Critical Evaluation

2.7 CONCLUSION

The current security indicator has been around for more than twenty years. However, to this day these security indicators differ between browsers. The https prefix and browser padlock only cover existence of valid TLS and as such are not a valid representation of the complexities of modern website security. The research performed in the background study shows that people are largely unaware that they can interact with the padlock. Users that did view the information in the padlock popup often failed to understand the certificate information when attempting to evaluate how trustworthy the website may be.

The background study informed the requirements and analysis stage of the relevant design and technical decisions to be made within the project. Specially, in areas surrounding which modules to write, how certain domain configurations should be scored and what weighting modules should have in regard to their impact on the overall score. It also helped determine that the Heimdall extension icon needed to be as useful to the user as possible, even without active user interaction. To this end, the icon would be large, strongly coloured and placed in standard position alongside other browser extension icons the user may utilise (such as their password manager or popup blocker). It was discovered that the proposed colour coded score classification is well understood (Temple and Fraser 2014) and used by many existing security tools which make it a good fit for Heimdall. During the study I could not find any single tool that meets the user stories for Heimdall.

3 METHODOLOGY

3.1 INTRODUCTION

The following chapter discusses the Software Development Life Cycle (SDLC) and project management methodology necessary for conducting a successful IT project and makes justified selections which are contrasted against viable alternatives as required. These two approached are combined to create the overall methodology used in the project.

3.2 OVERVIEW

The SDLC describes the stages in a software development project, and it focuses on the product rather than the process. The project management methodology focuses on the processes involved instead of the product being created. More specifically it focuses on resources, schedule, risk and quality.

Both the SDLC and project management methodology can be separated in to two broad categories of traditional and agile (Awad 2005). Traditional approaches are linear and follow predefined, inflexible or plan-driven stages. Typically, these are requirements, analysis, design, coding, testing and deployment stages. Agile approaches may follow roughly similar stages but they plan to, or allow for, project members to fall back to previous stages when required.

3.3 SOFTWARE DEVELOPMENT LIFE CYCLE

The chosen SDLC was a slightly modified version of Extreme Programming (XP) (Figure 3.1), where ‘Stand Up Meetings’ were replaced with ‘Introspection’, where I would consider previous XP cycles and how the knowledge gained from prior cycles may affect the following cycle. ‘Pair Negotiation’ was removed due to there being a single artefact author. A disadvantage of this was an inability to discuss different design approaches and collectively decide on the best course of action. Similar to other agile software development life cycles, XP aims to provide small iterative software releases throughout the project with continuous integration and code refactoring.

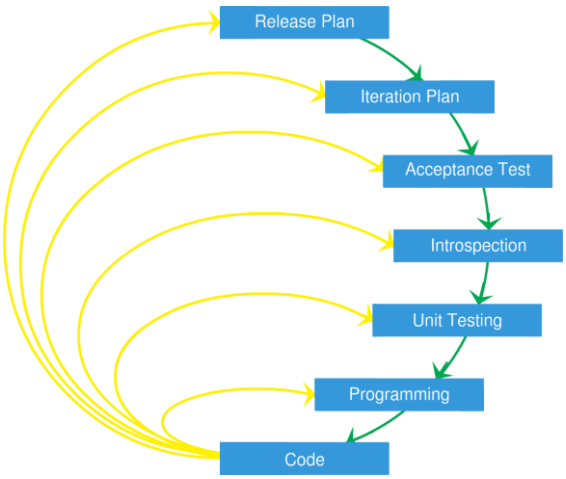


Figure 3.1 Modified Stages of Extreme Programming

The heavy component dependency model of Heimdall means that full iterative releases are infeasible until at least one module, module runner, and front-end are finished, so following a set of pre-defined stages, whilst allowing flexibility to return to previous stages when needed is appropriate. The project also has a feasibility study element, which lends itself to methodologies akin to exploratory prototyping, but with more structure through pre-defined stages which reduce risk. The continuous integration and unit testing of XP (Paulk 2001) pairs well with the modular design of the Heimdall library. The primary benefit of this modular design for unit testing is that the unit tests can be selectively executed when a component changes, in order to check for issues due to module failure or failures resulting from incompatibilities between components.

3.4 Project Management

I have chosen a modified version of Kanban as the project management methodology. Scrum was an alternative methodology given consideration. However, I quickly decided that Scrum was inappropriate for a number of reasons; Scrum is targeted to help manage team projects (Sutherland et al. 2007) and Heimdall has a single author and its sequential nature restricts the ability to quickly change the heavily integrated components of Heimdall. Additionally, Scrum is time-boxed, therefore a sprint will terminate at a fixed point regardless of how complete each Sprints backlog is (Scruminc 2017). This makes it impractical for the project due to the strict component dependency (i.e. the web extension is useless and cannot be developed without a working tray applet).

Kanban takes user stories and seeks to visualise and standardise the workflow of a project by grouping tasks based on these stories into easy to use categories which are organised on a project board (Anderson 2010). I added a category named ‘Tested’ to the typical Kanban categories of ‘To-do’, ‘Doing’ and ‘Done’ (Figure 3.2). This helps accommodate the necessity to test integration of components and ascertain how successful they are at fulfilling user story needs, as it requires each user story to be tested individually during development rather than only at the end of the project.

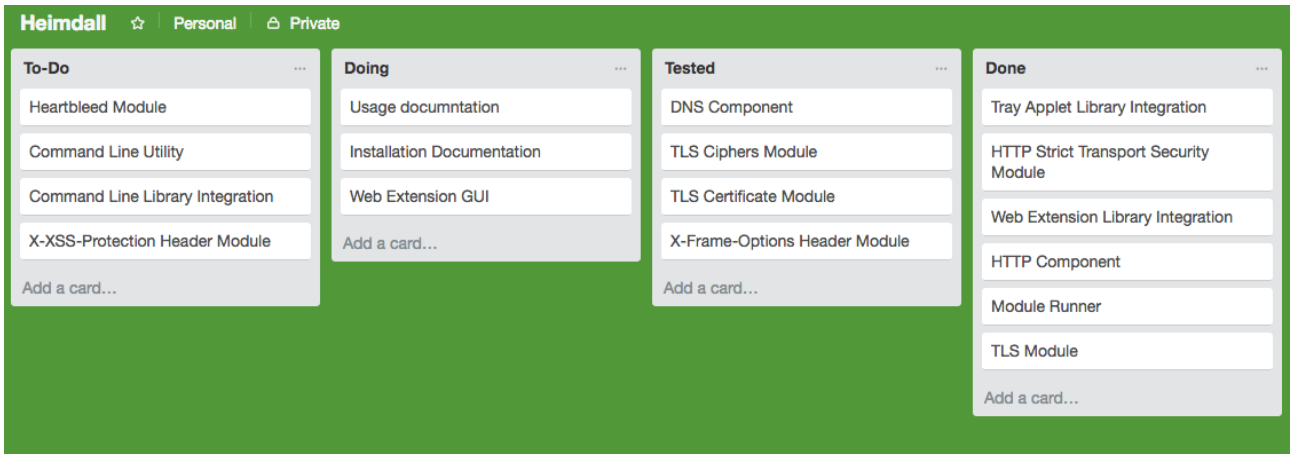


Figure 3.2 Simplified Trello Kanban Board Example

3.5 Project Management Methodology and SDLC Compatibility

A Kanban board provided me with a simple and effective progress indicator. This ability to monitor project progress is enhanced by the structure of XP, which is provided through pre-defined stages. Both Kanban and XP were useful to me as they allowed me to deliver features at fixed points of the project timeline, track my workflow and optimise my resource spending on the individual components which make up Heimdall. As minimal overhead was desired, both Kanban and XP are a good fit for a single author project, whilst still using approaches practiced in industry on larger-scale projects.

3.6 Summary

Extreme Programming was chosen as a SDLC as the project has a feasibility study element that lends itself to exploratory prototyping. The project required more risk reducing structure in order to deliver a good quality artefact in a timely manner, which XP helps to provide through requiring the use of pre-defined stages.

For project management methodology, a modified version of Kanban was chosen as it seeks to visualise and standardise the workflow of a project by grouping tasks generated from user stories into easy to use categories. Scrum was deemed inappropriate because it is intended to organise team projects, it is time-boxed and was not flexible enough to quickly go back to previous stages.

4 REQUIREMENTS AND ANALYSIS

4.1 INTRODUCTION

This chapter discusses the project risk analysis, requirements, what the objectives of the system and its components are and how the artefact will be evaluated against these objectives and requirements.

4.1.1 Project Requirements

Requirements were gathered as part of a hybrid release plan stage of XP. They are based on results of the background study.

4.1.1.1 Library Requirements

The library will take a response from a web server and process it using the collection of modules which each return a score. The lowest score from the modules will be taken as the final result for that website. Two processing modes are required, a “quick” mode which only runs the modules required in order to determine the overall score, and a “full” mode, which runs all modules regardless of whether they will affect the overall final score. The quick mode is needed to keep response time down in order to keep up with regular web browsing. The full mode is required to give a full report with additional information for technical users, developers or server administrators.

4.1.1.2 Tray Applet Requirements

The tray applet will expose the library via a web-socket API for use by the web browser extension. An icon will be added to the tray toolbar that can be used to configure and exit the application. The tray applet will have configuration options that controls auto-starting, the number of threads used for processing, and what address and port the API should use.

4.1.1.3 Browser Extension Requirements

The browser extension will use a subset of the library, or the tray applet to display a score for each request made. It will also be able to display a full report for more technical users, developers or site administrators. The browser extension will have configuration options that control which mode to use (Passive or Active), which port the tray applet API is bound to, and a whitelist/blacklist to allow the user to control which websites Heimdall will analyse.

4.1.1.4 Command-Line Application Requirements

The command-line application will return a detailed report in JavaScript Object Notation (JSON) for each address passed as an argument. The Command Line Interface (CLI) will allow automation without using the library directly and should be useful for gathering statistics about multiple domains. This will provide easier comparisons in reports by developers or server administrators.

4.1.2 Objectives

The overall system objective is to be able to effectively analyse the security of a website domain and to reduce this analysis into a simple, accurate and representative score. A sub-objective of the system is to fulfil the user stories of a normal user, advanced user, developer and system administrator (See APPENDIX E – Software Designs).

4.1.3 Evaluation

Heimdall will be evaluated on whether it can successfully complete its objectives. This evaluation will take place via unit, functional, performance, portability, reliability and usability testing. The black-box tests will be derived from the requirements specification and all white-box tests will be written with the requirements specification in mind. The final evaluation as to whether requirements were met relies upon all user stories being fulfilled and end-users being able to complete their objectives using the artefact. The outcome of this evaluation will be discussed in Chapter 8.

4.2 ANALYSIS

4.2.1 Requirements Analysis

The requirements gathering phase was conducted in order to analyse the scope of the project and relevant implementation details.

4.2.1.1 Library

4.2.1.1.1 Type

In order to allow the web extension to be useful without the tray applet, modules were split into passive and active categories. The aggressive category was added in order to restrict certain modules to only run when an explicit opt-in was present.

- Passive modules do not make any active requests, for example the HSTS module.
- Active modules make active requests to the server that precisely follow the intended purpose of the server, for example the TLS Ciphers module.
- Aggressive modules make active requests which do not follow the intended purpose of the server, for example the Heartbleed module.

4.2.1.1.2 Mode

Upon usage of the existing solutions discussed in the background study, it was apparent that speed would be a concern for Heimdall. A rough goal of two seconds response time was set (Nah 2004). In order to achieve this goal during normal web browsing two different modes would be needed; quick mode and full mode. In quick mode relevant modules would be processed in order of severity, and upon completing each module, the module runner would re-evaluate which modules to skip if those modules could not lower the overall score. In full mode all relevant modules would be ran regardless of whether they impacted the final score.

4.2.1.1.3 Module List

Previously developed software with similar features such as SSL Labs SSL/Server Test (Labs 2018) were evaluated in conjunction with standards and best practices pertaining to website domain security (see Chapter 2.6). From this evaluation, a module list was derived to determine what aspect of security each module had to test. In order to fit in the project duration, the list was narrowed down by evaluating how much time would likely be spent developing the module, and how useful, effective and representative the module could be in the security analysis. To see the final list of modules, see APPENDIX D – Software Requirements Specifications.

4.2.1.2 Web Extension and Tray Applet

By analysing the WebExtension (Mozilla 2018a) documentation I determined that due to limitations such as the inability to open a raw TCP or UDP socket, it would not be possible to write certain active or aggressive modules that require such functionality within the confines of a browser extension. The solution to this was to create a library of modules that could be used by the web extension and tray applet to access lower-level APIs. For the active and aggressive modules, the tray applet would need to expose the library via an API to the web extension.

4.2.1.3 Command-Line Interface

As the library was being developed, it was deemed beneficial to create a command-line interface (CLI) front-end application, targeted towards developers and system administrators. This facilitates automated use of the library through scripting without direct library integration, as is popular in UNIX environments which developers and system administrators favour (Stackexchange 2018). This contributes to fulfilling the developer and system administrator user stories.

4.2.1.4 Legal Requirements

I had originally intended to allow aggressive modules to be used across all front-ends, however, these modules would only run if an explicit opt-in from the domain owner was detected via the use of a custom HTTP header. Based on advice from my supervisor, I further restricted use of aggressive modules by only allowing them to run via the command-line application. This decreased risk through reduced accessibility to novice users, as such users are typically less comfortable with command line applications.

To further reduce legal risk, I added a blacklist and whitelist to the Heimdall browser extension. This meant a user could choose to add domains to their blacklist and whitelist to control which domains Heimdall would analyse. All domains used in design, development and testing of Heimdall are locally generated by the test suite with Docker and Nginx, are owned by the project author, or with the explicit permission of the appropriate domain owner.

4.2.2 Risk Analysis

A risk analysis was performed to evaluate what risks may impact the project, and how they could be mitigated or reduced.

Risk	Likelihood (L, M, H)	Impact (L, M, H)	Total Risk (L, M, H)	Effect	Risk Mitigation / Reduction
Inappropriate Technology Selection	Low	High	Medium	Artefact incomplete due to technical limitations or time constraints.	Careful analysis of requirements and study of relevant technology.
API Changes	Low	Medium	Low	Lost time, requires rebuilding components that no longer work.	Targeting specific standards such as WebExtensions, Node LTS and Electron; which are widely used and unlikely to change significantly.
Data Loss	Low	High	Medium	Lost data, requires restarting the project.	Syncthing (Borg and Butkevicius 2018) and Git (Chacon and Long 2018) distributed version control.
Inappropriate Project Management Methodology Selected	Low	High	Medium	Project failure, incomplete artefact, missing features.	Proper analysis of different project management methodologies.
Inappropriate System Development Life Cycle Selected	Low	High	Medium	Project failure, incomplete artefact, missing features.	Careful analysis of various software development lifecycles and proper selection based on problem domain knowledge,
Legal Issues	Low	High	Medium	Legal action against users & developer.	Aggressive mode only available via CLI and Library requires explicit opt-in from domain owner. All testing is done on local domains and remote domains with explicit permission from the owner.
Scope Creep	Low	Medium	Low	The project is not complete by the deadline.	Following a defined project plan and system requirements using the chosen project management methodology.

Figure 4.1 Risk Analysis Table

4.3 SUMMARY

A project risk analysis was performed to determine which risks may occur during the project and to determine their likelihood, impact and how they can be mitigated or reduced. Requirements were gathered by analysing existing solutions, compliance standards, technical documentation and feedback from the project supervisor. This informed legal requirements and helped to devise appropriate user stories.

The overall system objective is the ability to effectively analyse the security of a website domain and reduce this analysis into a simple, accurate and representative score. A library, web extension, tray applet and command-line application are used to fulfil the overall system objective. The artefact will be evaluated against these objectives and requirements through the testing phase, by comparing how well user stories are fulfilled through end-user testing.

5 DESIGN AND IMPLEMENTATION

5.1 INTRODUCTION

This chapter justifies chosen technologies and discusses how the components of Heimdall were designed and implemented. Multiple iterations were constructed until the final implementation was reached. Designs were created initially and subsequently updated throughout the project as necessary.

5.2 DESIGN

5.2.1 Library

The design of the library was an important factor in the scalability, portability and extensibility of the artefact. The library is the central hub of the application utilised by each front-end component. Figure 5.1 shows how the browser extension, tray applet and CLI tool utilise one or more modules. It is separated into a module runner and sets of modules which are categorised as either passive, active or aggressive. The browser extension uses the passive modules, the tray applet may use the passive and active modules and the CLI tool may use the passive, active and aggressive modules. The criteria used to test the library was chosen based on the background study of other relevant software, standards and commonly known best practices, see Chapter 2.

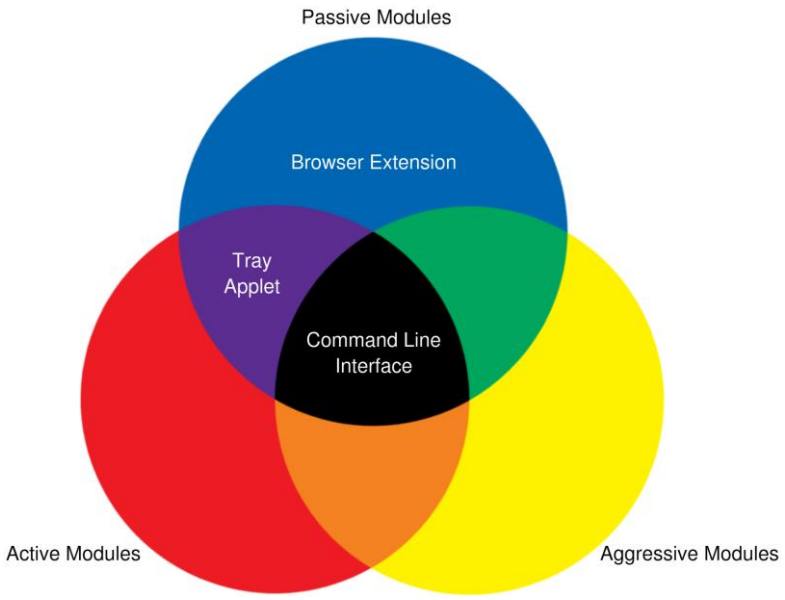


Figure 5.1 Diagram showing which library modules are exposed to which utility

5.2.1.1 Module Overview and Legality

When a standard web browser makes a HTTP GET request to a web server, the server sends a response back which is subsequently displayed in the web browser. If the Heimdall browser extension is installed in a web browser, after the page is displayed in the web browser, the response is sent on to the passive module to be processed and checked for security. The product of these checks is then returned and displayed to the user as a colour-coded score from *'A' – 'F'* (see Chapter 5.2.4). The passive modules are designed to only parse this response, and do not create any more connections or requests than would otherwise be made if Heimdall were not installed.

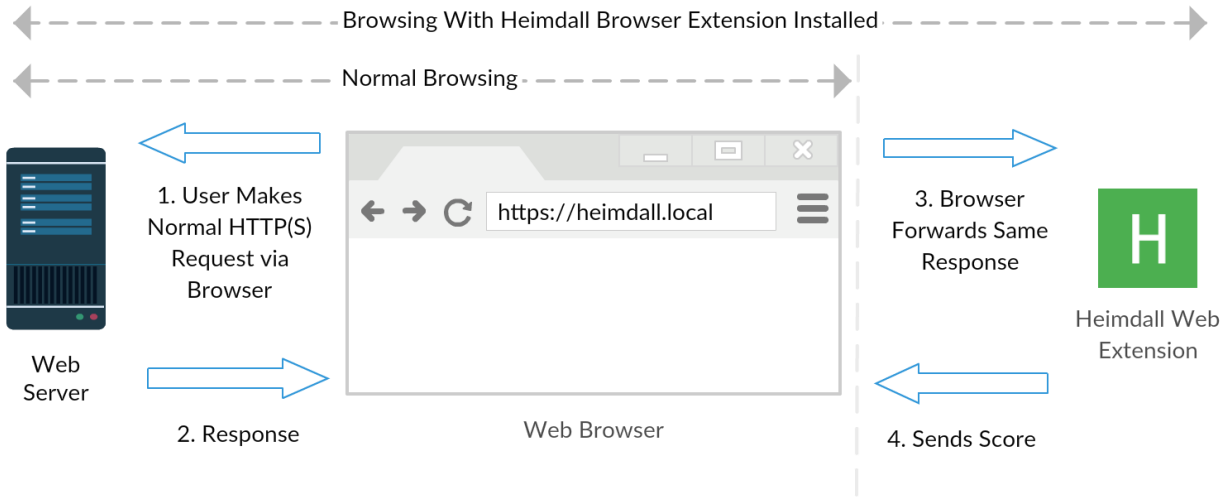


Figure 5.2 A diagram showing how the passive Heimdall browser extension works

The active modules differ from the passive modules, in that they are able to actively open connections and make requests in order to gather relevant information that cannot be passively gathered. However, it is by design that these requests are still fully within the scope of the intended purpose of the server.

The optional aggressive modules also actively open connections and make requests, however they differ from active modules by not matching the intended purpose of the server. In order to comply with UK Law, specifically the Computer Misuse Act 1990, section three "Unauthorised acts with intent to impair, or with recklessness as to impairing, operation of computer, etc." and 3ZA "Unauthorised acts causing, or creating risk of, serious damage" (Parliament 2015), the decision was made for the aggressive modules to require an explicit opt-in from the website domains owner. If a specific Heimdall header is not enabled by the owner of that domain, or aggressive is not enabled in Heimdall, aggressive modules will not run against the domain.

All testing during development of Heimdall was either done using the local test suite, or externally on domains the author has prior permission to use for testing purposes.

5.2.2 API Design

The WebSocket API was designed with simplicity in mind, where the web extension posts a JSON object to the tray applet API. This object is meant to contain a response from a web server using the WebExtension API, specifically the object returned from “webRequest.onCompleted” with the “extraInfoSpec” option containing “responseHeaders”. This approach is simpler than the alternative of creating a new object format, as future developers who wish to work on Heimdall need only to read and understand an existing format. The “responseHeaders“ option can either be an array or object of key-pair values. Any data attached to this object is sent back to the client in the Heimdall response. This suggested process allows tracking of which requests are relevant for each response and facilitates expandability of the API in future work.

5.2.3 Tray Applet

The tray applet shows the connection status of clients and also provides configuration options. On the context menu of the tray applet, there is a toggle for auto starting the applet, a button for opening the settings window (Figure 5.3). and an exit button.

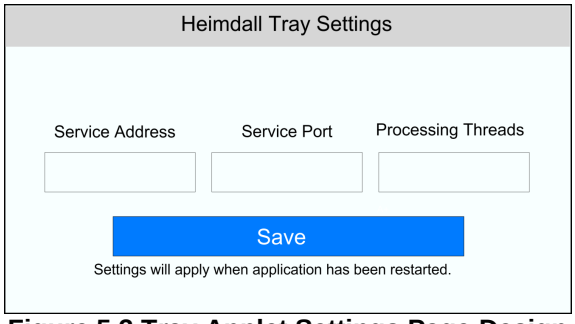


Figure 5.3 Tray Applet Settings Page Design

The settings window has an option to change which address and port the API should bind to. This allows users to choose a free port or expose the Heimdall API to their network. In addition, the settings window allows users to configure Heimdall resource usage, this is achieved by setting the number of threads available for processing. Altering thread count affects resource consumption which has a direct effect on response time, where more threads yield a faster response due to the asynchronous request model. The default number of processing threads is calculated as half the number of threads the computers Central Processing Unit possesses, down to a floor count of one. This was implemented to help Heimdall scale well across various types of hardware. The documentation for the Tray Applet Design can be found in APPENDIX E – Software Designs.

5.2.4 Web Extension

The web extension was designed to show a quick score by changing the icon of the extension button on the browser toolbar, this allows users to browse the web and simply glance at the icon to quickly determine the security of the domain.

The chosen scoring system, was originally an ‘*A’* through ‘*F’* grade with colour coding of green through red respectively. This was based on research from the background study, as it is easy for

an average user to understand that the greener a score is the better the result (Temple and Fraser 2014), and that an '*A'* is a better result than an '*F'*. As an incentive to properly harden domains, an '*A+'* on a blue background was added to show domains that have been hardened to the best Heimdall can analyse. In addition to this, a '*U'* score was also added to show that a module produced an error and thus a grade could not be given.

The scoring system was purposefully designed to show a per module score as well as an overall score, as this helps users understand the importance and weighting of modules. The choice was made for the overall score to be based on the lowest module score, rather than a sum or average of module results. This was chosen because security is a zero-sum game, for example, having TLS can be considered largely irrelevant if no HSTS header is used, as it is easier to strip via a man-in-the-middle attack (Dolnák and Litvik 2017).

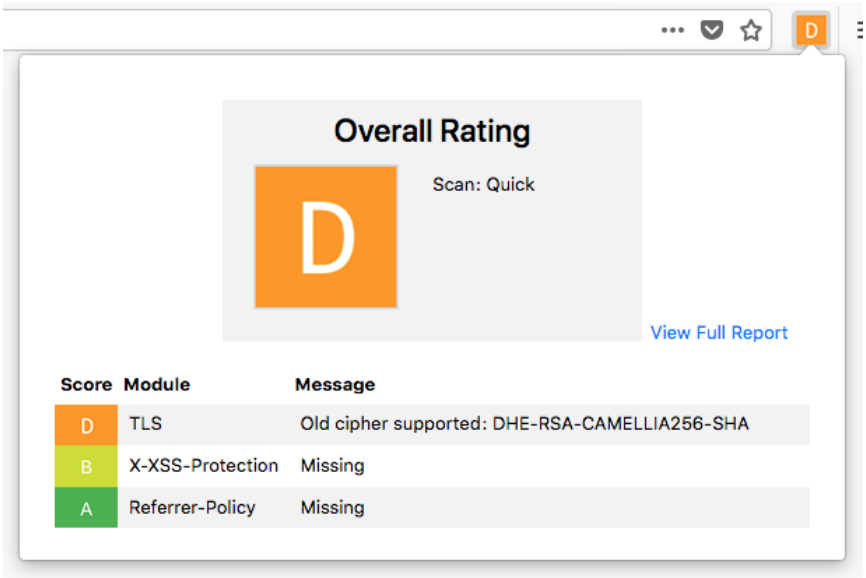


Figure 5.4 Web Extension Popup Design

Clicking on the Heimdall extension button in the browser toolbar gives a quick report breakdown showing which modules returned which grade (Figure 5.4). The quick report displays an overall rating for the currently open domain and a link to the full report. This link opens a new tab with the results of every module, message per module and a key.

The options pane of the Heimdall web extension natively integrates with the browser being used. The options available are scan type (Passive, Active), service address/port and a whitelist/blacklist to enable the user to control which domains the domain will analyse. The documentation for the full Web Extension Design can be found in APPENDIX E – Software Designs.

5.2.5 Command-Line Application

The command-line interface gives an end user access to the library without using a GUI application or WebSocket interface. The CLI takes logical arguments in order to produce response objects in JSON (the same format as the WebSocket API). The documentation for the CLI Design can be found in APPENDIX E – Software Designs.

The command-line interface was implemented to tightly integrate with the core library and not be dependent on the tray applet or browser extension, as such the CLI is the most portable and scriptable interface for Heimdall.

5.3 CHOICE OF TECHNOLOGY

5.3.1 WebSocket Protocol

The WebSocket protocol (Fette 2011) is a standard that enables two-way asynchronous communication between a client and a server. The protocol was chosen as the communication mechanism between the tray applet and web browser extension, as it allows the browser extension to send multiple asynchronous requests for each main frame page load and asynchronously receive responses from the multiple threads of the tray applet.

5.3.2 WebExtension Standard

The WebExtension standard (Mozilla 2018a) was chosen to develop the web browser extension, because it is the only standardised API that allows multiple web browsers from different vendors to use the same code. While there are small compliance inconsistencies between browsers, the cost of implementing workarounds is minimal. Given the tight time and resource constraints of the project, it was not feasible to develop an extension per each major browser.

5.3.3 Node

Node (Foundation 2018) is a JavaScript runtime built on the Google Chrome V8 engine. Node offers an event-driven, non-blocking I/O model that perfectly fits the asynchronous model for Heimdall of sending multiple messages to the tray applet and needing an asynchronous response. Node offers clustering as a method of multi-threading which is used by Heimdall to allow the processing of multiple concurrent requests. Node was mainly chosen as it allows portability of the core library across the tray applet, web browsers and command-line.

C was considered as a less resource intensive alternative. However, given the time and resource constraints, integrating / rewriting the library for each browser and operating system platform would not be viable given the project deadline.

5.3.4 Node Packages

Numerous Node Packages (Schlueter et al. 2018) were used when creating the Heimdall artefact, these can be found in the package.json files on the accompanying project CD. These packages allowed for more time to be focused on integration and application specific logic rather than excessive implementation details. Using Node packages allowed the author to complete the artefact within the project deadline, whilst maintaining a relatively large scope. This was achieved by slipstreaming development and retaining focus (Wittern et al. 2016).

5.3.4.1 Electron

Given the tight time and resource constraints of the project, it was not feasible to develop a native tray applet for each major operating system platform such as Linux, MacOS and Windows. The constraints also meant that the chosen GUI framework needed to be well documented. Electron (GitHub 2018) is a Node-based framework for creating cross-platform desktop applications using web technologies. Electron was chosen as it is a framework that meets the authors criteria of good documentation and cross platform compatibility. It allowed the tray applet to be built cross-platform within the project deadline. This would have been much more challenging when using frameworks written in lower-level languages. A further reason for choosing Electron, was that as its Node based, it allowed direct integration of the library into the tray applet via native Node APIs, without the need for external API bindings.

5.3.4.2 Mocha and Chai

The Mocha test framework (Holowaychuk 2018) and Chai assertion library (Luer 2018) were used to write the unit test suite for the library. Mocha and Chai are popular choices for JavaScript/Node that are well documented and easy to use. These tools were chosen over alternatives such as Jasmine, due to the authors prior positive experience with them in industry.

6 TESTING

6.1 INTRODUCTION

This chapter discusses the various methods used to test the library, web extension, tray applet and command-line interface. Testing was conducted throughout the project as part of the testing phase of Extreme Programming. A final dedicated testing phase was also conducted as part of the project plan. The overall test strategy was a mixture of functional and structural testing, with a focus on ease of automation for long-term maintainability.

6.2 STRUCTURAL TESTING

Extensive unit testing of the core library was performed, allowing repeatable tests with a large coverage. Unit tests most suited the core library, as it has no usable interface by itself and the inherent modular design allows unit tests to be written and ran on a whole library or per-module basis. Unit tests were derived from the source code. To see the final results of the unit tests, see APPENDIX J – Software Unit Test Results.

6.3 FUNCTIONAL TESTING

A mixture of error guessing and boundary value analysis was used (where appropriate) on the CLI, web extension and tray applet. Test plans were derived from the design documentation and user stories (see APPENDIX E – Software Designs). No failures were found as a result of this testing. This is likely due to the Extreme Programming SDLC requiring constant testing throughout development, especially during the integration phase of each cycle. Bugs that would have been found by these test plans were ironed out before functional tests in the dedicated testing phase were performed.

6.4 PERFORMANCE TESTING

Performance testing was undertaken throughout development of Heimdall, particularly in the case of quick mode. Additionally, performance testing was undertaken in the dedicated testing phase by using the command-line application under a realistic workload against two real-world domains that the author had prior permission to use (rhowell.io and athomas.uk). The author ran the benchmark for a duration of ten minutes on two different machines with dramatically different configurations. These were an AMD 1800X eight core desktop computer on a wired Gigabit Ethernet connection and a Lenovo X230 four core laptop computer utilising 5Ghz Wi-Fi connectively (see APPENDIX H – Performance Benchmarks).

From the requirements elicitation stage of the project, a rough goal of two seconds response time was set for Heimdall. Figure 6.1 shows that for the 1800X rhowell.io benchmark, the response time on average stayed within two seconds of the response time goal. As seen in Figure 6.2, the X230 wireless rhowell.io benchmark, the response on average still remained within two seconds of the response time goal, although the outliers vary more wildly, which is likely due to Wi-Fi instability.

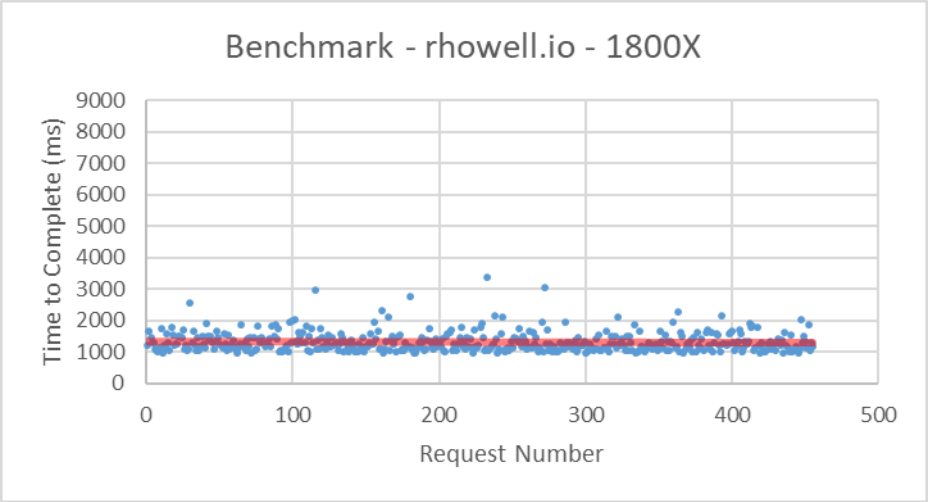


Figure 6.1 Benchmark - rhowell.io - 1800X

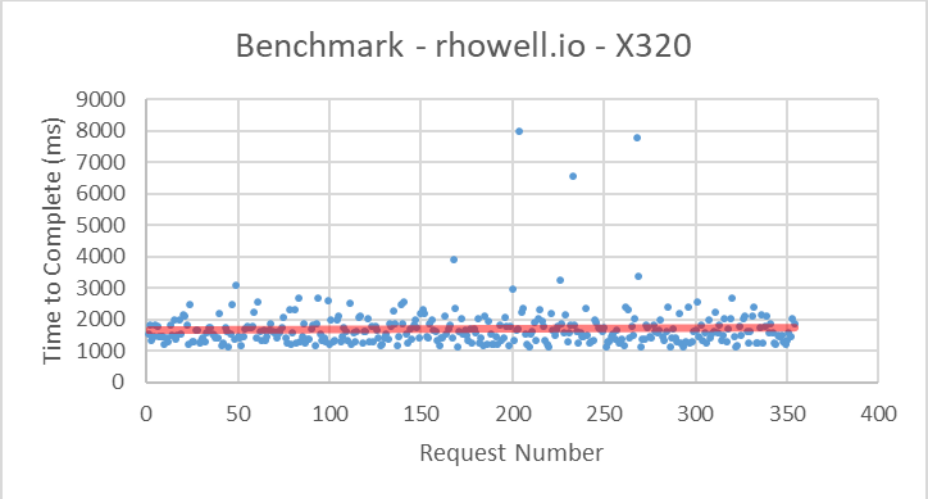


Figure 6.2 Benchmark - rhowell.io – X320

6.5 Portability Testing

Portability testing was performed by installing each Heimdall component in a range of website browser versions and operating system versions from multiple vendors. Each type of test was performed on the various browser and operating system combinations in order to assure the components were working correctly.

6.6 USABILITY TESTING

Usability testing was performed by giving a wide range of friends and family access to download the components along with installation and usage instructions. Particular attention was given to investing users in security. Users were passively observed during the installation and use of the software. Tweaks were made regarding any discovered usability issues based on the feedback, see APPENDIX K – Software Usability Test Results.

6.7 SUMMARY

Numerous methods were used to test the software project, with no major flaws or issues found in the final dedicated testing phase. However, small issues were fixed and improved throughout the project as part the testing phase of XP. Given the resource and time constraints of the project, the author determines that a reasonable amount of testing was performed, in order to ensure that the artefact is fit for purpose and thus able to fulfil user stories.

7 RESULTS AND DISCUSSION

7.1 INTRODUCTION

This chapter discusses and critiques the results of the project, as well as the progression of each component from conception to final build.

7.2 BACKGROUND STUDY

The background study phase of the project provided much insight into the what was necessary to properly secure a web server. Evaluating existing security scanners really helped to inform the scope of Heimdall.

The research into security indicators of popular browsers and HCI was especially useful when evaluating the intricacies of how users reacted and interacted with security indicators and how this could be improved by using Heimdall.

7.3 METHODOLOGY

The scope and time limitations of the project were well managed with the help of XP and Kanban. The necessity to provide a Gantt chart as a project deliverable does not, in the authors opinion, lend itself to these methodologies. That said, creating a Gantt chart really helped me to visualise dependencies and sequencing, as part of an overarching agile methodology. It was an invaluable organisation tool when paired with Kanban work item duration estimates. There were some deviations from the planned Gantt chart. However, the result was satisfactory as these were absorbed by the allocated contingency time with no negative impact on the overall project. See APPENDIX C – Project Plans.

7.4 REQUIREMENTS AND ANALYSIS

An analysis of the background study determined a web extension, tray applet and command-line interface was appropriate for the project. The result of this architecture performed well; indicating that appropriate requirements were produced from the background study.

Consideration was given to legal requirements that could arise from development and use of Heimdall. The test suite designed around these considerations allowed for suitable testing whilst meeting these requirements.

Numerous standards and best practices were analysed in order to create the list of Heimdall modules used to analyse domain security. These modules successfully analysed domains in much greater depth than regular security indicators in modern web browsers. During the requirements and analysis phase, user stories were created from project aims and objectives. Testing subsequently concluded that the project requirements fulfilled these user stories.

7.5 Design

The command-line interface remained consistent, as the design was made to be similar to other Unix programs; with a defined set of inputs and an associated output.

The initial user interface designs were followed closely, with some changes made as a result of testing, such as rounding corners of certain GUI elements to try make it more cosmetically attractive. The browser extension was simplified in an effort to make it more readable. The overall score in the popup and full report were made bigger and extraneous details in the popups table were removed, such as the key and modules that did not need to run in quick mode. Several iterations of the score icon colours were considered to maximise readability and attractiveness. The tray applet icons were tweaked to better fit varying operating systems, in particular a dark and light mode was added for MacOS. The results of these changes was a polished design and integrated application, giving a more native look and feel on each platform.

Based on the background study, a choice was made to display each modules score below the overall score, on both the popup menu and the full report page. Compared to existing security scanners, this provides the end user with a better understanding of the overall scores weighting, as a low importance module can fail, yet that module may be given an '*A'* because of its low importance to security. Conversely, if a module of highest importance fails, the module will be given an '*F'* and this will produce an overall score of '*F'*, even if every other module has passed with an '*A'*.

7.6 Artefact and Choice of Technology

During the development and testing of the library, some limitations of the chosen technologies became apparent. A restriction in Node was one such limitation, due to SSLv3 being disabled by default and it having less flexible cryptography APIs than if I were to directly link to the standard C APIs for OpenSSL. Another limitation was Electron breaking the normal node clustering model for multi-threading, instead requiring me to open a new hidden window per thread. The slightly slower performance and higher resource consumption due to this was still a reasonable trade-off given time restrictions.

When implementing and testing the web extension, it was discovered that Chrome handled HSTS differently than Firefox. Chrome would strip out the HSTS header from the final object before passing it to the listener. A workaround was implemented, that captures the redirect event with a status code specific to HSTS. This allowed some form of indication whether HSTS was used. However, as a result the ability to determine max-age (Jackson et al. 2012) was lost. The author discovered another bug with the Chrome extension API, where the function call for setting an icon of a tab does not always trigger the associated icon change. A lightweight refresh timer workaround was implemented, that re-called this function to change the icons for each tab at a fixed interval. This was a successful trade-off. Despite these issues with Chrome, the browser extension proved to be a good choice for displaying security information to the user.

The CLI interface has proved to be an effective means of scripting the Library without direct API use. The testing phase verified this through performance benchmarks. The library was a successful implementation of modules to determine domain security, and is easily extendable for future work. The approach of selecting which modules to run in quick mode was effective in reducing response time. The method of choosing the lowest score from the module results worked well, giving an accurate analysis of domain security.

The choice of technology for developing Heimdall provided interoperability across operating systems, with the artefact supporting Windows, MacOS and Linux on various CPU architectures. Although as Electron does not support BSD, a version of the tray applet for that operating system is currently unavailable. As such, a re-write or fork in a different GUI framework would be necessary to support this operating system. This was not considered for the first build because of time constraints and the low BSD desktop market share.

I had limited experience with the Electron library prior to commencing work, but ultimately this choice held up well. It allowed me to complete the project to a good degree of quality within the deadline, with the functionality defined in the artefact requirements in order to satisfy user stories (See APPENDIX D – Software Requirements Specifications). Overall the tray applet proved to be a good choice for this project, as it provides access to lower-level APIs than are available to a web extension.

User testing showed that communication of scores in Heimdall proved effective in allowing non-technical users to better understand the security of a domain. The final artefact is useful and one which myself and others who have tested it may continue to use during normal web browsing. The artefact met all requirements outlined in Chapter 4 and is fit for purpose based upon the testing detailed in Chapter 6.

7.7 TESTING

There were issues related to Mocha and Chai when writing the test suit. This was due to exception handling and runtime errors caused by intentionally bad test server configurations. Despite these manageable issues, Mocha and Chai worked very well for the project. To see the results of the test suite, see APPENDIX J – Software Unit Test Results.

During the testing phases, a Node based HTTP(S) server was integrated directly with the test suite. This worked well for simple tests, however, it was discovered that Node does not allow a severely insecure server to be configured, which is something Heimdall tests for. Instead, a test suite of Docker containers featuring various Nginx configurations was created. The relevant container for the type of test conducted is started and terminated when required.

The performance testing was simple to carry out via bash scripting (which also proves the usefulness of the command-line application) and gave results close to the initial target of two seconds response time. This could be improved with more time dedicated to performance refactoring. To see performance benchmarks, see APPENDIX H – Performance Benchmarks.

The end-user testing was vital in determining the success of the project. The artefact was given to friends, family and interested companies with the purpose of fulfilling the user stories of normal users, technical users, developers and system administrators. A change that resulted from this pertained to the web extensions quick and full report tables. Here it was suggested that links be provided to web pages containing information about the security standard or best practice the modules had analysed. This was subsequently implemented and received positive feedback from end users. End user testing determined that the artefact does fulfil the user stories defined in the software requirements. For feedback resulting from end-user testing, see APPENDIX K – Software Usability Test Results.

8 CONCLUSIONS

8.1 PROJECT OBJECTIVE FULFILMENT

The project has successfully produced an artefact that more accurately represents how secure a website domain is. The project was successfully managed with software engineering best practices following the Kanban and XP approaches that included design, build and testing phases to provide timely deliverables of a good standard.

The project has successfully completed its main objective of exploring the feasibility of analysing the many aspects of domain security, and reducing these aspects to a simple score which accurately conveys actual domain security in a simple and understandable way. All additional objectives in Chapter 1.5 have also been achieved.

8.2 SUCCESS AND FAILURES

Success of the project was primarily evaluated by the timely delivery of an artefact that solved the needs of user stories and met their satisfaction criteria. This was verified by end-user testing with family, friends and interested companies. This proved to be a good metric for assessing the project.

Secondly, there was the necessity to deliver an artefact which correctly analysed the security of website domains in line with the standards, best practices and compliance determined through background study. On both primary and secondary success criteria, the project has succeeded.

8.3 EXISTING SOLUTIONS AND HEIMDALL

There are tools with larger test suites than Heimdall. However, as detailed in Chapter 2.6.2, they do not convey information in a way that normal users may understand easily. The existing solutions are more aimed towards technical users scanning specific websites in order to generate full technical reports through a website interface. Heimdall caters for normal users who want a quick security summary in their browser interface during web browsing, whilst still allowing more technical users to get detailed information through the popup or full technical report.

Existing solutions are website based and require users to change their browsing habits in order to gather security information. The Heimdall web extension displays the current website domain score in its extension icon, and does not require uses to alter their workflow. This is more satisfying for the end user.

8.4 Personal

I have learned a great deal about server security configuration and gained experience with Electron and the WebExtension standard. I intend to continue using Heimdall throughout my normal browsing as I find it a useful tool which gives me the information I need to make better decisions surrounding website security and my browsing habits.

8.5 Future Work

8.5.1 Ports

As the browser extension is written in the WebExtension standard, therefore porting to other web browsers such as Microsoft Edge should be relatively trivial. However, some workarounds and additional testing will be required, as there are small inconsistencies across platforms.

8.5.2 Usability Improvements

Although the reports provide an on-hover description of each module with a link to more information, it would be of benefit to create small infographic videos that would clearly explain what each module covers, and why it matters to end-users.

8.5.3 Further Features

Additional modules can be written for Heimdall that may help improve accuracy. An analysis for each new module has to be performed, regarding how time-consuming the implementation would be, its resource consumption, and by what degree it would improve the accuracy of the score provided by Heimdall.

8.5.4 Technical Improvements

Given more time it would be beneficial to rewrite the module runner and certain modules (such as the TLS Ciphers module) in a lower level language such as C or C++. This would give lower-level access to APIs, a performance increase and decrease in memory usage. A trade off of this would be simplicity and ease of building the application for multiple platforms.

Word Count: 10861

REFERENCES

Anderson, D. J., 2010. *Kanban: successful evolutionary change for your technology business*. Blue Hole Press.

Apple, 2018. *Avoid fraud by using encrypted websites in Safari on Mac* [online]. Available from: https://support.apple.com/en-gb/guide/safari/avoid-fraud-by-using-encrypted-websites-sfri40697/mac [Accessed 2 Apr. 2018].

Awad, M., 2005. A comparison between agile and traditional software development methodologies. *University of Western Australia*.

Borg, J. and Butkevicius, A., 2018. *Syncthing* [online]. Available from: https://syncthing.net/ [Accessed 2 Apr. 2018].

Bridge, H.-T., 2018. *SSL/TLS Server Test | High-Tech Bridge* [online]. @htbridge. Available from: https://www.htbridge.com/ssl/ [Accessed 2 Apr. 2018].

C. Steward, J., Wahsheh, L. A., Ahmad, A., Graham, J. M., Hinds, C. V., Williams, A. T. and DeLoatch, S. J., 2012. Software Security: The Dangerous Afterthought, *2012 Ninth International Conference on Information Technology - New Generations* (pp. 815-818).

Chacon, S. and Long, J., 2018. *Git* [online]. Available from: https://git-scm.com/ [Accessed 2 Apr. 2018].

Christ, R. E., 1975. Review and analysis of color coding research for visual displays. *Human factors*, 17 (6), 542-570.

Comodo, 2018. *History of SSL Certificate | When was SSL Certificate Introduced* [online]. Available from: https://www.evsslcertificate.com/ssl/ssl-history.html [Accessed 2 Apr. 2018].

Council, P. S. S., 2018. *Official PCI Security Standards Council Site - Verify PCI Compliance, Download Data Security and Credit Card Security Standards* [online]. Available from: https://www.pcisecuritystandards.org/document_library?category=pcidss&document=pci%20dss [Accessed 2 Apr. 2018].

Dierks, T., 2008. *The Transport Layer Security (TLS) Protocol Version 1.2* [online]. Available from: https://tools.ietf.org/html/rfc5246 [Accessed 2 Apr. 2018].

Dolnák, I. and Litvik, J., 2017. Introduction to HTTP security headers and implementation of HTTP strict transport security (HSTS) header for HTTPS enforcing, *Emerging eLearning Technologies and Applications (ICETA), 2017 15th International Conference on* (pp. 1-4): IEEE.

Eastlake, D., 1999. *Domain Name System Security Extensions* [online]. Available from: https://tools.ietf.org/html/rfc2535 [Accessed 2 Apr. 2018].

Fette, I., 2011. *The WebSocket Protocol* [online]. Available from: https://tools.ietf.org/html/rfc6455 [Accessed 2 Apr. 2018].

Foundation, N. j., 2018. *Node.js* [online]. @nodejs. Available from: https://nodejs.org/en/ [Accessed 2 Apr. 2018].

GitHub, 2018. *Electron | Build cross platform desktop apps with JavaScript, HTML, and CSS.* [online]. @ElectronJS. Available from: https://electronjs.org/ [Accessed 2 Apr. 2018].

Google, 2018a. *Check if a site's connection is secure - Google Chrome Help* [online]. Available from: https://support.google.com/chrome/answer/95617?hl=en-GB [Accessed 2 Apr. 2018].

Google, 2018b. *Safe Browsing: malware and phishing – Google Transparency Report* [online]. Available from: https://transparencyreport.google.com/safe-browsing/overview [Accessed 2 Apr. 2018].

Google, 2018c. *Secure your site with HTTPS - Search Console Help* [online]. Available from: https://support.google.com/webmasters/answer/6073543 [Accessed 2 Apr. 2018].

Hallam-Baker, P. and Stradling, R., 2013. *DNS Certification Authority Authorization (CAA) Resource Record* [online]. Available from: https://tools.ietf.org/html/rfc6844 [Accessed 2 Apr. 2018].

Helme, S., 2018. *Analyse your HTTP response headers* [online]. Available from: https://securityheaders.io/ [Accessed 2 Apr. 2018].

Holowaychuk, T., 2018. *Mocha* [online]. Available from: https://mochajs.org/ [Accessed 2 Apr. 2018].

Jackson, C., Barth, A. and Hodges, J., 2012. *HTTP Strict Transport Security (HSTS)* [online]. Available from: https://tools.ietf.org/html/rfc6797 [Accessed 2 Apr. 2018].
Kelley, T. and Bertenthal, B. I., 2016. Real-world decision making: Logging into secure vs. insecure websites: USEC.
Korde, V., 2016. *The Importance of a Proper HTTP Strict Transport Security Implementation on Your Web Server | Qualys Blog* [online]. Available from: https://blog.qualys.com/securitylabs/2016/03/28/the-importance-of-a-proper-http-strict-transport-security-implementation-on-your-web-server [Accessed 2 Apr. 2018].
Labs, Q. S., 2018. *SSL Server Test* [online]. Available from: https://www.ssllabs.com/ssltest/index.html [Accessed 2 Apr. 2018].
Luer, J., 2018. *Chai* [online]. Available from: http://www.chaijs.com/ [Accessed 2 Apr. 2018].
Meier, B. J., 1988. ACE: a color expert system for user interface design, *Proceedings of the 1st annual ACM SIGGRAPH symposium on User Interface Software* (pp. 117-128): ACM.
Menn, J., 2013. *Exclusive: Secret contract tied NSA and security industry pioneer* [online]. @Reuters. Available from: https://www.reuters.com/article/us-usa-security-rsa/exclusive-secret-contract-tied-nsa-and-security-industry-pioneer-idUSBRE9BJ1C220131220 [Accessed 2 Apr. 2018].
Microsoft, 2017. *How to know whether to trust a website in Microsoft Edge* [online]. Available from: https://support.microsoft.com/en-gb/help/4027268/windows-how-to-know-whether-to-trust-a-website-in-microsoft-edge [Accessed 2 Apr. 2018].
Mozilla, 2018a. *Browser Extensions* [online]. Mozilla. Available from: https://developer.mozilla.org/en-US/docs/Mozilla/Add-ons/WebExtensions [Accessed 2 Apr. 2018].
Mozilla, 2018b. *How do I tell if my connection to a website is secure? | Firefox Help* [online]. Mozilla. Available from: https://support.mozilla.org/en-US/kb/how-do-i-tell-if-my-connection-is-secure [Accessed 2 Apr. 2018].
Mozilla, 2018c. *Security/Server Side TLS - MozillaWiki* [online]. repository on github. Available from: https://wiki.mozilla.org/Security/Server_Side_TLS [Accessed 2 Apr. 2018].
Nah, F. F.-H., 2004. A study on tolerable waiting time: how long are Web users willing to wait? *Behaviour & Information Technology*, 23 (3), 153-163.
ONS, O. f. N. S., 2017. *Internet access - households and individuals - Office for National Statistics* [online]. Available from: https://www.ons.gov.uk/peoplepopulationandcommunity/householdcharacteristics/homeinternetandsocialmediausage/datasets/internetaccesshouseholdsandindividualsreferencetables [Accessed 2 Apr. 2018].
OWASP, 2018a. *OWASP Secure Configuration Guide - OWASP* [online]. Available from: https://www.owasp.org/index.php/OWASP_Secure_Configuration_Guide [Accessed 2 Apr. 2018].
OWASP, 2018b. *SCG WS Apache - OWASP* [online]. Available from: https://www.owasp.org/index.php/SCG_WS_Apache [Accessed 2 Apr. 2018].
OWASP, 2018c. *SCG WS nginx - OWASP* [online]. Available from: https://www.owasp.org/index.php/SCG_WS_nginx [Accessed 2 Apr. 2018].
Parliament, 2015. *Computer Misuse Act 1990* [online]. Statute Law Database. Available from: https://www.legislation.gov.uk/ukpga/1990/18 [Accessed 2 Apr. 2018].
Paulk, M. C., 2001. Extreme programming from a CMM perspective. *IEEE Software*, 18 (6), 19-26.
Polk, T., McKay, K. and Chokhani, S., 2014. *Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations.*
Qualys, 2018. *ssllabs/research* [online]. Available from: https://github.com/ssllabs/research [Accessed 2 Apr. 2018].
Rescorla, E., 2000. *HTTP Over TLS* [online]. Available from: https://tools.ietf.org/html/rfc2818 [Accessed 2 Apr. 2018].
Schlueter, I., Voss, L. and Silverio, C., 2018. *npm* [online]. Available from: https://www.npmjs.com/ [Accessed 2 Apr. 2018].
Scruminc, 2017. *What is Timeboxing? | How is Timeboxing used in Scrum? | Scrum Inc.* [online]. Available from: https://www.scruminc.com/what-is-timeboxing/ [Accessed 2 Apr. 2018].
Sleevi, R., Evans, C. and Palmer, C., 2015. Public Key Pinning Extension for HTTP.
Stackexchange, 2018. Stack Overflow Developer Survey 2018.

Sutherland, J., Schwaber, K., Scrum, C.-c. O. and Sutherl, C. J., 2007. The scrum papers: Nuts, bolts, and origins of an agile process.

Temple, N. J. and Fraser, J., 2014. Food labels: A critical assessment. *Nutrition*, 30 (3), 257-260.

West, M., 2018. *Content Security Policy Level 3* [online]. Lists. Available from: http://www.w3.org/TR/CSP/ [Accessed 2 Apr. 2018].

Whalen, T. and Inkpen, K. M., 2005. Gathering evidence: use of visual security cues in web browsers, *Proceedings of Graphics Interface 2005* (pp. 137-144): Canadian Human-Computer Communications Society.

Wikimedia, 2018. *Dashiki: Simple Request Breakdowns* [online]. Available from: https://analytics.wikimedia.org/dashboards/browsers/#desktop-site-by-browser [Accessed 2 Apr. 2018].

Wittern, E., Suter, P. and Rajagopalan, S., 2016. A Look at the Dynamics of the JavaScript Package Ecosystem, *2016 IEEE/ACM 13th Working Conference on Mining Software Repositories (MSR)* (pp. 351-361).

APPENDIX A – PROJECT PROPOSAL

BU Computing Programmes 2017-2018

Undergraduate Project Proposal Form

Please refer to the **Project Handbook Section 4** when completing this form

Degree Title: SE	**Student's Name:** **Ryan Howell**
	Supervisor's Name: **Gernot Liebchen**
	Project Title/Area: **Scanning Services for End-User Web Browsing**

Section 1: Project Overview

1.1 Problem definition - use <u>one sentence</u> to summarise the problem:

Users and admins can sometimes have an unrealistic understanding of the security of a domain.

1.2 Background - please provide brief background information, e.g., client:

Non-technical end users and admins often have blind trust in the security of a domain when that site has an SSL certificate (especially EV) showing the green lock in their web browser – this project aims to give the user a more realistic view of the security of that domain.

1.3 Aims and objectives – what are the aims and objectives of your project?

To produce a system which can give a more realistic view of the security of a domain for an end-user or a site's administrator.

Edited by Dr Nan Jiang based on PH Section 4

BU Computing Programmes 2017-2018

Section 2: Artefact

2.1: What is the artefact that you intend to produce?

A library & web browser extension, clearly showing a simple colour dial (red, yellow, green), with a % of tests passed. Clicking on the extension should give more detailed information about failures.

2.2 How is your artefact actionable (i.e., routes to exploitation in the technology domain)?

The artefact can be used by a user to better understand the security of a domain.

Section 3: Evaluation

3.1 How are you going to evaluate your work?

I will evaluate my work by using the artefact within my normal web browsing usage, noting down any notable results that could help an end user / admin better understand the security of a domain.

I will also test my work against a set of services that are purposefully configured to cause test failures.

3.2 Why is this project honours worthy?

The project is relevant to my subject area (software engineering), because I am engineering a software system in order to solve a end-user need. The project can scale to as many work hours as available for development of the artefact, since there are countless services & configuration options that could be tested. The project involves independent research of services, best practices & the latest development stacks being used, with practical software engineering work being done to produce the artefact. The

Edited by Dr Nan Jiang based on PH Section 4

project will further my knowledge gained from the Advanced Development & Software Quality & Testing units.

3.3 How does this project relate to your degree title outcomes?

The project directly covers all areas of software engineering, an artefact will be produced that is relevant to the subject area and fulfils an end-user requirement.

The project will cover advanced development, by making use of modern technology/programming techniques such as asynchronous web requests/websockets.

The project will cover software quality and testing, both by testing the product itself, and the products purpose is to effectively test the security aspects of a domain.

The project will cover software systems modelling, because modelling will be used to accurately describe the systems that will be scanned, especially example use cases.

3.4 How does your project meet the BCS Undergraduate Project Requirements?

Production of the artefact shows the ability to apply practical and analytical skills, as a system that can be used practically by an end user will be produced and analysis of the services and best practices are required as a core part of the system..

3.5 What are the risks in this project and how are you going to manage them?

Hard drive failure – mitigated by using syncthing, local and cloud backups.

API changes – mitigated by targeting specific standards such as WebExtensions, which are widely used and unlikely to change significantly ensuring interoperability.

Users could get blacklisted because the system will have the ability to check if a port is open for specific service – services beyond basic http scanning will be an option in the settings for the system and not enabled by default, with a warning describing the risk to the user.

Edited by Dr Nan Jiang based on PH Section 4

BU Computing Programmes 2017-2018

Section 4: References

4.1 Please provide references if you have used any.

Section 5: Ethics (please delete as appropriate)

5.1 Have you submitted the ethics checklist to your supervisor?

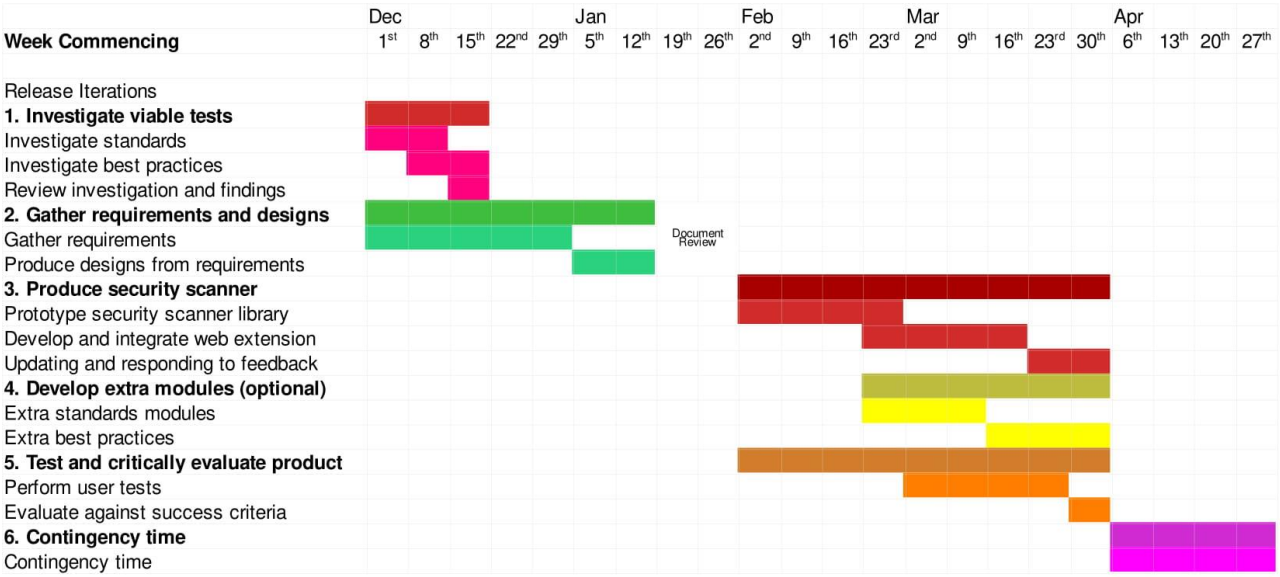
Yes

5.2 Has the checklist been approved by your supervisor? **Not received reply**

Section 6: Proposed Plan (please attach your Gantt chart below)

Edited by Dr Nan Jiang based on PH Section 4

BU Computing Programmes 2017-2018



Edited by Dr Nan Jiang based on PH Section 4

APPENDIX B – BOURNEMOUTH UNIVERSITY INITIAL RESEARCH ETHICS CHECKLIST

Bournemouth University

Research Ethics Checklist

Adapted for the use by Department of Computing and Informatics ONLY

1. Student Details

Name	Ryan Howell
School	Faculty of Science & Technology
Course	BSc Software Engineering
Have you received external funding to support this research project?	No
Please list any persons or institutions that you will be conducting joint research with, both internal to BU as well as external collaborators.	

2. Project Details

Title	Scanning Services for End-User Web browsing
Proposed Start Date	29-January-2018
Proposed End Date	31-August-2018 – **Note this is not the submission deadline!**
Supervisor	Gernot Liebchen

Summary (including detail on background methodology, sample, outcomes, etc.)

A system designed to make the security of a domain easy to understand and compare for non-technical end users.

Non-technical end users often have blind trust in the security of a domain when that site has an SSL certificate (especially EV) showing the green lock in their web browser. This project aims to give the user a more realistic view of the security of that domain.

This can be achieved by performing authorised analyses of the web server, to make sure they are properly configured according to common best practices and standards. E.g. HSTS, HPKP and CSP Headers.

The expected outcome is a library and web browser extension, clearly showing a simple colour dial (red, yellow, green), with a score of tests passed. Clicking on the browser extension should give more detailed information about success and failure.

Bournemouth University

Research Ethics Checklist

Adapted for the use by Department of Computing and Informatics ONLY

3. External Ethics Review (Answer "Yes" go to 4, "No" go to 5)

Does your research require external review through the NHS National Research Ethics Service (NRES) or through another external Ethics Committee?	No

4. External Ethics Review Continued

Answered "Yes" to question 3 will conclude the BU Ethics Review so you do not need to answer the following questions. Note you will need to obtain external ethical approval before commencing your research.

5. Research Literature (Answer "Yes" go to 6, "No" go to 7)

Is your research solely literature based?	Yes

6. Research Literature Continued (Either answer will conclude the review)

Will you have access to personal data that allows you to identify individuals OR access to confidential corporate or company data (that is not covered by confidentiality terms within an agreement or by a separate confidentiality agreement)?	No
Describe how you will collect, manage and store the personal data (taking into consideration the Data Protection Act and the Data Protection Principles).	

Bournemouth University

Research Ethics Checklist

Adapted for the use by Department of Computing and Informatics ONLY

7. Human Participants Part 1 (Answer "Yes" go to 8, "No" go to 12)

Will your research project involve interaction with human participants as primary sources of data (e.g. interview, observation, original survey)?	No

8. Human Participants Part 2 (Answer **any** "Yes" go to 9)

Does your research specifically involve participants who are considered vulnerable (i.e. children, those with cognitive impairment, those in unequal relationships—such as your own students, prison inmates, etc.)?	Choose an item.
Does the study involve participants age 16 or over who are unable to give informed consent (i.e. people with learning disabilities)? NOTE: All research that falls under the auspices of the Mental Capacity Act 2005 must be reviewed by NHS NRES.	Choose an item.
Will the study require the co-operation of a gatekeeper for initial access to the groups or individuals to be recruited? (i.e. students at school, members of self-help group, residents of Nursing home?)	Choose an item.
Will it be necessary for participants to take part in your study without their knowledge and consent at the time (i.e. covert observation of people in non-public places)?	Choose an item.
Will the study involve discussion of sensitive topics (i.e. sexual activity, drug use, criminal activity)?	Choose an item.

9. Human Participants Part 2 Continued

Describe how you will deal with the ethical issues with human participants?

Bournemouth University

Research Ethics Checklist

Adapted for the use by Department of Computing and Informatics ONLY

10. Human Participants Part 3 (Answer **any** "Yes" go to 11, **all** "No" go to 12)

Could your research induce psychological stress or anxiety, cause harm or have negative consequences for the participant or researcher (beyond the risks encountered in normal life)?	Choose an item.
Will your research involve prolonged or repetitive testing?	Choose an item.
Will the research involve the collection of audio materials?	Choose an item.
Will your research involve the collection of photographic or video materials?	Choose an item.
Will financial or other inducements (other than reasonable expenses and compensation for time) be offered to participants?	Choose an item.

11. Human Participants Part 3 Continued

Please explain below why your research project involves the above mentioned criteria (be sure to explain why the sensitive criterion is essential to your project's success). Give a summary of the ethical issues and any action that will be taken to address these. Explain how you will obtain informed consent (and from whom) and how you will inform the participant(s) about the research project (i.e. participant information sheet). A sample consent form and participant information sheet can be found on the Research Ethics website.

Bournemouth University

Research Ethics Checklist

Adapted for the use by Department of Computing and Informatics ONLY

12. Final Review

Will you have access to personal data that allows you to identify individuals OR access to confidential corporate or company data (that is not covered by confidentiality terms within an agreement or by a separate confidentiality agreement)?	No
Will your research take place outside the UK (including any and all stages of research: collection, storage, analysis, etc.)?	Yes
Please use the below text box to highlight any other ethical concerns or risks that may arise during your research that have not been covered in this form.	

Review Completion Date: 07-May-2018 – Double click to change it!

The following section is to be filled by the supervisor only

Supervisor's Review: Choose an item.

Please leave your comments:

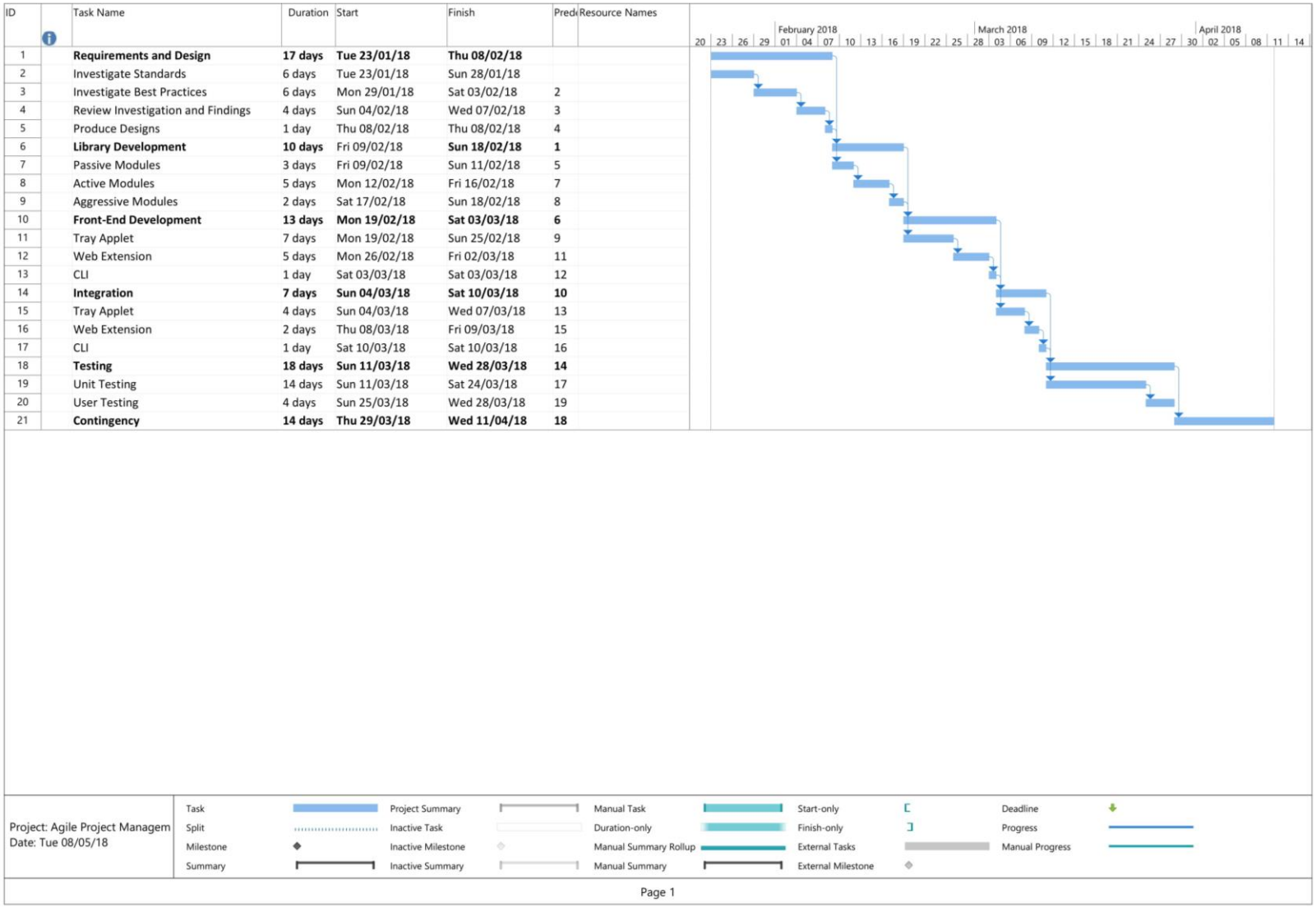
This is a technical investigation only, and it is carried out on test-servers.

07/05/2018

APPENDIX C – PROJECT PLANS

ID	Task Name	Duration	Start	Finish	Predecessors	Resource Names
1	**Requirements and Design**	**21 days?**	**Tue 23/01/18**	**Mon 12/02/18**		
2	Investigate Standards	7 days?	Tue 23/01/18	Mon 29/01/18		
3	Investigate Best Practices	7 days?	Tue 30/01/18	Mon 05/02/18	2	
4	Review Investigation and Findings	6 days?	Tue 06/02/18	Sun 11/02/18	3	
5	Produce Designs	1 day?	Mon 12/02/18	Mon 12/02/18	4	
6	**Library Development**	**21 days?**	**Tue 13/02/18**	**Mon 05/03/18**	**1**	
7	Passive Modules	5 days?	Tue 13/02/18	Sat 17/02/18	5	
8	Active Modules	11 days?	Sun 18/02/18	Wed 28/02/18	7	
9	Aggressive Modules	5 days?	Thu 01/03/18	Mon 05/03/18	8	
10	**Front-End Development**	**14 days?**	**Tue 06/03/18**	**Mon 19/03/18**	**6**	
11	Tray Applet	7 days?	Tue 06/03/18	Mon 12/03/18	9	
12	Web Extension	6 days?	Tue 13/03/18	Sun 18/03/18	11	
13	CLI	1 day?	Mon 19/03/18	Mon 19/03/18	12	
14	**Integration**	**7 days?**	**Tue 20/03/18**	**Mon 26/03/18**	**10**	
15	Tray Applet	4 days?	Tue 20/03/18	Fri 23/03/18	13	
16	Web Extension	2 days?	Sat 24/03/18	Sun 25/03/18	15	
17	CLI	1 day?	Mon 26/03/18	Mon 26/03/18	16	
18	**Testing**	**14 days?**	**Tue 27/03/18**	**Mon 09/04/18**	**14**	
19	Unit Testing	12 days?	Tue 27/03/18	Sat 07/04/18	17	
20	User Testing	2 days?	Sun 08/04/18	Mon 09/04/18	19	
21	**Contingency**	**14 days?**	**Tue 10/04/18**	**Mon 23/04/18**	**18**	

Planned Project Plan



ID	Task Name	Duration	Start	Finish	Pred	Resource Names
1	**Requirements and Design**	**17 days**	**Tue 23/01/18**	**Thu 08/02/18**		
2	Investigate Standards	6 days	Tue 23/01/18	Sun 28/01/18		
3	Investigate Best Practices	6 days	Mon 29/01/18	Sat 03/02/18	2	
4	Review Investigation and Findings	4 days	Sun 04/02/18	Wed 07/02/18	3	
5	Produce Designs	1 day	Thu 08/02/18	Thu 08/02/18	4	
6	**Library Development**	**10 days**	Fri 09/02/18	**Sun 18/02/18**	**1**	
7	Passive Modules	3 days	Fri 09/02/18	Sun 11/02/18	5	
8	Active Modules	5 days	Mon 12/02/18	Fri 16/02/18	7	
9	Aggressive Modules	2 days	Sat 17/02/18	Sun 18/02/18	8	
10	**Front-End Development**	**13 days**	**Mon 19/02/18**	**Sat 03/03/18**	**6**	
11	Tray Applet	7 days	Mon 19/02/18	Sun 25/02/18	9	
12	Web Extension	5 days	Mon 26/02/18	Fri 02/03/18	11	
13	CLI	1 day	Sat 03/03/18	Sat 03/03/18	12	
14	**Integration**	**7 days**	**Sun 04/03/18**	**Sat 10/03/18**	**10**	
15	Tray Applet	4 days	Sun 04/03/18	Wed 07/03/18	13	
16	Web Extension	2 days	Thu 08/03/18	Fri 09/03/18	15	
17	CLI	1 day	Sat 10/03/18	Sat 10/03/18	16	
18	**Testing**	**18 days**	**Sun 11/03/18**	**Wed 28/03/18**	**14**	
19	Unit Testing	14 days	Sun 11/03/18	Sat 24/03/18	17	
20	User Testing	4 days	Sun 25/03/18	Wed 28/03/18	19	
21	**Contingency**	**14 days**	**Thu 29/03/18**	**Wed 11/04/18**	**18**	

Actual Project Plan

APPENDIX D – SOFTWARE REQUIREMENTS SPECIFICATIONS

Modules	Module Type	Test Type	Description
Content Security Policy	Passive	Standard	Checks for and validates CSP HTTP header
HTTP Strict Transport Security	Passive	Standard	Checks for and validates HSTS HTTP header
Hypertext Transfer Protocol Secure	Passive	Standard	Checks whether https was used
Referrer Policy	Passive	Standard	Checks for and validates referrer policy HTTP header
Version Numbers	Passive	Best Practice	Checks for version numbers in HTTP header
Content Type Options	Passive	Standard	Checks for and validates xcontenttypeoptions HTTP header
Frame Options	Passive	Standard	Checks for and validates xframeoptions HTTP header
XSS Protection	Passive	Best Practice	Checks for and validates x-xss-protection HTTP header
Doman Name System	Active	Standard	Makes DNS requests, DS/CAA
Hypertext Transfer Protocol	Active	Best Practice	Makes an HTTP request, checks for status code
TLS Certificate	Active	Best Practice	Opens TLS socket, validates certificate
TLS Ciphers	Active	Best Practice	Opens TLS sockets, checks for supported cipher suites

TLS Online Certificate Status Protocol	Active	Standard	Opens TLS socket, checks for OCSP support/validates
TLS Perfect Forward Secrecy	Active	Best Practice	Opens TLS socket, checks for forward secrecy support
Heartbleed	Aggressive	Best Practice	Opens TLS socket, checks for Heartbleed
Ports	Aggressive	Best Practice	Checks for open ports, i.e. 3306

Module List

Normal User	Technical User	Developer	Server Administrator
As a normal user, I want to use a browser add-on to find more information about the security of websites so that I can make more informed decisions about the websites I use. I will be satisfied when the information is conveyed accurately and understandably via an icon in my browser and I can get further information by clicking on the icon should I wish.	As a technical user, I want to find out detailed TLS, HTTP Header and other vulnerability information about the security of websites, so that I am able to make better decisions about the websites I use. I will be satisfied when I can use a single tool to achieve this need. As a technical user, I want the ability to discover more information about the aspects of website domain security that I do not understand so I am able to learn new things. I will be satisfied when there are links to documentation in the full report page of the browser plugin.	As a developer, I want to be able to configure a security browser extension to switch between a quick mode and a full mode, so I can choose which types of analysis to run on my websites. I will be satisfied when I can choose this option either in the tray applet or the web extension and when the tray applet can auto-start for me when I turn on my computer.	As a server administrator, I want to use a security scanner so that I can check the security of all the domains which I own. I will be satisfied when I can use a command line scripting language to achieve this quickly and easily, as well as repeatedly if required.

User Stories

APPENDIX E – SOFTWARE DESIGNS

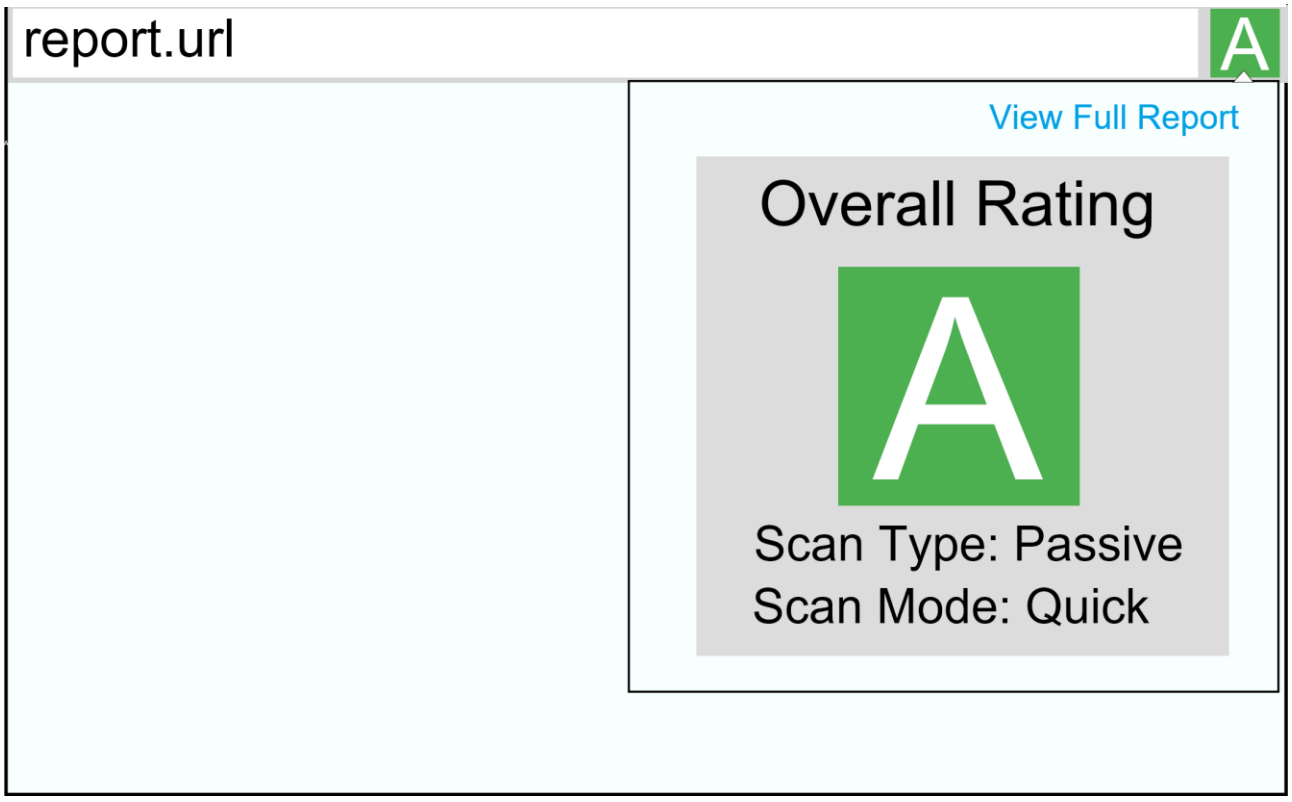
heimdall-cli

Usage: heimdall scan example1.com http://example2.com https://example3.com
Options:

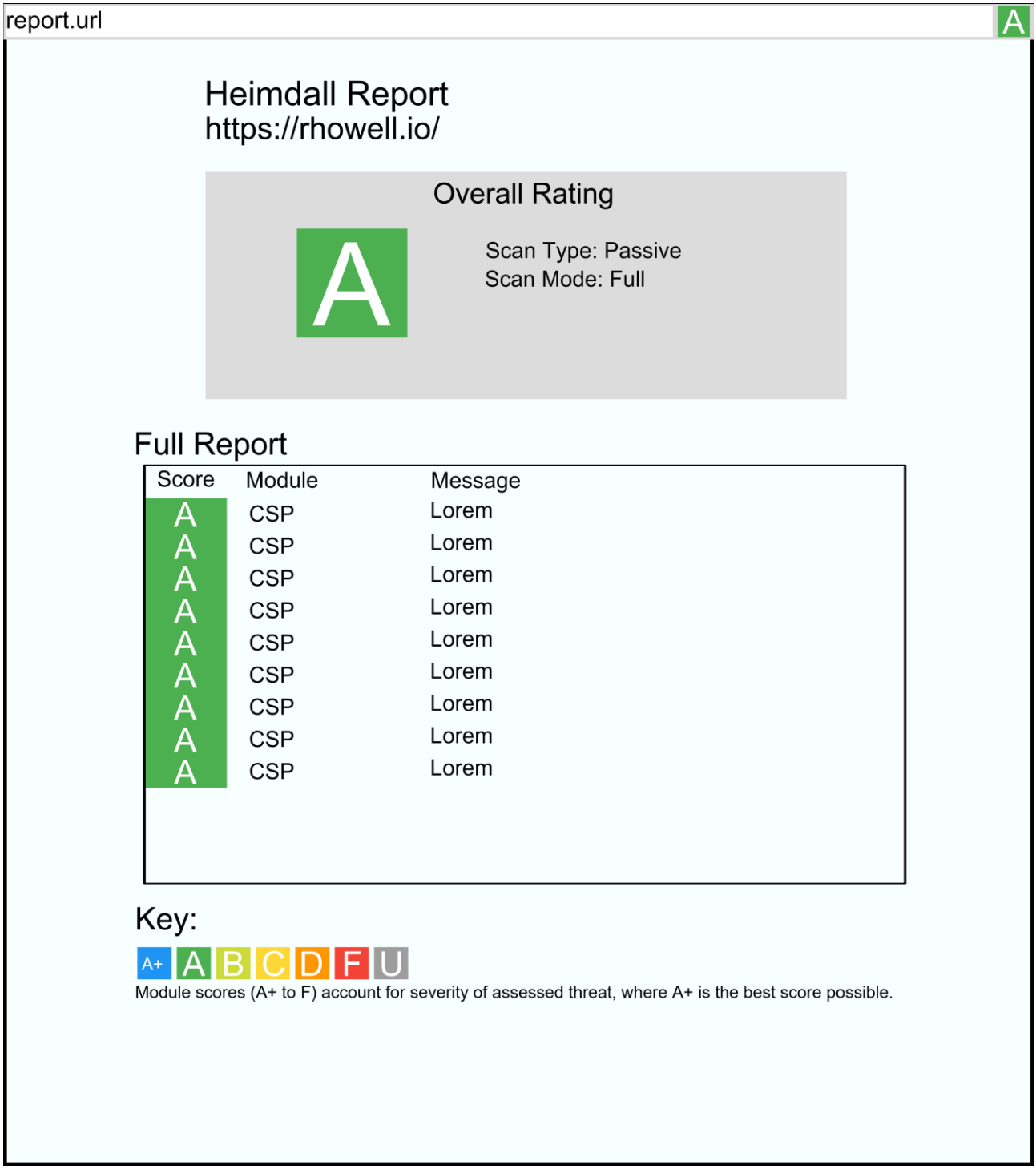
-V, --version	output the version number
-m --mode <mode>	Mode of scan (quick, full), defaults to quick.
-t --type <type>	Type of scan (passive, active, aggressive), defaults to passive.
-h, --help	output usage information

Commands:
scan <url> [otherUrls...]

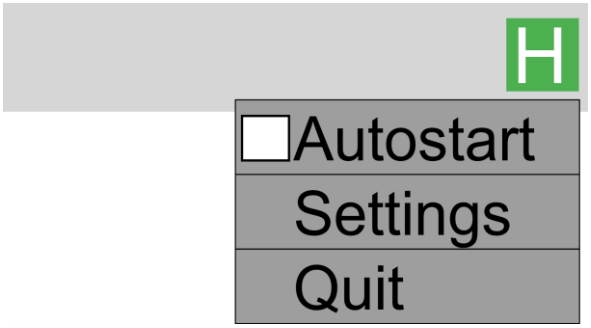
CLI Design



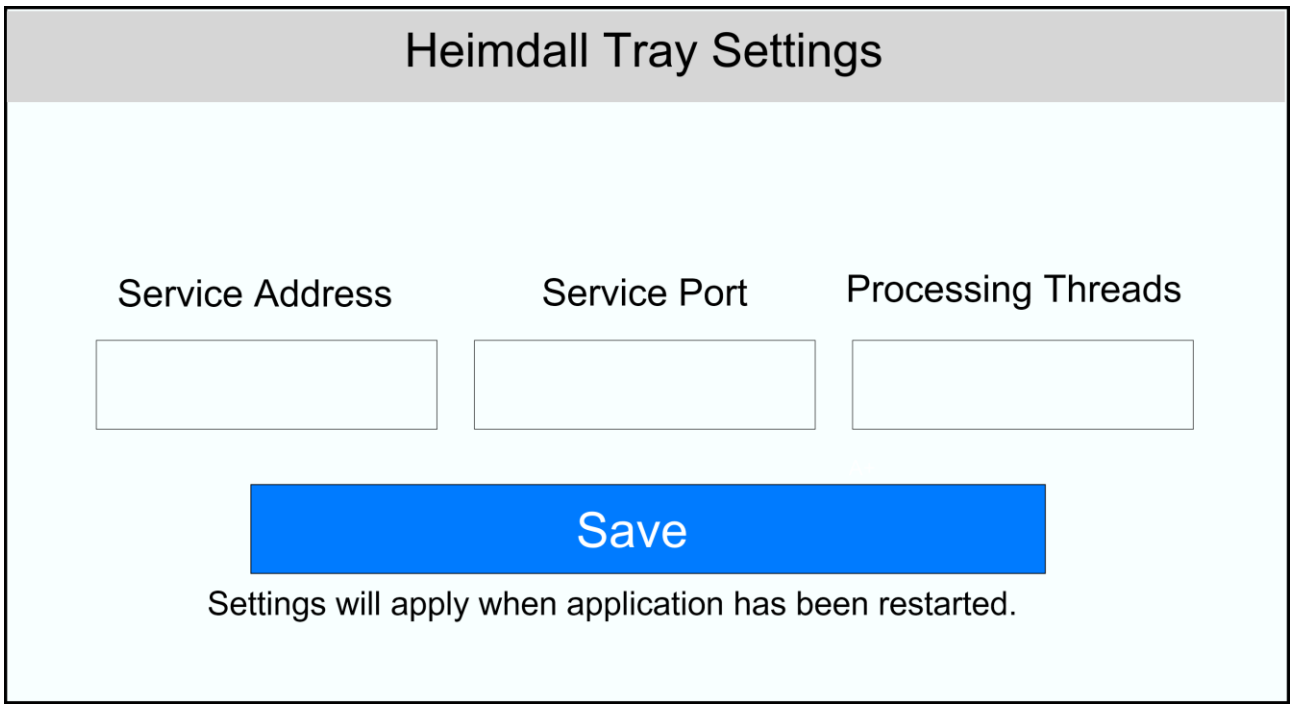
Web Extension Popup Design



Web Extension Report Design



Tray Applet Design



Tray Applet Settings

APPENDIX F – SOFTWARE DOCUMENTATION

Heimdall by Ryan Howell

Contents

Note from author: For ease of Installation, Windows and Firefox is recommended due to the pre-release nature of this software.

In a real-life scenario Heimdall would be published to the appropriate platform stores, i.e. Google Chrome Webstore, Firefox add-on store, MacOS brew etc.

On Linux, the tray applet requires app indicator support in your chosen desktop environment, e.g. Gnome TopIcons.

The chrome build of the web extension can only run on Linux for pre-release software. From google developer documentation: “Extensions hosted outside of the Chrome Web Store can only be installed by Linux users”.

For more information see: https://developer.chrome.com/extensions/linux_hosting

Installation

Windows 10 and Firefox

Web extension:

1: Open Mozilla Firefox.

2(a): Drag heimdall-*.xpi from the included CD to the Mozilla Firefox address bar.

- OR -

2(b): Open Firefox -> go-to about:addons (CTRL+SHIFT+A) -> Extensions -> Cog dropdown -> Install Add-on From File
Select heimdall-*.xpi from the included CD

3: Select *Add* from the addon popup that appears.

Tray Applet (Optional, used for active mode):

1: Double left click the relevant OpenSSL installer for your platform (Win32 or Win64).

2: When prompted select the default installation folder and the windows system directory for DLL location.

3: Double left click 'Heimdall Tray Setup*.exe' file to install it.

4: If Heimdall Tray Application is not present in the System tray (bottom right), run Heimdall Tray Application from the desktop icon.

5: Right click the Heimdall Tray Application icon in the system tray and left click *Autostart*.

6: Open Firefox and type '*about:addons*' in the address bar and then press the *enter* key.

7: Select *Extensions* from the left menu and then select *Options* under the Heimdall addon.

8: Change Scan Type to *Active* and click *Save*.

Command Line Interface (Optional, used for aggressive mode - intended for power users):

1: Launch a command prompt (cmd.exe)

2: Navigate to the directory containing heimdall-win.exe and type '*heimdall-win.exe --help*' and press *enter*

NB: All commands exclude quotations, for example one would type *heimdall-win.exe --help* and not '*heimdall-win.exe --help*'

Linux and Firefox

Web extension:

1: Open Mozilla Firefox.

2(a): Drag *heimdall-*.xpi* from the included CD to the Mozilla Firefox address bar.

- OR -

2(b): Open Firefox -> go-to *about:addons* (CTRL+SHIFT+A) -> Extensions -> Cog dropdown -> Install Add-on From File
Select *heimdall-*.xpi* from the included CD

3: Select '*Add*' from the addon popup that appears.

Tray Applet (Optional, used for active mode):

1: Install relevant package:
Deb: *sudo apt-get install ./heimdal-tray*.deb*
RPM: *sudo dnf install ./heimdal-tray*.rpm*

- OR -

sudo yum install ./heimdal-tray.rpm*
AppImage: Double click the *./heimdal-tray**.AppImage file

2: Run Heimdall Tray Application from your desktop environments App drawer

3: Right click the *Heimdall Tray* Application icon in your desktop environments app tray and left click '*Autostart*'.

4: Open Firefox and type '*about:addons*' in the address bar and then press the *enter* key.

5: Select *Extensions* from the left menu and then select *Preferences* under the Heimdall addon.

6: Change Scan Type to '*Active*' and click save.

Command Line Interface (Optional, used for aggressive mode - intended for power users):

1: Open a terminal

2: Copy *heimdall-macos* from the included data CD to the local computer

3: Navigate in Terminal to the directory containing *heimdall-macos* file

4: Type '*chmod +x heimdall-linux*'

5: Type '*./heimdall-macos --help*'

NB: All commands exclude quotations, for example one would type *heimdall-win.exe --help* and not '*heimdall-win.exe --help*'

MacOS High Sierra and Firefox

Web extension:

1: Open Mozilla Firefox

2(a): Drag *heimdall-*.xpi* from the included CD to the Mozilla Firefox address bar.

- OR -

2(b): Open Firefox -> go-to *about:addons* (SHIFT+CMD+A) -> Extensions -> Cog dropdown -> Install Add-on From File *heimdall-*.xpi* from the included CD

3: Select '*Add*' from the addon popup that appears

Tray Applet (Optional, used for active mode):

1: Open a terminal window from Applications -> Utilities Folder

2: Install Homebrew if you do not already have it installed, for more details https://brew.sh/

3: To install the OpenSSL dependency via brew, type the following in the Terminal window: '*brew install openssl*'

4: Install '*Heimdall Tray*.dmg*' file

5: If Heimdall Tray Application is not present in the System Bar (top right), run Heimdall Tray from MacOS Applications folder.

6: Left click the Heimdall Tray Application icon in the System Bar (top right) and left click '*Autostart*'.

7: Open Firefox and type '*about:addons*' in the address bar and then press the enter key.

7: Select 'Extensions' from the left menu and then select '*Preferences*' under the Heimdall addon.

8: Change Scan Type to '*Active*' and click *save*.

Command Line Interface (Optional, used for aggressive mode - intended for power users):

1: Open a terminal window from Applications -> Utilities Folder

2: Install Homebrew if you do not already have it installed, for more details https://brew.sh/

3: To install the OpenSSL dependency via brew, type the following in the Terminal window: '*brew install openssl*'

4: Copy *heimdall-macos* from the included data CD to the local computer

3: Navigate in Terminal to the directory containing *heimdall-macos* file

4: Type '*chmod +x heimdall-macos*'

5: Type '*./heimdall-macos --help*'

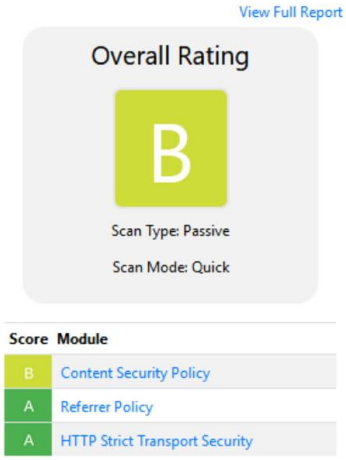
NB: All commands exclude quotations, for example one would type *brew install openssl* and not '*brew install openssl*'

Usage

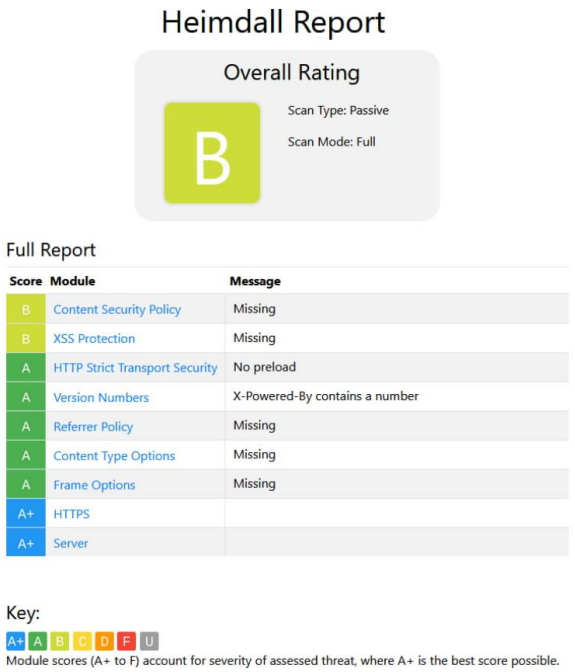
Extension

Open https://rhowell.io in Firefox, the extension Heimdall icon, by default located in the top right, will change when the quick report runs at page load. To see the quick report click the Heimdall extension icon in your browser window. By Default, this is on the right-hand side of the Mozilla Firefox or Google Chrome Browsers. To see the full report click the Heimdall icon and click "View Full Report".

Heimdall Extension Icon

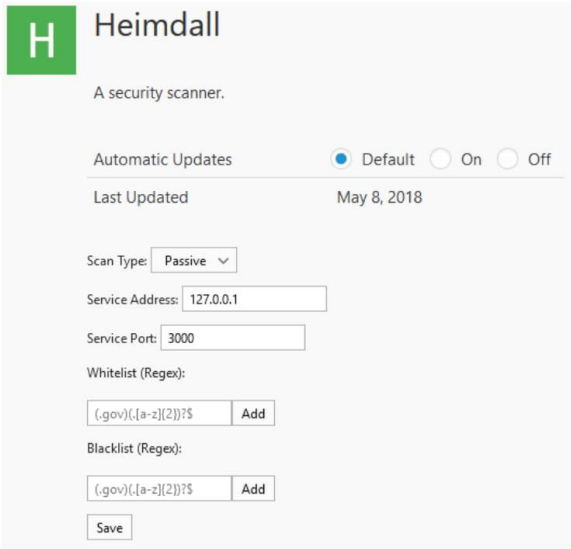


Example Quick Report Window



Example Full Report Window

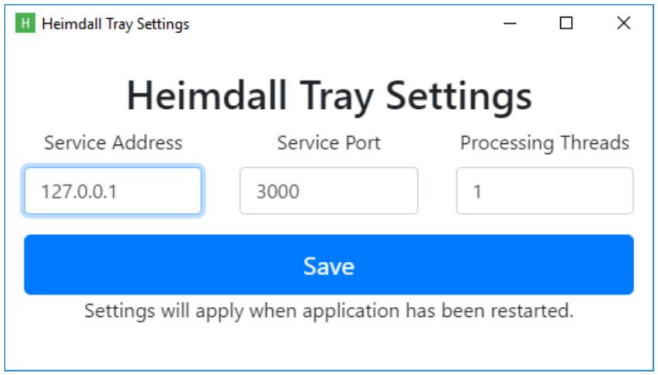
Open the add-on settings to change scan type, service address/port and the whitelist/blacklist. In order to use the active scan type the tray applet must be running and both the web extension and tray applet must be configured with matching address/port.



Heimdall Extension Settings

Tray Applet

Right-click on the tray applet to change settings or enable auto-start. The tray icon will change to a *H* when a connection is active.



Heimdall Tray Settings

CLI

Usage: heimdall scan example1.com http://example2.com https://example3.com

Options:

-V, --version	output the version number
-m --mode <mode>	Mode of scan (quick, full), defaults to quick.
-t --type <type>	Type of scan (passive, active, aggressive), defaults to passive.
-h, --help	output usage information

Commands:

scan <url> [otherUrls...]

APPENDIX G – SOFTWARE TEST PLANS

Test Number	Arguments	Expected Output	Actual Output	Notes
1	-V	Outputs the version number	Outputs the version number	
2	--version	Outputs the version number	Outputs the version number	
3	-m quick scan rhowell.io	Quick passive scan of rhowell.io	Quick passive scan of rhowell.io	
4	-m full scan rhowell.io	Full passive scan of rhowell.io	Full passive scan of rhowell.io	
5	-t passive scan rhowell.io	Quick passive scan of rhowell.io	Quick passive scan of rhowell.io	
6	-t active scan rhowell.io	Quick active scan of rhowell.io	Quick active scan of rhowell.io	
7	-t aggressive scan rhowell.io	Quick active scan of rhowell.io	Quick active scan of rhowell.io	allowaggressive flag false
8	-t aggressive scan rhowell.io	Quick aggressive scan of rhowell.io	Quick aggressive scan of rhowell.io	allowaggressive flag true
9	-t passive -m quick scan rhowell.io	Quick passive scan of rhowell.io	Quick passive scan of rhowell.io	
10	-t active -m quick scan rhowell.io	Quick active scan of rhowell.io	Quick active scan of rhowell.io	
11	-t aggressive -m quick scan rhowell.io	Quick active scan of rhowell.io	Quick active scan of rhowell.io	allowaggressive flag false
12	-t aggressive -m quick scan rhowell.io	Quick aggressive scan of rhowell.io	Quick aggressive scan of rhowell.io	allowaggressive flag true
13	-t passive -m full scan rhowell.io	Full passive scan of rhowell.io	Full passive scan of rhowell.io	
14	-t active -m full scan rhowell.io	Full active scan of rhowell.io	Full active scan of rhowell.io	
15	-t aggressive -m full scan rhowell.io	Full active scan of rhowell.io	Full active scan of rhowell.io	allowaggressive flag false
16	-t aggressive -m full scan rhowell.io	Full aggressive scan of rhowell.io	Full aggressive scan of rhowell.io	allowaggressive flag true
17	-h	Help dialog	Help dialog	
18	--help	Help dialog	Help dialog	

CLI Test Plan

Test Number	Action	Expected Output	Actual Output
1	Install extension.	Icon appears in toolbar, add-on appears in about:addons.	Icon appears in toolbar, add-on appears in settings.
2	Modify service address.	Different service address is used.	Different service address is used.
3	Modify service port.	Different service port is used.	Different service port is used.
4	Change scan type to passive.	Passive mode is used, no connections to tray applet made.	Passive mode is used, no connections to tray applet made.
5	Change scan type to active.	Active mode is used, no connections to tray applet made.	Active mode is used, no connections to tray applet made.
6	Add website to white-list.	White-list is used, can only load websites that are in white-list and not black-list.	White-list is used, can only load websites that are in white-list and not black-list.
7	Add website to black-list.	Black-list is used, can only load websites that are not black-list.	Black-list is used, can only load websites that are not black-list.
8	Load website	Heimdall runs, score displayed on icon.	Heimdall runs, score displayed on icon.
9	Click on icon	Quick report displayed.	Quick report displayed.
10	Click on “View Full Report”	Full report displayed.	Full report displayed.

Web Extension Test Plan

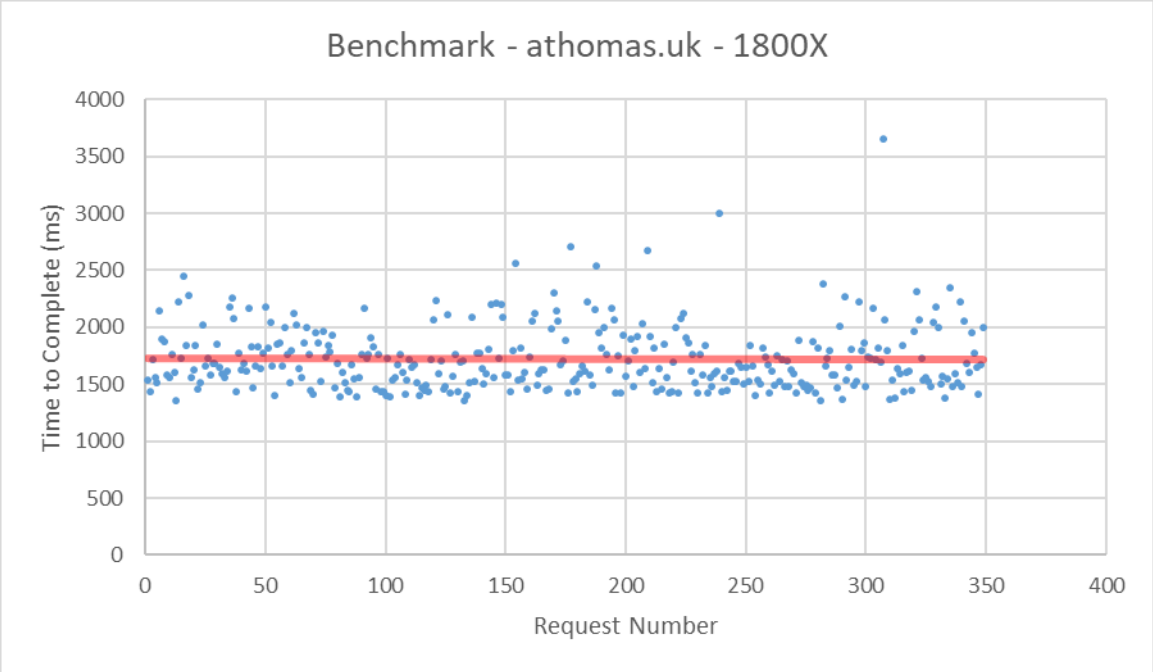
Test Number	Action	Expected Output	Actual Output
1	Launch tray applet	Tray applet is launched, icon appears in tray.	Tray applet is launched, icon appears in tray.
2	Enable auto-start.	Application auto-starts on login.	Application auto-starts on login.
3	Click exit.	Application exits.	Application exits.
4	Click settings.	Settings window appears.	Settings window appears.
5	Change service address.	Different service address is used.	Different service address is used.
6	Change service port.	Different service port is used.	Different service port is used.
7	Change number of processing threads to 0.	Cannot save form.	Cannot save form.
8	Change number of processing threads to 1.	One processing thread is used.	One processing thread is used.
9	Change number of processing threads to 2.	Two processing threads are used.	Two processing threads are used.
10	Change number of processing threads to higher than CPU thread count.	Cannot save form.	Cannot save form.
11	Click save in settings window.	Settings are saved. Application restarts.	Settings are saved. Application restarts.

Tray Applet Test Plan

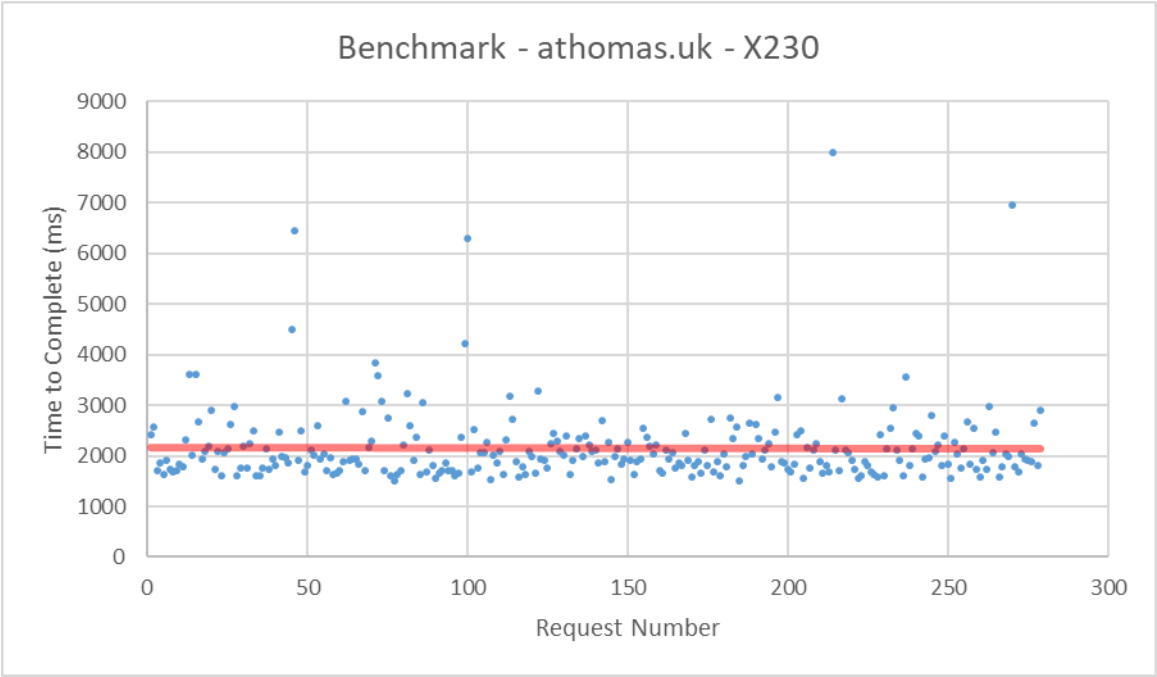
APPENDIX H – PERFORMANCE BENCHMARKS

Platform	RAM	Connection	Notes
AMD Ryzen 7 1800X Desktop	32GB	One gigabit wired.	The router was a Linksys 1900ACS running OpenWRT connected to a 40mb/10mb ADSL connection.
Lenovo X230 Intel Core i7-3520M Laptop	16GB	5GHz wireless.	The laptop was placed on a desk within one meter of the router and without physical obstructions between them.

Benchmark Hardware

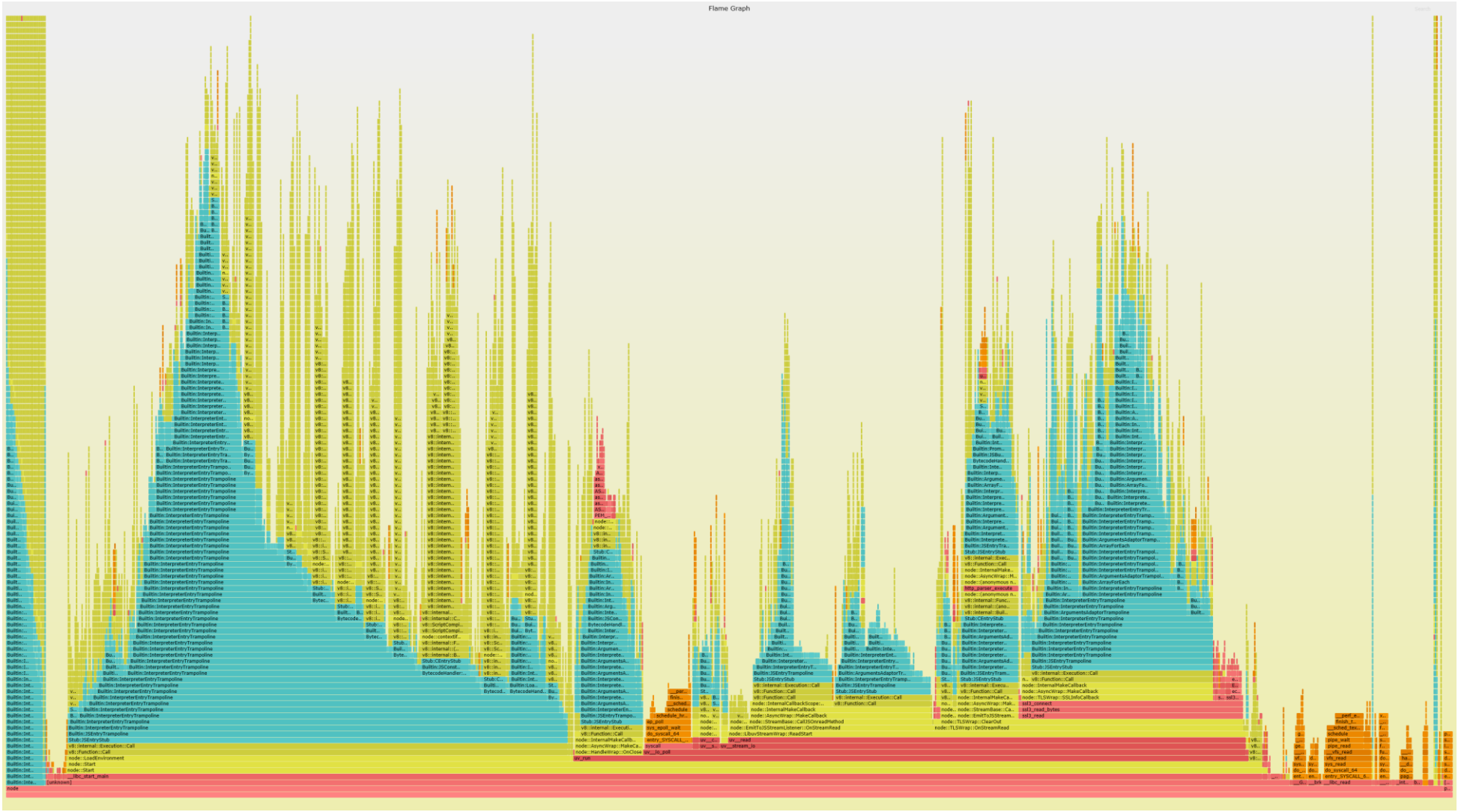


Benchmark - athomas.uk - 1800X



Benchmark - athomas.uk - X320

APPENDIX I – COMMAND-LINE APPLICATION CPU FLAME-GRAPH



APPENDIX J – SOFTWARE UNIT TEST RESULTS

```
dns
  run
    ✓ DS (7ms)
    ✓ CAA (1ms)
    ✓ DS CAA (1ms)
  failures
    ✓ nothing
    ✓ timeout (2005ms)

http
  run
    ✓ no redirect (1038ms)
    ✓ 301 redirect (983ms)
    ✓ 302 redirect (995ms)
  failures
    ✓ invalid url (1ms)
    ✓ invalid protocol
    ✓ nothing (7ms)

tlscert
  run
    ✓ rsa4096 (1030ms)
    ✓ rsa3072 (997ms)
    ✓ rsa2048 (1016ms)
    ✓ rsa1024 (1005ms)
    ✓ rsa768 (937ms)
    ✓ rsa512 (910ms)
    ✓ prime256v1 (1007ms)
    ✓ selfsigned (979ms)
    ✓ expired (947ms)
    ✓ untrusted (979ms)
  failures
    ✓ invalid url (2ms)
    ✓ nothing (7ms)

tlsciphers
  run
    ✓ modern (1325ms)
    ✓ intermediate (1482ms)
    ✓ old (1878ms)
    ✓ discard (1765ms)
    ✓ selfsigned (1634ms)
    ✓ expired (1290ms)
    ✓ untrusted (2330ms)
  failures
    ✓ invalid url (587ms)
    ✓ nothing (617ms)

tlsocsp
  run
    ✓ none (1ms)
    ✓ good (1ms)
    ✓ bad
    ✓ bad2
  failures
    ✓ invalid url

tlspfs
  run
    ✓ prime256v1 (1706ms)
    ✓ dh2048 (1345ms)
    ✓ dh1024 (1716ms)
    ✓ none (1551ms)
  failures
    ✓ invalid url (791ms)
    ✓ nothing (325ms)
```

```
heartbleed
  run
    ✓ vulnerable (991ms)
    ✓ not vulnerable (1079ms)

ports
  run
    ✓ none (6ms)
    ✓ 3306 (11ms)
    ✓ 5432 (4ms)
    ✓ 1433 (4ms)
    ✓ 1434 (6ms)
    ✓ 3050 (5ms)
    ✓ 2375 (4ms)
    ✓ 2376 (4ms)
    ✓ 3306 5432 (4ms)

index
  run
    ✓ rhowell.io allowagressive full (2870ms)
    ✓ rhowell.io allowagressive quick (6126ms)
    ✓ rhowell.io full (2065ms)
    ✓ rhowell.io quick (1459ms)
    ✓ athomas.uk allowagressive full (2294ms)
    ✓ athomas.uk allowagressive quick (4094ms)
    ✓ athomas.uk full (1468ms)
    ✓ athomas.uk quick (971ms)
  failures
    ✓ no url
    ✓ invalid url
    ✓ nothing allowagressive full (675ms)
    ✓ nothing allowagressive quick (172ms)
    ✓ nothing full (835ms)
    ✓ nothing quick (175ms)

csp
  run
    ✓ should exist
    ✓ should return promise (1ms)
    promise
      ✓ should return truthy value (1ms)
      code
        ✓ should exist
        ✓ should be a number
        ✓ should be at least 0 (1ms)
        ✓ should be at most 6
      name
        ✓ should exist
        ✓ should be string (3ms)
      message
        ✓ should exist
        ✓ should be string
  lowestState
    ✓ should exist
    ✓ should be a number
    ✓ hould be at least 0
    ✓ should be at most 6
  type
    ✓ should exist
    ✓ should be a number
    ✓ should be at least 0 (1ms)
    ✓ should be at most 6
  name
    ✓ should exist
    ✓ should be a string
  depends
    ✓ should exist
    ✓ should be an array
  states
    ✓ no header (1ms)
```

```
    ✓ exists
    ✓ perfect (1ms)
    default-src
      ✓ * (1ms)
      ✓ unsafe
      ✓ data (1ms)
    script-src
      ✓ *
      ✓ unsafe
      ✓ data
    style-src
      ✓ *
      ✓ data (1ms)
    font-src
      ✓ *
    connect-src
      ✓ *
    media-src
      ✓ * (1ms)
    object-src
      ✓ *
    child-src
      ✓ *
    frame-src
      ✓ *
    worker-src
      ✓ * (1ms)
    frame-ancestors
      ✓ *
    form-action
      ✓ *
    manifest-src
      ✓ *
    upgrade-insecure-requests
      ✓ missing (1ms)
    block-all-mixed-content
      ✓ missing
    disown-opener
      ✓ missing (1ms)
    require-sri-for
      ✓ missing
      ✓ missing script (1ms)
      ✓ missing style
    reflected-xss
      ✓ missing
      ✓ allow
    referrer
      ✓ missing
      ✓ missing

hsts
  run
    ✓ should exist
    ✓ should return promise
    promise
      ✓ should return truthy value
      code
        ✓ should exist
        ✓ should be a number (1ms)
        ✓ should be at least 0
        ✓ should be at most 6
      name
        ✓ should exist
```

```
        ✓ should be string
      message
        ✓ should exist
        ✓ should be string (1ms)
  lowestState
    ✓ should exist
    ✓ should be a number
    ✓ hould be at least 0 (1ms)
    ✓ should be at most 6
  type
    ✓ should exist
    ✓ should be a number (1ms)
    ✓ should be at least 0
    ✓ should be at most 6
  name
    ✓ should exist
    ✓ should be a string (1ms)
  depends
    ✓ should exist
    ✓ should be an array
  states
    ✓ no header
    ✓ no preload
    ✓ low max-age
    ✓ exact max-age with preload
    ✓ exact max-age without preload
    ✓ perfect
    ✓ bypass

https
  run
    ✓ should exist (1ms)
    ✓ should return promise
    promise
      ✓ should return truthy value (1ms)
      code
        ✓ should exist
        ✓ should be a number
        ✓ should be at least 0
        ✓ should be at most 6
      name
        ✓ should exist
        ✓ should be string
      message
        ✓ should exist
        ✓ should be string
  lowestState
    ✓ should exist
    ✓ should be a number (1ms)
    ✓ hould be at least 0
    ✓ should be at most 6
  type
    ✓ should exist
    ✓ should be a number
    ✓ should be at least 0
    ✓ should be at most 6
  name
    ✓ should exist
    ✓ should be a string
  depends
    ✓ should exist
    ✓ should be an array
  states
    ✓ http (1ms)
    ✓ https
```

```
  failures
    ✓ invalid url
    ✓ invalid protocol

poweredby
  run
    ✓ should exist
    ✓ should return promise (1ms)
    promise
      ✓ should return truthy value
      code
        ✓ should exist
        ✓ should be a number
        ✓ should be at least 0
        ✓ should be at most 6
      name
        ✓ should exist
        ✓ should be string (1ms)
      message
        ✓ should exist
        ✓ should be string
  lowestState
    ✓ should exist (1ms)
    ✓ should be a number
    ✓ hould be at least 0
    ✓ should be at most 6
  type
    ✓ should exist
    ✓ should be a number
    ✓ should be at least 0
    ✓ should be at most 6
  name
    ✓ should exist
    ✓ should be a string
  depends
    ✓ should exist
    ✓ should be an array
  states
    ✓ no header (1ms)
    ✓ exists
    ✓ contains number

referrerpolicy
  run
    ✓ should exist
    ✓ should return promise
    promise
      ✓ should return truthy value (1ms)
      code
        ✓ should exist
        ✓ should be a number
        ✓ should be at least 0
        ✓ should be at most 6
      name
        ✓ should exist
        ✓ should be string
      message
        ✓ should exist
        ✓ should be string
  lowestState
    ✓ should exist
    ✓ should be a number
    ✓ hould be at least 0
    ✓ should be at most 6
  type
    ✓ should exist
    ✓ should be a number (1ms)
    ✓ should be at least 0
    ✓ should be at most 6 (1ms)
  name
```

```
    ✓ should exist
    ✓ should be a string
  depends
    ✓ should exist
    ✓ should be an array (1ms)
  states
    ✓ no header
    ✓ exists (1ms)
    ✓ no-referrer
    ✓ unsafe-url
    ✓ origin

server
  run
    ✓ should exist
    ✓ should return promise (1ms)
    promise
      ✓ should return truthy value
      code
        ✓ should exist
        ✓ should be a number
        ✓ should be at least 0
        ✓ should be at most 6 (1ms)
      name
        ✓ should exist
        ✓ should be string
      message
        ✓ should exist (1ms)
        ✓ should be string
  lowestState
    ✓ should exist
    ✓ should be a number (1ms)
    ✓ hould be at least 0
    ✓ should be at most 6
  type
    ✓ should exist
    ✓ should be a number
    ✓ should be at least 0
    ✓ should be at most 6 (1ms)
  name
    ✓ should exist
    ✓ should be a string
  depends
    ✓ should exist
    ✓ should be an array
  states
    ✓ no header
    ✓ exists (1ms)
    ✓ contains number

xcontenttypeoptions
  run
    ✓ should exist
    ✓ should return promise
    promise
      ✓ should return truthy value
      code
        ✓ should exist
        ✓ should be a number (1ms)
        ✓ should be at least 0
        ✓ should be at most 6
      name
        ✓ should exist
        ✓ should be string
      message
        ✓ should exist
        ✓ should be string
  lowestState
    ✓ should exist
    ✓ should be a number
```

```
    ✓ hould be at least 0
    ✓ should be at most 6
  type
    ✓ should exist
    ✓ should be a number
    ✓ should be at least 0
    ✓ should be at most 6
  name
    ✓ should exist
    ✓ should be a string
  depends
    ✓ should exist
    ✓ should be an array
  states
    ✓ no header
    ✓ exists
    ✓ nosniff

xframeoptions
  run
    ✓ should exist
    ✓ should return promise
    promise
      ✓ should return truthy value (1ms)
      code
        ✓ should exist
        ✓ should be a number
        ✓ should be at least 0
        ✓ should be at most 6
      name
        ✓ should exist
        ✓ should be string
      message
        ✓ should exist (1ms)
        ✓ should be string
  lowestState
    ✓ should exist
    ✓ should be a number
    ✓ hould be at least 0
    ✓ should be at most 6
  type
    ✓ should exist
    ✓ should be a number
    ✓ should be at least 0
    ✓ should be at most 6
  name
    ✓ should exist
    ✓ should be a string (4ms)
  depends
    ✓ should exist
    ✓ should be an array
  states
    ✓ no header
    ✓ exists

xgenerator
  run
    ✓ should exist
    ✓ should return promise (1ms)
    promise
      ✓ should return truthy value
      code
        ✓ should exist
        ✓ should be a number
        ✓ should be at least 0
        ✓ should be at most 6
      name
        ✓ should exist
        ✓ should be string
      message
```

```
        ✓ should exist
        ✓ should be string
    lowestState
      ✓ should exist
      ✓ should be a number
      ✓ hould be at least 0
      ✓ should be at most 6
    type
      ✓ should exist
      ✓ should be a number
      ✓ should be at least 0
      ✓ should be at most 6
    name
      ✓ should exist
      ✓ should be a string
    depends
      ✓ should exist
      ✓ should be an array
    states
      ✓ no header
      ✓ exists (1ms)
      ✓ contains number

  xxssprotection
    run
      ✓ should exist
      ✓ should return promise
      promise
        ✓ should return truthy value
        code
          ✓ should exist
          ✓ should be a number (1ms)
          ✓ should be at least 0
          ✓ should be at most 6
        name
          ✓ should exist
          ✓ should be string
        message
          ✓ should exist
          ✓ should be string
    lowestState
      ✓ should exist (1ms)
      ✓ should be a number
      ✓ hould be at least 0
      ✓ should be at most 6
    type
      ✓ should exist
      ✓ should be a number (1ms)
      ✓ should be at least 0
      ✓ should be at most 6
    name
      ✓ should exist
      ✓ should be a string
    depends
      ✓ should exist
      ✓ should be an array
    states
      ✓ no header
      ✓ exists (1ms)
      ✓ off
      ✓ off mode
      ✓ off mode=invalid
      ✓ off mode=block
      ✓ on
      ✓ on mode
      ✓ on mode=invalid (1ms)
      ✓ on mode=block


  369 passing (1m)
```

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
All files	92.97	85.32	96.12	92.95	
heimdall-library	92.91	91.3	92.86	92.91	
index.js	92.91	91.3	92.86	92.91	... 241,245,246
heimdall-library/modules/active	87.94	73.53	95	87.87	
dns.js	98.41	84.38	100	98.41	106
http.js	100	86.67	100	100	65,72
tlscert.js	82.35	64.18	94.44	82.05	... 215,228,236
tlsciphers.js	78.67	70.83	92.86	78.67	... 128,153,157
tlsocsp.js	100	91.67	100	100	43
tlspfs.js	86.54	70	88.89	86.54	... ,95,112,116
heimdall-library/modules/aggressive	100	100	100	100	
heartbleed.js	100	100	100	100	
ports.js	100	100	100	100	
heimdall-library/modules/passive	100	98.11	100	100	
csp.js	100	98.36	100	100	37
hsts.js	100	90	100	100	41
https.js	100	100	100	100	
poweredby.js	100	100	100	100	
referrerpolicy.js	100	100	100	100	
server.js	100	100	100	100	
xcontenttypeoptions.js	100	100	100	100	
xframeoptions.js	100	100	100	100	
xgenerator.js	100	100	100	100	
xxssprotection.js	100	100	100	100	

APPENDIX K – SOFTWARE USABILITY TEST RESULTS

Heimdall User Survey

1. What type of user are you?

		Response Percent	Response Total
1	Normal User	33.33%	3
2	Technical User	22.22%	2
3	Developer	22.22%	2
4	System Administrator	22.22%	2
Analysis	Mean: 2.33 Std. Deviation: 1.15 Satisfaction Rate: 44.44 Variance: 1.33 Std. Error: 0.38	answered	9
		skipped	0

2. Which operating system do you use?

		Response Percent	Response Total
1	Windows	44.44%	4
2	Linux	33.33%	3
3	MacOS	22.22%	2
Analysis	Mean: 1.78 Std. Deviation: 0.79 Satisfaction Rate: 38.89 Variance: 0.62 Std. Error: 0.26	answered	9
		skipped	0

3. Which web browser do you use?

		Response Percent	Response Total
1	Google Chrome	22.22%	2
2	Mozilla Firefox	77.78%	7
Analysis	Mean: 1.78 Std. Deviation: 0.42 Satisfaction Rate: 77.78 Variance: 0.17 Std. Error: 0.14	answered	9
		skipped	0

4. Browser Extension

	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree	Response Total
The browser extension was easy to install	0.0% (0)	11.1% (1)	22.2% (2)	55.6% (5)	11.1% (1)	9
The browser extension was easy to configure	0.0% (0)	0.0% (0)	11.1% (1)	55.6% (5)	33.3% (3)	9

4. Browser Extension

	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree	Response Total
The browser extension was easy to use	0.0% (0)	0.0% (0)	11.1% (1)	22.2% (2)	66.7% (6)	9
I found the web extension useful	0.0% (0)	0.0% (0)	11.1% (1)	44.4% (4)	44.4% (4)	9
I felt better informed about website security when browsing the internet	0.0% (0)	0.0% (0)	0.0% (0)	44.4% (4)	55.6% (5)	9
					answered	9
					skipped	0

Further Comments: (9)

1	Better than I was expecting!
2	Add-on would be simpler to install via the webstore.
3	Running the unpacked extension was hard, needs uploading to the web-store. Would be more understandable/useful information if you described what the modules did and gave a link for further reading.
4	Reports are hard to understand, don't know what each thing is.
5	Thankyou I had brew already so the installation was easier
6	Good to see sites where I spent time hardening getting a better score.
7	Really easy to install and use, I was impressed with what you'd done here.
8	-
9	Install went smoothly

Matrix Charts

4.1. The browser extension was easy to install			Response Percent	Response Total
1	Strongly Disagree		0.0%	0
2	Disagree		11.1%	1
3	Neither Agree nor Disagree		22.2%	2
4	Agree		55.6%	5
5	Strongly Agree		11.1%	1
Analysis	Mean: 3.67 / Std. Deviation: 0.82 / Satisfaction Rate: 66.67 / Variance: 0.67 / Std. Error: 0.27		answered	9

4.2. The browser extension was easy to configure			Response Percent	Response Total
1	Strongly Disagree		0.0%	0

4.2. The browser extension was easy to configure		Response Percent	Response Total
2	Disagree	0.0%	0
3	Neither Agree nor Disagree	11.1%	1
4	Agree	55.6%	5
5	Strongly Agree	33.3%	3
Analysis	Mean: 4.22 Std. Deviation: 0.63 Satisfaction Rate: 80.56 Variance: 0.4 Std. Error: 0.21	answered	9

4.3. The browser extension was easy to use		Response Percent	Response Total
1	Strongly Disagree	0.0%	0
2	Disagree	0.0%	0
3	Neither Agree nor Disagree	11.1%	1
4	Agree	22.2%	2
5	Strongly Agree	66.7%	6
Analysis	Mean: 4.56 Std. Deviation: 0.68 Satisfaction Rate: 88.89 Variance: 0.47 Std. Error: 0.23	answered	9

4.4. I found the web extension useful		Response Percent	Response Total
1	Strongly Disagree	0.0%	0
2	Disagree	0.0%	0
3	Neither Agree nor Disagree	11.1%	1
4	Agree	44.4%	4
5	Strongly Agree	44.4%	4
Analysis	Mean: 4.33 Std. Deviation: 0.67 Satisfaction Rate: 83.33 Variance: 0.44 Std. Error: 0.22	answered	9

4.5. I felt better informed about website security when browsing the internet		Response Percent	Response Total
1	Strongly Disagree	0.0%	0
2	Disagree	0.0%	0
3	Neither Agree nor Disagree	0.0%	0
4	Agree	44.4%	4
5	Strongly Agree	55.6%	5
Analysis	Mean: 4.56 Std. Deviation: 0.5 Satisfaction Rate: 88.89 Variance: 0.25 Std. Error: 0.17	answered	9

5. Tray Applet						
	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree	Response Total
The tray applet was easy to install	0.0% (0)	0.0% (0)	0.0% (0)	44.4% (4)	55.6% (5)	9
The tray applet was easy to configure	0.0% (0)	0.0% (0)	0.0% (0)	44.4% (4)	55.6% (5)	9
The tray applet was easy to use	0.0% (0)	0.0% (0)	0.0% (0)	33.3% (3)	66.7% (6)	9
Heimdall's active mode provided more informative results	0.0% (0)	0.0% (0)	22.2% (2)	22.2% (2)	55.6% (5)	9
					answered	9
					skipped	0

Further Comments: (4)

1	Great simple tray applet.
2	It made more modules appear in reports and scores lower in places.
3	Great tray applet, adds more detailed information.
4	-

Matrix Charts

5.1. The tray applet was easy to install			Response Percent	Response Total
1	Strongly Disagree		0.0%	0
2	Disagree		0.0%	0
3	Neither Agree nor Disagree		0.0%	0
4	Agree		44.4%	4
5	Strongly Agree		55.6%	5
Analysis	Mean: 4.56 | Std. Deviation: 0.5 | Satisfaction Rate: 88.89 Variance: 0.25 | Std. Error: 0.17		answered	9

5.2. The tray applet was easy to configure			Response Percent	Response Total
1	Strongly Disagree		0.0%	0
2	Disagree		0.0%	0
3	Neither Agree nor Disagree		0.0%	0
Analysis	Mean: 4.56 | Std. Deviation: 0.5 | Satisfaction Rate: 88.89 Variance: 0.25 | Std. Error: 0.17		answered	9

5.2. The tray applet was easy to configure			Response Percent	Response Total
4	Agree		44.4%	4
5	Strongly Agree		55.6%	5
Analysis	Mean: 4.56 | Std. Deviation: 0.5 | Satisfaction Rate: 88.89 Variance: 0.25 | Std. Error: 0.17		answered	9

5.3. The tray applet was easy to use			Response Percent	Response Total
1	Strongly Disagree		0.0%	0
2	Disagree		0.0%	0
3	Neither Agree nor Disagree		0.0%	0
4	Agree		33.3%	3
5	Strongly Agree		66.7%	6
Analysis	Mean: 4.67 | Std. Deviation: 0.47 | Satisfaction Rate: 91.67 Variance: 0.22 | Std. Error: 0.16		answered	9

5.4. Heimdall's active mode provided more informative results			Response Percent	Response Total
1	Strongly Disagree		0.0%	0
2	Disagree		0.0%	0
3	Neither Agree nor Disagree		22.2%	2
4	Agree		22.2%	2
5	Strongly Agree		55.6%	5
Analysis	Mean: 4.33 | Std. Deviation: 0.82 | Satisfaction Rate: 83.33 Variance: 0.67 | Std. Error: 0.27		answered	9

6. Command Line Interface

	Not Applicable	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree	Response Total
The command line interface was easy to use	55.6% (5)	0.0% (0)	0.0% (0)	0.0% (0)	22.2% (2)	22.2% (2)	9
The script I wrote in conjunction with the Heimdall's CLI helped me to automate the gathering of results	66.7% (6)	0.0% (0)	0.0% (0)	0.0% (0)	0.0% (0)	33.3% (3)	9

6. Command Line Interface

	Not Applicable	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree	Response Total
Heimdall's active mode provided more informative results	55.6% (5)	0.0% (0)	0.0% (0)	0.0% (0)	22.2% (2)	22.2% (2)	9
						answered	9
						skipped	0

Further Comments: (4)

1	I didn't use this sorry
2	All parameters are non-interactive, made scripting very easy.
3	Simple consistent CLI, only wrote a simple bash script but I see how powerful this could be with further scripting.
4	The CLI was a nice touch

Matrix Charts

6.1. The command line interface was easy to use		Response Percent	Response Total
1	Not Applicable	55.6%	5
2	Strongly Disagree	0.0%	0
3	Disagree	0.0%	0
4	Neither Agree nor Disagree	0.0%	0
5	Agree	22.2%	2
6	Strongly Agree	22.2%	2
Analysis	Mean: 3 Std. Deviation: 2.26 Satisfaction Rate: 40 Variance: 5.11 Std. Error: 0.75	answered	9

6.2. The script I wrote in conjunction with the Heimdall's CLI helped me to automate the gathering of results		Response Percent	Response Total
1	Not Applicable	66.7%	6
2	Strongly Disagree	0.0%	0
3	Disagree	0.0%	0
4	Neither Agree nor Disagree	0.0%	0
5	Agree	0.0%	0
6	Strongly Agree	33.3%	3
Analysis	Mean: 2.67 Std. Deviation: 2.36 Satisfaction Rate: 33.33 Variance: 5.56 Std. Error: 0.79	answered	9

6.3. Heimdall's active mode provided more informative results			Response Percent	Response Total
1	Not Applicable		55.6%	5
2	Strongly Disagree		0.0%	0
3	Disagree		0.0%	0
4	Neither Agree nor Disagree		0.0%	0
5	Agree		22.2%	2
6	Strongly Agree		22.2%	2
Analysis	Mean: 3 / Variance: 5.11	Std. Deviation: 2.26 / Std. Error: 0.75	Satisfaction Rate: 40	answered 9

7. Scoring System and reports

	Not Applicable	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree	Response Total
I found the scoring system easy to understand	0.0% (0)	0.0% (0)	0.0% (0)	0.0% (0)	22.2% (2)	77.8% (7)	9
The icon colour coding was more useful than the A-F letter values	0.0% (0)	0.0% (0)	0.0% (0)	44.4% (4)	44.4% (4)	11.1% (1)	9
The quick report was easy to understand	0.0% (0)	0.0% (0)	0.0% (0)	11.1% (1)	55.6% (5)	33.3% (3)	9
The quick report provided me with the information I wanted	0.0% (0)	0.0% (0)	0.0% (0)	22.2% (2)	44.4% (4)	33.3% (3)	9
The full report was easy to understand	22.2% (2)	0.0% (0)	0.0% (0)	11.1% (1)	44.4% (4)	22.2% (2)	9
The full report provided me with the information I wanted	22.2% (2)	0.0% (0)	0.0% (0)	11.1% (1)	22.2% (2)	44.4% (4)	9
						answered	9
						skipped	0

Further Comments: (9)

1	I didn't view the full report but the quick report was well laid out.
2	Simple to understand, like SSL Labs I've used before.
3	You need to explain what the modules do in the reports.
4	Reports are hard to understand, don't know what each thing does.
5	I found both the colour and letter on the icon useful in equal measure
6	Good reports, will be useful to show colleagues the difference hardening makes.

7. Scoring System and reports

	Not Applicable	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree	Response Total
7	On the whole I found the reports to be very insightful, they provide a similar style of information that we're used to seeing when working with security partners to analyse our domains and applications. Particularly flagging things like TLS signature algorithms, HTTP redirects and the contents of headers is very helpful. Those are elements of configuration that we have a good amount of control over and as such can be remedied quite quickly.						
8	Since it sounds like you want to inform users about security, it'd help non-nerds if you make the module names in the reports into links so they could get more info						
9	The links on the report pages were useful to me as I didn't understand what the names meant. I liked the colour coding of the score symbol too and that it was next to my popup blocker icon so I knew where to look for it.						

Matrix Charts

7.1. I found the scoring system easy to understand			Response Percent	Response Total
1	Not Applicable		0.0%	0
2	Strongly Disagree		0.0%	0
3	Disagree		0.0%	0
4	Neither Agree nor Disagree		0.0%	0
5	Agree		22.2%	2
6	Strongly Agree		77.8%	7
Analysis	Mean: 5.78 / Std. Deviation: 0.42 / Satisfaction Rate: 95.56 / Variance: 0.17 / Std. Error: 0.14		answered	9

7.2. The icon colour coding was more useful than the A-F letter values			Response Percent	Response Total
1	Not Applicable		0.0%	0
2	Strongly Disagree		0.0%	0
3	Disagree		0.0%	0
4	Neither Agree nor Disagree		44.4%	4
5	Agree		44.4%	4
6	Strongly Agree		11.1%	1
Analysis	Mean: 4.67 / Std. Deviation: 0.67 / Satisfaction Rate: 73.33 / Variance: 0.44 / Std. Error: 0.22		answered	9

7.3. The quick report was easy to understand	Response Percent	Response Total

7.3. The quick report was easy to understand		Response Percent	Response Total
1	Not Applicable	0.0%	0
2	Strongly Disagree	0.0%	0
3	Disagree	0.0%	0
4	Neither Agree nor Disagree	11.1%	1
5	Agree	55.6%	5
6	Strongly Agree	33.3%	3
Analysis	Mean: 5.22 Std. Deviation: 0.63 Satisfaction Rate: 84.44 Variance: 0.4 Std. Error: 0.21	answered	9

7.4. The quick report provided me with the information I wanted		Response Percent	Response Total
1	Not Applicable	0.0%	0
2	Strongly Disagree	0.0%	0
3	Disagree	0.0%	0
4	Neither Agree nor Disagree	22.2%	2
5	Agree	44.4%	4
6	Strongly Agree	33.3%	3
Analysis	Mean: 5.11 Std. Deviation: 0.74 Satisfaction Rate: 82.22 Variance: 0.54 Std. Error: 0.25	answered	9

7.5. The full report was easy to understand		Response Percent	Response Total
1	Not Applicable	22.2%	2
2	Strongly Disagree	0.0%	0
3	Disagree	0.0%	0
4	Neither Agree nor Disagree	11.1%	1
5	Agree	44.4%	4
6	Strongly Agree	22.2%	2
Analysis	Mean: 4.22 Std. Deviation: 1.81 Satisfaction Rate: 64.44 Variance: 3.28 Std. Error: 0.6	answered	9

7.6. The full report provided me with the information I wanted		Response Percent	Response Total
1	Not Applicable	22.2%	2
Analysis	Mean: 4.44 Std. Deviation: 1.95 Satisfaction Rate: 68.89 Variance: 3.8 Std. Error: 0.65	answered	9

7.6. The full report provided me with the information I wanted		Response Percent	Response Total
2	Strongly Disagree	0.0%	0
3	Disagree	0.0%	0
4	Neither Agree nor Disagree	11.1%	1
5	Agree	22.2%	2
6	Strongly Agree	44.4%	4
Analysis	Mean: 4.44 | Std. Deviation: 1.95 | Satisfaction Rate: 68.89 | Variance: 3.8 | Std. Error: 0.65	answered	9

8. Did you find Heimdall useful?

		Response Percent	Response Total
1	Yes	100.00%	9
2	Somewhat	0.00%	0
3	No	0.00%	0
Analysis	Mean: 1 | Std. Deviation: 0 | Satisfaction Rate: 0 | Variance: 0 | Std. Error: 0	answered	9
		skipped	0

Further Comments: (5)

1	Gives an accurate representation of how good the security configuration is.
2	Gives me an indicator of how secure a site is.
3	It's good I plan to keep on using it
4	Found a few things we didn't expect, see above
5	-

9. Do you plan to use Heimdall in the future?

		Response Percent	Response Total
Analysis	Mean: 3.44 | Std. Deviation: 0.83 | Satisfaction Rate: 61.11 | Variance: 0.69 | Std. Error: 0.28	answered	9
		skipped	0

Further Comments: (7)

1	I thought it was good and the little icon was easy to read, just a shame it isnt out for my ipad
2	Useful tool as a developer, will also run during my normal browsing usage.
3	Will continue to run on my laptop at work.
4	Useful for comparing site security, choosing which to use.
5	Installed on my laptop and desktop.
6	I will use it when we need to check something.
7	Both for personal browsing and the CLI scripting stuff.

9. Do you plan to use Heimdall in the future?

		Response Percent	Response Total
1	Never	0.00%	0
2	Occasionally	11.11%	1
3	Fairly Often	44.44%	4
4	Always	33.33%	3
5	Undecided	11.11%	1
Analysis	Mean: 3.44 | Std. Deviation: 0.83 | Satisfaction Rate: 61.11 Variance: 0.69 | Std. Error: 0.28	answered	9
		skipped	0

Further Comments: (7)

1	I thought it was good and the little icon was easy to read, just a shame it isnt out for my ipad
2	Useful tool as a developer, will also run during my normal browsing usage.
3	Will continue to run on my laptop at work.
4	Useful for comparing site security, choosing which to use.
5	Installed on my laptop and desktop.
6	I will use it when we need to check something.
7	Both for personal browsing and the CLI scripting stuff.

10. Any other comments?

		Response Percent	Response Total
1	Open-Ended Question	100.00%	7

1	No
2	Would be great to be able to configure the extension to block rendering of a page below a certain threshold. Would be useful as a developer to have the the ability to sideload new modules into Heimdall without recompilation.
3	Great project, look forward to public release with easier installation.
4	Great work, would recommend to others when fully released.
5	In terms of suggestion for improvement, I'd say that as well as identifying configuration issues it'd be good to suggest improvements and best practice. It's good that you show a message indicating the nature of the problem but some suggestion on how best to fix it would be good for users who maybe not as technically proficient. As I say the quality of information provided is excellent, really nice job.
6	Good work Ryan. It will be interesting to see where you take this if you carry the work on after University.
7	Thanks ryan it was good.

	answered	7
	skipped	2

APPENDIX L – DISK CONTENTS AND ACCESS INSTRUCTIONS

File/Folder	Description
CLI	Folder containing CLI binaries.
CLI/heimdall-linux	Heimdall CLI binary for linux.
CLI/heimdall-macos	Heimdall CLI binary for MacOS.
CLI/heimdall-win.exe	Heimdall CLI binary for Windows.
Web Extensions	Folder containing web extensions.
Web Extensions/heimdall-0.0.14-an+fx.crx	Heimdall Chrome extension.
Web Extensions/heimdall-0.0.14-an+fx.xpi	Heimdall Firefox add-on.
Tray Applet	Folder containing Heimdall Tray Applet.
Tray Applet/Linux	Folder containing Heimdall Tray Applet Linux packages.
Tray Applet/Linux/heimdall-tray_0.0.8_amd64.deb	Heimdall Tray Applet DEB package, AMD64.
Tray Applet/Linux/heimdall-tray_0.0.8_i386.deb	Heimdall Tray Applet DEB package, i386.
Tray Applet/Linux/heimdall-tray-0.0.8.i686.rpm	Heimdall Tray Applet RPM package, i686.
Tray Applet/Linux/heimdall-tray-0.0.8.x86_64.rpm	Heimdall Tray Applet RPM package, x86_64.
Tray Applet/Linux/heimdall-tray-0.0.8-i386.AppImage	Heimdall Tray Applet AppImage package, i386.
Tray Applet/Linux/heimdall-tray-0.0.8-x86_64.AppImage	Heimdall Tray Applet AppImage package, x86_64.
Tray/MacOS	Folder containing Heimdall Tray Applet MacOS packages.
Tray/MacOS/Heimdall Tray-0.0.8.dmg	Heimdall Tray Applet DMG package.
Tray/Windows	Folder containing Heimdall Tray Applet Windows installer, and OpenSSL dependency.
Tray/Windows/Heimdall Tray Setup 0.0.8.exe	Heimdall Tray Applet installer for Windows.
Tray/Windows/Win32OpenSSL-1_1_0g.exe	OpenSSL Installer, 1.1.0g, Win32.
Tray/Windows/Win64OpenSSL-1_1_0g.exe	OpenSSL Installer, 1.1.0g, Win64.
README.pdf	Heimdall ReadMe file, contains installation and usage instructions.

APPENDIX M – GLOSSARY

Name	Description	Link
Address	An address in Internet Protocol, i.e. 127.0.0.1 or 1.1.1.1	https://en.wikipedia.org/wiki/IP_address
Agile	A method of software development focused on flexibility in responding to change.	https://en.wikipedia.org/wiki/Agile_software_development
Apache	A popular HTTP server, used for serving web pages and content.	https://en.wikipedia.org/wiki/Apache_HTTP_Server
Application Programming Interface (API)	A defined structured interface that allows one application to communicate to another, with consistent variable / function position and naming scheme.	https://en.wikipedia.org/wiki/Application_programming_interface
BEAST	A method of attack against TLS.	https://en.wikipedia.org/wiki/Transport_Layer_Security#BEAST_attack
Black-box	Tests that are not derived from source code.	https://en.wikipedia.org/wiki/Black-box_testing
Blacklist	A list of entities to be avoided.	https://en.wikipedia.org/wiki/Blacklisting
Certificate Algorithm	The algorithm used within the certificate, e.g. RSA, ECC.	https://en.wikipedia.org/wiki/RSA_(cryptosystem)
Certificate Authority	An entity that issues certificates.	https://en.wikipedia.org/wiki/Certificate_authority
Certificate Key	A cryptographic key used within a certificate, e.g. an RSA public key.	https://en.wikipedia.org/wiki/Public_key_certificate
Certificate Transparency	A standard for monitoring and auditing the issuing of certificates.	https://en.wikipedia.org/wiki/Certificate_Transparency
Certification Authority Authorization (CAA)	A standard for indicating what Certificate Authorities can issue certificates for a domain.	https://en.wikipedia.org/wiki/DNS_Certification_Authority_Authorization
Cipher	An algorithm for performing encryption or decryption.	https://en.wikipedia.org/wiki/Cipher
Click-jacking	A malicious method of hijacking the users cursor to perform an action undesired by the user.	https://en.wikipedia.org/wiki/Clickjacking
Command Line Interface (CLI)	A text-based interaction model for computer programs.	https://en.wikipedia.org/wiki/Command-line_interface
Content Security Policy (CSP)	A standard for indicating what content the browser should be allowed to display for a domain.	https://en.wikipedia.org/wiki/Content_Security_Policy
Content Type Options Header	A standard for indicating whether content type sniffing algorithms should be used.	https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Content-Type-Options

Central Processing Unit (CPU) Architectures	A model which translates a set of instructions into physical operations performed by the CPU.	https://en.wikipedia.org/wiki/Instruction_set_architecture
Cross-Site Scripting (XSS)	A method of attack that injects client side scripts into web pages.	https://en.wikipedia.org/wiki/Cross-site_scripting
Cryptography	The practice of secure communication, i.e. encryption.	https://en.wikipedia.org/wiki/Cryptography
Domain Name System (DNS)	A system for translating domain names into IP Addresses.	https://en.wikipedia.org/wiki/Domain_Name_System
Domain Name System Security Extensions (DNSSEC)	A method of cryptographically authenticating DNS entries.	https://en.wikipedia.org/wiki/Domain_Name_System_Security_Extensions
Embedded Content	Content that is not part of the root page, i.e. an iframe.	https://en.wikipedia.org/wiki/HTML_element
Extended Validation Certificate	A type of digital certificate that requires the Certificate Authority to verify the legal identity of the entity requesting a certificate.	https://en.wikipedia.org/wiki/Extended_Validation_Certificate
Extreme Programming (XP)	A software development technique that focuses on software quality and flexibility.	https://en.wikipedia.org/wiki/Extreme_programming
Frame Options Header	A standard for indicating whether or not a browser should be allowed to render a page in an iframe, frame or object.	https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
Gantt Chart	A type of bar chart for illustrating a project schedule.	https://en.wikipedia.org/wiki/Gantt_chart
Git	A distributed version control system for tracking changes in computer files.	https://en.wikipedia.org/wiki/Git
Heartbleed	A vulnerability within OpenSSL that allowed reading the server memory.	https://en.wikipedia.org/wiki/Heartbleed
Heimdall	A Software System to Improve Analysis and Communication of Domain Security. Named after the marvel character based on Norse mythology, an all-seeing, all-hearing Asgardian that guards the Bifrost Bridge, a rainbow bridge used to enter Asgard.	N/A
Health Insurance Portability and Accountability Act (HIPAA)	A US act which required national standards for electronic health care.	https://en.wikipedia.org/wiki/Health_Insurance_Portability_and_Accountability_Act
Hypertext Transfer Protocol (HTTP)	A protocol for transmitting web content and headers.	https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol

HTTP Public Key Pinning (HPKP)	A standard for indicating what public keys must appear within the certificate path for HTTPS.	https://en.wikipedia.org/wiki/HTTP_Public_Key_Pinning
HTTP Secure Transport Security (HSTS)	A standard for indicating that further connections to a domain should only happen over HTTPS.	https://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security
HTTPS	The secure version of the HTTP protocol which adds TLS encryption.	https://en.wikipedia.org/wiki/HTTPS
Human-Computer Interaction (HCI)	A field of research around the design and use computer technology, focused on the interfaces between users and computers.	https://en.wikipedia.org/wiki/Human%E2%80%93computer_interaction
JavaScript Object Notation (JSON)	A file format for storing data which can be serialized.	https://en.wikipedia.org/wiki/JSON
Kanban	A method of tracking project progress via a board of items.	https://en.wikipedia.org/wiki/Kanban
Long term support (LTS)	A term used to describe a type of software with a longer support cycle than normal.	https://en.wikipedia.org/wiki/Long-term_support
Mime type	A standard for indicating the nature and format of a document., i.e. image/png	https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types
Man In The Middle (MITM)	A type of attack where an attacker relays and possibly alters communication between two parties.	https://en.wikipedia.org/wiki/Man-in-the-middle_attack
Name-server	An application that implements a network service for providing responses to queries against a directory service.	https://en.wikipedia.org/wiki/Name_server
Nginx	A popular HTTP server, used for serving web pages and content.	https://en.wikipedia.org/wiki/Nginx
National Institute of Standards and Technology (NIST)	A measurement standards laboratory, and a non-regulatory agency of the United States Department of Commerce.	https://en.wikipedia.org/wiki/National_Institute_of_Standards_and_Technology
OpenSSL CCS flaw	A vulnerability within OpenSSL that potentially allowed an attacker to decrypt and modify traffic.	https://access.redhat.com/security/cve/cve-2014-0224
OpenSSL Padding-oracle flaw	A vulnerability within OpenSSL that potentially allowed an attacker to retrieve plain-text from encrypted packets.	https://access.redhat.com/security/cve/cve-2016-2107
PCI-DSS	An information security standard that handles branded credit cards from the major card schemes.	https://en.wikipedia.org/wiki/Payment_Card_Industry_Data_Security_Standard

POODLE	A vulnerability within OpenSSL that allowed an attacker to downgrade the encrypted connection to legacy SSL.	https://en.wikipedia.org/wiki/POODLE
Port	The port an application binds to for communication.	https://en.wikipedia.org/wiki/Port_(computer_networking)
Ransomware	A type of malicious software that holds the victim ransom.	https://en.wikipedia.org/wiki/Ransomware
Scrum	A team-based agile software development framework, with an emphasis on flexibility.	https://en.wikipedia.org/wiki/Scrum_(software_development)
Secure Sockets Layer (SSL) / Transport Layer Security (TLS)	A protocol that provides secure communication over a network.	https://en.wikipedia.org/wiki/Transport_Layer_Security
Subversion	A version control system for tracking changes in computer files.	https://en.wikipedia.org/wiki/Apache_Subversion
Syncthing	An application for syncing files between multiple machines in a peer-to-peer distributed manner.	https://syncthing.net/
Thread	The smallest sequence of programmed instructions that can be independently managed by a scheduler.	https://en.wikipedia.org/wiki/Thread_(computing)
UNIX	A family of operating systems that derive from AT&T UNIX.	https://en.wikipedia.org/wiki/Unix
Web socket	A communications protocol for providing full-duplex communication over a single TCP connection.	https://en.wikipedia.org/wiki/WebSocket
WebExtensions	A standard for creating extensions to web browsers.	https://wiki.mozilla.org/WebExtensions
White box	A method of testing where tests are derived with knowledge of source code.	https://en.wikipedia.org/wiki/White-box_testing
Whitelist	A list of entities to be accepted.	https://en.wikipedia.org/wiki/Whitelisting
XSS Protection Header	A header for protecting against cross site scripting attacks.	https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-XSS-Protection