



LETTERKENNY INSTITUTE OF TECHNOLOGY

A thesis submitted in partial fulfilment of
The requirements for the Master of Science in Computing in
Information Systems and Software Security
Letterkenny Institute of Technology

*An Implementation and Evaluation of PCI
DSS 3.0 for E-Commerce in a Testing
Environment*

Author:

Kelly McWeeney

Supervisor:

Mr. John McGarvey

Submitted to Quality and Qualifications Ireland (QQI) Dearbhú Cáilíochta agus Cáilíochtaí
Eireann) March 2015

Declaration

I hereby certify that the material, which I now submit for assessment on the programmes of study leading to the award of Master of Science in Computing in Enterprise Application Development, is entirely my own work and has not been taken from the work of others except to the extent that such work has been cited and acknowledged within the text of my own work. No portion of the work contained in this thesis has been submitted in support of an application for another degree or qualification to this or any other institution.

Signature of candidate: _____

Date: _____

Acknowledgments

This thesis would not have been possible without the support of many people at the Letterkenny Institute of Technology. The author would like to acknowledge and express her sincere gratitude to her supervisor John McGarvey for his guidance and support. The author would also like to extend her gratitude to the Head of Computing Department Mr. Thomas Dowling, and all the staff in Letterkenny Institute of Technology. Finally the author would also like to thank Sean McWeeney, Darina McWeeney, Shane McWeeney, Enda Cassidy, Gary Cassidy and Anthony Caldwell for their support.

Abstract

Threats to personal payment card information are continually on the rise. To implement a formal process for protecting this information PCI DSS was created. However credit card fraud is still rising. This leads to the question of whether or not PCI DSS is effective in protecting this data. This dissertation implements the 12 PCI DSS 3.0 requirements and tests the effectiveness in regard to the complexity of implementation.

While there are clearly issues within the standard: existing ambiguities, speed of adapting to industry findings: the main issue with the standard is not with the standard itself but with a failure to comply and still accepting electronic payments. The self-assessment options as found in testing would have left many network vulnerabilities that would remain undiscovered without full standard implementation.

Table of Figures

Figure 1: PCI Security Standards Council (2010) requirements evolution	15
Figure 2: Data Breach Statistics (Sullivan, 2013)	20
Figure 3: Full Network Design for PCI DSS 3.0 Compliance Testing.....	26
Figure 4: Network Switches	31
Figure 5: Ubuntu Hard Disk Partitions	31
Figure 6: Removal of Default PhpMyAdmin Users	32
Figure 7: Audit Ubuntu Server Checklist (Center of Internet Security, 2015)	33
Figure 8: Apache Server Confirmation.....	34
Figure 9: CIS Audit Apache Server Checklist (Centre of Internet Security, 2012)	34
Figure 10: ClearOS Interfaces.....	35
Figure 11: Smoothwall Configuration and Logging Options.....	36
Figure 12: Implementation of RSA SSL.....	37
Figure 13: TLS v1.2 configuration.....	37
Figure 14: Successful MySQL SSL configuration	38
Figure 15: php.ini configuration	39
Figure 16: Comodo Installation.....	40
Figure 17: Self-Assessment Versions	40
Figure 18: Wireshark TLS handshake	41
Figure 19: HTTPS Communications.....	42
Figure 20: Nikto Configuration.....	42
Figure 21: Scan for Wireless Routers	43
Figure 22: De-Authorisation Packet Injection.....	44
Figure 23: Wireshark capture of AES keys	44
Figure 24: Failed Aircrack Attack	45
Figure 25: PCI DSS Requirement 11.3	47

Table of Acronyms

Acronym	Meaning	Acronym	Meaning
ACL	Access Control Lists	NAT	Network Address Translation
AES	Advanced Encryption Standard	NIC	Network Interface Controller
CDE	Card Data Environment	NIST	National Institute of Standards and Technology
CIS	Centre for Internet Security	OS	Operating System
DMZ	De-Militarised Zone	PCI	Payment Card Industry
DPA	Data Protection Act	PCI DSS	Payment Card Industry Data Security Standard
EAPOL	Extensible Authentication Protocol over LAN	SHA	Secure Hash Algorithm
GUI	Graphical User Interface	SDLC	Systems Development Life Cycle
IDS	Intrusion Detection Systems	SSL	Secure Sockets Layer
IP	Internet Protocol	TLS	Transport Layer Security
ISO	International Organization for Standardization	VLAN	Virtual Local Area Network
IEC	International Electro-Technical Commission	VM	Virtual Machine
MD5	Message Digest Algorithm	WPA	Wi-Fi Protected Access

Table of Contents

Acknowledgments.....	3
Abstract.....	4
Table of Figures.....	5
Table of Acronyms	6
1. Introduction.....	12
1.1 Purpose	12
1.2 Background	13
1.3 Outline of Report	13
1.4 Research Question	13
1.5 Research Objectives.....	13
2. Literature Review	14
2.1 Introduction	14
2.2 The History and Evolution of PCI DSS	14
2.2.1 The Founding and Early Conceptions of the PCI Model.....	14
2.2.2 PCI DSS 2.0	15
2.2.3 PCI DSS 3.0	16
2.3 The Importance of the Payment Card Industry Data Security Standard	17
2.3.1 Protecting Data	17
2.3.2 Implications of Non-Compliance.....	18
2.3.3 Lack of Understanding	18
2.4 PCI DSS and Legislation	19
2.5 Data Breaches	19
2.6 Implementing the Standard	20
2.7 PCI DSS and the Security Community	21
2.8 PCI DSS 3.0 and Security Education	22
2.9 The Future of PCI DSS.....	24

2.10	Conclusion.....	24
3.	Requirements Specification.....	25
3.1	Introduction	25
3.2	Network Design.....	25
3.3	Network Tools.....	26
3.3.1	VSphere.....	26
3.3.2	Client, Server and Database OS	27
3.3.3	Firewalls	27
3.3.4	Router	27
3.4	Website Implementation	27
3.5	Compliance Testing.....	28
3.6	Testing of the Standard.....	28
3.6.1	Kali.....	28
3.7	Summary	29
4.	Methodology	30
4.1	Introduction	30
4.2	Virtual Network.....	30
4.3	CDA and DMZ Initial Configuration	31
4.4	Database Server	32
4.5	Web Server Configuration.....	33
4.6	ClearOS Setup	35
4.7	Smoothwall Setup	35
4.8	Security Protocols	36
4.9	Wireless Security.....	38
4.10	Database Encryption	38
4.11	Secure Programming.....	39
4.12	Anti-Virus	39
4.13	Self-Assessment	40

5	Testing	41
5.1	Secure Network Communications	41
5.2	Vulnerability Scanning	42
5.2.1	Findings	43
5.3	Wireless Testing	43
5.3.1	Aircrack	43
5.3.2	Findings	45
5.4	Penetration Testing.....	45
5.4.1	Ubuntu Servers Findings	46
5.4.2	ClearOS Findings	46
5.4.3	Smoothwall Findings	46
5.5	Results Conclusion	47
5.5.1	SAQ A v3.0.....	48
5.5.2	SAQ A-EP v3.0.....	48
5.5.3	SAQ for Payment Terminals	48
6	Conclusion	49
6.1	Slow to Adapt.....	50
6.2	PCI DSS 3.0 Ambiguities	50
6.3	Self-Assessment	51
6.4	Failure to comply	51
6.5	PCI DSS 3.1	52
6.5	Restrictions	52
6.6	Further Work.....	52
	Bibliography	53
	Appendix A - VCG Source Code Scan Results.....	59
	Open Cart Scan Results	59
	Commercial Site1 Scan Results	64

Commercial Site2 Scan Results	65
Appendix B - PhpmyAdmin configuration files	66
Appendix C - Ubuntu CIS Audit Configurations.....	68
Fstab - Partition Rules	68
Sysctl.conf – Set Routing to Disabled per CIS Guidelines	69
Auditd.conf – Configure System for Logging	70
Audit .rules – Set Rules for Audit Log Files	71
Password Implementation.....	73
Login.defs - Logging Incorrect Attempts and Setting Password LifeCycle	73
Pam.d - Set Passwords to Required	75
Shadow Configuration Sets Password Attempts Limits	76
Appendix D - Apache CIS Audit Configuration	78
000-default.conf - Local Host Sites Configuration File.....	78
Default-ssl.conf - HTTPS Local Host Configuration File.....	79
Appendix E – MySQL Security Protocols	80
My.Conf enforcing TLS certifactes to communicate with WebServer	80
Appendix E - ClearOS Configurations	82
IP Tables	82
Intrusion Detection System	83
Appendix F – SmoothWall Iptables.....	87
Appendix G - Metasploit Scans	88
Ubuntu Server Metasploit Results.....	88
ClearOS Metasploit Results.....	91
SmoothWall Metasploit Results.....	91
Appendix H – Nikto Vulnerability Scan	94
Appendix I – Table of Requirements.....	95
Build and Maintain a Secure Network and Systems	95

Protect Cardholder Data 95

Maintain a Vulnerability Management Program 95

Implement Strong Access Control Measures..... 95

Regularly Monitor and Test Networks 96

Maintain an Information Security Policy 96

1. Introduction

Key trend indicators from industrial sources have tracked the progressive escalation in security incidents worldwide (Trustwave, 2013; Verizon, 2013; PWC, 2014; Ponemon, 2014). Both the domestic end-user and contemporary computing professional are faced with persistent threats to their personal data in today's Internet enabled world. The existing principles about which security is oriented such as confidentiality, integrity and availability face considerable challenges. From its inception, the Payment Card Industry Data Security Standards (PCI DSS) was established as a means of securing payment card information and there has been considerable debate as regards the effectiveness. With the continual rise of security breaches drawing attention to PCI DSS, the effectiveness of the standard is under much debate. With many large companies such as Target suffering security breaches the level of protection offered by the standard has come under question (Plato, 2014)

Malicious attacks of increasing sophistication and complexity are hampering today's security professionals and while no single methodology, technology or attitude shift will ensure security, it is recommended that a defence in-depth strategy is adopted. Central to this strategy is the PCI DSS framework which, if properly implemented, can offer an organization, institution or end-user a much needed layer of support to help mitigate for the scale of attacks currently experienced. This thesis offers considerable insight into the application of PCI DSS by testing its strength on an experimental but compliant network built to accept secure payment card information. The literature review focuses on the history of PCI DSS and why it is a necessary and progressive component of today's contemporary security infrastructures.

1.1 Purpose

The strengths and weaknesses of PCI DSS 3.0 will be tested with a view to proposing some potential mitigation techniques. While PCI DSS 3.0 aims to provide a further level of clarity than its predecessors, ambiguity in its communication can have a detrimental impact particularly as regards its implementation which is critical in successfully obtaining compliance. Therefore, the primary focus of this thesis is to assess the effectiveness of the standard with special reference to the complexity of the implementation process.

1.2 Background

When a business accepts credit card payments, whether online or in a physical environment the retailer is required to be PCI DSS compliant. PCI DSS was originally developed by Visa, MasterCard, American Express and JBC to help protect sensitive payment card information. Since the standards original release in 2004 data breaches and credit card fraud have been rapidly increasing. The effectiveness for PCI DSS 3.0 is essential to protect sensitive information.

With the standard being of such importance an evaluation of its strengths, weaknesses and implementation process is necessary. With the threat of data breaches continuously on the rise (Sullivan, 2013) it is not enough to blindly trust a security standard; therefore the standard requires in-depth review and assessment.

1.3 Outline of Report

Chapter 2 Literature review

Chapter 3 Requirements design

Chapter 4 Methodology

Chapter 5 Testing

Chapter 6 Conclusion, limitations and suggestions for further research

1.4 Research Question

Is PCI DSS 3.0 an effective method of protecting cardholder data? Is the rise of data breaches due to weaknesses in the standard or to a lack of implementation?

1.5 Research Objectives

- Build and secure a network with web server to accept payment card information
- Test the security measures in place with penetration tests
- Assess and secure any weaknesses found
- Evaluation of the strength of PCI DSS 3.0

2. Literature Review

2.1 Introduction

Considerable debate surrounding the effectiveness of the PCI DSS has highlighted the importance of clarity regarding documentation and implementation procedures (Kedgley, 2014). Since the release of PCI DSS 3.0 in 2013 there is a dearth of information in this respect. What follows is a review of the history and evolution of PCI DSS, its importance and data breaches.

2.2 The History and Evolution of PCI DSS

2.2.1 The Founding and Early Conceptions of the PCI Model

Prior to the release of PCI DSS, payment card industries comprising of Mastercard, Visa, American Express, JCB and Discover relied upon individual policies regarding the safe processing and storage of payment card data. This created significant challenges as regards an integrated approach to the issues surrounding PCI. Prior to the initial release of the standard the Payment Card Industry Security Standards Council was founded, the PCI Security Standards Council combined these individual policies resulting in the release of PCI DSS version 1.0 in December 2004 (Roebuck, 2012). As a result, procedures were put in place to mitigate for sensitive payment card data being breached by ensuring merchants who accept this data are both aware of and enforce the necessary security measures (Chuvakin & Williams, 2011).

Early documentation from the American Bar Association (2008) and some more recent research from Chuvakin and Williams (2011) and Bakay et al, (2014) suggested that a climate of growing concern surrounding the protection of sensitive credit card data led to the PCI DSS being devised by MasterCard and Visa to alleviate concerns surrounding the safe handling of cardholder data in a practical step by step process (American Bar Association, 2008, Bakay et al, 2014). An important feature driving the initial stages of the standard's creation was the need for compliance and the importance of clarity within its process and documentation. This early phase of the standards creation contained numerous flaws subsequently rectified by version 1.2.1, version 1.2.1 had many requirement additions to address these issues, providing more guidance on wireless

technologies, virtualisation and encryption methods, resulting in a higher level of security than its predecessor (Bhargav, 2014). However, PCI DSS version 2 became the industry standard in active use until December 2014.

2.2.2 PCI DSS 2.0

In 2010, PCI DSS 2.0 was released and became the industry standard in 2011. While PCI DSS 2.0 did not introduce major changes from PCI DSS 1.2.1, it contained more subtle and practical enhancements which more appropriately dealt with technological advances and how security should be approached. Research by Chuvakin and Williams (2011) documented these differences, highlighting important changes particularly as regards wireless networks such as WEP authentication which was no longer deemed acceptable, the introduction of virtualization and a significant change in outlook as regards the implementation of risk management. Figure 1 below shows an example of how the PCI Security Standards Council charted the evolution of the standard from PCI DSS 1.2.1 to PCI DSS 2.0.


			
Section or Requirement		Change	Type ⁱ
Old	New		
General	General	Scope of Assessment for Compliance with PCI DSS Requirements <ul style="list-style-type: none"> Added “<i>virtualization components</i>” to the definition of “<i>system components</i>.” Clarified that the cardholder data environment is comprised of “<i>people, processes and technology that store, process, or transmit cardholder data or sensitive authentication data</i>.” 	Additional Guidance
General	General	Scope of Assessment for Compliance with PCI DSS Requirements Added detailed paragraph to clarify that the first step of a PCI DSS review is to accurately determine the scope of the assessment, by identifying all locations and flows of cardholder data and ensuring that all such locations are included in the assessment.	Additional Guidance
General	General	Network Segmentation <ul style="list-style-type: none"> Added clarifications including that segmentation may be achieved through physical or logical means. Minor replacements to some wording to clarify meaning. 	Clarification
General	General	Wireless Clarified focus on presence of a WLAN rather than a LAN.	Clarification

Figure 1: PCI Security Standards Council (2010) requirements evolution

As useful as this standard was in its initial stages, weaknesses in PCI DSS 2.0 began to emerge that may be still be prevalent in the updated standard. For example, Blackwell (2008) noted the importance of auditing in organizations and warns that it may be bypassed via physical access which employees have to many critical systems. Responding to this, PCI Security Standards Council included important guidelines for requirements and resources that may be used when determining potential weaknesses (PCI Security Standards Council, 2013). For those companies that are compliant with PCI DSS 2, these guidelines, while welcome, lack details as regards implementation and other sources are required. For example, the standard highlights those areas prone to security breaches along with aiding merchants to adapt their systems so that implementation is more readily achieved to gain compliance. While the PCI Security Standards Council (2013) effectively describes the aims of PCI DSS they do not provide sufficient research or merchant feedback on the success levels of these aims.

2.2.3 PCI DSS 3.0

PCI DSS 3.0 was released in November 2013 but will not be in full effect until 1st of January 2015. Due to the fact that PCI DSS 3.0 is not yet in full effect, information on the effectiveness of very limited. However, Chuvakin and Williams (2014) while providing insights into all previous versions also stay up to date with the standard releasing an updated edition of '*PCI Compliance*' to take into account the implementation changes in the new standard. Keeping the same structure as the other editions previously mentioned, this provides a useful guide for measuring the scope of PCI DSS 3.0 and provides useful implementation techniques.

While Shihab and Misdianti (2014) do not focus on implementing PCI DSS 3.0 from the beginning like Chuvakin and Williams (2014), they do provide recommendations on updating network from a previous PCI version to PCI DSS 3.0. Shihab and Misdianti (2014) further their research by providing a break-down of the requirement changes, providing a list of requirements that are necessary when updating to the new standard. This is very useful as it separates out clarifications and evolving requirements, providing a concise list of changes that will need to be made to network to update to PCI DSS 3.0.

Identified Problems with PCI DSS 3.0

Kedgley (2014) states there are no stark contrasts between PCI DSS 3.0 and its predecessor. While PCI compliance is a requirement for merchants taking credit card payments data breaches are still increasing, Kedgley (2014) asks are the subtle changes in PCI DSS 3.0 enough to reduce online credit card fraud? This is reiterated by explaining that the clarifications in the new standard seem to be of greater importance than strengthening mitigation techniques and that more than clarifications must be implemented to successfully protect card holder data. While Kedgley (2014) does provide a document that offers a well detailed standpoint on the new standard, as there currently is still a dearth of information on PCI DSS 3.0 there is little documentation to concrete or dispute this view.

While the argument made by Kedgley is logical there is no data to back up the hypothesis. PCI DSS 3.0 is a clarification on PCI DSS 2.0 as can be seen from the list provided by the PCI Security Standards Council (2013), however this does not mean that there are no changes between the standards, and it may be premature to write off the effectiveness of the standard at such an early stage.

While implementing PCI DSS 3.0 involves technical complexity it is not the only issue facing businesses implementing the standard. Lovric and Sedinic (2013) also evaluate the financial implications of gaining PCI DSS compliance. A network is built for a fictitious company 'The First Company Inc.' and a study on implementing PCI DSS as well as ISO/IEC 2700 is undertaken for the methodology. Lovric and Sedinic (2013) also outline the specifications of the First Company Inc. stating that it processes more than 1 million credit card transactions per annum. The findings show that to achieve compliance would have a financial impact of up to \$250,000. This provides a severe financial impact, and for smaller industries may be unfeasible. This document is particularly useful as it takes a different angle of research by not studying effectiveness or efficiency but rather the financial implications which is of great importance to merchants.

2.3 The Importance of the Payment Card Industry Data Security Standard

2.3.1 Protecting Data

Recent research from Atay et al (2014) and Grobler et al (2014) highlighted the prevalence and rapid adoption of credit card payments online with the concordant importance of how

the payment card industry has dealt with the protection of cardholder data. Of significant concern were the growing incompatibilities between the PCI standard and the diversity of vendors' systems and networks. While there is continuing research being conducted to mitigate for data breaches in both the public and private sector, PCI DSS remains the only officially implemented standard to specifically protect credit card holder data (Grobler et al, 2014). Highlighting the importance of effective security to protect cardholder data, Kedgley (2014) reported that while the total value of card payment transactions exceeds \$21trillion annually, this is unfortunately overshadowed by losses in excess of \$11billion due to fraudulent transactions. This emphasises the importance of the PCI DSS standards since, to date, this is the only payment security standard enforced today. Therefore, its effectiveness is paramount if the protection of cardholders' data is to be a priority. Non-compliance to these standards faces considerable financial penalties.

2.3.2 Implications of Non-Compliance

PCI DSS is not only important in terms of protecting data; there are consequences for non-compliance. Kidd (2008) discusses the effects of non-compliance where vendors who are found to be non-compliant after a breach are subject to fines and/or the implementation of restrictions as regards taking further credit card payments online. An important matter is highlighted in this respect since not only can non-compliance result in fines of up to \$500,000 for each individual breach (Douthit and Huang, 2008), the loss of consumer confidence can have a significant negative impact upon the reputation of a firm to be found non-compliant. Beyond the standards, requirements and recommendations there are legal and regulatory frameworks already in place to deal with data protection.

2.3.3 Lack of Understanding

Contrary to the importance of the standard, there is a lack of understanding for the need of PCI compliance among merchants and their employees, particularly in regards to previous versions. As industry research shows (Trustwave, 2013; Verizon, 2013; PWC, 2014; Ponemon, 2014) non-compliance is the main cause of data breaches, and lack of awareness and understanding seemed to be the main factor in regards to non-compliance. However PCI DSS 3.0 aims to alleviate this confusion by enforcing training methods for staff in the updated requirements (Mehta, 2014). This awareness is essential for improving security when handling sensitive payment card data.

2.4 PCI DSS and Legislation

Accepting the PCI DSS requirements, merchants must also be aware of other legislation which may have an impact on their implementation. Andress (2014) goes to considerable efforts to note that the necessary legislation may be divided by jurisdiction thus providing the merchant an easier means of identifying those which are most important for their particular business. Of some value then is the Governance, Risk Management and Compliance (GRC) framework which takes an international perspective surrounding the enforcement of measures to ensure all required standards and legislations are met along with legal implications (Gordon, 2015). For example in Ireland, the framework provides a mechanism to take into consideration the Data Protection Act (DPA) (2011) and furthermore at a European level via the 'Electronic Communications Networks and Services Privacy and Electronic Communications Regulations' which stipulates that consumers are made aware and give permission to allow their data to be stored (Morgan & Boardman, 2012).

Within the context of increasing security trends indicating the escalation of data breaches specifically targeting consumer data (Trustwave, 2013; Verizon, 2013; PWC, 2014; Ponemon, 2014) it is important to recognise the importance of the 2011 DPA when building and an e-commerce website within compliance regulations.

2.5 Data Breaches

The data gathered by Sullivan (2013) from 2005 to 2010 regarding the prevalence of attacks leveraged against organisations storing sensitive payment card data reveals the heavy financial impact of data breaches (see Figure 2). While the statistics are dated as only take into account up until 2010, Sullivan (2013) provides great detail on the financial impact that incurs from a data breach, but furthermore shows details of many high-level breaches like that of TJX that spurred changes in the standard. Of some concern is the worrying trend to sell the information extracted to perpetrate fraud. The cost of a data breach outweighs the cost of compliance and with credit card breaches. On this basis, it is important to note that PCI DSS 3.0 must provide comprehensive assurance to merchants and consumers alike.

Chart 2

RECORDS COMPROMISED FROM PUBLICLY DISCLOSED DATA BREACHES IN THE UNITED STATES

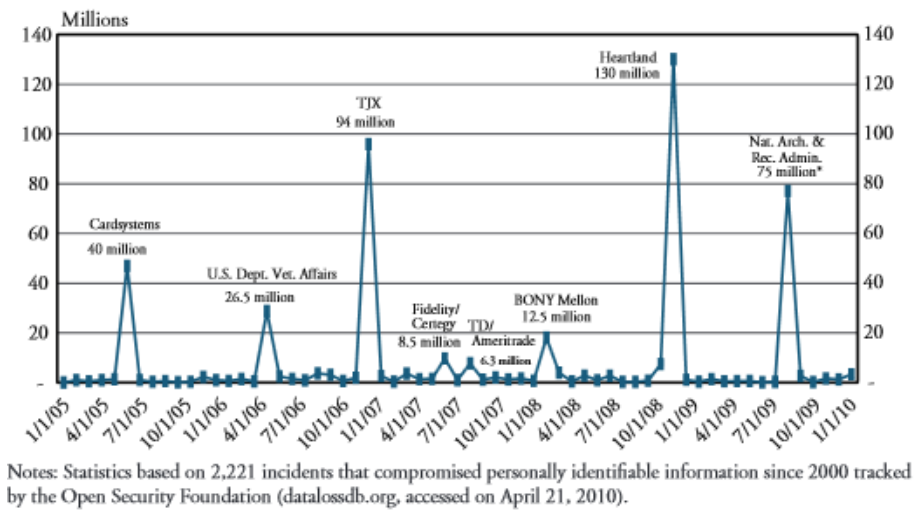


Figure 2: Data Breach Statistics (Sullivan, 2013)

While Sakharova (2012) does not provide the in-depth statistics provided by Sullivan (2013), there are figures provided on the use of credit cards. While these are based in the U. S. it does show the rise of debit and credit cards use. Sakharova (2012) utilises figures released from the U.S. Census Bureau which state that in 2008 there were approximately 176 million credit card users and 181 million debit card holders, with these numbers constantly rising. This illustrates the need for effective security standards given the volume of payment card data, highlighting the impact that payment card fraud has on industry costing \$8.6 billion per annum and rising.

2.6 Implementing the Standard

Aside from the standard requirements released by the PCI Security Standards Council there is a lack of literature on the implementation process for PCI DSS 3.0. Bhargav (2014) provides a guide on implementing PCI DSS 3.0 with information on each requirement and how to effectively implement these to achieve compliance. Due to the lack of information on PCI DSS 3.0 this guide is a well-written and concise resource that provides a walk through guide to the implementation process.

While Johnson (2014) does not aim to provide a definitive guide on the implementation of PCI DSS 3.0 he does provide insight into the more difficult issues of the implementation process. Johnson (2014) details the difficulty of network segmentation as required by the

new standard. Another difficulty highlighted by Johnson (2014) is the added complexity of the encryption standards necessary, as in PCI DSS 3.0 it is required to also encrypt sensitive information at rest. This is a valuable resource that will be utilised when evaluating the complexity of the implementation process. Johnson also provides a clear guide on IT security best practices and is a valuable resource as many of these best practices are used in the implementation process for PCI DSS 3.0.

Chuvakin and Williams (2011) discuss the implementation process for PCI DSS 2.0 which will no longer be in effect from 1st of January 2014. Despite this, a checklist is provided for business which is particularly useful for the planning stages of the compliance process. Chuvakin and Williams (2011) detail the importance of training staff to reduce the likelihood of a data breach to ensure that compliance can be maintained at all managerial and administrative levels of an organisation. While this document was published for PCI DSS 2.0 the point that is made is still valid as education will always be a form of data breach mitigation.

PCI DSS 3.0 requirements 6.5.1 to 6.5.10 do not directly reference the Owasp top ten, this list provides an extensive resource on how to implement these PCI DSS 3.0 standards. Owasp (2014) provides mitigation techniques for these top ten weaknesses but many other techniques that should be considered good practice. Owasp is invaluable as it provides clear descriptions of the weaknesses, how these weaknesses can be targeted and mitigation techniques for data breaches and attacks. The Owasp top ten list provides in depth details on injection attacks, session authentication and management, cross-site scripting, direct object referencing, security misconfiguration, data exposure, function level access controls, CSRF attacks, component vulnerabilities and un-validated redirects and forwards (Owasp, 2014). Owasp is a vital resource and provides mitigation techniques for the listed attack types across a range of coding disciplines recommended for not only PCI DSS 3.0 but implementing best practiced for secure coding techniques.

2.7 PCI DSS and the Security Community

As shown, the evolution and progression of PCI DSS has been well documented, there is an area of research that has been yet to be addressed, the debate on the effectiveness and

reliability of the standard. While it is clear that the need for such a standard is paramount, there are those who believe that the state of the current standard is not enough.

Jeffrey Hall (2014) highlights issues within the standard, many obtaining compliance for compliance sake, rather than focusing on obtaining an adequate layer of security. While not negating the importance of the standard, or what it aims to achieve, the main obstacle is the human element. The implementation complexity can lead to unreliable testing procedures and monitoring. However, Hall (2014) does provide a balanced argument, security is not absolute, it is mitigation. For PCI DSS to provide a higher mitigation level an active approach to security needs to be undertaken, going beyond the minimal scope of the standard.

Shier (2014) iterates this point further, providing views both on the successes and failures of the standard the defining point made is that PCI DSS is not a failsafe security standard. When storing sensitive data, merely achieving compliance is not enough. Shier (2014) states that PCI DSS should be used as a security baseline and not be seen as the holy grail of security. Another integral point is the self-assessment element. Should a company that is being trusted by the consumer to store their private data be the ones that determine the effectiveness of the security measures they have in place?

While Hulme (2009) presents a well-grounded article on the PCI DSS it is based on the predecessor to PCI DSS 3.0, however the points made are still relevant. The argument presented like both Hall (2014) and Shier (2014) is not dismissive of the standard. It is clear that a standard to protect sensitive information needs to be implemented, however the issue that Hulme (2009) presents is that compliance is not security. In contrast to building a service and then trying to enforce compliance it is essential that companies instead enforce security during the build. Comprehensive coding, testing and logging should be developed. PCI DSS should not be the overall aim; instead as part of a secure builds PCI DSS should naturally be implemented.

2.8 PCI DSS 3.0 and Security Education

In the research conducted by Coburn (2010) the importance of education to not only achieve but also maintain compliance is well documented. The importance of education as a culture is also discussed, while this importance is mainly discussed as part of PCI DSS

lifecycle it is also recommended to create a culture of education for all security and technological needs. Coburn (2010) effectively shows how vital training is, as once compliance has been achieved it also needs to be maintained, educating employees in acceptable procedures and best practice techniques is integral for compliance.

PCI DSS 3.0 introduces a more education focused approach to security. While not focused on PCI DSS education, NIST successfully highlights the importance of education frameworks in the entirety of IT security. The outlook on education is not only provided from an IT perspective but also from a business perspective, not only for protecting consumers' details, but company resources, information and reputability. NIST goes even further in depth, providing education frameworks that focus on understanding based education. It is not simply enough to inform staff about security practices. For security to be implemented effectively security professionals must not only understand the implementation but also the purpose (Guttman and Roback, 1995). Even though published in 1995 and in such a fast paced and evolving industry, the importance of a progressive and continuous education model is recommended by many contemporary security professionals.

Kamala et al (2012) further the point made by Hall (2014) previously, that the human factor and poor understanding lead to vulnerabilities. However, Kamala et al (2014) also provide business justification for education. Both education and security are expensive, but with the growing threat of sensitive data being compromised the loss of business due to reputation and re-configuration post breach is even more detrimental. While constant training can not only be a monetary issue, it can also be tedious for staff. A study on different business sectors showed that the most effective methodology for maintaining successful training programs is an environment of reward and results based encouragement.

While this source does not directly refer to PCI DSS and the standards educational requirements, it does provide an extensive analysis of education in the IT security industry as a whole. Alshomrani (2012) discusses the vital need for security education in the IT industry. This is partly due to the fact that many coding experts may not be aware of potential security risks that may be introduced into source code, but also as the security industry needs evolve at such a high rate it is imperative that IT professionals be aware of emerging threats. Alshomrani (2012) also provides resources that can be utilised to ensure

security education, many of which directly influence PCI DSS, such as network and coding security certifications.

2.9 The Future of PCI DSS

PCI DSS is constantly evolving, with new revisions and version being constantly updated through its lifecycle. PCI DSS follows a three year lifecycle, meaning that the longest time one version of the standard will be in full effect for is a three year period (Chuvakin & Williams, 2014; PCI Security Standards Council, 2010). As technologies rapidly evolve PCI DSS will need to address these. For example, with the growth of mobile technologies PCI DSS 3.0 does make greater reference to mobile to its predecessor, however these are still minor references in requirements 1 and 11. As these technologies grow PCI DSS may have to further and adapt the scope of the standard to allow for this (Chuvakin & Williams, 2014).

At the moment, PCI DSS is not legislation, even though there are penalties for non-compliance, one opinion is that PCI DSS will evolve to become law, so when a merchant decides to take payment card information they will be legally obliged to enforce the standard (Carpenter, 2010). Given the link between PCI DSS and the both EU and US privacy legislations this is a logical step in the evolution of the standard.

2.10 Conclusion

PCI DSS was developed to aid in alleviating growing concerns for private data utilised for credit card payments and as noted by Grobler et al (2014) it is important that these standards be highly effective given that they are the only existing formal standard in place for protecting credit card holder data. On this basis, PCI DSS 3.0 needs to be a significant deterrent to malicious intent and effectively protect sensitive data. The importance of the standard and documentation of its evolution is well written and can be easily accessed, however, Sullivan (2013) does show the rise of online credit card fraud and the economic impact of data breaches. This shows that even though PCI DSS is a required standard, either it is not being fully enforced or the standard is not effective enough to mitigate for data breaches. The key value of the work presented here is the in-depth study of PCI DSS which represents a significant contribution to the field of PCI DSS 3.0 compliance.

3. Requirements Specification

3.1 Introduction

The primary focus of this thesis is to assess the effectiveness of the PCI DSS 3.0 standard, given the lack of research regarding an effective implementation method to achieve compliance; the methodology and implementation process will provide an important contribution to the literature surrounding PCI compliance. Coburn (2010) and Alshomrani (2012) emphasised the need for effective security training and while Kedgley (2014) noted that there were no significant differences PCI DSS 3.0 and its predecessor, this should not prevent a continuing education programme from developing. Considerable difficulties are faced by the security professional as regards PCI DSS implementation given the diversity of organizational activities, the products and services offered and their awareness of security processes. To date, no automated solutions exist which might make the job of PCI compliance more efficient; however PCI DSS may not lend itself to automation. This section will outline the specification requirements necessary to build a PCI DSS 3.0 compliant network for testing. The requirements for this network include a switch, web server, router and multiple firewalls for DMZ implementation. This will be followed by the integration of a fully functioning e-commerce website linked to the host web server as a testing platform to establish the efficacy of PCI DSS 3.0.

3.2 Network Design

To establish a baseline for compliance and to rigorously test the effectiveness of PCI DSS 3.0, it is necessary to test on a compliant network and website. It is important to note that for PCI DSS 3.0 network compliance, segregation is key. To accomplish this the network design is that of a switched network which will consist of three separate switches to achieve the level of segregation necessary. One switch will host the internal LAN, another for the CDE and another for the DMZ. The LAN switch will host three separate business units which are set up on their own separate VLANs simulating a small/medium business. The access granted on any network should be the minimum necessary to fulfil the functions of its users. In this regard, the LAN design here consists of admin, sales and production with administrative users having a higher level of access. There will also be an Apache web server implemented that will be contained in the DMZ and protected via firewall. The database

server that will contain the sensitive payment information will be hosted on the CDE switch also protected with configured firewalls. For network connectivity each switch via routers, the CDE and LAN will both connect to the outside world via the DMZ but will have no contact with each other. Figure 3 below details the full network design with the IP addressing; the DMZ and CDE are to be configured with minimal hosts per subnet to aid in preventing exploitation of open ports.

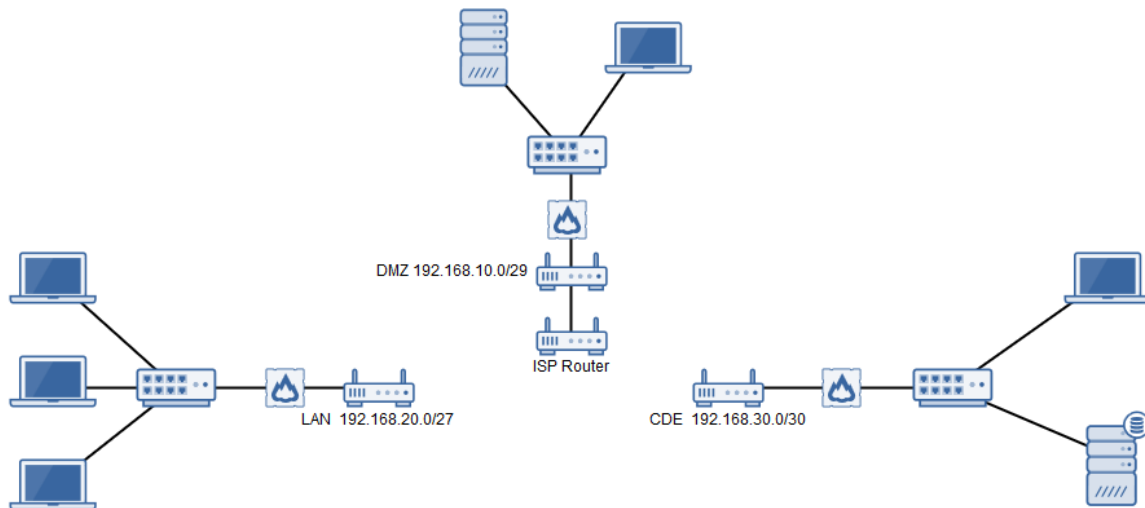


Figure 3: Full Network Design for PCI DSS 3.0 Compliance Testing

With a network design established testing the network will require the use of specialised tools to capture the network state for further analysis.

3.3 Network Tools

3.3.1 VSphere

Due to financial constraints the acquisition of all the physical hardware necessary for full network implementation is unfeasible. The network build will be completed utilising VSphere and VMWare Workstation 11. There are considerable advantages associated with VSphere such as the ability to set up virtualised networking including switch configuration and implementation, the prioritization of access to network resources based on business rules and load balancing features. The primary feature of significant value to this work is the network snapshot component which captures the network state and can be used for reverting or comparing previous development states.

3.3.2 Client, Server and Database OS

Both the server and database will utilize Ubuntu Server with Apache as the web server combined with a MySQL database server for storing of sensitive data. This configuration allows full Lammpp stack implementation or installing single aspects. This aspect is useful as the web and database server functionality can be segregated without taking up valuable resources. Due to resource limitations, the open source client desktop images in the business unit VLANs will be implemented as Linux Ubuntu desktops.

3.3.3 Firewalls

Standard guides surrounding PCI DSS compliance requires the implementation of firewalls (Chuvakin and Williams, 2011; Bhargav, 2014; Johnson 2014). Using these recommendations as a baseline, ClearOS will be utilised as an external facing firewall with Smoothwall configured with router and firewall functionality in the internal LAN and CDE. Best practices indicate that a combination of supporting systems is a more robust solution rather than a singular source, in this respect different firewall systems will be deployed in this network so that limitations in one firewall are overcome by advantages in other.

3.3.4 Router

ClearOS as well as offering Firewall functionality can also be used as a network router. ClearOS also supports VLAN routing so allows communication between VLANs but also allows ACLs to be applied to the communications. ClearOS also has the capability to link virtualised network to physical ISP routers, providing the network with internet access.

3.4 Website Implementation

Due to code complexity the network build and the time constraints of the project the website build will consist of obtaining an ecommerce website. To choose the code base that would be utilised for the project, two commercial code bases are tested along with one open source code base, Opencart (2015) version 2.0.2.0. The code base for each cart is tested with a static code scan; see Appendix A for full scan results. While the two commercial projects were advertised as PCI DSS compliant many weaknesses were found in the code base, all three contained MD5 hashing algorithms along with insecure php coding techniques. While OpenCart also had these weaknesses, it is an open source code base that is permissive of modification and was chosen for the ecommerce functionality. The code

base for OpenCart will be secured in the methodology section of the dissertation. When utilising open source in industry there are implications that must be considered. Some license types like the MIT and Free BSD License are very permissive, allowing redistribution and modification, however some other license types referred to as copyleft license types contain a clause that if modification and distribution occur all code must be donated back to the open source community (Yingkui , et al., 2010).

3.5 Compliance Testing

The systems development life cycle (SDLC) is widely used throughout the IT industry and while considerable debate surrounding its configuration will continue (McMurtrey, 2013) its ability to adapt and evolve in established and some emergent environments makes it an ideal choice to integrate full compliance testing. While the network will be built utilising a SDLC to ensure security is kept at the forefront of the project, upon completion of the network build and implementation of the website full compliance will be tested.

3.6 Testing of the Standard

3.6.1 Kali

Maintained and funded by Offensive Security Ltd, Kali is a digital forensics and penetration testing suite that offers over 700 penetration tools accumulating in a powerful testing suite (Binder and Broad, 2003). The issue met was narrowing down the available options provided by Kali to a concise list including:

1. Metasploit - an open source software that offers many features that will be utilized in the testing phase. As it can be used on networks, devices and web applications it will be valuable during penetration testing.
2. Nikto - this is a powerful vulnerability scanning tool that will be utilised to assess vulnerabilities on the webserver and hosted site. Nikto is continually updated so contains details of new means of network exploitation.
3. Nmap - a versatile tool that will assist in assessing many of the aspects of PCI DSS 3.0. Nmap will be utilized to assess the strength of firewall configurations and any IP filters that may be on the network. Furthermore Nmap can gain vital information about the physical network. This tool has the ability to port scan and gain data on operating systems and network protocols.

4. Aircrack specializes in accessing WLAN data and breaking WEP and WPA keys. This is a very useful tool when attempting to infiltrate a network. Aircrack uses ARP packets to gain information on both WEP and WPA keys.
5. Wireshark – Wireshark is a network packet analyser which captures network packets and displays packet data as detailed as possible; in particular, the ability to capture live network traffic combined with filtering systems that will be useful to focus on specific protocols such as SSL and aid in decryption. Vandeven (2013) successfully shows how Wireshark can be utilised to test encryption methods and traffic flows via a network. Secure data flow and encryption methods are necessities for compliance.

These tools have been chosen as they are recommended to assist in testing each aspect of PCI DSS (Koster, 2012; Chuvakin & Williams, 2011), while there are alternatives which are more user friendly, they do not provide the comprehensive level of testing offered by the tools selected (Koster, 2012; Chuvakin & Williams, 2011). They will assist in evaluating the strength of the standard from attacks both inside and outside the network. To assess the effectiveness of PCI DSS 3.0 these tools will be used to perform high level penetration tests on the network, server and traffic from the web application. As the goal of a malicious user is to gain access to sensitive data, the aim of the testing method is to do the same. If the data is breached PCI DSS 3.0 will be assessed by comparing the complexity of the breach with the information that has been obtained.

3.7 Summary

The network topology, tools and testing strategy presented here represent a comprehensive plan to gather data on the efficacy of a PCI compliant network. While standard guidelines have been used a series of progressive tools and techniques have been leveraged together to augment the current PCI compliance strategies available.

4. Methodology

4.1 Introduction

The project implementation explains the necessary steps taken to build a PCI DSS 3.0 compliant network for testing. The first step is the network build, the layout of the network as highlighted in the planning section, showing the segregation of the network. This segregation will be implemented in network development and as a result fulfilling requirements 1.3.1-1.3.3 and 1.3.7

4.2 Virtual Network

The first step in configuring the network is setting up the network switches. To implement the high level of segregation the switches for the LAN, DMZ and CDE are implemented within vSphere. The switches are configured as separate networks and all communications will be restricted via router and firewall configurations. Unlike physical switches rather than configuring VLAN IDs and IP subnetting via serial port the virtual network requires the IP ranges and subnets to be configured upon set-up and then the VMs can be added to the appropriate VLANs. The CDE has a stricter set of subnetting rules to restrict the number of devices enabled on a switch and avoiding open ports as shown in Figure 4. vSphere allows for simple scalability as if more active ports are required it does not require a full-network configuration but rather only an adjustment of the properties tab meaning a higher level of security as negating the need for unnecessary open ports.

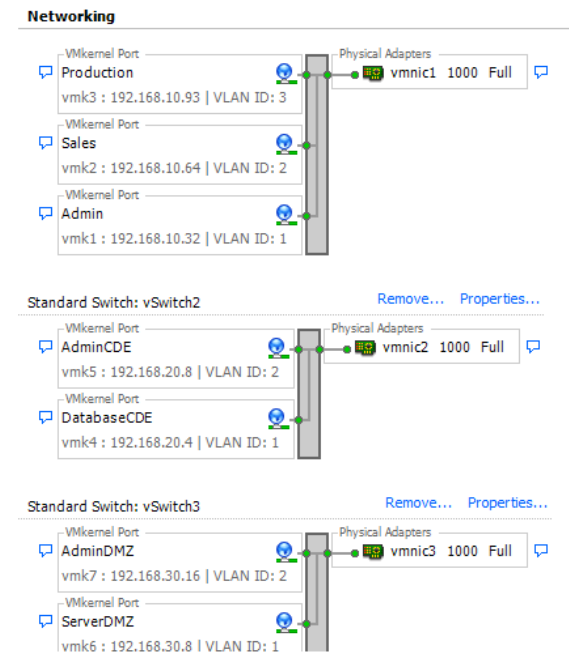


Figure 4: Network Switches

4.3 CDA and DMZ Initial Configuration

Ubuntu Server is being implemented as the CDE and DMZ servers. Two new virtual machines are created and the Ubuntu Server image is chosen as the OS. The country, language and keyboard layout are chosen appropriately once prompted. At this point the server is configured to comply with the first CIS audit (Center of Internet Security, 2015) requirements, while this requirement is part of PCI DSS requirement 2, this aspect of the CIS audit is easier to configure during the installation process due to disk partitioning. The partitions are set up as can be seen in Figure 5. This implements segregates application and OS information, however in production these disk spaces would be much larger due to building many VMs for the testing environment resources are applied sparingly.

NAME	FSTYPE	SIZE	MOUNTPOINT	LABEL
sda		30G		
├─sda1		1K		
├─sda2	ext4	20.2G	/	
├─sda5	swap	487M	[SWAP]	
├─sda6	ext4	1.9G	/home	
├─sda7	ext4	1.9G	/tmp	
├─sda8	ext4	1.9G	/var	
├─sda9	ext4	1.9G	/var/log	
├─sda10	ext4	1.9G	/var/log/audit	
sr0		1024M		

Figure 5: Ubuntu Hard Disk Partitions

The remaining CIS audit techniques will be configured post-server installation due to access limitations post-audit. This initial configuration is utilised for both the web server and database server on the network.

4.4 Database Server

For the database server in the CDE PhpMyAdmin and MySql-Server are installed. On installation PhpMyAdmin and MySql require passwords to be set. This is completed on the prompt screen. The next step is to log into PhpMyAdmin with the default root username and password. For PCI DSS default log in credentials must be removed, PhpMyAdmin has three default user accounts. To remove the default login settings new users will first have to be created. This is completed by selecting new user and adding a new username and password. For the database server two users are assigned as shown in Figure 6, one for administration and one for maintenance.

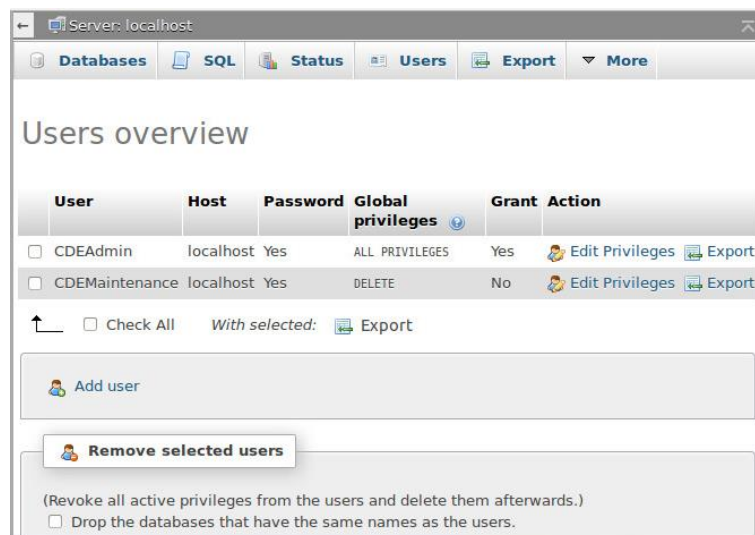


Figure 6: Removal of Default PhpMyAdmin Users

The administration user is granted administrative privileges in the database. The second user maintenance will be utilised for removing non-required data after 6 months, this user is granted no write privileges only delete, to ensure that this is done only as required the resource limits are set accordingly. The resource settings limits the frequency one user can connect and how many queries they can enter in a period of time. The phpMyAdmin configuration files can be viewed in Appendix B. To implement the SQL tables to interact with OpenCart a new database is created within phpMyAdmin. The SQL provided with OpenCart is loaded into the database to create the fields required. Upon completion of the

database set up the remaining CIS Ubuntu audit (Center of Internet Security, 2015) is completed to secure the OS, Figure 7 shows all of the areas secured by this audit. Implementing the CIS audit on Ubuntu virtual Machines fulfils requirement 2.2 of PCI DSS 3.0 and provides a higher level of OS security as the audit techniques harden the systems OS to industry standard.

Task	Status
Patching and Software Updates	Completed
File System Configuration	Completed
Secure Boot Settings	Completed
Additional Process Hardening	Completed
OS Services	Completed
Special Purpose Services	Completed
Network Configuration and Firewalls	Completed
Logging and Auditing	Completed
System Access and Authorisation	Completed
User Accounts and Environments	Completed
Warning Banners	Completed
Verify System File Permissions	Completed
Review User Group and Settings	Completed

Figure 7: Audit Ubuntu Server Checklist (Center of Internet Security, 2015)

The CIS audit as outlined in PCI DSS requirement 2.2 on the servers also implements many other of the standards requirements. Logging, file integrity management, user groups and password requirements are all configured on the server via the audits guidelines and full configuration files can be viewed in Appendix C. While these guidelines do enforce logging, IDS and file integrity, this is only implemented on a host level, further security measures will be implemented on a network level in the firewall configurations.

4.5 Web Server Configuration

After the Initial OS installation as explained previously, apache2 is installed on the server by entering `sudo apt-get install apache2`. The functionality of the server is then tested by

entering localhost into the browser window and the below page appears confirming the successful installation of Apache2

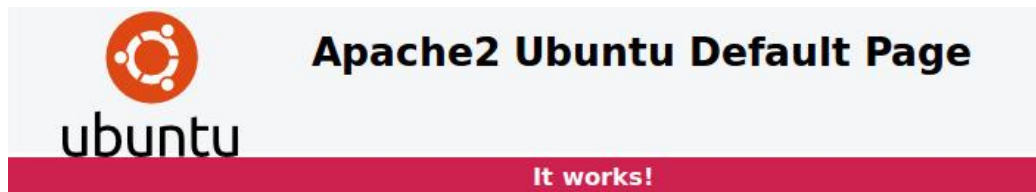


Figure 8: Apache Server Confirmation

PhpMyAdmin is also installed on this server to allow communication to the CDE server by entering `sudo apt-get install phpmyadmin`. After successful installation the php configuration settings need to be updated. The 'phpmyadmin/conf.inc.php' file is edited to connect to the database server located at 192.168.40.10 rather than the default of 12.0.0.7. This means that all sensitive information will be located in the CDE rather than on a MySQL server within the DMZ. PhpMyAdmin default users are removed using the same process as used with the database server.

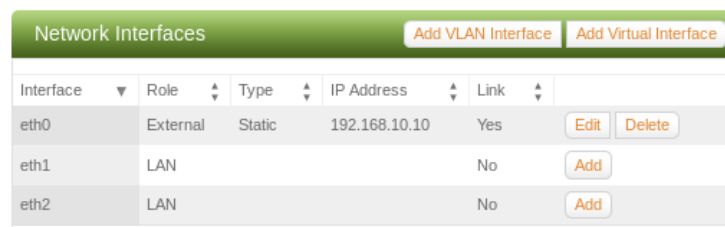
The server is also hardened to comply with the CIS Apache audit (Center of Internet Security, 2012), areas of security that are added to the server are represented in Figure 9. The apache2 affected configuration files are located in Appendix D.

Task	Notes	Status
Planning Checklist	Will be completed with IDS and Firewall	Pending
Minimize Apache Modules		Completed
Permission and Ownership Configuration		Completed
Access Control Configuration		Completed
Minimize Features and Options		Completed
Logging Monitoring and Maintenance		Completed
Use SSL/TLS		Completed
Information Leakage		Completed
DDOS Mitigation		Completed
Request Limits		Completed

Figure 9: CIS Audit Apache Server Checklist (Centre of Internet Security, 2012)

4.6 ClearOS Setup

To allow communications between the Virtual Network and the ISP router ClearOS is installed as the external facing network router. Clear OS is configured with three NICs one with a NAT address to allow communications to the physical ISP router and the other two are the host only NICs, one for the LAN and the other the CDE. To allow for ClearOS to successfully communicate between the internal and external network a static IP from the ISP is required and assigned to eth0. The new virtual machine is created with the three NICs and ClearOS is selected as the disk image. Once the virtual disk is configured with the OS the router and firewall can be modified. The initial step is to ensure ClearOs is set in gateway mode so traffic is filtered. ClearOS automatically assigned the NICs correctly with eth0 external and eth1 and eth2 internal as shown in Figure 10.



Interface	Role	Type	IP Address	Link	
eth0	External	Static	192.168.10.10	Yes	Edit Delete
eth1	LAN			No	Add
eth2	LAN			No	Add

Figure 10: ClearOS Interfaces

To further protect the network both firewall and IDS configurations are applied on ClearOS via the marketplace GUI. The ClearOS firewall utilises iptables for firewall protection and Snort for IDS. These are implemented via the ClearOS guidelines for firewall (ClearCentre, 2014a) and IDS (ClearCentre, 2015b) integration.

The last thing required for the firewall is to configure logging. The ClearOS firewall comes with logging viewer installed; to implement the feature log viewer is activated. This keeps files of all the firewall logs and also provides a filter system for system administrators, for easier log management. Clear OS configuration files are located in Appendix E.

4.7 Smoothwall Setup

As the network design also requires firewall routers for communications, two further routers are configured on the network. Smoothwall also requires two NICs one facing the internal network and one facing the DMZ. This regulates the flow of traffic so everything from the internal LAN flows through the Smoothwall routers and is forwarded to Clear OS

router. As there is no external facing NICs Smoothwall unlike the Smoothwall configuration they are not automatically assigned so are chosen manually again with eth0 as external facing the DMZ and eth1 internal facing the LAN on the first router and on the second Smoothwall router eth1 faces the CDE. As Smoothwall is a Linux distribution it also utilises iptables, the full configuration can be seen in Appendix F.

```

[root@smoothwall etc]# ls
adjtime          fstab            ld.so.conf      nsswitch.conf   services
apcupsd/         group           localtime@     options         shadow
at.deny          group-          login.defs      passwd          shadow-
bashrc           gshadow         logrotate.conf  pcmcia/         snort/
certparams      hosts@          logrotate.d/    ppp/            snort.conf
cron.d/          httpd/          mime.types      profile         syslog.conf
cron.daily/     ibod.cf@       miniupnpd.conf protocols        threshold.conf
cron.hourly/    inittab        miniupnpd.uuid rc.d/            udev/
cron.monthly/   inputrc        mke2fs.conf    resolv.conf     whois/
cron.often/     ioptions       modules         resolv.conf.dnsmasq
cron.weekly/    iproute2/     modules.conf   rpc
crontab         ipsec.conf@   mtab@          sensors.d/
eciads/         ld.so.cache   nicdevices     sensors3.conf
[root@smoothwall etc]# _

```

Figure 11: Smoothwall Configuration and Logging Options

4.8 Security Protocols

To allow for secure communications from the servers OpenSSL is installed by entering `sudo apt-get install openssl`. Due to the known weaknesses with SSL, the PCI Security Standards Council are releasing an updated version of PCI DSS v3 to remove SSL as a compliant protocol. So these changes won't affect the network it is configured with TLS. This step is completed by entering the command `sudo nano /etc/ssl.conf` and locating the line that says `Protocols All`. This line is changed to: `Protocols +TLSv1.2, +TLSv.1`. In the `ssl.conf` file the cipher suite values are also ensured that no weak encryption ciphers are utilised. For the apache server to communicate using this method encrypted keys need to be created and imported into the apache2 SSL folder, the keys are created as shown in Figure 12.

```

Generating a 2048 bit RSA private key
.....+++
.....+++
writing new private key to '/etc/apache2/ssl/apache.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:IE
State or Province Name (full name) [Some-State]:Donegal
Locality Name (eg, city) []:Letterkenny
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Student
Organizational Unit Name (eg, section) []:Student
Common Name (e.g. server FQDN or YOUR name) []:L00046121
Email Address []:l00046121@student.lyit.ie

```

Figure 12: Implementation of RSA SSL

To ensure that the TLS protocol has been successfully applied to the localhost a connection to the localhost is tested with the command `openssl s_client - connect localhost: 443`. The output as shown in Figure 13 shows the successful configuration of TLS on the server.

```

SSL-Session:
  Protocol  : TLSv1.2
  Cipher    : ECDHE-RSA-AES256-GCM-SHA384
  Session-ID: 57451F3FCC7AFC885FD03510A15E4827FB3E5D698FE07831A4DE55FA20994345
  Session-ID-ctx:
  Master-Key: 049E08F79937C654123BCDCAC8CD3442B972B164AB703A7BF0946C9173973E3E789C365A61D01DAEF23F5957F4763CBC
  Key-Arg   : None
  PSK identity: None
  PSK identity hint: None
  SRP username: None
  TLS session ticket lifetime hint: 300 (seconds)
  TLS session ticket:
0000 - a5 c4 11 34 1c 52 2a d4-8d 64 d4 f0 0b bb 13 7c   ...4.R*.d....|
0010 - fa 79 99 33 12 24 2d 03-5d 12 77 71 36 eb 15 07   .y.3.$-.]wq6...
0020 - 38 ce 07 4a f6 96 9c e8-80 fc 49 34 12 ab a1 ca   8..J.....I4....
0030 - 66 f3 11 ce 68 df b7 f8-c5 38 bf 8c 0f 1f c6 ea   f...h...8.....
0040 - fb b7 7c 59 f8 58 db 20-be a2 9b 70 d5 c8 e0 e7   ..|Y.X. ...p....
0050 - e2 cd 07 26 ca b7 f9 b8-38 af 3c c5 89 f0 6f 70   ...&....8.<...op
0060 - 77 c4 5f 20 b9 14 0f 87-d1 6a df 2b 56 e8 5e 75   w._ .....j.+V.^u
0070 - 2d d3 c0 e5 c5 be 32 8f-a8 22 35 6c 68 06 19 6a   -....2..]5lh..j
0080 - 10 36 0b aa c2 39 8c b0-da b4 60 b0 0f 7f d1 c1   .6...9....`.....
0090 - 17 08 14 e6 fe 99 5b 16-12 db 41 a3 d8 9b 07 01   .....[...A.....
00a0 - 30 24 c9 df 5a ab 1d 30-1d 62 0d b4 45 0a db 3d   0$.Z..0.b...E...=
00b0 - 51 f2 74 2f c1 61 ac 42-e8 30 9b ab 94 9b 03 e1   Q.t/.a.B.0.....

```

```

Start Time: 1429976022
Timeout    : 300 (sec)
Verify return code: 18 (self signed certificate)

```

Figure 13: TLS v1.2 configuration

While this will secure all communications via https to the localhost, communications between the web server and database in the CDE also need to be secure. To complete this, a SSL certificate is created on the CDE database in the same manner as completed with the apache server. When the key is created the SQL configuration file `my.conf` is altered to

utilise this cert for secure communications. To ensure MySQL SSL configurations are successful the SSL variables are verified as shown in Figure 14.

```
mysql> show variables like '%ssl%';
+-----+-----+
| Variable_name | Value                               |
+-----+-----+
| have_openssl  | YES                                 |
| have_ssl      | YES                                 |
| ssl_ca        | /etc/mysql/ca-cert.pem            |
| ssl_capath    |                                     |
| ssl_cert      | /etc/mysql/server-cert.pem       |
| ssl_cipher    |                                     |
| ssl_key       | /etc/mysql/server-key.pem        |
+-----+-----+
7 rows in set (0.01 sec)
```

Figure 14: Successful MySQL SSL configuration

4.9 Wireless Security

The ISP router providing connectivity is set to use WPA2 secured connections as default. The encryption method used was AES which. The username and passwords are changed from the default setting as required by entering the routers IP address into the address bar of a connected device. The password set is a 16 character alpha-numeric password that contains no dictionary words to increase the level of security on the router

4.10 Database Encryption

OpenCart enables the sensitive data (payment and password information) being passed through the website to be salted and hashed, so sensitive data entering the database is already encrypted, however this was an issue in each website scanned, the hashing algorithms utilised in the code base are MD5. As PCI DSS 3.0 requires using strong cryptographic algorithms in the PHP files responsible for applying the hash values the algorithm used is changed to SHA256 for stronger encryption. The SALT is created with the open-ssl random number generator. While using SALT on the password does have performance implications it does add a much higher level of security as brute forcing of password attacks on passwords become much more complex due to the randomly generated values appended to the passwords, therefore the level of security outweighs the performance issues that will be encountered.

4.11 Secure Programming

While the report from the static code scan showed many lines of secure coding weaknesses, many of the results were for the same issues: shell commands, cross site scripting from non-sanitised input and the utilisation of MD5 and pseudo-random number generators. Before the website is connected to the server these vulnerabilities are mitigated for. The MD5 hashing algorithms are changed to use the stronger sha256 hashing algorithm and the 00random number generator `mt_rand()` is replaced with the more secure open-ssl random number generator as the generated values are less predictable.

The code base is secured further by editing the `php.ini` file on the Apache server to disallow shell commands to be executed.

```
; This directive allows you to disable certain functions for security reasons.
; It receives a comma-delimited list of function names. This directive is
; *NOT* affected by whether Safe Mode is turned On or Off.
; http://php.net/disable-functions
disable_functions = exec,passthru,shell_exec,system,proc_open,popen,curl_exec,c$
```

Figure 15: `php.ini` configuration

The remaining vulnerability is mitigated for by changing `register_globals` from the default ON to OFF. For an extra level of security the Toolkit package is downloaded. This is an open source library to help in the securing of php and it includes libraries for escape management. This is successfully added by downloading the toolkit and linking to the php code base.

4.12 Anti-Virus

To fulfil the PCI DSS 3.0 requirement Comodo anti-virus software for Linux systems is installed on the network devices. As Comodo is designed to be used on Linux OS it can be applied to both servers and desktops. Unlike other anti-virus software available for Linux platforms Comodo has the ability to quarantine affected files to a folder on the system for analysis, meaning that the affected files will not have access to system files, further securing network devices.

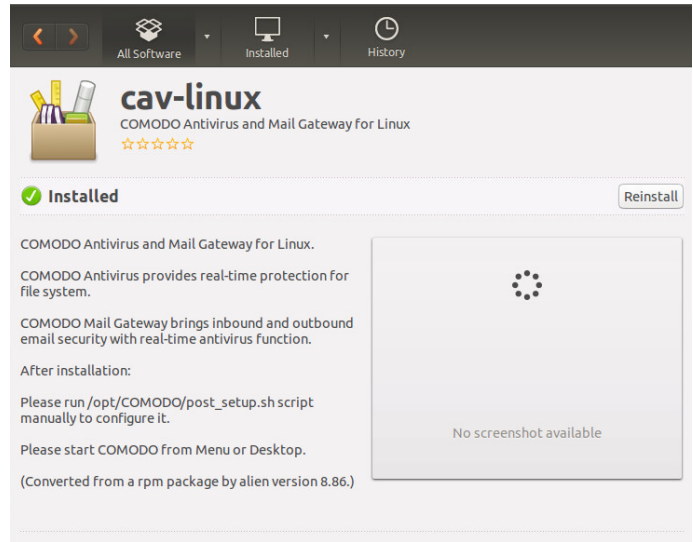


Figure 16: Comodo Installation

4.13 Self-Assessment

While this methodology successfully implements the full PCI DSS 3.0 standard there are also many self-assessment options available as shown in Figure 17. Many of these do not require the full implementation of all twelve PCI DSS 3.0 requirements. The testing section will not only test the network for weaknesses but along with this any found weaknesses will be compared to the to the self-assessment options to verify the strength of these alternative compliance options.

SAQ Version	Criteria	Necessary Requirements
A v3.0	Card not present merchants, all cardholder data functions fully outsourced	Requirements 9 and 12
A-EP v3.0	Partially outsourced e-commerce merchants using a third-party website for payment processing	All requirements necessary, however each aspect of the requirements is not.
B v3.0	Merchants with only imprint machines or only standalone dial-out terminals	Requirements 3, 4, 7, 9 and 12
B-IP v3.0	Merchants with standalone, IP-connected, point-of-interaction	All requirements except 5
C v3.0	Merchants with payment application systems connected to the Internet	All requirements necessary, however each aspect of the requirements is not.
C-VT v3.0	Merchants with web based virtual payment terminals	All requirements except 8,10 and 11
D Merchant v3.0	All other SAQ eligible merchants	All requirements necessary, however each aspect of the requirements is not.
D Service Provider v3.0	SAQ eligible service providers	All requirements necessary, however each aspect of the requirements is not.
P2PE-HW v3.0	Hardware payment terminals in a PCI - listed P2PE solution only	Requirements 3, 4, 9 and 12

Figure 17: Self-Assessment Versions

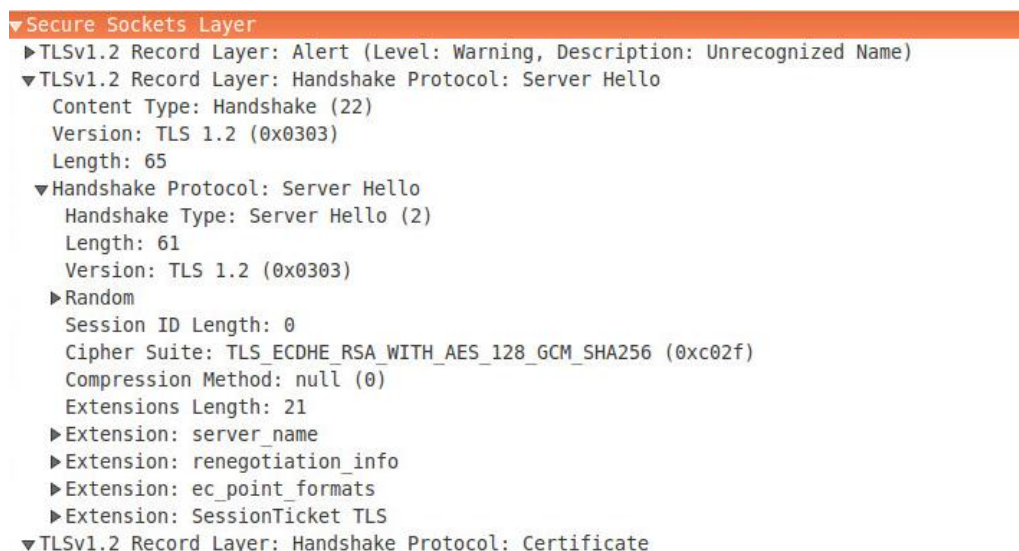
5 Testing

The testing methodologies that are explained below are based on the NIST SP 800-15 (Cody, et al., 2008) standard. Initially focusing on network communications then wireless access points, vulnerability scanning and finally penetration testing. The order completed testing the network from the outside inwards, as to initially harden the network from malicious outside traffic.

5.1 Secure Network Communications

To ensure communications via the web server are secure Wireshark is utilised on the webserver. To start the capture the loopback connection for the localhost server is selected as the target connection, once this is selected localhost is entered into a browser window and the capture is started. On the website an item is added to the shopping cart and the payment process is started.

Upon completion the Wireshark capture is stopped and the capture file of the transaction is opened. Figure 23 shows the successful implementation of TLS for secure encrypted communications. The TLS handshake is initiated before any transactions between the web server and client are undertaken.

A screenshot of a Wireshark network capture showing a TLS handshake. The top bar is orange and labeled 'Secure Sockets Layer'. Below it, the capture list shows several packets. The first packet is a TLSv1.2 Record Layer: Alert (Level: Warning, Description: Unrecognized Name). The second packet is a TLSv1.2 Record Layer: Handshake Protocol: Server Hello, with details expanded to show Content Type: Handshake (22), Version: TLS 1.2 (0x0303), and Length: 65. The third packet is a Handshake Protocol: Server Hello, with details expanded to show Handshake Type: Server Hello (2), Length: 61, Version: TLS 1.2 (0x0303), and various extensions including Random, Session ID Length: 0, Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f), Compression Method: null (0), and Extensions Length: 21. The fourth packet is a TLSv1.2 Record Layer: Handshake Protocol: Certificate.

```
▼ Secure Sockets Layer
  ▶ TLSv1.2 Record Layer: Alert (Level: Warning, Description: Unrecognized Name)
  ▼ TLSv1.2 Record Layer: Handshake Protocol: Server Hello
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 65
  ▼ Handshake Protocol: Server Hello
    Handshake Type: Server Hello (2)
    Length: 61
    Version: TLS 1.2 (0x0303)
    ▶ Random
    Session ID Length: 0
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
    Compression Method: null (0)
    Extensions Length: 21
    ▶ Extension: server_name
    ▶ Extension: renegotiation_info
    ▶ Extension: ec_point_formats
    ▶ Extension: SessionTicket TLS
  ▼ TLSv1.2 Record Layer: Handshake Protocol: Certificate
```

Figure 18: Wireshark TLS handshake

While this shows the initial TLS handshake to ensure all communications are secured the protocols for the entire transaction are captured. As highlighted in Figure 24 all packets to and from the server were completed using https for secure communications. All data sent to

and from the server is using this secure communication technique, mitigating for sensitive data being sniffed in clear text from the network communications.

```

10 0.453602000  :::1  :::1  TCP 86 38492 > https [ACK]
12 0.454387000  :::1  :::1  TCP 86 https > 38492 [ACK]
13 0.456669000  :::1  :::1  TCP 94 38493 > https [SYN]
14 0.456696000  :::1  :::1  TCP 94 https > 38493 [SYN]
15 0.456713000  :::1  :::1  TCP 86 38493 > https [ACK]
17 0.457036000  :::1  :::1  TCP 86 https > 38493 [ACK]
19 0.541927000  :::1  :::1  TCP 86 38493 > https [ACK]
21 0.542782000  :::1  :::1  TCP 86 38492 > https [ACK]

```

```

12 0.454387000  :::1  :::1  TCP 86 https > 38492 [ACK] Seq=1 Ack=230 Win=44800 Len=0 TSval=28
Transmission Control Protocol, Src Port: https (443), Dst Port: 38492 (38492), Seq: 1, A
Source port: https (443)
Destination port: 38492 (38492)
[Stream index: 1]
Sequence number: 1 (relative sequence number)
Acknowledgment number: 230 (relative ack number)
Header length: 32 bytes
▼Flags: 0x010 (ACK)

```

Figure 19: HTTPS Communications

5.2 Vulnerability Scanning

Nikto is utilised to assess vulnerabilities on the web server. To initiate the vulnerability scan the Nikto configuration file first needs to be edited to enable scans of localhost proxies. This is completed by removing the comments from the proxy configuration in the Nikto config.text file as show in Figure 22.

```

GNU nano 2.2.6 File: config.txt
# Proxy settings -- still must be enabled by -useproxy
PROXYHOST=127.0.0.1
PROXYPORT=8080
PROXYUSER=proxyuserid
PROXYPASS=proxypassword

# Cookies: send cookies with all requests
# Multiple can be set by separating with a semi-colon, e.g.:
# "cookie1="cookie value";"cookie2"="cookie val"
#STATIC-COOKIE=

# The below allows you to vary which HTTP methods are used to check whether an S
# is running. Some web servers, such as the autopsy web server do not implement$
CHECKMETHODS=HEAD GET

# If you want to specify the location of any of the files, specify them here
# EXECDIR=/opt/nikto
# PLUGINDIR=/opt/nikto/plugins
# TEMPLATEDIR=/opt/nikto/templates

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell

```

Figure 20: Nikto Configuration

Once the changes above have been completed the scan is completed by inserting the command `nikto -h localhost`.

5.2.1 Findings

Weaknesses were discovered in the web server utilising Nikto, please refer to Appendix H for the outputted Nikto scan results. Seven vulnerabilities were returned by the scanner, all of which are cross site scripting weaknesses

1. `phpimageview.php` located in PHP Image View 1.0 is vulnerable to Cross Site Scripting
2. `links.php` in myphpnuke is vulnerable to Cross Site Scripting
3. `modules.php` in Post Nuke 0.7.2.3-Phoenix is vulnerable to Cross Site Scripting
4. `members.asp` in Web Wiz Forums 7.01 and below is vulnerable to Cross Site Scripting
5. `forum_members.asp` in Web Wiz Forums 7.01 and below is vulnerable to Cross Site Scripting

5.3 Wireless Testing

Kali is utilised to test the wireless security on the network. Kali comes with Aircrack already installed and this piece of software will be utilised in an attempt to hack into the wireless network.

5.3.1 Aircrack

To utilise in Aircrack in Kali a wireless network adapter must be installed on the virtual machine. To initiate the testing `iwconfig` is entered to ensure the virtual machine successfully recognised the wireless network adapter, furthermore to ensure `wlan0` is active the command `airmon-ng start wlan0` is entered. This command also creates a `mon0` testing interface. The next step is to scan for wireless interfaces. This is completed with the command `airodump-ng mon0`. This returns a screen as shown in Figure 18 that shows the name of the wireless router, the encryption protocols utilised and the channel.

```
CH 14 ][ Elapsed: 40 s ][ 2015-04-21 15:59
BSSID          PWR Beacons  #Data, #/s  CH  MB  ENC  CIPHER AUTH ESSID
90:EF:68:3E:7F:4E -51    127      313   0   3  54e  WPA2 CCMP  PSK  eirco
BSSID          STATION    PWR   Rate    Lost   Frames  Probe
(not associated) 7C:E9:D3:75:12:87 -67   0 - 1     0     5  eircom20370
90:EF:68:3E:7F:4E FC:0F:E6:D9:3A:DD -1    54 - 0     0    307
```

Figure 21: Scan for Wireless Routers

Aircrack injects packets to force a client to reconnect to the wireless router. To be able to capture this process a Wireshark file of the process is required to be captured. To complete this `airodump-ng -bssid 90:EF:68:3E:7F:4E -channel 3 write wificapture mon0`. This creates a Wireshark file to store the data of the packet injection. To insert the packets `aireplay-ng -0 3 -a 90:EF:68:3E:7F:4E -c FC:0F:E6:D9:3A:DD mon0` is entered into the terminal. Figure 19 shows the successful injection of the de-authorisation packets between the two devices.

```

root@kali:/# aireplay-ng -0 3 -a 90:EF:68:3E:7F:4E -c 7C:E9:D3:75:12:87 mon1
16:18:32 Waiting for beacon frame (BSSID: 90:EF:68:3E:7F:4E) on channel 3
16:18:32 Sending 64 directed DeAuth. STMAC: [7C:E9:D3:75:12:87] [ 0 | 0 ACKs]
16:18:33 Sending 64 directed DeAuth. STMAC: [7C:E9:D3:75:12:87] [ 0 | 0 ACKs]
16:18:33 Sending 64 directed DeAuth. STMAC: [7C:E9:D3:75:12:87] [ 0 | 0 ACKs]
root@kali:/#

```

Figure 22: De-Authorisation Packet Injection

The wireshark file is filtered by the EAPOL protocol to ensure the packet injection was successful. This files show the re-authentication of the client to the wireless router. The packet injection was successful and the communications are encrypted via AES.

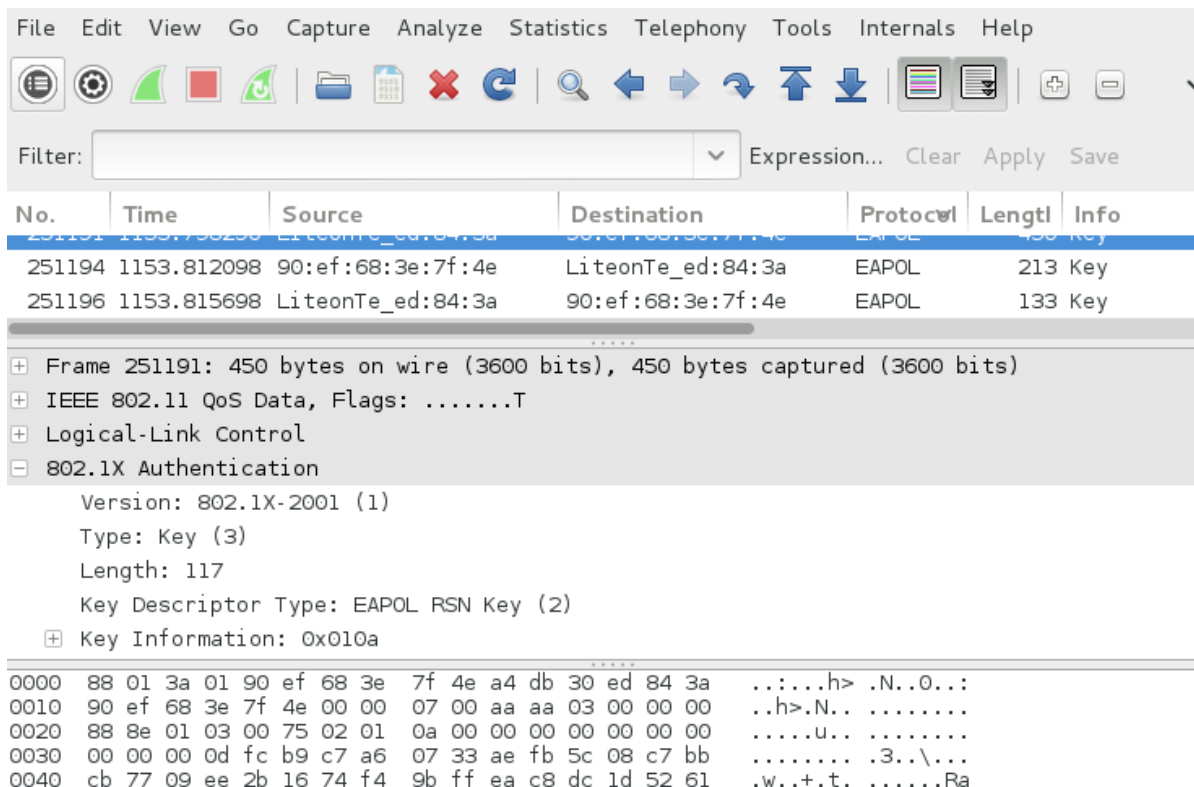


Figure 23: Wireshark capture of AES keys

To attempt to crack the password on the router a password cracker is run through aircrack, this compares encrypted values to the AES encryption values. To complete this darkc0de.lst is downloaded, it contains a comprehensive password list to compare to the Wireshark encryption values. To run the password cracking the command `aircrack-ng wificapture -w /usr/share/wordlists/darkc0de.lst`.

5.3.2 Findings

As illustrated in Figure 21 the WPA2 cracking attempt was unsuccessful, both dictionary and alpha-numeric values were entered as password values. As the wireless access key is comprised of a 14 characters, is alpha-numeric with no dictionary values the complexity of the password was not able to be cracked showing utilising this method.

```
Current passphrase: zwingliane

Master Key   : 29 05 DA DB FE F7 8E FB 7F 45 5B D0 81 D8 CF C0
              CB E4 D6 98 C8 66 68 74 AA 0B C2 74 51 D7 15 90

Transient Key : D3 8F C8 E0 9F 4F 01 1C AB 20 94 8A EA CD 39 0D
              08 86 04 50 6A 62 1B 1F 69 4F F6 9A 3A 1A 7F 5C
              72 E9 D0 66 F8 F3 89 5B AA F6 AD 2A 90 07 F1 27
              67 38 53 45 C1 37 12 DF 4E 42 F3 A5 00 37 74 43

EAPOL HMAC   : F9 14 89 7D 20 F2 83 89 30 AC DD 6C E0 B7 8A 55

Passphrase not in dictionary
```

Figure 24: Failed Aircrack Attack

5.4 Penetration Testing

Armitage and Metasploit are used together from the Kali virtual machine to penetration test all network devices. Armitage and Metasploit are used in the same manner on all devices on the network. To initiate the penetration testing Metasploit and PostgreSQL are started on the virtual machine and when both applications have successfully loaded `armitage` is entered into the terminal. To determine the operating system being utilised on the target machine it is selected in the Armitage window and an Nmap OS scan is completed.

Nmap successfully detects the operating system of the client and Metasploit uses this data to customise the exploits to the host OS. To see the available exploits `show payloads` is entered into the terminal, this shows all available payloads on the target system. To verify if the host machine is vulnerable the `check` command can be utilised to verify this. While the

process of finding exploits on each target machine is the machine, the results vary based on the OS. The full Metasploit scan findings can be viewed in Appendix G.

5.4.1 Ubuntu Servers Findings

While the servers were secured to comply with both CIS audits and PCI DSS 3.0 before testing, there were still vulnerabilities found on both machines. Both servers returned the same vulnerabilities with no result differences.

Cuteflow File Upload Exploit

This exploit was found on both Ubuntu servers as is a component that is automatically installed upon mounting operating system. This exploit required no credentials and allows the attacker to upload php files into the system and run them as executables, this is very dangerous as will allow these files to gain access to all information installed on host machine.

Skybluecanvas Exec Exploit

This vulnerability also does not require any login credentials for execution it allows attackers to execute shell commands via php editing database credentials when the PID value is equal to 4. The php.ini file on both servers has been configured to disallow shell commands aiding to mitigate for this type of attack, however to further secure systems the functionality should be assessed and removed if not required.

5.4.2 ClearOS Findings

Tectia Password Change Request Exploit

This vulnerability is found in UNIX systems that utilise Tectia Server for SSH functionality. The exploit allows an attacker to brute force the username, once this has been found a password reset request is sent and accepted by the server. This allows the attacker to change login credentials to ones of their own choosing and therefore after gaining access change the firewall rules to maintain network access.

5.4.3 Smoothwall Findings

Gitorious Graph Exploit

Metasploit successfully found and exploited the gitorious graph vulnerability on the Smoothwall firewall. This vulnerability exists in the gitorious component pre-installed on the

Smoothwall ISO. This allows for an attacker to enter un-validated input to start a command line session and gain control of the system.

5.5 Results Conclusion

While the network was configured with security in mind, the testing phase still successfully uncovered vulnerabilities in the network. In both the CDE and DMZ servers the functionality of cuteflow and skybluecanvas are not required so the files are purged from the systems. The same is the case for the SmoothWall firewall and vulnerable files are purged. For ClearOS as the user name was brute forced and no password was required, to mitigate for this the username is updated to comply with password best practices, utilising numbers letters and special characters.

While this form of testing is required on full implementation of the standard this not the case for all SAQ options. Furthermore, the issue with lack of guidance on testing frameworks provided may result in a network that is only as strong as the research carried out by the tester on other suitable frameworks. To highlight this particular issue with the standard, requirement 11.3 of PCI DSS 3.0 requires penetration testing be completed on the network. Figure 25 shows that framework NIST SP800-115 may be used however no others are listed for comparison.

11.3 Implement a methodology for penetration testing that includes the following:

- **Is based on industry-accepted penetration testing approaches (for example, NIST SP800-115)**

Figure 25: PCI DSS Requirement 11.3

For example on the network built in the methodology section if only CDE connections were tested vulnerabilities in firewalls and servers would remain. The vulnerabilities that would have remained would allow bypassing of external firewall due to password exploitations and the server would vulnerable to malicious php that could then be executed on the server.

Not only do many of the self-assessments fail to uncover the issues as outlined above there is a vast difference between the security levels enforced when comparing the full standard

and the SAQs. While network and code weaknesses in this implementation would have successfully uncovered these weaknesses pre-production only fulfilling SAQ requirements may lead to weak coding and networking practices.

5.5.1 SAQ A v3.0

This self-assessment version while only requires requirement 9 and 12 to be fulfilled the prerequisite for this self-assessment is that all payment information is fully outsourced to a third party, requirement 12 does require that a contract is in place so the third party vendors accept responsibility for ensuring their products are secured to the full standard, therefore the merchant is not responsible for any weaknesses that would have been uncovered in both the methodology and testing procedures.

5.5.2 SAQ A-EP v3.0

SAQ-EP v3 while covers all twelve of the PCI DSS requirements to be fulfilled, not every aspect of each requirement are necessary. The documentation of user privileges, access and information security are not required as per the full standard, meaning in the event of a breach inside from inside the network assisting documentation to assist in finding the source is severely limited.

5.5.3 SAQ for Payment Terminals

SAQ B v3.0, SAQ B-IP v3.0 and AQ P2PE-HW v3.0 self-assessment options are available for card payment machines and are not an option for e-commerce payment applications, however with these options for payment card machines there is a lack of required documentation for physical access to devices. With physical tampering of devices being a high risk, a secure network build and documentation of access should have a higher implementation requirement than covered by these self-assessment versions.

6 Conclusion

At its core, PCI DSS v3.0 represents best practice guidelines which, if followed, mitigate the risk from many cyber threats. Its implementation and methodological approach to hardening the security profile of an organisation offer considerable benefits. Indeed, recognizing the protection of privacy and civil liberties has come to engender the central issue regarding trust between the general public and organisations. However, 'one-size-fits-all' will not work given the diversity of company objectives and practices. Added to this, the security professional is faced with considerable challenges from given the sophisticated nature of new threats (Trustwave, 2013; Verizon, 2013; PWC, 2014; Ponemon, 2014). Despite the wealth of experience and lessons learned by security professionals, more work is needed. Existing defences can be improved by developing the most appropriate security posture for an organisation which may be aimed at preventing common types of attack from damaging the business assets. Recently, the National Institute of Standards and Technology (NIST) published the "Framework for Improving Critical Infrastructure Cybersecurity," (NIST, 2014) and represents a valuable tool to help organizations assess and improve their security systems. In conjunction with this, security training and certification are required and may be implemented relatively quickly so that any gaps in knowledge may be filled. Benefits to organizations include, but are not limited to, increased customer confidence, standardized sharing and storage of sensitive information and the simplification of practices and operations across international regimes. The work presented offers a progressive and reasonable addition to security education, since there is a dearth of research into the advantages of PCI DSS 3.0 implementation the table provided in Appendix I provides an in-depth review on how each implementation step fulfilled each requirement. On this basis, several areas which are worthy of consideration for those intent on implementing PCI DSS v3.0 from a compliance perspective, the perceptions of industry, the importance of third party tools, how delays in adoption and subsequent adaptation are costly, the importance of self-assessment, some inherent ambiguities in the standard that could be changed and the current restrictions of the standard.

6.1 Slow to Adapt

The emergence of Heartbleed and POODLE PCI DSS has prompted the revision of the standard. However Heartbleed was discovered in April 2014 after which POODLE came to light in October 2014, the PCI Security Standards Council stated in February 2015 that a revised standard will be released ending the acceptance of SSL as secure. This comes ten months after the initial vulnerability has come to light. In comparison Chokhani et Al (2014) released the NIST guide to TLS in April 2014. This document states that SSL is not an approved means of encryption and updated this standard to TLS. This shows a vast gap in both re-action times and standard strength. NIST tested and adapted within a month, however PCI DSS have only just formally released version 3.1 a full year later and are also giving merchants another year to update their systems. While PCI DSS 3.0 has officially been retired, the updated requirements of 3.1 will not be fully enforced until June 2016 (PCI Security Standards Council, 2015).

6.2 PCI DSS 3.0 Ambiguities

While PCI DSS 3.0 does provide a more secure implementation and testing guideline than its predecessor there is still vagueness in certain aspects of the standard. PCI DSS 3.0 does require that many aspects like penetration testing, implementing cryptographic algorithms be completed to industry best practice, there is very little information given on the industry best practice guidelines to be followed. It is understood why this is left vague as if a new vulnerability is found, an outlined practice may be outdated quickly, but rather than leaving room for interpretation these aspects should be more specific. Stating that penetrations should be completed to an industry framework such as NIST or another industry framework leaves ambiguity to which testing standard can be utilised. Providing a guideline such as systems should be tested utilising the most recent NIST framework, means the framework will stay up to date and there is no ambiguity as to which industry framework should be utilised.

The same issue exists in the cryptographic section, it is required that strong hashing algorithms be utilised with no explanation of a strong hashing algorithm. It is well known that MD5 is broken and should not be used, so updating the standard to be more specific and require that SHA256 and above be utilised for hashing data leaves much less room for interpretation and therefore implementation errors. This was highlighted during the static

code scan of all code bases, each code base while being advertised as PCI DSS 3.0 compliant utilised MD5 hashing algorithms.

6.3 Self-Assessment

It is clear that enforcing security is an expensive task and the self-assessment procedures have been enforced to aid smaller businesses enforce security without the expense of a full audit. If a business chooses to handle consumers sensitive information, having that business itself assess the security measures in place is not good enough. While the self-assessment document is utilised for when card holder information is not stored, if the data is being taken from the consumer communicated across the network, surely the network communications and security levels would be tested to the same extent as when the data is being stored. Data breaches are not just confined to storage breaches, but man in the middle breaches are just as severe and all communications with sensitive data should be protected and evaluated to the same extent. As Figure 17 showed previously there are many security requirements that would not be implemented with the self-assessment options. Requirement 11 helped secure the network further, using common attack tools a variety of weaknesses were uncovered which would remain in self-assessment versions resulting in a vulnerable infrastructure.

6.4 Failure to comply

The largest issue with PCI DSS 3.0 is not with the standard itself rather than a lack of compliance, or understanding of the standard. While the standard cannot 100% guarantee that a breach will not happen, no security standard or compliance document can, Verizon (2015) does show there is a large gap in compliance, with all post-breach companies assessed by Verizon in 2014 being found to be non-compliant and only 33% of QSA found to pass the testing requirement.

When a company makes the decision to accept and store cardholder information PCI DSS should be a pre-requisite not a choice. While there are fines for breaches and non-compliance this does not seem to be enough to encourage companies to enforce security measures. With sensitive and private data at risk the standard needs to be enforced from the start.

6.5 PCI DSS 3.1

PCI DSS is constantly evolving with many iterations of a standard being issued throughout its lifetime. Close to the completion of this dissertation PCI DSS 3.1 was released implementing new guidelines and clarifications on PCI DSS 3.0. However this dissertation focused on ground up approach to security, therefore remaining valid for PCI DSS 3.1. As can be seen in the guidelines submitted by the PCI Security Standards Council (2015) the revised version mainly focus on standard clarifications, the only requirements change is the removal of SSL as a secure communications method.

6.5 Restrictions

While further tests with alternative hardware and software configurations would have yielded data for comparison, this was restricted by financial and time issues. However, given that the tests were performed on a relatively common setup further work could be carried out using industry grade platforms and tools.

6.6 Further Work

There is much further work that can be initiated to further the work of this dissertation. This dissertation focused assessing the standard in a testing environment. To assess the standard even further the next step would be to move the network to a physical environment and assess any differences in the findings that may come to light. Furthermore the full network and site can be further assessed in a live production environment.

To further evaluate any ambiguities and testing issues in regard to requirement 11, further testing is required on the network with varying testing frameworks to evaluate their strengths and weaknesses. This will aid in discovering the varying levels of security that the different frameworks provide.

With the new PCI DSS 3.1 being released in April, assessing the changes, how they further the security that PCI compliance enforces. The major change being enforced in this standard is how data in transit is secured. Since Heartbleed and POODLE have been discovered as large weaknesses in SSL communications, PCI DSS have updated the standard to try and mitigate this by enforcing the move from SSL to TLS. Further work would be to enforce this change on a network utilising SSL and fully assess the TLS standard itself. As well as assessing any further changes that may be enforced in the revised standard.

Bibliography

Alshomrani, M., 2012. The Importance and Dilemmas of Security Education in Information System. *WIAR '2012; National Workshop on Information Assurance Research*, 1(1), p. 1-5.

American Bar Association, 2008. *Data Security Handbook*. 1st ed. Chicago: ABA Publishing.

Atay, L., Bahtiyar, S. & Gur, G., 2014. Security Assessment of Payment Systems under PCI DSS Incompatibilities. *CT Systems Security and Privacy Protection*, 428(2), p. 395-402.

Bakay, O., Dudykevych, V. & Lakh, Y., 2013. Investigation of Payment Cards Systems Information Security Control. *IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems*, 2(651-654), p. 7.

Bhargav, A., 2014. *PCI Compliance: The Definitive Guide*. 1st ed. Florida: CRC Press.

Blackwell, C., 2008. The Management of Online Credit Card Data using the Payment Card Industry Data Security Standard. *Digital Information Management*, 3(840-842), p. 3.

Carpenter, T., 2010. *SQL Server 2008 Administration*. 1st ed. New Jersey: John Wiley & Sons.

Center of Internet Security, 2012. *Security Configuration Benchmark for Apache HTTP Server*.

[Online]

Available at:

https://benchmarks.cisecurity.org/tools2/apache/CIS_Apache_HTTP_Server_Benchmark_v3_1.0.pdf

[Accessed 2 February 2015].

Center of Internet Security, 2015. *CIS Ubuntu 14.04 LTS Server Benchmark*. [Online]

Available at:

https://benchmarks.cisecurity.org/tools2/linux/CIS_Ubuntu_14.04_LTS_Server_Benchmark_v1.0.0.pdf

[Accessed 02 February 2015].

Chokhani, S., McKay, K. & Polk, T., 2014. Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations. *NIST Special Publication*, 1(1), p. 9-67.

Chuvakin, A. & Williams, B., 2011. *PCI Compliance: Understand and Implement Effective PCI Data Security Standard Compliance*. 3rd ed. Massachusetts: Syngress Publishing.

Chuvakin, A. & Williams, B., 2014. *PCI Compliance: Understand and Implement Effective PCI Data Security Standard Compliance*. 4th ed. Massachusetts: Syngress Publishing.

Clear Center, 2014a. *Clear Center*. [Online]

Available at:

http://www.clearcenter.com/support/documentation/clearos_enterprise_5.2/user_guide/custom_firewall

[Accessed 10 February 2015].

Clear Center, 2014b. *Clear Center*. [Online]

Available at:

http://www.clearcenter.com/support/documentation/user_guide/intrusion_detection

[Accessed 10 February 2014].

Coburn, A., 2010. Fitting PCI DSS Within a Wider Governance Framework.. *Computer Fraud & Security*, 9(2), p. 11-13.

Cody, A., Orebaugh, A., Scarfone, K. & Souppaya, M., 2008. *NIST*. [Online]

Available at: <http://csrc.nist.gov/publications/nistpubs/800-115/SP800-115.pdf>

[Accessed 21st May 2015].

Creti, A. & Verdie, M., 2014. Fraud, investments and liability regimes in payment platforms. *International Journal of Industrial Organization*, 35(7), p. 84-93.

Douthit, P. & Huang, K., 2008. ST&E is the Most Effective Measure to Comply with Payment Card Industry (PCI) Data Security Standard. *Financial Cryptography and Data Security*, 12(18), p. 321-322.

Fernandes, J. J., 2014. Get ready for PCI DSS 3.0 with real-time monitoring. *Computer Fraud & Security*, 2015(2), p. 17-18.

Grobler, M., Li, C. & Mouhtaropoulos, A., 2014. Digital Forensic Readiness: Are We There Yet?. *Journal of International Commercial Law and Technology*, 9(3), p. 173-179.

Guttman, B. & Roback, E., 1995. *An Introduction to Computer Security: the NIST Handbook*. 1 ed. Gaithersburg: National Institute of Standards & Technology.

Hall, J., 2014. *PCI Guru*. [Online]

Available at: <https://pciguru.wordpress.com/2014/08/01/the-dilemma-of-pci-scoping/>

[Accessed 12 November 2014].

Hulme, G., 2009. *Dark Reading*. [Online]

Available at: <http://www.darkreading.com/risk-management/the-death-of-pci-dss-dont-be-silly/d/d-id/1076031?>

[Accessed 13 November 2014].

Johnson, R., 2014. *Security Policies and Implementation Issues*. 2nd ed. Massachusetts: Jones & Bartlett Publishers.

Kedgley, M., 2014. PCI DSS Version 3.0: new standard, but same problems?. *Computer Fraud & Security*, 2(1), pp. 5-9.

Kidd, R., 2008. Counting the cost of non-compliance with PCI DSS. *Computer Fraud & Security*, 11(6), p. 13-14.

Koster, J., 2012. *The SANS Insititute*. [Online]

Available at: <http://www.sans.org/reading-room/whitepapers/compliance/in-house-penetration-testing-pci-dss-33930>

[Accessed 22 May 2015].

Li, C. & Thornton, B., 2013. The applicability of network management systems in small businesses. *IEEE Joint International Computer Science and Information Technology Conference*, 13(2), p. 1-7.

Lovric, Z. & Sedinic, I., 2013. Influence of Established Information Security Governance and Infrastructure on Future Security Certifications. *Information & Communication Technology Electronics & Microelectronics*, 36(4), p. 1112-1115.

McMurtrey, M., 2013. A Case Study of the Application of the Systems Development Life Cycle (SDLC) in 21st Century Health Care: Something Old, Something New?. *Journal of the Southern Association for Information Systems*, 1(1-3), p. 1.

Mehta, L., 2014. *Infosec*. [Online]

Available at: <http://resources.infosecinstitute.com/pci-dss-3-0-key-drivers/>
[Accessed 21st May 2015].

Morgan, R. & Boardman, R., 2012. *Data Protection Strategy: Implementing Data Protection Compliance*. 2nd ed. Andover: Sweet & Maxwell.

OpenCart, 2015. *OpenCart*. [Online]

Available at: <http://www.opencart.com/index.php?route=common/home>
[Accessed 31st March 2015].

PCI Security Standards Council, 2010. *PCI Security Standards*. [Online]

Available at:

https://www.pcisecuritystandards.org/pdfs/pci_lifecycle_for_changes_to_dss_and_padss.pdf

[Accessed 22 May 2015].

PCI Security Standards Council, 2010. *Summary of Changes from PCI DSS Version 1.2.1 to 2.0*. [Online]

Available at:

https://www.pcisecuritystandards.org/documents/pci_dss_v2_summary_of_changes.pdf

[Accessed 14 October 2014].

PCI Security Standards Council, 2013. *Requirements and Security Assessment Procedures*. [Online]

Available at: https://www.pcisecuritystandards.org/documents/PCI_DSS_v3.pdf

[Accessed 28 September 2014].

PCI Security Standards Council, 2014. *PCI Security Standards*. [Online]

Available at: https://www.pcisecuritystandards.org/documents/PCI_DSS_v3.pdf

[Accessed 3rd October 2014].

PCI Security Standards Council, 2015. *PCI Security Standards*. [Online]
Available at: https://www.pcisecuritystandards.org/documents/PCI_DSS_v3-1_Summary_of_Changes.pdf
[Accessed 15 April 2015].

Ponemon, 2014. *Ponemon Institute Releases 2014 Cost of Data Breach: Global Analysis*. [Online]
Available at: <http://www.ponemon.org/blog/ponemon-institute-releases-2014-cost-of-data-breach-global-analysis>
[Accessed 26th January 2015].

Roche, T., 2014. *File Integrity AIDE*. [Online]
Available at: <https://help.ubuntu.com/community/FileIntegrityAIDE>
[Accessed 2 February 2015].

Sakharova, I., 2012. Payment Card Fraud: Challenges and Solutions. *Intelligence and Security Informatics*, 1(2), p. 227-234.

Shier, J., 2014. *Naked Security*. [Online]
Available at: <https://nakedsecurity.sophos.com/2014/04/23/pci-dss-why-it-fails/>
[Accessed 12 November 2014].

Shihab, M. R. & Misdianti, F., 2014. Moving towards PCI DSS 3.0 compliance: A case study of credit card data security audit in an online payment company. *Advanced Computer Science and Information Systems (ICACSIS)*, 34(1), p. 151 - 156.

Sullivan, R., 2013. The US Adoption of Computer-Chip Payment Cards: Implications for Payment Fraud. 2013(1), 101-127.. *Economic Review: Federal Reserve Bank of Kansas City*, 13(1), p. 101-127.

TrustWave, 2014. *TrustWave Global Security Report*. [Online]
Available at:
https://www2.trustwave.com/rs/trustwave/images/2014_Trustwave_Global_Security_Report.pdf
[Accessed 28th January 2015].

Verizon, 2013. *Verizon Annual Report*. [Online]

Available at: http://www.verizon.com/about/sites/default/files/2013_vz_annual_report.pdf

[Accessed 25th January 2015].

Verizon, 2015. *Verizon PCI Report*. [Online]

Available at: http://www.verizonenterprise.com/resources/reports/rp_pci-report-2015_en_xg.pdf

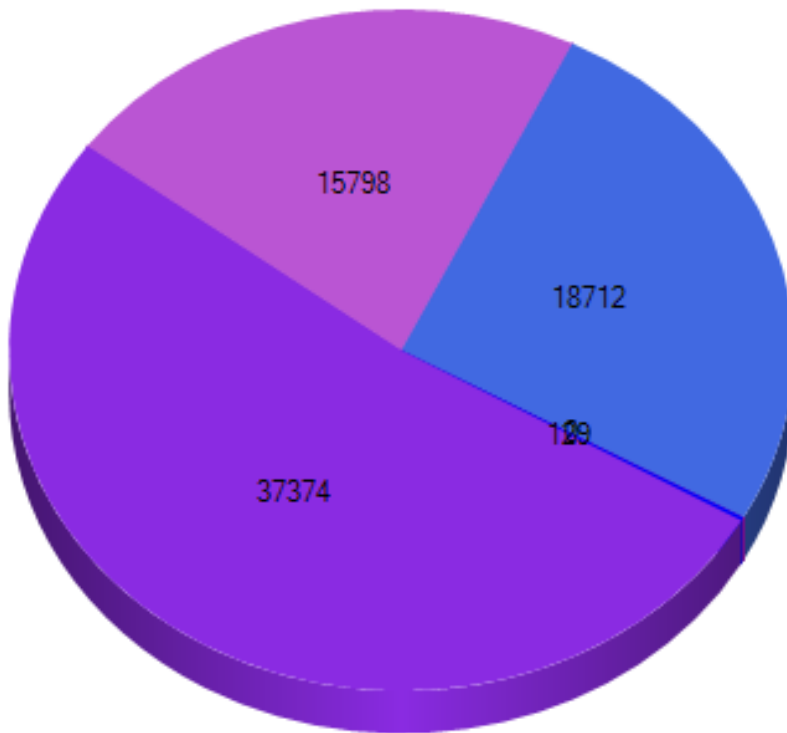
[Accessed 2 March 2015].

Yingkui , Z., Jing, Z. & Liye, W., 2010. Justification of Free Software and Its Enlightenment.

WRI World Congress on Software Engineering, 2010(2), p. 171-173.

Appendix A - VCG Source Code Scan Results

Open Cart Scan Results



- Overall code (37374 lines)
- Overall whitespace (15798 lines)
- Overall comments (18712 comments)
- Potentially broken/unfinished flags (0 Counts)
- Potentially dangerous code (129 Counts)

Result Findings

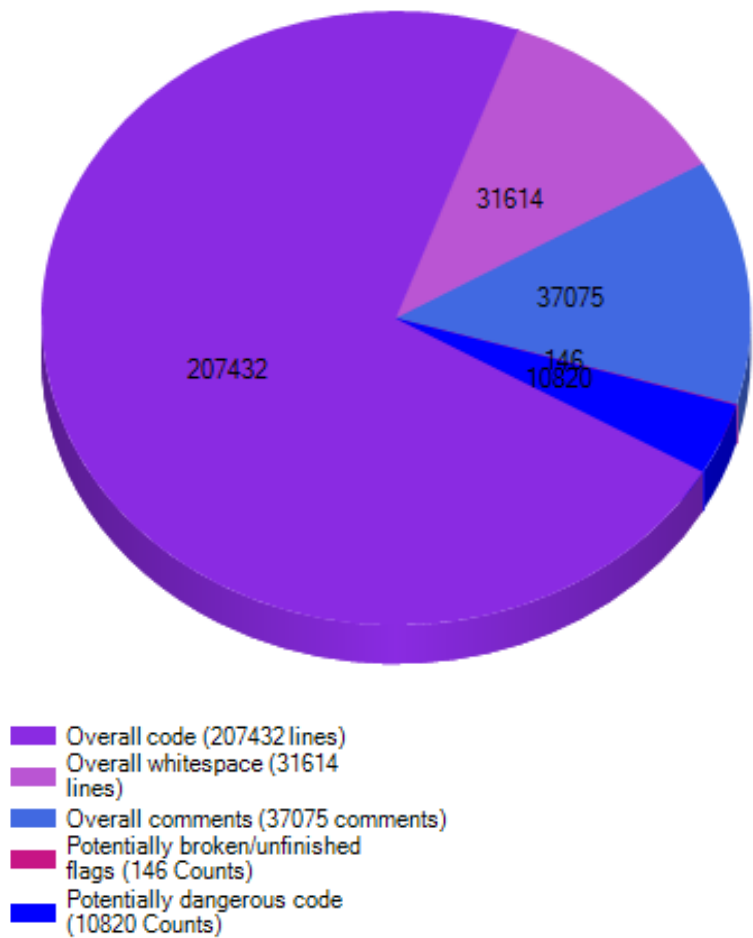
Severity	Title	Description	File Name
Medium	Application Variable Used on System Command Line	The application appears to allow the use of an unvalidated variable when executing a command.	ebay_settings.php
Medium	Application Variable Used on System Command Line	The application appears to allow the use of an unvalidated variable when executing a command.	ebay_settings.php
Medium	Application Variable Used on System Command Line	The application appears to allow the use of an unvalidated variable when executing a command.	ebay_settings.php
Medium	Application Variable Used on System Command Line	The application appears to allow the use of an unvalidated variable when executing a command.	openbay.php
Medium	md5	MD5 Hashing algorithm.	cba.php
Medium	md5	MD5 Hashing algorithm.	cba.php

Medium	md5	MD5 Hashing algorithm.	cba.php
Medium	md5	MD5 Hashing algorithm.	confirm.php
Medium	md5	MD5 Hashing algorithm.	document.php
Medium	md5	MD5 Hashing algorithm.	paymate.php
Medium	mt_rand	The application uses pseudo-random number generation that is not cryptographically secure.	amazon_checkout.php
Medium	mt_rand	The application uses pseudo-random number generation that is not cryptographically secure.	amazon_login_pay.php
Medium	mt_rand	The application uses pseudo-random number generation that is not cryptographically secure.	confirm.php
Medium	mt_rand	The application uses pseudo-random number generation that is not cryptographically secure.	download.php
Medium	mt_rand	The application uses pseudo-random number generation that is not cryptographically secure.	paymate.php
Medium	mt_rand	The application uses pseudo-random number generation that is not cryptographically secure.	pp_express.php
Medium	mt_rand	The application uses pseudo-random number generation that is not cryptographically secure.	pp_express.php
Medium	mt_rand	The application uses pseudo-random number generation that is not cryptographically secure.	pp_pro_iframe.php
Medium	mt_rand	The application uses pseudo-random number generation that is not cryptographically secure.	pp_pro_iframe.php
Medium	mt_rand	The application uses pseudo-random number generation that is not cryptographically secure.	sagepay_direct.php
Medium	mt_rand	The application uses pseudo-random number generation that is not cryptographically secure.	sagepay_server.php
Medium	mt_rand	The application uses pseudo-random number generation that is not cryptographically secure.	worldpay.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	amazon.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	amazonus.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	amazonus_product.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	amazonus_product.php

Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	response.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	sagepay_direct.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	sagepay_direct.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	sagepay_direct.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	sagepay_server.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	sagepay_server.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	sagepay_server.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	securetrading_pp.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	securetrading_pp.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	securetrading_pp.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	securetrading_ws.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	securetrading_ws.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	securetrading_ws.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	securetrading_ws.php
Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation or sanitisation.	worldpay.php
Standard	exec	This function allows execution of commands. It is dangerous when used with user controlled parameters and may facilitate direct attacks against the web server.	cli_install.php
Standard	preg_replace	This function will evaluate PHP code. It is dangerous when used with user controlled parameters and may facilitate direct attacks against the web server.	action.php
Standard	preg_replace	This function will evaluate PHP code. It is dangerous when used with user controlled parameters and may	address.php

		facilitate direct attacks against the web server.	
Standard	preg_replace	This function will evaluate PHP code. It is dangerous when used with user controlled parameters and may facilitate direct attacks against the web server.	download.php
Standard	system	This function allows execution of commands. It is dangerous with user controlled parameters and may facilitate direct attacks against the web server.	ebay_settings.php
Standard	system	This function allows execution of commands. It is dangerous with user controlled parameters and may facilitate direct attacks against the web server.	openbay.php
Standard	Use of Deterministic Pseudo-Random Values	The resulting values are predictable and may be enumerated by a skilled and determined attacker, although this is partly mitigated by a seed that does not appear to be time-based.	confirm.php
Standard	Use of Deterministic Pseudo-Random Values	The resulting values are predictable and may be enumerated by a skilled and determined attacker, although this is partly mitigated by a seed that does not appear to be time-based.	download.php
Standard	Use of Deterministic Pseudo-Random Values	The resulting values are predictable and may be enumerated by a skilled and determined attacker, although this is partly mitigated by a seed that does not appear to be time-based.	paymate.php
Low	Variable Used as FileName	The application appears to use a variable name in order to define a filename used by the application. It is unclear whether this variable can be controlled by the user	action.php
Low	Variable Used as FileName	The application appears to use a variable name in order to define a filename used by the application. It is unclear whether this variable can be controlled by the user	amazon_checkout.php
Low	Variable Used as FileName	The application appears to use a variable name in order to define a filename used by the application. It is unclear whether this variable can be controlled by the user	backup.php

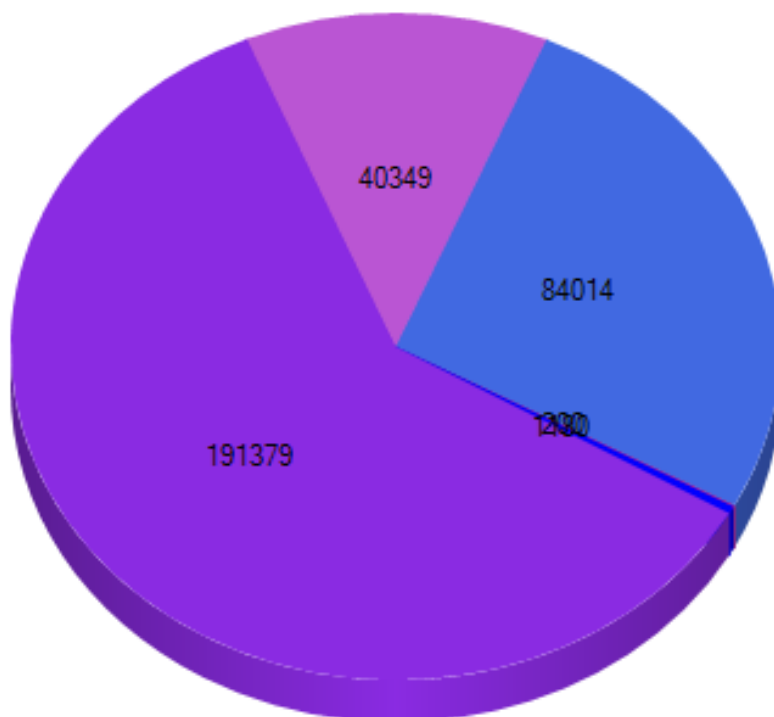
Commercial Site1 Scan Results



Summary of results

Priority	Severity	Title	Description
3	Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation
3	Medium	MD5	MD5 hashing algorithm utilised
3	Medium	Unsafe Password Management	The application appears to handle passwords in a case in-sensitive manner
3	Medium	Application Variable Used on System Command Line	The application appears to allow the use of variable within backticks
4	Standard	System	This function allows execution of commands
4	Standard	Preg_replace	This function will evaluate PHP code. It is dangerous when used with user controlled input
4	Standard	Create_funtion	This function allows execution of commands
5	Low	Variable Used as Filename	The application appears to use a variable name in order to define a filename
6	Suspicious	Comment indicates potentially unfinished code	The comment includes some wording which indicates that the developer regards as unfinished

Commercial Site2 Scan Results



- Overall code (191379 lines)
- Overall whitespace (40349 lines)
- Overall comments (84014 comments)
- Potentially broken/unfinished flags (200 Counts)
- Potentially dangerous code (1180 Counts)

Summary of results

Priority	Severity	Title	Description
3	Medium	Potential XSS	The application appears to reflect data to the screen with no apparent validation
3	Medium	MD5	MD5 hashing algorithm utilised
3	Medium	Application Variable Used on System Command Line	The application appears to allow the use of variable within backticks
3	Medium	Mt_rand	The function uses pseudo-random number generation that is not cryptographically secure
4	Standard	System	This function allows execution of commands
4	Standard	Assert	This function will evaluate PHP code. It is dangerous when used with user controlled input
4	Standard	Preg_replace	This function will evaluate PHP code. It is dangerous when used with user controlled input
4	Standard	Create_funtion	This function allows execution of commands
4	Standard	Exec	This function allows execution of commands
5	Low	Variable Used as Filename	The application appears to use a variable name in order to define a filename
6	Suspicious	Comment indicates potentially unfinished code	The comment includes some wording which indicates that the developer regards as unfinished

Appendix B - PhpmyAdmin configuration files

```
<?php
/**
 * Debian local configuration file
 * This file overrides the settings made by phpMyAdmin interactive setup
 * utility.
 * For example configuration see
 * /usr/share/doc/phpmyadmin/examples/config.sample.inc.php
 * or
 * /usr/share/doc/phpmyadmin/examples/config.manyhosts.inc.php
 * NOTE: do not add security sensitive data to this file (like passwords)
 * unless you really know what you're doing. If you do, any user that can
 * run PHP or CGI on your webserver will be able to read them. If you still
 * want to do this, make sure to properly secure the access to this file
 */

if (!function_exists('check_file_access')) {
    function check_file_access($path)
    {
        if (is_readable($path)) {
            return true;
        } else {
            error_log(
                'phpmyadmin: Failed to load ' . $path
                . ' Check group www-data has read access and open_basedir
restrictions.'
            );
            return false;
        }
    }
}

// Load secret generated on postinst
if (check_file_access('/var/lib/phpmyadmin/blowfish_secret.inc.php')) {
    require('/var/lib/phpmyadmin/blowfish_secret.inc.php');
}

// Load autoconf local config
if (check_file_access('/var/lib/phpmyadmin/config.inc.php')) {
    require('/var/lib/phpmyadmin/config.inc.php');
}

/* Server(s) configuration */
$i = 0;
// The $cfg['Servers'] array starts with $cfg['Servers'][1]. Do not use
$cfg['Servers'][0].
// You can disable a server config entry by setting host to ''.
$i++;

/*Read configuration from dbconfig-common */
if (check_file_access('/etc/phpmyadmin/config-db.php')) {
    require('/etc/phpmyadmin/config-db.php');
}

/* Configure according to dbconfig-common if enabled */
if (!empty($dbname)) {
    /* Authentication type */
    $cfg['Servers'][$i]['auth_type'] = 'cookie';
    /* Server parameters */
    if (empty($dbserver)) $dbserver = 'localhost';
}
```

```

$cfg['Servers'][$i]['host'] = $dbserver;

if (!empty($dbport) || $dbserver != 'localhost') {
    $cfg['Servers'][$i]['connect_type'] = 'tcp';
    $cfg['Servers'][$i]['port'] = $dbport;
}
//$cfg['Servers'][$i]['compress'] = false;
/* Select mysqli if your server has it */
$cfg['Servers'][$i]['extension'] = 'mysqli';
/* Optional: User for advanced features */
$cfg['Servers'][$i]['controluser'] = $dbuser;
$cfg['Servers'][$i]['controlpass'] = $dbpass;

$i++;
}

$i++;
$cfg['Servers'][$i]['connect_type'] = 'tcp';
$cfg['Servers'][$i]['extension'] = 'mysql';
$cfg['Servers'][$i]['compress'] = FALSE;
$cfg['Servers'][$i]['auth_type'] = 'config';
$cfg['Servers'][$i]['user'] = 'CDEAdmin';
$cfg['Servers'][$i]['password'] = 'Brainmc7';

/*End of servers configuration */

/*Directories for saving/loading files from server */
$cfg['UploadDir'] = '';
$cfg['SaveDir'] = '';

/* Support additional configurations */
foreach (glob('/etc/phpmyadmin/conf.d/*.php') as $filename)
{
    include($filename);
}

```

Appendix C - Ubuntu CIS Audit Configurations

Fstab - Partition Rules

```
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).

# <file system> <mount point> <type> <options> <dump> <pass>
# / was on /dev/sda2 during installation
UUID=644ac70b-6978-4f48-8d20-78784701b807 / ext4
errors=remount-ro 1

# /home was on /dev/sda6 during installation
UUID=686b06d9-df88-459f-8dfc-9d82b5961a78 /home ext4
defaults,nodev,noexec,nosuid 0 2

# /tmp was on /dev/sda7 during installation
UUID=ff8cc2a2-d6c6-4068-bd4c-04ec509adb20 /tmp ext4
defaults,nodev,noexec,nosuid 0 2

# /var was on /dev/sda8 during installation
UUID=55674176-0a77-4003-87ff-061166706425 /var ext4
defaults,nodev,noexec,nosuid 0 2

# /var/log was on /dev/sda9 during installation
UUID=9ef999e4-9723-4e84-9e81-d2436c4242be /var/log ext4 defaults
0 2

# /var/log/audit was on /dev/sda10 during installation
UUID=4fca5fb3-1939-4f69-b08b-d9e25a6bbf6d /var/log/audit ext4 defaults
0 2

# swap was on /dev/sda5 during installation
UUID=8bc7e44f-8c17-449f-9395-8477bdc5301f none swap sw
0 0
/dev/fd0 /media/floppy0 auto rw,user,noauto,exec,utf8 0 0
```

Sysctl.conf – Set Routing to Disabled per CIS Guidelines

```
# /etc/sysctl.conf - Configuration file for setting system variables

#kernel.domainname = example.com

# Uncomment the following to stop low-level messages on console
#kernel.printk = 3 4 1 3

#####3
# Functions previously found in netbase
#

# Turn on Source Address Verification in all interfaces to
# prevent some spoofing attacks
#net.ipv4.conf.default.rp_filter=1
#net.ipv4.conf.all.rp_filter=1

# See http://lwn.net/Articles/277146/
# Note: This may impact IPv6 TCP sessions too
#net.ipv4.tcp_syncookies=1

#net.ipv4.ip_forward=1

# Enabling this option disables Stateless Address Autoconfiguration
# based on Router Advertisements for this host
#net.ipv6.conf.all.forwarding=1

#####
# Additional settings - these settings can improve the network
# security of the host and prevent against some network attacks
# including spoofing attacks and man in the middle attacks through
# redirection. Some network environments, however, require that these
# settings are disabled so review and enable them as needed.
#
# Do not accept ICMP redirects (prevent MITM attacks)
#net.ipv4.conf.all.accept_redirects = 0
#net.ipv6.conf.all.accept_redirects = 0
# _or_
# Accept ICMP redirects only for gateways listed in our default
# gateway list (enabled by default)
# net.ipv4.conf.all.secure_redirects = 1
#
# Do not send ICMP redirects (we are not a router)
#net.ipv4.conf.all.send_redirects = 0
#
# Do not accept IP source route packets (we are not a router)
#net.ipv4.conf.all.accept_source_route = 0
#net.ipv6.conf.all.accept_source_route = 0
#
# Log Martian Packets
#net.ipv4.conf.all.log_martians = 1
```

Auditd.conf – Configure System for Logging

```
# This file controls the configuration of the audit daemon
#

log_file = /var/log/audit/audit.log
log_format = RAW
log_group = root
priority_boost = 4
flush = INCREMENTAL
freq = 20
num_logs = 5
disp_qos = lossy
dispatcher = /sbin/audispd
name_format = NONE
##name = mydomain
max_log_file = 6
max_log_file_action = keep_logs
space_left = 75
space_left_action = email
action_mail_acct = root
admin_space_left = 50
admin_space_left_action = HALT
disk_full_action = SUSPEND
disk_error_action = SUSPEND
##tcp_listen_port =
tcp_listen_queue = 5
tcp_max_per_addr = 1
##tcp_client_ports = 1024-65535
tcp_client_max_idle = 0
enable_krb5 = no
krb5_principal = auditd
##krb5_key_file = /etc/audit/audit.key
```

Audit .rules – Set Rules for Audit Log Files

```
# This file contains the auditctl rules that are loaded
# whenever the audit daemon is started via the initscripts.
# The rules are simply the parameters that would be passed
# to auditctl.
# First rule - delete all
-D
# Increase the buffers to survive stress events.
# Make this bigger for busy systems
-b 320
-a always,exit -F arch=b64 -S adjtimex -S settimeofday -k time-change
-a always,exit -F arch=b32 -S adjtimex -S settimeofday -S stime -k time-
change
-a always,exit -F arch=b64 -S clock_settime -k time-change
-a always,exit -F arch=b32 -S clock_settime -k time-change
-w /etc/localtime -p wa -k time-change
-w /etc/group -p wa -k identity
-w /etc/passwd -p wa -k identity
-w /etc/gshadow -p wa -k identity
-w /etc/shadow -p wa -k identity
-w /etc/security/opasswd -p wa -k identity
-a exit,always -F arch=b64 -S sethostname -S setdomainname -k system-locale
-a exit,always -F arch=b32 -S sethostname -S setdomainname -k system-locale
-w /etc/issue -p wa -k system-locale
-w /etc/issue.net -p wa -k system-locale
-w /etc/hosts -p wa -k system-locale
-w /etc/network -p wa -k system-locale
-w /etc/selinux/ -p wa -k MAC-policy
-w /var/log/faillog -p wa -k logins
-w /var/log/lastlog -p wa -k logins
-w /var/log/tallylog -p wa -k logins
-w /var/run/utmp -p wa -k session
-w /var/log/wtmp -p wa -k session
-w /var/log/btmp -p wa -k session
-a always,exit -F arch=b64 -S chmod -S fchmod -S fchmodat -F auid>=500 \
-F auid!=4294967295 -k perm_mod
-a always,exit -F arch=b32 -S chmod -S fchmod -S fchmodat -F auid>=500 \
-F auid!=4294967295 -k perm_mod
-a always,exit -F arch=b64 -S chown -S fchown -S fchownat -S lchown -F
auid>=500 \
-F auid!=4294967295 -k perm_mod
-a always,exit -F arch=b32 -S chown -S fchown -S fchownat -S lchown -F
auid>=500$
-F auid!=4294967295 -k perm_mod
-a always,exit -F arch=b64 -S setxattr -S lsetxattr -S fsetxattr -S
removexattr -S \ lremovexattr -S fremovexattr -F auid>=500 -F
auid!=4294967295 -k perm_mod
-a always,exit -F arch=b32 -S setxattr -S lsetxattr -S fsetxattr -S
removexattr$
-a always,exit -F arch=b64 -S creat -S open -S openat -S truncate -S
ftruncate \
-F exit=EACCES -F auid>=500 -F auid!=4294967295 -k access
-a always,exit -F arch=b32 -S creat -S open -S openat -S truncate -S
ftruncate \
-F exit=EACCES -F auid>=500 -F auid!=4294967295 -k access
-a always,exit -F arch=b64 -S creat -S open -S openat -S truncate -S
ftruncate \
-F exit=EPERM -F auid>=500 -F auid!=4294967295 -k access
-a always,exit -F arch=b32 -S creat -S open -S openat -S truncate -S
ftruncate \
```

```
-F exit=EPERM -F auid>=500 -F auid!=4294967295 -k access
-a always,exit -F arch=b64 -S mount -F auid>=500 -F auid!=4294967295 -k
mounts
-a always,exit -F arch=b32 -S mount -F auid>=500 -F auid!=4294967295 -k
mounts
-a always,exit -F arch=b64 -S unlink -S unlinkat -S rename -S renameat -F
auid>=500 \
-F auid!=4294967295 -k delete
-a always,exit -F arch=b32 -S unlink -S unlinkat -S rename -S renameat -F
auid>$
-F auid!=4294967295 -k delete
-w sudoers -p wa -k scope
-w /var/log/sudo.log -p wa -k actions
-w /sbin/insmod -p x -k modules
-w /sbin/rmmod -p x -k modules
-w /sbin/modprobe -p x -k modules
-a always,exit -F arch=b64 -S init_module -S delete_module -k modules
-e 2
```


Password Implementation

Login.defs - Logging Incorrect Attempts and Setting Password LifeCycle

```
# /etc/login.defs - Configuration control definitions for the login
package.
# Three items must be defined: MAIL_DIR, ENV_SUPATH, and ENV_PATH.
# Modified for Linux. --marekm

# REQUIRED for useradd/userdel/usermod
# Directory where mailboxes reside, _or_ name of file, relative to the
# home directory. If you _do_ define MAIL_DIR and MAIL_FILE,
# MAIL_DIR takes precedence.
# Essentially:
# - MAIL_DIR defines the location of users mail spool files
# (for mbox use) by appending the username to MAIL_DIR as defined
# below.
# - MAIL_FILE defines the location of the users mail spool files as the
# fully-qualified filename obtained by prepending the user home
# directory before $MAIL_FILE
MAIL_DIR /var/mail

# Enable logging and display of /var/log/faillog login failure info.
# This option conflicts with the pam_tally PAM module.
FAILLOG_ENAB yes

# Enable display of unknown usernames when login failures are recorded.
# WARNING: Unknown usernames may become world readable.
# See #290803 and #298773 for details about how this could become a
# security concern
LOG_UNKFAIL_ENAB no

# Enable logging of successful logins
LOG_OK_LOGINS no

# SYSLOG_SU_ENAB does the same for newgrp and sg.
SYSLOG_SU_ENAB yes
SYSLOG_SG_ENAB yes

# If defined, login failures will be logged here in a utmp format
# last, when invoked as lastb, will read /var/log/btmp, so...
FTMP_FILE /var/log/btmp

# If defined, the command name to display when running "su -". For
# example, if this is defined as "su" then a "ps" will display the
# command is "-su". If not defined, then "ps" would display the
# name of the shell actually being run, e.g. something like "-sh".
SU_NAME su

# If defined, file which inhibits all the usual chatter during the login
# sequence. If a full pathname, then hushed mode will be enabled if the
# user's name or shell are found in the file. If not a full pathname, then
# hushed mode will be enabled if the file exists in the user's home
# directory.
HUSHLOGIN_FILE .hushlogin

# *REQUIRED* The default PATH settings, for superuser and normal users.
# (they are minimal, add the rest in the shell startup files)
ENV_SUPATH
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

```

ENV_PATH    PATH=/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games

# Login configuration initializations:
#   ERASECHAR    Terminal ERASE character ('\010' = backspace).
#   KILLCHAR     Terminal KILL character ('\025' = CTRL/U).
#   UMASK        Default "umask" value.
ERASECHAR    0177
KILLCHAR     025
UMASK        077

# Password aging controls:
#   PASS_MAX_DAYS    Maximum number of days a password may be used.
#   PASS_MIN_DAYS    Minimum number of days allowed between password
#                   changes.
#   PASS_WARN_AGE    Number of days warning given before a password
#                   expires.
PASS_MAX_DAYS    60
PASS_MIN_DAYS    7
PASS_WARN_AGE    7

# Min/max values for automatic uid selection in useradd
UID_MIN          1000
UID_MAX          60000

# Min/max values for automatic gid selection in groupadd
GID_MIN          1000
GID_MAX          60000

# Max number of login retries if password is bad. This will most likely be
# overridden by PAM, since the default pam_unix module has it's own built
# in of 3 retries. However, this is a safe fallback in case you are using
# an authentication module that does not enforce PAM_MAXTRIES.
LOGIN_RETRIES    5

# Max time in seconds for login
LOGIN_TIMEOUT    60

# Which fields may be changed by regular users using chfn - use
# any combination of letters "frwh" (full name, room number, work
# phone, home phone).  If not defined, no changes are allowed.
# For backward compatibility, "yes" = "rwh" and "no" = "frwh".
CHFN_RESTRICT    rwh

# Should login be allowed if we can't cd to the home directory?
# Default in no.
DEFAULT_HOME     yes

# Enable setting of the umask group bits to be the same as owner bits
# (examples: 022 -> 002, 077 -> 007) for non-root users, if the uid is
# the same as gid, and username is the same as the primary group name.
# If set to yes, userdel will remove the user's group if it contains no
# more members, and useradd will create by default a group with the name
# of the user.
USERGROUPS_ENAB yes

# Note: It is recommended to use a value consistent with
# the PAM modules configuration.
#
ENCRYPT_METHOD    SHA512

```

Pam.d - Set Passwords to Required

```
# /etc/pam.d/common-password - password-related modules common to all
# services
# This file is included from other service-specific PAM config files,
# and should contain a list of modules that define the services to be
# used to change user passwords.  The default is pam_unix.

# As of pam 1.0.1-6, this file is managed by pam-auth-update by default.
# To take advantage of this, it is recommended that you configure any
# local modules either before or after the default block, and use
# pam-auth-update to manage selection of other modules.  See
# pam-auth-update(8) for details.

# here are the per-package modules (the "Primary" block)
Password [success=1 default=ignore] pam_unix.so remember=5 obscure sha512

# here's the fallback if no module succeeds
Password requisite pam_deny.so

# prime the stack with a positive return value if there isn't one already;
# this avoids us returning an error just because nothing sets a success
# code since the modules above will each just jump around
Password required pam_permit.so

# and here are more per-package modules (the "Additional" block)
Password optional pam_gnome_keyring.so

password required pam_cracklib.so retry=3 minlen=14 dcredit=-1 ucredit=-1
ocredit=-1 lcredit=-1

password sufficient pam_unix.so remember=5

# end of pam-auth-update config
```

Shadow Configuration Sets Password Attempts Limits

```
# The PAM configuration file for the Shadow `login' service
# Enforce a minimal delay in case of failure (in microseconds).
# (Replaces the `FAIL_DELAY' setting from login.defs)
# Note that other modules may require another minimal delay. (for example,
# to disable any delay, you should add the nodelay option to pam_unix)
auth optional pam_faildelay.so delay=3000000

# Outputs an issue file prior to each login prompt
auth required pam_tally2.so onerr=fail audit silent deny=5 unlock_time=900

# Disallows root logins except on tty's listed in /etc/securetty
# root will not be prompted for a password on insecure lines.
# if an invalid username is entered, a password is prompted (but login
# will eventually be rejected)
# You can change it to a "requisite" module if you think root may mis-type
# her login and should not be prompted for a password in that case. But
# this will leave the system as vulnerable to user enumeration attacks.
# You can change it to a "required" module if you think it permits to
# guess valid user names of your system (invalid user names are considered
# as possibly being root on insecure lines), but root passwords may be
# communicated over insecure lines.
auth [success=ok new_authtok_reqd=ok ignore=ignore user_unknown=bad
default=die] pam_securetty.so

# Disallows other than root logins when /etc/nologin exists
# (Replaces the `NOLOGINS_FILE' option from login.defs)
Auth requisite pam_nologin.so

# SELinux needs to be the first session rule. This ensures that any
# lingering context has been cleared. Without out this it is possible
# that a module could execute code in the wrong domain.
# When the module is present, "required" would be sufficient
session [success=ok ignore=ignore module_unknown=ignore default=bad]
pam_selinux.so close

# This module parses environment configuration file(s)
# and also allows you to use an extended config
# file /etc/security/pam_env.conf.
# parsing /etc/environment needs "readenv=1"
Session required pam_env.so readenv=1

# locale variables are also kept into /etc/default/locale in etch
# reading this file *in addition to /etc/environment* does not hurt
Session required pam_env.so readenv=1 envfile=/etc/default/locale

# Standard Un*x authentication.
@include common-auth

# This allows certain extra groups to be granted to a user
# based on things like time of day, tty, service, and user.
auth optional pam_group.so

# Uncomment and edit /etc/security/time.conf if you need to set
# time restraint on logins.
# account requisite pam_time.so

# Sets up user limits according to /etc/security/limits.conf
# (Replaces the use of /etc/limits in old login)
```

```
Session required pam_limits.so
auth required pam_tally2.so onerr=fail audit silent deny=5 unlock_time=900

# Prints the last login info upon succesful login
# (Replaces the `LASTLOG_ENAB' option from login.defs)
session optional pam_lastlog.so

# Prints the message of the day upon succesful login.
# (Replaces the `MOTD_FILE' option in login.defs)
# This includes a dynamically generated part from /run/motd.dynamic
# and a static (admin-editable) part from /etc/motd.
Session optional pam_motd.so motd=/run/motd.dynamic nouupdate
Session optional pam_motd.so

# Prints the status of the user's mailbox upon succesful login
# This also defines the MAIL environment variable
# However, userdel also needs MAIL_DIR and MAIL_FILE variables
# in /etc/login.defs to make sure that removing a user
# also removes the user's mail spool file.
Session optional pam_mail.so standard

# Standard Un*x account and session
@include common-account
@include common-session
@include common-password

# SELinux needs to intervene at login time to ensure that the process
# starts in the proper default security context. Only sessions which are
# intended to run in the user's context should be run after this.
session [success=ok ignore=ignore module_unknown=ignore default=bad]
pam_selinux.so open
```

Appendix D - Apache CIS Audit Configuration

000-default.conf - Local Host Sites Configuration File

```
<VirtualHost *:80>
    # The ServerName directive sets the request scheme, hostname and port
    # that the server uses to identify itself.
    # This is used when creating redirection URLs. In the context of
    # virtual hosts, the ServerName specifies what hostname must appear
    # in the request's Host: header to match this virtual host.
    # For the default virtual host (this file) this value is not
    # decisive as it is used as a last resort host regardless.
    # However, you must set it for any further virtual host explicitly.
    #ServerName www.example.com

    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/
    ServerName localhost
    Redirect / https://localhost/

    <Directory /var/www/>
        Order allow,deny
        Allow from all
        AllowOverride None
        Options None
        TraceEnable Off
        <LimitExcept GET POST OPTIONS>
            deny from all
        </LimitExcept>
    </Directory>

    # Available loglevels: trace8,trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    LogLevel notice

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host.

</VirtualHost>
```

Default-ssl.conf - HTTPS Local Host Configuration File

```
<IfModule mod_ssl.c>
  <VirtualHost _default_:443>
    ServerAdmin webmaster@localhost

    DocumentRoot /var/www/
    LogLevel notice
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # SSL Engine Switch:
    # Enable/Disable SSL for this virtual host.
    SSLEngine on

    # SSLCertificateFile directive is needed.
    SSLCertificateFile      /etc/apache2/ssl/apache.pem

    # Certificate Authority (CA):
    # Set the CA certificate verification path where to find CA
    # certificates for client authentication or alternatively one
    # huge file containing all of them (file must be PEM encoded)
    #SSLCACertificatePath /etc/ssl/certs/
    #SSLCACertificateFile /etc/apache2/ssl.crt/ca-bundle.crt

    # Certificate Revocation Lists (CRL):
    # Set the CA revocation path where to find CA CRLs for client
    # authentication or alternatively one huge file containing
    # all of them (file must be PEM encoded)
    # Note: Inside SSLCARevocationPath you need hash symlinks
    # to point to the certificate files.
    # SSLCARevocationPath /etc/apache2/ssl.crl/
    # SSLCARevocationFile /etc/apache2/ssl.crl/ca-bundle.crl

    # Client Authentication (Type):
    # Client certificate verification type and depth. Types are
    # none, optional, require and optional_no_ca. Depth is a
    # number which specifies how deeply to verify the certificate
    # issuer chain before deciding the certificate is not valid.
    #SSLVerifyClient require
    #SSLVerifyDepth 10

    # SSL Engine Options:
    #SSLOptions +FakeBasicAuth +ExportCertData +StrictRequire
    <FilesMatch "\.(cgi|shtml|phtml|php)$">
        SSLOptions +StdEnvVars
    </FilesMatch>
    <Directory /usr/lib/cgi-bin>
        SSLOptions +StdEnvVars
    </Directory>

    # SSL Protocol Adjustments:

    BrowserMatch "MSIE [2-6]" \
        nokeepalive ssl-unclean-shutdown \
        downgrade-1.0 force-response-1.0
    # MSIE 7 and newer should be able to use keepalive
    BrowserMatch "MSIE [17-9]" ssl-unclean-shutdown

  </VirtualHost>
</IfModule>
```

Appendix E – MySQL Security Protocols

My.Conf enforcing TLS certificates to communicate with WebServer

```
# The MySQL database server configuration file.
# This will be passed to all mysql clients
# Remember to edit /etc/mysql/debian.cnf when changing the socket location.
[client]
port                = 3306
socket              = /var/run/mysqld/mysqld.sock

# Here is entries for some specific programs
# The following values assume you have at least 32M ram
# This was formally known as [safe_mysqld]. Both versions are currently
# parsed.
[mysqld_safe]
socket              = /var/run/mysqld/mysqld.sock
nice                = 0

[mysqld]

# * Basic Settings
user                = mysql
pid-file            = /var/run/mysqld/mysqld.pid
socket              = /var/run/mysqld/mysqld.sock
port                = 3306
basedir             = /usr
datadir             = /var/lib/mysql
tmpdir              = /tmp
lc-messages-dir    = /usr/share/mysql
skip-external-locking

# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address        = 127.0.0.1

# * Fine Tuning
key_buffer          = 16M
max_allowed_packet = 16M
thread_stack        = 192K
thread_cache_size   = 8

# This replaces the startup script and checks MyISAM tables if needed
# the first time they are touched
myisam-recover      = BACKUP
#max_connections    = 100
#table_cache        = 64
#thread_concurrency = 10

# * Query Cache Configuration
query_cache_limit   = 1M
query_cache_size    = 16M

# * Logging and Replication
# Both location gets rotated by the cronjob.
general_log_file    = /var/log/mysql/mysql.log
general_log         = 1

# Error log - should be very few entries.
log_error           = /var/log/mysql/error.log
```



```
expire_logs_days = 10
max_binlog_size = 100M

# * Security Features
ssl-cipher=DHE-RSA-AES256-SHA
ssl-ca=/etc/mysql/ca-cert.pem
ssl-cert=/etc/mysql/server-cert.pem
ssl-key=/etc/mysql/server-key.pem

[mysqldump]
quick
quote-names
max_allowed_packet = 16M

[isamchk]
key_buffer = 16M

# * IMPORTANT: Additional settings that can override those from this file!
# The files must end with '.cnf', otherwise they'll be ignored.

!includedir /etc/mysql/conf.d/
```

Appendix E - ClearOS Configurations

IP Tables

```
#Default drop rule
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT
# Flush (-F) all specific rules
iptables -F INPUT
iptables -F FORWARD
iptables -F OUTPUT
iptables -F -t nat
#Forward internal network traffic
iptables -A FORWARD -i eth1 -o eth0 -j ACCEPT
iptables -A FORWARD -i eth2 -o eth0 -j ACCEPT
#Allow and forward only established connections inbound
iptables -A FORWARD -i eth0 -o eth1 -m state --state ESTABLISHED,RELATED -j
ACCEPT
iptables -A FORWARD -i eth0 -o eth2 -m state --state ESTABLISHED,RELATED -j
ACCEPT
iptables -A INPUT -i eth0 -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A INPUT -i eth1 -s 0/0 -d 0/0 -j ACCEPT
iptables -A INPUT -i eth2 -s 0/0 -d 0/0 -j ACCEPT
iptables -A INPUT -i lo -s 0/0 -d 0/0 -j ACCEPT
iptables -A POSTROUTING -t nat -o eth0 -j MASQUERADE
#Disallow traffic from external network with spoofed internal ip
iptables -A INPUT -i eth0 -s 192.168.10.0/29-j DROP
iptables -A INPUT -i eth0 -s 192.168.30.0/30 -j DROP
iptables -A INPUT -i eth0 -s 192.168.20.0/27 -j DROP
iptables -A INPUT -i eth0 -s 127.0.0.0/8 -j DROP
```

Intrusion Detection System

```
# Stop generic decode events:
config disable_decode_alerts

# Stop Alerts on experimental TCP options
config disable_tcpopt_experimental_alerts

# Stop Alerts on obsolete TCP options
config disable_tcpopt_obsolete_alerts

# Stop Alerts on T/TCP alerts
config disable_tcpopt_ttcp_alerts

# Stop Alerts on all other TCPOption type events:
config disable_tcpopt_alerts

# Stop Alerts on invalid ip options
config disable_ipopt_alerts

# Alert if value in length field (IP, TCP, UDP) is greater th e length of
the packet

#config enable_decode_oversized_alerts

# Configure IP / TCP checksum mode
config checksum_mode: all

# Configure the base detection engine. For more information, see
README.decode

# Configure PCRE match limitations
config pcre_match_limit: 3500
config pcre_match_limit_recursion: 1500

# Configure the detection engine See the Snort Manual, Configuring Snort -
Includes - Config
config detection: search-method ac-split search-optimize max-pattern-len 20

# Configure the event queue. For more information, see README.event_queue
config event_queue: max_queue 8 log 5 order_events content_length

# Configure protocol aware flushing
config paf_max: 16000

# Configure dynamic loaded libraries.
# path to dynamic preprocessor libraries
dynamicpreprocessor directory /usr/lib/snort_dynamicpreprocessor/

# Target-based IP defragmentation. For more inforation, see README.frag3
```

```

preprocessor frag3_global: max_frags 65536

preprocessor frag3_engine: policy windows detect_anomalies overlap_limit 10
min_fragment_length 100 timeout 180

# Target-Based stateful inspection/stream reassembly. For more information,
see README.stream5

preprocessor stream5_global: track_tcp yes, \
    track_udp yes, \
    track_icmp no, \
    max_tcp 262144, \
    max_udp 131072, \
    max_active_responses 2, \
    min_response_seconds 5

# path to base preprocessor engine
dynamicengine /usr/lib/snort_dynamicengine/libsf_engine.so

# performance statistics. For more information, see the Snort Manual,
Configuring Snort - Preprocessors - Performance Monitor

# preprocessor perfmonitor: time 300 file /var/snort/snort.stats pktcnt
10000

# HTTP normalization and anomaly detection. For more information, see
README.http_inspect

preprocessor http_inspect: global iis_unicode_map unicode.map 1252
compress_depth 65535 decompress_depth 65535

preprocessor http_inspect_server: server default \

http_methods { GET POST PUT SEARCH MKCOL COPY MOVE LOCK UNLOCK NOTIFY POLL
BCOPY BDELETE BMOVE LINK UNLINK OPTIONS HEAD DELETE TRACE TRACK CONNECT
SOURCE SUBSCRIBE UNSUBSCRIBE PROPFIND PROPPATCH BPROPFIND BPROPPATCH
RPC_CONNECT PROXY_SUCCESS BITS_POST CCM_POST SMS_POST RPC_IN_DATA
RPC_OUT_DATA RPC_ECHO_DATA } \

    chunk_length 500000 \
    server_flow_depth 0 \
    client_flow_depth 0 \
    post_depth 65495 \
    oversize_dir_length 500 \
    max_header_length 750 \
    max_headers 100 \
    max_spaces 200 \
    small_chunk_length { 10 5 } \

```

```

ports { 36 80 81 82 83 84 85 86 87 88 89 90 311 383 555 591 593 631 801 808
818 901 972 1158 1220 1414 1533 1741 1830 1942 2231 2301 2381 2809 2980
3029 3037 3057 3128 3443 3702 4000 4343 4848 5000 5117 5250 5600 6080 6173
6988 7000 7001 7071 7144 7145 7510 7770 7777 7778 7779 8000 8008 8014 8028
8080 8081 8082 8085 8088 8090 8118 8123 8180 8181 8222 8243 8280 8300 8333
8344 8500 8509 8800 8888 8899 8983 9000 9060 9080 9090 9091 9111 9290 9443
9999 10000 11371 12601 13014 15489 29991 33300 34412 34443 34444 41080
44449 50000 50002 51423 53331 55252 55555 56712 } \

    non_rfc_char { 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 } \

    enable_cookie \

    extended_response_inspection \

    inspect_gzip \

    normalize_utf \

    unlimited_decompress \

    normalize_javascript \

    apache_whitespace no \

    ascii no \

    bare_byte no \

    directory no \

    double_decode no \

# Back Orifice detection.

preprocessor bo

Portscan detection.  For more information, see README.sfportscan

preprocessor sfportscan: proto { all } memcap { 10000000 } sense_level {
low }

# ARP spoof detection.  For more information, see the Snort Manual -
Configuring Snort - Preprocessors - ARP Spoof Preprocessor

preprocessor arpspoof

preprocessor arpspoof_detect_host: 192.168.40.1 f0:0f:00:f0:0f:00

#Configure output plugins

# syslog

output alert_syslog: LOG_AUTHPRIV LOG_ALERT

#Customize Rules

include $RULE_PATH/gpl/attack_response.rules

include $RULE_PATH/gpl/chat.rules

include $RULE_PATH/gpl/dns.rules

include $RULE_PATH/gpl/exploit.rules

```

```
include $RULE_PATH/gpl/ftp.rules
include $RULE_PATH/gpl/imap.rules
include $RULE_PATH/gpl/misc.rules
include $RULE_PATH/gpl/p2p.rules
include $RULE_PATH/gpl/pop3.rules
include $RULE_PATH/gpl/rpc.rules
include $RULE_PATH/gpl/scan.rules
include $RULE_PATH/gpl/shellcode.rules
include $RULE_PATH/gpl/smtp.rules
include $RULE_PATH/gpl/snmp.rules
include $RULE_PATH/gpl/sql.rules
include $RULE_PATH/gpl/tftp.rules
include $RULE_PATH/gpl/web_client.rules
include $RULE_PATH/gpl/web_server.rules
include $RULE_PATH/gpl/web_specific_apps.rules
```

Appendix F – SmoothWall Iptables

```
#Default drop rule

iptables -P INPUT DROP

iptables -P FORWARD DROP

iptables -P OUTPUT ACCEPT

# Flush (-F) all specific rules

iptables -F INPUT

iptables -F FORWARD

iptables -F OUTPUT

iptables -F -t nat

#Allow and forward only established connections inbound

iptables -A FORWARD -i eth0 -o eth1 -m state --state ESTABLISHED,RELATED -j
ACCEPT

iptables -A INPUT -i eth0 -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -A INPUT -i eth1 -s 0/0 -d 0/0 -j ACCEPT

iptables -A INPUT -i lo -s 0/0 -d 0/0 -j ACCEPT

iptables -A POSTROUTING -t nat -o eth0 -j MASQUERADE
```

Appendix G - Metasploit Scans

Ubuntu Server Metasploit Results

Exploit	Result
exploit hagent_untrusted_hsdata	[-] No reply from the target, this is not a vulnerable system
exploit cuteflow_fileupload	[*] 192.168.20.5:80 - The target appears to be vulnerable.
Checking unix/webapp/arkeia_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/awstats_configdir_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/awstatstotals_multisort	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/barracuda_img_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/base_qry_common	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/basilic_diff_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/carberp_backdoor_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/coppermine_piceditor	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/datalife_preview_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/egallery_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/flashchat_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/get_simple_cms_upload_exec	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking unix/webapp/google_proxystylesheet_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/graphite_pickle_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/hastymail_exec	[-] Check failed: The following options failed to validate: USER, PASS.
Checking unix/webapp/havalite_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/horde_unserialize_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/instantcms_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/invision_pboard_unserialize_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/libretto_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/mambo_cache_lite	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/mitel_awc_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/nagios_graph_explorer	[-] Check failed: The following options failed to validate: PASSWORD.
Checking unix/webapp/narcissus_backend_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/open_flash_chart_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/openemr_sqli_privesc_upload	[-] Check failed: The following options failed to validate: USER, PASS.
Checking unix/webapp/openemr_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/opensis_modname_exec	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking unix/webapp/openx_banner_edit	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking unix/webapp/oscommerce_filemanager	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/php_charts_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/php_vbulletin_template	[*] 192.168.20.5:80 - The target is not exploitable.

Checking unix/webapp/php_wordpress_infusionsoft	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/php_wordpress_lastpost	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/php_xmlrpc_eval	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/phpmyadmin_config	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/projectpier_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/projectsend_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/redmine_scm_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/simple_e_document_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/skybluecanvas_exec	[*] 192.168.20.5:80 - The target appears to be vulnerable.
Checking unix/webapp/sphpblog_file_upload	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/squash_yaml_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/sugarcrm_unserialize_exec	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking unix/webapp/tuleap_unserialize_exec	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking unix/webapp/vbulletin_vote_sqli_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/vicialial_manager_send_cmd_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/webtester_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/wp_admin_shell_upload	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking unix/webapp/wp_advanced_custom_fields_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/wp_downloadmanager_upload	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/wp_platform_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/wp_wptouch_file_upload	[-] Check failed: The following options failed to validate: PASSWORD, USER.
Checking unix/webapp/xoda_file_upload	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/webapp/zeroshell_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/activecollab_chat	[-] Check failed: The following options failed to validate: USER, PASS.
Checking multi/http/ajaxplorer_checkinstall_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/apprain_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/auxilium_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking linux/http/centreon_sqli_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/cisco_dcnm_upload	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/coldfusion_rds	[*] 192.168.20.5:80 - The target is not exploitable.
Checking unix/http/contentkeeperweb_mimecode	[*] 192.168.20.5:80 - The target is not exploitable.
Checking linux/http/dolibarr_cmd_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking linux/http/dreambox_openpli_shell	[*] 192.168.20.5:80 - The target is not exploitable.
Checking linux/http/esva_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/extplorer_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking linux/http/fritzbox_echo_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/glossword_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/horde_href_backdoor	[*] 192.168.20.5:80 - The target is not exploitable.

Checking multi/http/kordil_edms_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking linux/http/linksys_wrt110_cmd_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/log1cms_ajax_create_folder	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/mobilecartly_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/mutiny_subnetmask_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/openx_backdoor_php	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/opmanager_socialit_file_upload	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/oracle_reports_rce	[*] 192.168.20.5:80 - The target is not exploitable.
Checking linux/http/pandora_fms_sqli	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/pandora_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/php_cgi_arg_injection	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/phpldapadmin_query_engine	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/phpmoadmin_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/phpscheduleit_start_date	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/phptax_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/polarcms_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/qdpm_upload_exec	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking linux/http/railo_cfml_rfi	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/sflog_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/sit_file_upload	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking linux/http/sophos_wpa_iface_exec	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking multi/http/stunshell_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/sun_jsws_dav_options	[*] 192.168.20.5:80 - The target is not exploitable.
Checking linux/http/symantec_web_gateway_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking linux/http/symantec_web_gateway_file_upload	[*] 192.168.20.5:80 - The target is not exploitable.
Checking linux/http/symantec_web_gateway_lfi	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/testlink_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/traq_plugin_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/v0pcr3w_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/vbseo_proc_deutf	[*] 192.168.20.5:80 - The target is not exploitable.
Checking linux/http/vcms_upload	[*] 192.168.20.5:80 - The target is not exploitable.
Checking linux/http/wanem_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking linux/http/webid_converter	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/webpagetest_upload_exec	[*] 192.168.20.5:80 - The target is not exploitable.
Checking multi/http/wikka_spam_exec	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD, PAGE.
Checking multi/http/x7chat2_php_exec	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking multi/http/zabbix_script_exec	[*] 192.168.20.5:80 - The target is not exploitable.

ClearOS Metasploit Results

Exploit	Result
f5_bigip_known_privkey	[-] 192.168.30.17:22 SSH - Failed authentication
loadbalancerorg_enterprise_known_privkey	[-] 192.168.30.17:22 SSH - Failed authentication
array_vxag_vapv_privkey_privesc	[-] Exploit aborted due to failure: no-access: 192.168.30.17:22 SSH - Failed authentication
array_vxag_vapv_privkey_privesc	[-] Exploit aborted due to failure: no-access: 192.168.30.17:22 SSH - Failed authentication
quantum_dxi_known_privkey	[-] 192.168.30.17:22 SSH - Failed authentication
quantum_vmpro_backdoor	[-] 192.168.30.17:22 SSH - Failed authentication
msf exploit(symantec_smg_ssh) > exploit -j	[-] 192.168.30.17:22 SSH - Failed authentication
tectia_passwd_changereq	[*] 192.168.30.17:22 - Auths that can continue: 51
ssh_login	[*] Auxiliary module execution completed

SmoothWall Metasploit Results

Exploit	Result
Checking multi/http/activecollab_chat	[-] Check failed: The following options failed to validate: USER, PASS.
Checking multi/http/ajaxplorer_checkinstall_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/apprain_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/auxilium_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/cisco_dcnm_upload	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/coldfusion_rds	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/http/contentkeeperweb_mimencode	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/cuteflow_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/dexter_casinolader_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/explorer_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/familycms_less_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/freenas_exec_raw	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/gitlab_shell_exec	[-] Exploit aborted due to failure: no-access: 192.168.20.9:80 - Login failed
Checking multi/http/gitorious_graph	[*] The server returned: 301 Moved Permanently
Checking multi/http/glossword_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/ispconfig_php_exec	[-] Exploit aborted due to failure: Error getting initial page.: No reason given
Checking multi/http/kordil_edms_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/http/lifesize_room	[-] Exploit aborted due to failure: not-found: Could not obtain a Session ID
Checking multi/http/log1cms_ajax_create_folder	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/mobilecartly_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/mutiny_subnetmask_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/nas4free_php_exec	[-] Exploit aborted due to failure: Login failed
Checking multi/http/op5_license	[*] 192.168.20.9:443 - The target is not exploitable.

Checking multi/http/openx_backdoor_php	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/opmanager_socialit_file_upload	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/oracle_reports_rce	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/pandora_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/php_cgi_arg_injection	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/php_volunteer_upload_exec	[-] 192.168.20.9:80 - Login failed with "admin:volunteer"
Checking multi/http/phpmoadmin_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/phpscheduleit_start_date	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/phptax_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/pmwiki_pagelist	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/polarcms_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/processmaker_exec	[-] 192.168.20.9:80 - Authenticating as user 'admin' failed
Checking multi/http/qdpm_upload_exec	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking multi/http/rails_secret_deserialization	[-] Check failed: The following options failed to validate: SECRET.
Checking multi/http/sflog_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/sit_file_upload	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking multi/http/stunshell_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/sun_jsws_dav_options	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/testlink_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/traq_plugin_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/http/twiki_debug_plugins	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/v0pcr3w_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/vbseo_proc_deutf	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/http/vmturbo_vmtadmin_exec_noauth	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/vtiger_php_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/webpagetest_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking multi/http/wikka_spam_exec	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD, PAGE.
Checking multi/http/x7chat2_php_exec	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking unix/webapp/actualanalyzer_ant_cookie_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/awstats_migrate_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/barracuda_img_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/basilic_diff_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/carberp_backdoor_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/citrix_access_gateway_exec	[*] 192.168.20.9:443 - The target is not exploitable.
Checking unix/webapp/coppermine_piceditor	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/datalife_preview_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/dogfood_spell_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/egallery_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/freepbx_config_exec	[*] 192.168.20.9:80 - The target is not exploitable.

Checking unix/webapp/get_simple_cms_upload_exec	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking unix/webapp/google_proxystylesheet_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/graphite_pickle_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/hastymail_exec	[-] Check failed: The following options failed to validate: USER, PASS.
Checking unix/webapp/havalite_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/horde_unserialize_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/instantcms_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/invision_pboard_unserialize_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/libretto_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/mybb_backdoor	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/narcissus_backend_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/open_flash_chart_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/openemr_sql_privesc_upload	[-] Check failed: The following options failed to validate: USER, PASS.
Checking unix/webapp/opensis_modname_exec	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking unix/webapp/openx_banner_edit	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking unix/webapp/php_charts_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/php_vbulletin_template	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/php_xmlrpc_eval	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/phpbb_highlight	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/projectpier_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/redmine_scm_exec	[*] The server returned: 301 Moved Permanently
Checking unix/webapp/seportal_sql_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/simple_e_document_upload_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/skybluecanvas_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/sphpblog_file_upload	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/squash_yaml_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/sugarcrm_unserialize_exec	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking unix/webapp/tikiwiki_graph_formula_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/tikiwiki_jhot_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/tuleap_unserialize_exec	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking unix/webapp/vbulletin_vote_sql_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/vicialial_manager_send_cmd_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/webtester_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/wp_admin_shell_upload	[-] Check failed: The following options failed to validate: USERNAME, PASSWORD.
Checking unix/webapp/wp_platform_exec	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/xoda_file_upload	[*] 192.168.20.9:80 - The target is not exploitable.
Checking unix/webapp/zeroshell_exec	[*] 192.168.20.9:80 - The target is not exploitable.

Appendix H – Nikto Vulnerability Scan

- Nikto v2.1.4

```
-----
+ Target IP:      127.0.0.1
+ Target Hostname: localhost
+ Target Port:    80
+ Start Time:    2015-04-26 21:11:04
-----
+ Server: Apache/2.4.7 (Ubuntu)
+ Root page / redirects to: https://localhost/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ OSVDB-27071: /phpimageview.php?pic=javascript:alert(8754): PHP Image View 1.0 is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ OSVDB-3931: /myphpnuke/links.php?op=search&query=[script>alert('Vulnerable');[/script]?query=: myphpnuke is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ OSVDB-3931: /myphpnuke/links.php?op=MostPopular&ratenum=[script>alert(document.cookie);[/script]&ratetype=percent : myphpnuke is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+
+ /modules.php?op=modload&name=FAQ&file=index&myfaq=yes&id_cat=1&categories=%3Cimg%20src=javascript:alert(9456);%3E&parent_id=0: Post Nuke 0.7.2.3-Phoenix is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+
+ /modules.php?letter=%22%3E%3Cimg%20src=javascript:alert(document.cookie);%3E&op=modload&name=Members_List&file=index: Post Nuke 0.7.2.3-Phoenix is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ OSVDB-4598: /members.asp?SF=%22;}alert('Vulnerable');function%20x(){v%20=%22: Web Wiz Forums ver. 7.01 and below is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ OSVDB-2946: /forum_members.asp?find=%22;}alert(9823);function%20x(){v%20=%22: Web Wiz Forums ver. 7.01 and below is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ 6448 items checked: 0 error(s) and 7 item(s) reported on remote host
+ End Time:      2015-04-26 21:11:22 (18 seconds)
-----
+ 1 host(s) tested
```

Appendix I – Table of Requirements

Build and Maintain a Secure Network and Systems

Install and maintain a firewall configuration to protect cardholder data

Implementation and configuration ClearOS and Smoothwall Router/Firewalls at integral network points.

Network Diagram from planning section.

Firewall configurations ensure no direct access to CDE from outside networks and ensures all traffic to CDE is only allowed on established connections.

Do not use vendor-supplied defaults for system passwords and other security parameters

Removal of default usernames and passwords on OS installation.

Removal of default usernames and passwords for databases and wireless connections.

Implementation of CIS hardening standards.

Network protocols are limited to only those required for functionality.

Protect Cardholder Data

Protect stored cardholder data

Salting and Hashing of sensitive information stored in database.

Three digit verification number not stored.

Implementation of strong cryptographic keys (RSA). Keys are also given limited lifespan and rendered useless when expired.

Encrypt transmission of cardholder data across open, public networks

Implementing TLS to secure communications.

Wireless connections secured via WPA2 wireless connections, encrypted with AES.

Maintain a Vulnerability Management Program

Protect all systems against malware and regularly update anti-virus software or programs

Installation of anti-virus and anti-malware software on network.

Develop and maintain secure systems and application

Static code analysis conducted to evaluate weaknesses in the code base.

Code base strengthened by implementing necessary secure coding measures as required by the scan results.

Implement Strong Access Control Measures

Restrict access to cardholder data by business need to know

Cardholder can only be accessed by two administrative users.

These users both have different functionality and do not have the same access levels, so only one user can complete one task either server maintenance or data removal as required.

Base access control rules implemented to deny all.

Identify and authenticate access to system components

User groups implemented with access levels as required for job functionality

Users automatically assigned a unique ID.

Accounts disabled after 3 month period if not accessed.

Accounts locked out after 5 unsuccessful log in attempts.

Password requirements set to include strength and lifecycle limitations.

Restrict physical access to cardholder data

Could not be implemented due to the nature of the testing environment.

Regularly Monitor and Test Networks

Track and monitor all access to network resources and cardholder data

Logging implemented on all network devices to track access.

Regularly test security systems and processes

Completion of:

Network Communications Testing

Vulnerability Analysis

Wireless Testing

Penetration Testing

Maintain an Information Security Policy

Maintain a policy that addresses information security for all personnel

Authentication required for security personnel.

All devices on the network listed in network diagram.

Wireless access points also listed in network diagram.