

2016

## Penetration Testing Frameworks and methodologies: A comparison and evaluation

Aleatha Shanley  
*Edith Cowan University*

Follow this and additional works at: [https://ro.ecu.edu.au/theses\\_hons](https://ro.ecu.edu.au/theses_hons)



Part of the [Information Security Commons](#)

Shanley, A., & Johnstone, M. N. (2015). Selection of penetration testing methodologies: A comparison and evaluation. 13th Australian Information Security Management Conference, held from the 30 November – 2 December, 2015 (pp. 65-72), Edith Cowan University Joondalup Campus, Perth, Western Australia. <https://ro.ecu.edu.au/ism/182/>

---

### Recommended Citation

Shanley, A. (2016). *Penetration Testing Frameworks and methodologies: A comparison and evaluation*. [https://ro.ecu.edu.au/theses\\_hons/1553](https://ro.ecu.edu.au/theses_hons/1553)

This Thesis is posted at Research Online.  
[https://ro.ecu.edu.au/theses\\_hons/1553](https://ro.ecu.edu.au/theses_hons/1553)

# Edith Cowan University

## Copyright Warning

You may print or download ONE copy of this document for the purpose of your own research or study.

The University does not authorize you to copy, communicate or otherwise make available electronically to any other person any copyright material contained on this site.

You are reminded of the following:

- Copyright owners are entitled to take legal action against persons who infringe their copyright.
- A reproduction of material that is protected by copyright may be a copyright infringement. Where the reproduction of such material is done without attribution of authorship, with false attribution of authorship or the authorship is treated in a derogatory manner, this may be a breach of the author's moral rights contained in Part IX of the Copyright Act 1968 (Cth).
- Courts have the power to impose a wide range of civil and criminal sanctions for infringement of copyright, infringement of moral rights and other offences under the Copyright Act 1968 (Cth). Higher penalties may apply, and higher damages may be awarded, for offences and infringements involving the conversion of material into digital or electronic form.

## USE OF THESIS

The Use of Thesis statement is not included in this version of the thesis.

# Penetration Testing Frameworks and Methodologies: A comparison and evaluation

A Thesis for a dissertation in partial fulfilment of the requirements for the degree of

Bachelor of Science (Security) Honours

Aleatha Shanley

School of Computer and Security Science  
Edith Cowan University

Supervisor: Dr. Mike Johnstone

Date of Submission: Semester 2: 2015

## COPYRIGHT AND ACCESS DECLARATION

I certify that this thesis does not, to the best of my knowledge and belief:

- I. Incorporate without acknowledgement any material previously submitted for a degree or diploma in any institution of higher education;
- II. Contain any material previously published or written by another person except where due reference is made in the text; or
- III. Contain any defamatory material

Signed ..



Date

4/5/16.....

## **Abstract**

*Cyber security is fast becoming a strategic priority across both governments and private organisations. With technology abundantly available, and the unbridled growth in the size and complexity of information systems, cyber criminals have a multitude of targets. Therefore, cyber security assessments are becoming common practice as concerns about information security grow. Penetration testing is one strategy used to mitigate the risk of cyber-attack. Penetration testers attempt to compromise systems using the same tools and techniques as malicious attackers thus, aim to identify vulnerabilities before an attack occurs. Penetration testing can be complex depending on the scope and domain area under investigation, for this reason it is often managed similarly to that of a project necessitating the implementation of some framework or methodology. Fortunately, there are an array of penetration testing methodologies and frameworks available to facilitate such projects, however, determining what is a framework and what is methodology within this context can lend itself to uncertainty. Furthermore, little exists in relation to mature frameworks whereby quality can be measured. This research defines the concept of “methodology” and “framework” within a penetration testing context. In addition, the research presents a gap analysis of the theoretical vs. the practical classification of nine penetration testing frameworks and/or methodologies and subsequently selects two frameworks to undergo quality evaluation using a real-world case study. Quality characteristics were derived from a review of four quality models, thus building the foundation for a proposed penetration testing quality model. The penetration testing quality model is a modified version of an ISO quality model whereby the two chosen frameworks underwent quality evaluation.*

*Defining methodologies and frameworks for the purposes of penetration testing was achieved. A suitable definition was formed by way of analysing properties of each category respectively, thus a Framework vs. Methodology Characteristics matrix is presented. Extending upon the nomenclature resolution, a gap analysis was performed to determine if a framework is actually a framework, i.e., it has a sound underlying ontology. In contrast, many “frameworks” appear to be simply collections of tools or techniques. In addition, two frameworks OWASP’s Testing Guide and Information System Security Assessment Framework (ISSAF), were employed to perform penetration tests based on a real-world case study to facilitate quality evaluation based on a proposed quality model. The research suggests there are various ways in which quality for penetration testing frameworks can be measured; therefore concluded that quality evaluation is possible.*

## Table of Contents

Abstract.....	5
List of Figures.....	9
List of Tables .....	9
Chapter 1 : Introduction.....	10
1.1 Background:.....	10
1.2 Significance: .....	13
1.3 Purpose: .....	14
1.4 Research questions: .....	14
1.5 Definition of Common Terms:.....	14
1.6 Thesis Structure .....	17
1.7 Summary.....	18
Chapter 2 : Review of the Literature .....	19
2.1 Existing evaluations of Frameworks and Methodologies.....	19
2.2 Defining Methodologies and Frameworks for the purpose of pen testing .....	20
2.3 Review of Nine Penetration Testing Methodologies and Frameworks .....	21
2.3.1 Information System Security Assessment Framework (ISSAF) .....	21
2.3.2 Open Source Security Testing Methodology Manual (OSSTMM) .....	22
2.3.3 NIST 800-115 .....	22
2.3.4 Open Web Application Security Project (OWASP).....	23
2.3.5 Building Security in Maturity Model (BSIMM) .....	24
2.3.6 Penetration Testing Framework 0.59 (PTF) .....	25
2.3.7 Penetration Testing Execution Standard (PTES).....	25
2.3.8 Metasploit Framework (MSF) .....	26
2.3.9 Browser Exploitation Framework (BeEF).....	27
2.4 Quality Models .....	28
2.4.1 McCall model .....	28
2.4.2 Boehm Model .....	28

2.4.3 Dromey Model.....	29
2.4.4 ISO 9126.....	29
2.4.5 ISO/IEC 25010:2013 .....	30
2.4.6 Summary.....	30
2.5 Conclusion .....	31
Chapter 3 : Research Methods and Design:.....	33
3.1 Research Approach: .....	33
3.1.1 Object of Interest .....	34
3.2 Research Method: .....	34
3.3 Research Design .....	35
3.3.1 Case Study Description.....	36
3.4 Materials: .....	38
3.5 Data Analysis: .....	38
3.6 Limitations:.....	39
3.7 Summary.....	40
Chapter 4 : Analysis and Discussion .....	41
4.1 Nomenclature Resolution .....	42
4.1.1 Defining Methodologies and Frameworks for the purpose of pen testing .....	42
4.1.2 Nomenclature Definition .....	43
4.2 Gap Analysis.....	44
4.2.1 Theoretical Review .....	45
4.2.2 Gap Analysis Matrix .....	50
4.2.3 Candidate Selection .....	50
4.3 Quality Metrics and Evaluation.....	51
4.3.1 Quality Metrics Defined .....	53
4.3.2 Quality Model.....	55
4.4 Framework Quality Evaluation.....	56
4.4.1 Measuring Extensibility.....	57



4.4.2 Measuring Maintainability.....	58
4.4.3 Measuring Domain Coverage.....	61
4.4.4 Measuring Usability.....	63
4.4.5 Measuring Availability.....	69
4.4.6 Measuring Reliability .....	72
4.5 Discussion of Results.....	77
4.6 Amendments to the Research Environment.....	78
Chapter 5 : Conclusion .....	80
5.1 Research Outcomes .....	81
5.2 Critical Review of the Research .....	83
5.3 Future Work .....	84
References.....	86
Appendix A: Operability Metrics .....	89
Appendix B: OTG Field Notes .....	90

## List of Figures

Figure 1.1: Relationship between Methodologies and Frameworks .....	12
Figure 3.1: Research Design.....	38
Figure 4.1: Abstract Quality Model (adapted from ISO/IEC 25010:2013).....	56
Figure 4.2: Penetration Testing Quality Model adapted from (ISO/IEC 25010:2013) .....	56
Figure 4.3: OSMM Maturity Model (Navica, 2004).....	72

## List of Tables

Table 2.1: Quality Characteristics Comparison .....	31
Table 3.1: Information Systems Research Approaches (Galliers, 1990).....	33
Table 3.2: Research Question Phase Mapping .....	37
Table 4.1: Factors for Classification .....	43
Table 4.2: Framework vs. Methodology Characteristics .....	44
Table 4.3: Gap Analysis Matrix .....	50
Table 4.4: Classification of Penetration Testing Frameworks and Methodologies .....	51
Table 4.5: Quality Metric Comparison .....	52
Table 4.6: Extensibility .....	58
Table 4.7: Revision Frequency .....	60
Table 4.8: Domain Coverage .....	63
Table 4.9: Framework Appropriateness .....	64
Table 4.10: Learnability Metrics (ISO & IEC, 2002).....	65
Table 4.11: Amended Learnability Metrics .....	65
Table 4.12: ISSAF and OTG Learnability Result.....	66
Table 4.13: Operability Metrics .....	67
Table 4.14: Fog Index Scores .....	69
Table 4.15: Third-party Tool Availability .....	71
Table 4.16: Maturity Evaluation Template (Navica, 2004) .....	73
Table 4.17: OTG Maturity Score .....	74
Table 4.18: ISSAF Maturity Score .....	75
Table 4.19: Reliability Summary .....	77
Table 5.1: Research Questions and Proposed Solution .....	82

## **Chapter 1 : Introduction**

This chapter presents an overview of the topic of this thesis; an evaluation of penetration testing methodologies and frameworks. It begins by providing the necessary background information to contextualise the research questions this thesis aims to address. Next, the significance of this research study is discussed followed by the purpose to add further insight to the thesis topic. The research questions to be explored are presented with the aim of defining concepts and evaluation of currently available penetration testing methodologies and frameworks. Following the research questions a definition of terms is provided. This chapter concludes with the thesis structure.

### **1.1 Background:**

Cyber security is a widely used term along with similar terminology, i.e., information security. Cyber security can be defined as the ability to protect or defend the use of cyberspace from cyber attacks or similarly; measures taken to protect computers, networks, or data from unauthorised attack (Kissel, 2013). The rate of cyber security threats detected for business and government has increased rapidly with close to 7,300 incidents reported to the Computer Emergency Response Team (CERT) Australia in 2012 and approximately 8,500 incidents reported by August 2013 (Patteson, 2013). Fortunately, mitigation strategies exist for organisations, governments, and individuals to minimise risk. One mitigation strategy commonly used within the cyber security industry is penetration testing, commonly referred to as pen testing.

Pen testing helps evaluate information security measures through the eyes of a potential attacker with the aim of testing the security posture of a system (Midian, 2003). Pen testing is often employed by organisations as a mitigation strategy to reduce the risk of an attack on computer resources or in some cases critical infrastructure. Pen testing attempts to ensure weaknesses and vulnerabilities in a networked environment are detected and can be addressed before they are exploited in a real-world attack (Tang, 2014). In a typical penetration testing project, a security practitioner (pen tester) will conduct a series of tests in an attempt to gain access to a system and exploit vulnerabilities or security flaws using the same tools and techniques that simulate a malicious attack, but do so in a controlled manner (Yeo, 2013). The difference between an outside attacker and a pen tester is that the pen tester has been granted permission from the resource owner to carry out security tests in the interests of identifying vulnerabilities and securing systems before an attack occurs. As an information security strategy, pen testing offers a promising look into security defences through the eyes of potential attackers. The significance of a properly scoped and deployed pen test is that it can be an invaluable tool to assess the ability of a system to survive malicious attack (Valli, Woodward, Hannay, & Johnstone, 2014).

From a pen testing perspective a crucial success factor is its underlying methodology. A well-defined methodology is paramount to achieve a good result in identifying security flaws, in other words, gaps can be identified within the security posture and an organisation can validate the robustness of their security controls (Yeo, 2013). Without an established methodology or framework within which to conduct a pen test, identifying vulnerabilities consistently can become difficult (Frankland, 2009). Wilhelm (2009, p. 154) asserts that penetration tests are projects that need to be developed using effective and repeatable processes for improvements to be made, businesses goals to be met, and quality improved. This suggests that penetration testing is achieving some level of maturity, akin to software engineering, although the lack of attention paid to software vulnerabilities in initial system release may be due to the fixation of project managers on visible functionality as noted by Johnstone (2009).

In response to the growing need for security assessments and mitigation strategies, a range of pen testing methodologies and frameworks have been developed with the aim of developing a structured approach to pen testing projects. Avison and Fitzgerald (2006, p. 418) discuss in detail the loose but extensive use of the term “methodology” and argue that there is very little agreement as to what it means other than at a very general level. Furthermore, there appears to be very limited literature addressing frameworks and methodologies for the purposes of penetration testing specifically. Consequently, pen testing methodologies and frameworks appear to be poorly defined. Despite this confusion of terms there are many pen testing methodologies/frameworks available. Some are free to use, whereas others require some form of membership, payment or contribution, for example technical input to the framework or methodology. Several pen testing methodologies and frameworks widely available in particular include: Open Source Security Testing Methodology Manual (OSSTMM), Information Systems Security Assessment Framework (ISSAF), Open Web Application Security Project (OWASP), Metasploit Framework, and Building Security in Maturity Model (BSIMM) also known as Software Security Framework (SSF).

Similar to any project, one of the first steps in a pen test is defining an appropriate scope. Depending on the scope, it can often be difficult to decide which methodology or framework to use, moreover whether a framework should be used over a methodology or vice versa; or perhaps a combination of the two. To complicate things even more, domain coverage of penetration testing projects can vary in degree depending on the scope. Domain coverage can comprise of: web application testing, mobile phones, wireless networks, physical networks, operating systems, cloud computing and software applications. In addition, processes and tasks of projects can encompass

port scanning, vulnerability assessment, information gathering, password cracking and application exploitation. With the aforementioned variations of domain coverage, processes, and tasks it would therefore be beneficial to properly understand the concept of methodology and framework in this context. Methodologies and frameworks often provide structure and effective workflow for pen testing procedures (Holik, Horalek, Marik, Neradova, & Zitta, 2014). On the other hand frameworks should be implemented at a more abstract level, consequently a methodology might be used to implement the framework (Mnkandla, 2009), as shown in figure 1.1.

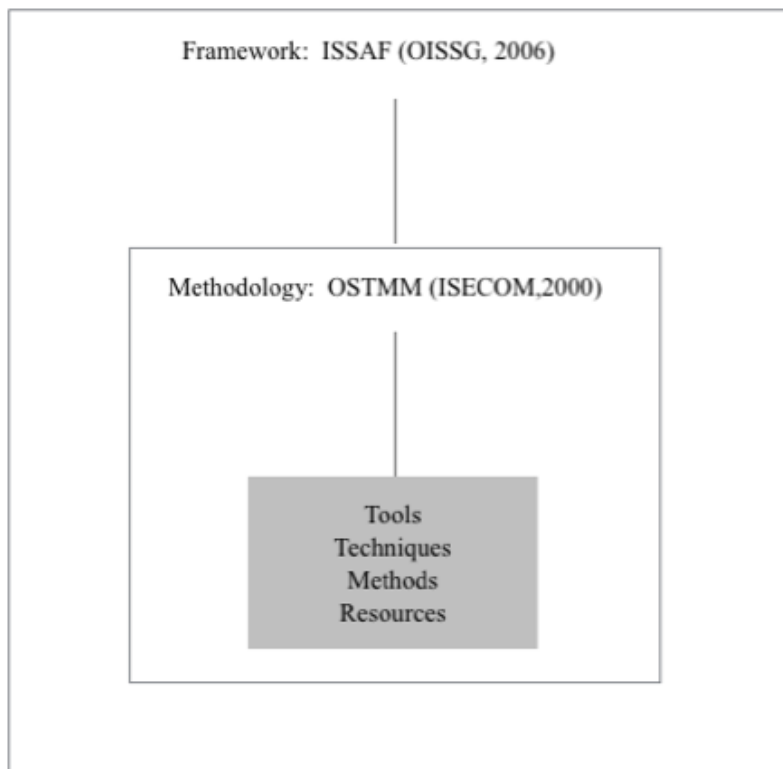


Figure 1.1: Relationship between Methodologies and Frameworks

While there are an array of methodologies and frameworks available for cyber security professionals, little research has been undertaken to determine what characteristics define a high quality framework or methodology, moreover evaluation methods aimed at penetration testing methodologies and frameworks for quality seem to be non-existent.

Defining quality characteristics is considered important for two primary reasons. First, technology is constantly changing; as a result bringing forth new security risks, for example, with the uptake of IPv6 changes to the pen testing processes are necessary (Ottow, van Vliet, de Boer, & Pras, 2012). Second, new devices are introduced into the mix of various domains, for instance, the mobile

phone, thus changing the security landscape. As a result, additional domain areas and/or hardware devices will require research and testing as new vulnerabilities are discovered. Defining a set of quality metrics might prove beneficial for the cyber security discipline overall as penetration testing grows in maturity. Quality characteristics for consideration in particular are: extensibility, maintainability, usability, domain coverage, and reliability.

To summarise, the framework of choice should provide enough flexibility to incorporate the latest vulnerabilities and have the capability of extension to allow for new domain coverage and class of problems without changing the architecture of the framework. Moreover, defining quality characteristics will provide pen testers with a means of evaluating frameworks more effectively before implementation. In addition, a comparison of the different methodologies and frameworks followed by a clear distinction of terms as to what differentiates a framework from a methodology would be beneficial for pen testers.

## **1.2 Significance:**

The cyber security industry is growing at a rapid rate with worldwide spending expected to rise with an estimated \$348 million by the year 2016 rising to \$547 million by the year 2018 (Gartner, 2016). Australian security and intelligence agencies have stated publically that Australia is experiencing an increase in sophisticated cyber attacks in both government and business, originating from an array of sources, mainly individuals, organised criminals, and foreign intelligence services (Patteson, 2013). Director-General of the Australian Secret Intelligence Service (ASIS) Nick Warner, stated “The field of cyber operations is one of the most rapidly evolving and potentially serious threats to our national security in the coming decade” (Warner, 2012, p. 6). CERT released the Cyber Crime and Security Survey in 2013 that showed an overall increase from 56 organisations reporting cyber security incidents in 2012 to 76 in 2013 (CERT, 2013).

With increased awareness of cyber security on the rise, pen testing has come to be one primary mitigation strategy to counter cyber attack (Tang, 2014). The Australian Signals Directorate (ASD) provides penetration testing services for Australian Government Agencies and further recommends organisations and business implement the “Top four mitigation strategies to protect your ICT system” as part of cyber security measures (Australian Signals Directorate, 2014). While most pen testing methodologies and frameworks are open source and obtainable in the public domain, there is little research comparing these methodologies in terms of quality and operational applicability as being valid and applicable for today’s cyber threats. In other words, is a methodology extensible, whereby it accommodates for new additions to the documentation or software, for instance,

extending its coverage for the uptake of IPv6. Similarly, is the framework maintainable whereby it can easily adapt to industry demands, and more importantly, do the creators support it? Furthermore, an understanding of the concepts by means of nomenclature resolution could help security professionals, new to the discipline, identify characteristics of a mature framework and methodology that is based on a sound underlying ontology.

### **1.3 Purpose:**

The purpose of this research is to evaluate a subset of nine penetration testing methodologies and frameworks using quality metrics. The research will begin with nomenclature resolution to clear any confusion that may exist relating to the terms “methodology” and ‘framework”. To extend upon this, the nine frameworks and methodologies will be classified into an appropriate category based on the definition to determine if they are indeed a framework, methodology or, for example, a resource. In addition, the research aims to define a set of quality metrics (by means of a quality model review) suitable for evaluating penetration testing frameworks. Subsequently, a quality model is proposed. Next, quality evaluation of a sub-set of the selected frameworks will be undertaken using data analysis and a real-world case study. As a result, the research will generate a detailed reference for cyber security professionals when exercising judgement on pen testing frameworks or methodologies.

### **1.4 Research questions:**

1. How to differentiate between methodologies and frameworks for the purposes of penetration testing
  - a) What characteristics define a penetration testing framework?
  - b) What characteristics define a penetration testing methodology?
2. What quality metrics can be defined for penetration testing frameworks?
3. How can penetration testing frameworks be evaluated using quality metrics?

### **1.5 Definition of Common Terms:**

**Attack surface:** A set of ways in which an adversary can enter a system and potentially cause damage (Manadhata, 2008) or; any reachable or exploitable vulnerabilities within a system (Northcutt, 2011). In penetration testing an attack surface can include several types, for example, any software or hardware that responds to a request. For instance, an open port on a server, an operating system or application on the target system; an employee may also be considered an attack surface by means of socially engineering.

**Black-box Testing:** In penetration testing, there are three main strategies used for security assessment; black-box, grey-box, and white-box. Black-box testing means the tester has no prior

knowledge of the system to be targeted, in other words, the pen tester will simulate a real-world attack whereby the tester is expected to figure out all the system details and loopholes with no prior understanding of the system (Shah & Mehtre, 2014).

**Domain:** Generically a domain is defined as “a sphere of control, influence or concern” (Oxford, 2008). Within a pen testing context, the domain is a particular area of interest for the purposes of a security assessment. Multiple domain areas may be included in a penetration test, for example; wireless, mobile phone, internal network, or web application. The scope and business objectives of a security assessment will likely determine the domain areas that will undergo a penetration test. Similarly pen testing methodologies and frameworks may include or exclude particular domains.

**Extensible:** Laplante (2001, p. 173) defines extensibility as “the capacity of a system, component, or class to be readily and safely extended in its behavioural specification/operational and/or structural capabilities”. Within a methodology and framework context, extensible pertains to the capacity of the framework or methodology to be readily and safely extended with minimal effects on its structure or, have the ability to accommodate new changes with or without instruction.

**Framework:** A Framework can be defined as “a domain specific application, shell, or skeleton software system” (Laplante, 2001, p. 197), or, as defined by Mnkandla (2009), “a skeletal abstraction of a solution to a number of problems that have some similarities”. Penetration testing frameworks can thus be defined as, a reusable shell or skeleton whereby new processes can be added in order to tailor them for new requirements; a reusable template or outline for the domain of penetration testing that may take the form of a documented framework or software framework. It is concerned with “what” needs to be done and allows the security professional to determine “how” methods are employed.

**Grey-box Testing:** Similarly to black-box testing, grey-box testing is an approach to security testing a system for vulnerabilities, however, unlike black-box testing, grey-box testing provides the pen tester with partial disclosure of information and resources about the target system (Shah & Mehtre, 2014). This approach can be thought of as a combination of white-box and black-box testing whereby some information is known, however, the pen tester must gather additional information by means of conducting further tests.

**Maintainable:** Within a software context, to maintain means the ease in which software can be understood, corrected, adapted and/or enhanced (Laplante, 2001, p. 293). A more generic definition is to enable or cause something to continue, or to keep in good condition by checking or repairing regularly ("Maintain", 2015). Sommerville (2007) defines maintainability as “software that can be adapted economically to cope with new requirements and where there is a low probability that making changes will introduce new errors into the system”. In relation to a maintainable methodology or framework the definition can be interpreted as, a methodology or framework that



can be adapted to cope with new requirements, and revisions/updates are applied regularly by its authors. For example, new and emerging technologies may be added and out-of-date practices removed. This implies the document or software undergoes regular updates and revisions.

**Method:** Often method and methodology are used interchangeably as Avison and Fitzgerald (2006) argue that what differentiates the two terms is the underlying philosophy of a methodology. In other words, a method is devoid of a philosophy. For the purposes of penetration testing, a method is a preferred way of performing a task suited to a particularly unique situation.

**Methodology:** Avison and Fitzgerald (2006, p. 10) define a methodology as “a collection of procedures, techniques, tools and documentation aids, which will help the systems developers in their efforts to implement a new information system. A methodology consists of phases, themselves consisting of sub phases, which will guide the systems developers in their choice of the techniques that might be appropriate at each stage of the project and also help plan, manage, control and evaluate information systems projects”. This definition can be adapted within a penetration testing context stating that a methodology is a collection of procedures, techniques, tools, and documentation aids that assist a penetration tester in his/her effort to perform a complete penetration test. A methodology consists of phases, themselves consisting of sub phases, which will guide penetration testers in their choice of techniques that might be appropriate at each stage of the project. It may also detail how to perform a particular task providing tools and techniques, and, assist with planning, management, and control as part of the methodology. It is concerned with “how” and “why” things are done.

**Security Assessment:** A security assessment is the measurement of the security posture of an organisation or a system (Miles, Rogers, Fuller, Hoagberg, & Dykstra, 2004). The security posture relates to the way the information system security is implemented and relies on three primary interrelated methods, such as review, examination, and testing. These methods combined can accurately assess the technology, people, and processes (Abdel-Aziz, 2011). Review relates to information gathering, whereas examination relates closely to identifying vulnerabilities. Finally, testing involves penetration testing the vulnerabilities identified in the examination phase; therefore a security assessment is the overall security posture, different from penetration testing. Penetration testing therefore, is one facet of a security assessment.

**Technique:** A way of carrying out a particular task, put another way, a skilful or efficient way of achieving something ("Technique", 2015). According to Avison and Fitzgerald (2006, p. 14), a technique is defined as “a way of doing a particular activity in the information systems development process and any particular methodology may recommend techniques to carry out many of these activities”. The abovementioned definition closely relates, and can be adapted to a penetration testing context, thus a technique is a way of doing a particular activity when performing a

penetration test, and any particular methodology or framework may recommend techniques to carry out these activities.

**Tool:** Software tools, as defined by the Oxford Dictionary of Computing, is “a program that is employed in the development, repair, or enhancement of other programs or hardware” (Oxford, 2008, p. 476). Penetration testing utilises a multitude of tools, therefore in this context it is defined as software that can assist in all activities of all phases of a penetration test for a specific task.

**White-box- Testing:** Contrary to black-box testing, white-box testing provides the pen tester with all the necessary information and resources about the system. This approach attempts to audit the internal security arrangements, thus simulating the actions and procedures of internal threats, for instance, disgruntled employees who have system access (Shah & Mehtre, 2014).

## 1.6 Thesis Structure

The remainder of this thesis is divided into four chapters. Chapter two provides a literature review, which describes existing evaluations relating to penetration testing methodologies and frameworks, and delves into literature concerned with nomenclature resolution, thus reviews the concept of framework and methodology in various contexts. This is followed by a detailed review of nine penetration testing methodologies and frameworks that are evaluated in chapter four. Finally, four existing quality models are reviewed.

Chapter three outlines the research methods and design of the thesis. It begins by identifying a suitable approach primarily in the field of information systems. Subsequently, an analysis of approaches of a particular topic area (object) of study is discussed with the aim of selecting the research approach. Next, research design includes five-phases presenting the design process in order of completion, followed by the materials required to conduct the research. Finally, the methods employed for data analysis are discussed. The chapter concludes with an outline of limitations identified for this research.

Chapter four provides an analysis and discussion of the research that is structured in four sections. Each section addresses the research questions in order respectively, thus section 4.1 delves into nomenclature resolution, whereby a solution for defining methodologies and frameworks for the purpose of penetration testing is proposed. Section 4.2 extends upon section 4.1 and presents a gap analysis of the nine penetration testing methodologies and frameworks reviewed, thus identifying framework vs. methodology characteristics that are subsequently used for the selection of two candidate frameworks employed for further evaluation. Section 4.3 addresses research question two which asks what quality metrics can be defined for penetration testing frameworks. It presents a

quality model of which provides the foundation for framework quality evaluation. Finally, section 4.4 performs a quality evaluation of the two selected frameworks that employs qualitative and quantitative means for measuring quality. The chapter concludes with a discussion of results and review of the research amendments.

This thesis concludes with chapter five, where the research questions and findings are summarised. The chapter commences with a discussion on the outcome of the research framed in relation to each research question respectively. Next, a critical review of the research is presented addressing complications and difficulties encountered throughout the research process. Finally, future work is identified that subsequently puts forth further questions and topic areas that could potentially expand upon this research, thus extending upon the knowledge of the research topic.

## **1.7 Summary**

This chapter presented the background information relating to the evaluation of penetration testing frameworks and methodologies highlighting the areas to be explored throughout this research. Subsequently, the significance and purpose were discussed followed by the research questions this thesis will address. A definition of terms was provided to add context to the terms frequently used throughout this thesis, concluding with the thesis structure. Chapter two begins by a review of the relevant literature required to commence this research.

## **Chapter 2 : Review of the Literature**

The global security testing market is experiencing rapid growth with reports estimating from (US) \$4.96 billion by the year 2019 ("Security Testing Market Worth \$4.96 Billion by 2019," 2014) to (US) \$170 billion by 2020 (*Cyber Security Market worth \$170.21 Billion by 2020* 2015). Penetration testing is one field within the security testing market benefitting from this growth and is not expected to slow down, thus it is not surprising that penetration testing firms are achieving success in the market place. For organisations, business applications have become vital information assets therefore securing applications has come to the fore due to the increase in cybercrime. For this reason penetration testing methodologies are paramount for successful identification of vulnerabilities, i.e., successful identification will largely depend on the underlying methodology (Frankland, 2009). This research aims to evaluate penetration testing methodologies and/or frameworks for their potential use in practice or research and in addition, determines a set of quality characteristics that can be applied to evaluate a methodology or framework for real world use.

The literature review is partitioned into four primary areas. First, a review of literature that attempts to evaluate existing methodologies and frameworks, including the differentiation of frameworks, tools, and techniques. The second section will discuss the confusion of definition of terms between methodology and framework. Avison and Fitzgerald (2006, p. 418) discuss in detail the loose but extensive use of the term "methodology" and argue that there is very little agreement as to what it means other than at a very general level. Furthermore, there is limited literature addressing frameworks and methodologies for the purposes of penetration testing specifically. Consequently, penetration testing methodologies and frameworks appear to be poorly defined. Therefore, the literature review will focus on generic definitions and more specific definitions drawn from the field of information systems. Third, a review of nine penetration testing methodologies and frameworks will be presented to objectively examine the diverse range of penetration testing methodologies and frameworks that exist. The format attempts to follow author, description, advantages, comparison to other frameworks and/or methodologies, coupled with a summary. Finally, a review of quality models that could potentially be used as an underlying model to effectively evaluate penetration testing frameworks will be included.

### **2.1 Existing evaluations of Frameworks and Methodologies**

A range of evaluations concerning penetration testing exists in the literature, however, it appears evaluation of more than two or three methodologies is minimal; furthermore, any evaluation of quality of these methodologies is non-existent. Kang (2008) asserts the importance of an underlying methodology specifically for penetration testing whereby the discussion is focused around three

methodologies in particular, namely, NIST, OSSTMM, and ISSAF. Kang does not address quality, however proposes that a methodology should include three main properties: sufficient information, an experienced team, and adequate tools for the job. Kang concludes that lack of a formal methodology will inherently result in inconsistency and that a methodology should provide a disciplined foundation to complete an accurate penetration test. Similarly, Holik et al. (2014) discuss the importance of penetration tests with a focus on two methodologies, namely, OWASP and OSTMM. Holik et al. do differentiate between methodologies and tools offering a discussion on various tools available, for example, Kali Linux and Nessus. Nevertheless the primary focus is Metasploit Framework whereby a case study is used as an example for conducting penetration tests. The authors conclude that there are multiple approaches to penetration testing consisting of methodologies, frameworks, and software tools, thus a security professional should invest a significant amount of time learning what tools and techniques exist. While NIST-800-115 is a penetration testing guide, it does discuss well-known methodologies, for instance OSTMM and OWASP (NIST, 2008), however comparison of methodologies is not addressed. NIST-800-115 distinguishes between tools and methodologies, thus demonstrating a distinction between the two. It is also important to note that while research is lacking in regards to comparing the various methodologies and frameworks or discussion on quality among them, there is sufficient literature on certain methodologies individually, therefore they can be researched in isolation.

## **2.2 Defining Methodologies and Frameworks for the purpose of pen testing**

Although the literature provides definitions for methodologies and frameworks in various contexts, primarily information systems, there appears to be dissonance when relating these terms to penetration testing. For instance, Avison and Fitzgerald (2006) define a methodology as "a collection of procedures, techniques, tools and documentation aids, which will help the systems developers in their efforts to implement a new information system. A methodology consists of phases, themselves consisting of sub phases, which will guide the systems developers in their choice of the techniques that might be appropriate at each stage of the project and also help them plan, manage, control and evaluate information systems projects". In generic terms the Oxford Dictionary defines a framework as "An essential supporting structure of a building, vehicle or object", or similarly more software focussed; "a basic structure underlying a system, concept, or text" ("Framework", 2015). Johnson (1997, p. 1) states, "a framework is the skeleton of an application that can be customised by an application developer" and "a framework is a reusable design of all or part of a system that is represented by a set of abstract classes and the way their instances interact". Depending on the context, the definition can vary as illustrated from the aforementioned definitions. From an application development context, frameworks are used as a

skeleton from which software applications can be built (Land, 2002). For penetration testing however, software applications are typically not the final product; moreover, the final product is generally a report that details results of a series of tests undertaken throughout a pen test consequently presented to the client as a report. Frameworks within a pen testing context, according to Wilhelm (2009), focus primarily on processes, activities and tasks, whereas a methodology will encapsulate processes, activities and tasks. Similarly, Mnkandla (2009) states that a framework should be implemented at a more abstract level, thus a methodology might be used to implement the framework.

In summary, methodologies and frameworks consist of various phases, characteristics, processes, and techniques providing structure and guidance to complex tasks. It can thus be said that a reliable pen test should consist of well-defined, repeatable processes; in other words treated as a project. With an underlying methodology in place results can be repeated and verified (Wilhelm, 2009), therefore defining what a methodology means within the boundaries of pen testing would be crucial; however, it is not clear from the literature what defines a methodology and what defines a framework within a pen testing context.

## **2.3 Review of Nine Penetration Testing Methodologies and Frameworks**

### **2.3.1 Information System Security Assessment Framework (ISSAF)**

ISSAF is an Open Source, peer-reviewed, penetration testing framework created by the Open Information Systems Security Group (OISSG). ISSAF is described as a framework and encapsulates multiple methodologies (draft 0.2.1B). The ISSAF appears to be a detailed and comprehensive framework covering a variety of domains whereby each is allowed to have its own methodology. Domain areas include, but are not limited to: password security testing, switch and router security, firewall security, intrusion detection systems assessments, VPN security, web application security, and windows security. Although not all domains are listed here it is suffice to say that the ISSAF attempts to cover all possible domains of a penetration test from conception to completion. The penetration testing methodology of the framework is divided into three primary phases, namely: planning and preparation, assessment, reporting and clean up. One advantage of the ISSAF in particular is that the distinct relationship between tasks and the associated tools for each task are shown, in addition screenshots of expected outputs are provided, together with explanations of “how” and “why” each task is performed. The ISSAF is large in comparison to other frameworks but assumes it is easier for organisations to delete material rather than develop it from the ground up (OISSG, 2005). Two disadvantages of the ISSAF are support and maintenance. The framework has not undergone any updates post 2006, thus, includes out-dated systems, for instance, Novell

Netware and Lotus Notes. Furthermore, some pen testing tools recommended within the ISSAF are no longer used in industry. Taking into account that additional tools and techniques have been developed since 2006, a user should be cautious when relying on this framework alone.

### **2.3.2 Open Source Security Testing Methodology Manual (OSSTMM)**

OSSTMM is an open source security testing methodology introduced in 2000 by the Institute for Security and Open Methodologies (ISECOM). OSSTMM was developed under peer-review and benefits from open source licensing, however, access to the latest version (version 4) require paid membership. The OSSTMM (version 3) is defined as a methodology that encapsulates modules and channels (ISECOM, 2000). OSSTMM classifies a domain area of interest as a channel. The OSSTMM consists of five channels: human, physical, wireless, telecommunications, and data network security. In these channels the methodology includes current environments namely, cloud computing and virtualisation. Unlike the ISSAF, OSSTMM does not recommend what tools to use, rather, best practices to follow. The OSSTMM refers to each repeatable phase as a module and assumes a security professional possesses sufficient knowledge of tools and techniques required to perform each module. OSSTMM provides adequate guidelines accompanied by templates for reporting; it also includes trust metrics that allow risk assessment for other factors that cannot be tested. That means cloud, vendor contracts and services, products, and even employee hiring can all be measured for their risk and attack surface, which could be considered an improvement compared to ISSAF. Attack surface can measure and point out the places where an attacker might attack, and which types of attacks would likely be successful.

In summary the OSSTMM is an auditing methodology designed to be consistent and repeatable at an operational level by providing accurate measures of security by means of a security audit. Both OSTMM and ISSAF provide guidance but the latter suggests tools or methods for completing modules. OSTMM is a valuable auditing resource that can be used to satisfy regulatory requirements for corporate assets provided security auditors have sufficient skills to complete each module.

### **2.3.3 NIST 800-115**

NIST 800-115 is a technical guide for information security testing and assessment and is maintained by the National Institute of Standards and Technology (NIST) for federal government agencies in the U.S. It is free for use in the private sector and is not subject to copyright though attribution is desired (NIST, 2008). NIST-800-115 provides guidelines for planning and conducting information security tests. By comparison, it is not as detailed as the OSSTMM and ISSAF. NIST offers structured documentation on how to implement a methodology with repeatable processes for

security assessments primarily aimed at organisations. In contrast to ISSAF and OSSTMM, NIST does not focus on pen testing alone, moreover, a focus on security assessments that include pen testing as part of the whole process. NIST 800-115 explains in detail planning, execution, and post execution techniques with a focus on identifying vulnerabilities. Similar to the OSSTMM, NIST 800-115 does not recommend particular tools but presents an appendix of suggested tools used in the field of cyber security for particular tasks. NIST 800-115 assumes the security professional possesses the knowledge and skills required for conducting penetration tests. The NIST 800-115 Guide was designed with the aim of providing organisations with a means of developing a methodology that can be adapted for security assessments thus fostering repeatable processes, planning, execution, and post-execution techniques for organisations. In summary, NIST 800-115 is a valuable guide that can be implemented as a security strategy for organisations. The documentation assists organisations in the development of an information security testing methodology, how to accurately plan and execute an assessment, and how to conduct analysis reporting. NIST 800-115 does not identify itself as a methodology; it assists in the development of a methodology for the purposes of information security assessments.

#### **2.3.4 Open Web Application Security Project (OWASP)**

OWASP is a not-for-profit organisation focused on improving software security. OWASP provides numerous tools, guides and testing methodologies for cyber security under the open source license. One of scores of security guides available is the OWASP Testing Guide (OTG), obtainable from the OWASP website. OTG encapsulates a testing framework for software development, web application security testing methodology for penetration testing web applications, and a reporting guide. The OTG details tasks and techniques appropriate for various phases of the software development lifecycle focussed on developing software with security in mind as opposed to identifying vulnerabilities after development. The OWASP Testing Guide is divided into three primary sections, namely, the OWASP testing framework for web application development, the web application testing methodology, and reporting. The OWASP Testing Guide encapsulates the Web Application Methodology (section four) that provides a strong focus on web applications. The methodology can be used independently or in conjunction with the testing framework; in other words, a developer can use the guide to build a web application with security in mind followed by a penetration test (web application methodology) to test the design. The OWASP methodology for web applications brings together tools and techniques used for each phase of a penetration test, however, it does not provide details of each tool, it instead assumes knowledge of tools and techniques. The recommended tools for each phase are detailed throughout; in addition web links to relevant information of tools and



techniques are offered to assist with any knowledge gaps that might exist. The target audience is intermediate to advanced level cyber security professionals with a strong focus on web application technologies. What is unique about this methodology is the discussion on protocol behaviour. In contrast to the methodologies/frameworks discussed thus far, the OTG shows what results might be expected from a test. The web application methodology presents as a well-defined structure that follows a consistent format, namely: phase summary, objective, how to test, recommended tools, references and/or whitepapers, and remediation. The tests cover both client-side and server-side testing coupled with examples. OWASP provides an additional resource for security professionals known as WEBGoat, a project that enables penetration testers to load a vulnerable website in a controlled environment and test these techniques against a live system.

To summarise, the OTG has a strong focus on web application security throughout the software development lifecycle as opposed to the ISSAF and OSSTMM, both of which are aimed at security testing an entire system post-implementation. The OTG is suited specifically to one domain area, that of web applications. It offers a web application testing methodology, whereby pen testers are expected to have appropriate knowledge of tools in the field and how to use them, but does not assume complete knowledge. Therefore, a list of recommended tools is provided.

### **2.3.5 Building Security in Maturity Model (BSIMM)**

Building Security in Maturity Model (BSIMM) is a software security framework (SSF) licensed under Creative Commons and is free to use as a resource (McGraw, Miguez, & West, 2009). The authors quantified the security practices implemented from sixty-seven highly successful companies including; Microsoft, Google, Adobe, McAfee, Sony mobile, HP, and Goldman Sachs, that form the benchmark for BSIMM. The authors describe BSIMM as a study of existing software security initiatives (McGraw et al., 2009). The approach was data first, then model. BSIMM is about what organisations actually do in the real world. BSIMM consists of 112 activities divided into twelve practices that support four domains, mainly governance, intelligence, SSDL touch points, and deployment. BSIMM is designed to help organisations understand, plan, and measure a software security initiative. The framework does not focus on what an organisation should do, instead, what successful organisations are doing. An observation learned from the study was that approximately 1% of the software developer size of an organisation made up a software security group (SSG). Once an SSG was formed within an organisation, they then take the BSIMM and implement its activities into their software security initiative. In comparison to the ISSAF, OSTMM and NIST, BSIMM does not set out what tools to use or how to use them, but the alternative, i.e., what practices successful companies are using to secure their environment. Pen testing is one practice

identified within the BSIMM framework and is only one of many processes recommended; therefore the focus is not penetration testing alone, moreover, a security assessment that includes penetration testing as one activity. BSIMM offers an alternative approach to that of other penetration testing methodologies and frameworks inasmuch as observing “what” other companies are implementing, not what companies “should” be doing.

### **2.3.6 Penetration Testing Framework 0.59 (PTF)**

PTF is defined as a penetration testing framework (Lawson, Byrne, Doraiswamy, & Ouchn, n.d) that contains fifteen sections. The first four sections are concerned with network foot printing, enumeration, password cracking, and vulnerability assessment. The remainder covers specific domain areas, for instance, wireless penetration testing, physical security, Bluetooth specific testing, and VoIP services. In addition, templates are provided for documentation and checklists. Tools and techniques are offered throughout each phase, however, the pen tester is expected to have the knowledge and skill to perform each test. Information outlined in the PTF is out-dated, such as out-dated tools and broken links, thus the framework cannot be considered up-to-date. In comparison to the aforementioned frameworks, PTF lacks structure and the authors do not appear to provide support. Furthermore, documentation regarding the PTF is difficult to obtain. In summary, PTF has the potential to be further developed into a reliable framework; however, the lack of structure and documentation as of this research was obvious. PTF would provide a useful resource of information worth keeping in any pen tester’s resource kit, however, implementation of the PTF as a choice of framework alone may be questionable.

### **2.3.7 Penetration Testing Execution Standard (PTES)**

Penetration Testing Execution Standard (PTES) is a penetration testing standard originally created in 2009 by a group of information security professionals who were of the view there was a lack of quality and guidance within the cyber security testing industry. The goal was to provide both business and security providers with a common language and scope for performing penetration tests with the aim of establishing a baseline to define the boundaries of a penetration test (Nickerson et al., n.d). In other words, to specify what is the minimum set of tests, processes and outputs that should be undertaken before a penetration test could be considered professional and complete. PTES includes pre-engagement interactions, intelligence gathering, threat modelling, vulnerability analysis, exploitation, post exploitation, and reporting. The last document update was April 2012. PTES provides an extensive list of tools and techniques used in each of the seven sections accompanied by a description for each section. Screen shots and links to resources are detailed throughout the document. PTES is not complete as of this writing with contributions required throughout the documentation. Overall this guideline is a valuable resource and has potential to

become a solid foundation for a pen testing framework. PTES takes advantage of other resources with the approach of incorporating other frameworks within it, for example, OWASP for web application testing is referenced and recommended for use when testing web applications. PTES attempts to fill the gaps of NIST-800-115 and OSTMM by providing guidelines, methods, tools and techniques in the one document. It also attempts to create a baseline for a penetration test. The target audience is security professionals with knowledge of tools used in the field, however, it does not assume complete knowledge. Organisations can refer to the document for understanding of what each phase attempts to accomplish for a specific test and what tests should be carried out.

### **2.3.8 Metasploit Framework (MSF)**

Metasploit in generic terms is a suite of penetration testing and intrusion detection tools designed to identify and exploit vulnerabilities on a target system. Metasploit was originally an open source project developed by HD More in 2003. It was acquired in 2009 by Rapid7 who is now responsible for its development and support (Holik et al., 2014). Metasploit is commonly known as the Metasploit Framework with four different versions available, namely, Metasploit Framework (MSF), Metasploit Community Edition, Metasploit Express, and Metasploit Pro. Metasploit Framework (MSF) is a free open source command line only version, whereas the Metasploit Community edition is bundled with a graphical user interface. MSF and the Community Edition versions are free for both commercial and private use. Finally, Metasploit Pro and Metasploit Express are both commercial versions. Metasploit Pro is bundled with extra features, thus suitable for the more advanced pen tester. Metasploit Pro provides advanced usability features, for instance, quick pen testing wizards, enhanced workflow productivity features, and comprehensive reporting tools. In addition, Metasploit Pro provides functionality to import vulnerabilities from vulnerability lists with the intent of testing system users to determine if they are susceptible to phishing attacks by means of simulation. All four editions of Metasploit provide an array of tools for specific tasks, in particular; information gathering, vulnerability scanning, exploitation, maintaining access, and reporting. However the free versions are not as comprehensive as the commercial versions. Updates to all Metasploit editions occur regularly and support is available depending on the edition. With reference to the MSF edition specifically; MSF is a platform for writing, testing, and using exploit code to deliver payloads to target systems for the purposes of penetration testing. It uses a command line interface with an underlying module based architecture and is built with extensibility in mind by allowing different modules to perform different tasks, thus new modules can be added to the framework (Holik et al., 2014). The basic process for the pen tester is first to choose an exploit that exists on the target system. Next, determine if the target is vulnerable to the exploit, followed by selection and configuration of a payload; finally, attempt to exploit the target. Specific information

about the target is required, thus other tools are often used in combination to obtain certain information, for instance; the target operating system, open ports, and/or potential vulnerabilities. MSF is suitable for the more advanced security professional who has a solid understanding of penetration testing and is competent using command line tools. In comparison to ISSAF and OSSTMM, MSF is a practical solution that provides a suite of tools rather than a documented outline of process and methods to follow. Rapid7 has packaged the four Metasploit editions as an off-the-shelf product that pen testers can take advantage of, however it is not clear whether any or all of the editions available are frameworks as the name suggests. Moreover, MSF could be considered an application that encompasses a suite of tools that facilitate a penetration test.

### **2.3.9 Browser Exploitation Framework (BeEF)**

BeEF is an open source browser exploitation framework used primarily as a penetration testing tool for browser based vulnerabilities and web applications. The framework was originally published in 2006 (Alcorn, Orru, Coles, Passmore, & Pilkington, 2012). Alcorn (2007) concluded that encapsulation of an exploit within one protocol can be used in an application using a different protocol, put another way, inter-protocol exploitation; for instance, using JavaScript to encapsulate an exploit within an HTTP request. BeEF comes packaged with a web interface, command-line application interface, and Metasploit integration. Similar to other frameworks, BeEF encompasses information gathering, social engineering, and network discovery as packaged modules. BeEF provides an experienced pen tester with the means of assessing the security posture of an environment by using client side attack vectors. The BeEF architecture consists of two interfaces. First, the user interface whereby a pen tester can observe all online and offline browsers to conduct exploits against. Second, the communications server. The communication server transmits via HTTP information concerning hooked browsers. BeEF covers only one domain area, that of the web browser. Similar to Metasploit, BeEF is a supported off the shelf application ready to use for experienced pen testers. In comparison to the ISSAF, OSSTMM, NIST 800-115, BeEF is not a document of processes and guidelines, rather a downloadable product specifically aimed at web applications.

In summary, there is a diverse range of methodologies and frameworks available. Each consists of unique characteristics and takes a distinct approach to a penetration testing project. The literature suggests a difference in the way terminology is applied to each framework or methodology respectively, thus used interchangeably. For instance, the ISSAF is defined as a framework however throughout the documentation it refers to methodology as the primary approach. Metasploit on the other hand is described as a framework, whereas it could be considered a software application

encompassing a suite of tools. Therefore clarification on the classification of common methodologies and frameworks is essential to avoid confusion.

## **2.4 Quality Models**

The Institute of Electrical and Electronics Engineers (1990) defines quality as the degree to which a system, component, or process meets specified requirements and the degree to which a system, component, or process meets customer or user needs or expectations. Similarly, Standards Australia (2000) defines quality as “the totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs”. Assessing a product’s quality generally relates to how “good” a product is, thus the use of product quality models is an acceptable means to determine a product’s overall quality (Miguel, Mauricio, & Rodriguez, 2014). Fortunately, the discipline of software engineering has available a range of quality models. Whilst focussed on software quality, these models can be drawn upon and applied to the field of penetration testing. Potentially suitable quality models include: McCall’s, Boehm’s, and Dromey’s Models. In addition, ISO 9126 and ISO/IEC 25010:2013 among others (Al-Qutaish & Abran, 2011).

### **2.4.1 McCall model**

Originally produced for the U.S. Air force, the McCall model was devised as a hierarchical model as a way to derive metrics for quantifying software for objective analysis (McCall, Richards, & Walters, 1977). The model is partitioned into three main perspectives for characterising quality attributes for software, namely, product review, product operation, and product transition, each containing sub characteristics referred to as factors (Miguel et al., 2014). Product review is concerned with the product’s ability to undergo changes (maintainability, flexibility and testability). Product operation addresses product operational characteristics (correctness, reliability, efficiency, integrity and usability). Finally, product transition is concerned with the product’s ability to adapt to new environments (portability, reusability and interoperability) (Al-Badareen, Selamat, Jabar, Jamilah, & Sherzod, 2011). The aim of the model is to determine the overall quality of software by means of yes/no metrics. In other words, if 50% of the quality metrics are ‘yes’, then the overall quality criteria achieved is 50%. A consideration worth noting however, is the answers to the questions relating to quality criteria depend largely on the evaluator’s judgement (Berander et al., 2005). Another drawback of McCall’s model is that user experience is not considered.

### **2.4.2 Boehm Model**

The Boehm model, similarly to the McCall model, is hierarchical in nature. The Boehm model attempts to improve upon the McCall model by adding an extra level to the hierarchy (Al-Badareen et al., 2011), thus a three level structure defining software quality at the base level. The higher level

(level one) includes three primary factors: portability, “as-is” utility, and maintainability (Miguel et al., 2014). Portability maps directly to level three factors, therefore does not contain sub-factors, and is concerned with the product’s ability to adapt to a new environment. Maintainability and as-is utility, on the other hand, map directly to level two factors that subsequently map to level three in the hierarchy. As-is utility consists of three sub factors: efficiency, usability, and reliability and is concerned primarily with the extent to which the software can be used “as-is”. Maintainability attempts to address the ease of modification and identifying change, thus the sub factors are; testability, understandability, and flexibility. Level three is further refined encompassing eighteen primitive factors providing a foundation for defining quality metrics. Ultimately, the goal of the Boehm model was to represent one or more metrics measuring a particular factor (Berander et al., 2005), however, the Boehm model does not offer a measurement methodology.

### **2.4.3 Dromey Model**

The Dromey model focuses primarily on product software, code, and product implementation. The Dromey model is empirical in nature with the idea of being broad enough to work with different systems (Al-Badareen et al., 2011). The model attempts to establish a link between tangibly quality characteristic and less tangible quality attributes (Dromey, 1995). According to Dromey (1995), other models fail to deal with product characteristics adequately, in addition, fail to link quality attributes with the corresponding product characteristics. The Dromey model suggests that quality evaluation is different for each product, thus proposes a more dynamic model compared to McCall’s and Boehm’s (Al-Badareen et al., 2011). Similarly to McCall and Boehm, Dromey is hierarchal and consists of three tiers, although it could be considered a two-tier model considering the first tier is the software product (implementation). The second tier is concerned with product properties that influence quality, therefore it identifies four product properties, namely: correctness, internal, contextual, and descriptive. Each product property maps directly to the third tier, whereby the corresponding quality attributes are identified. Correctness directly relates to quality attributes of functionality and reliability whereas internal relates to maintainability, efficiency, and reliability. Contextual shares two quality attributes with the internal product property, namely, maintainability and reliability, in addition, it includes reusability and portability. Similarly, the descriptive product property shares quality attributes with its predecessors, such as, maintainability, reusability and portability, with the addition of usability as an extra quality characteristic.

### **2.4.4 ISO 9126**

The ISO 9126 standard is a hierarchical model based on the Boehm and McCall models (Berander et al., 2005). The model is described in two parts. Part one consists of internal and external quality attributes that define software by its functionality, thus encapsulates quality characteristics from an

internal and external perspective. Quality attributes are partitioned into six characteristics: functionality, reliability, usability, efficiency, maintainability, and portability. Each of the top-level characteristics consists of sub characteristics that can be measured by internal or external metrics (Standards Australia, 2005). Part two describes a quality in use model that is intended to represent the users point of view. Quality in use details four attributes: effectiveness, productivity, safety, and satisfaction. Standards Australia (2005, p. 20) describes quality in use as “the user's view of the quality of an environment containing software, and is measured from the results of using the software in the environment, rather than properties of the software itself”. It therefore implies that the level of quality in the users' environment may be different from that of the developers' environment. Additionally, ISO 9126 suggests how metrics can be selected for quantitative measures and asserts that metrics selection will likely depend on business objectives and needs of the evaluator, thus supports a variety of evaluation requirements. ISO 9126 states it is not practically possible to measure all internal and external characteristics, therefore using the quality model is recommended as a checklist of issues relating to quality.

#### **2.4.5 ISO/IEC 25010:2013**

The ISO/IEC 25010 quality standard (Standards Australia, 2013) was derived from ISO 9126, and is considered to be a revised version of the latter. ISO/IEC 25010 is identical to ISO 9126 inasmuch as product quality and quality in use models are described. ISO/IEC 25010 defines a product quality model composed of eight characteristics as opposed to six, thus the taxonomy contains: functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability, and portability. Similarly, ISO/IEC 25010 is hierarchical and further divides sub-characteristics that relate to static properties of software respectively. Security, which is the primary focus of this research, is defined as a sub-characteristic of functionality in ISO 9126. This is a departure from the commonly held belief that security is solely within the domain of non-functional requirements. The ISO/IEC 25010 standard (Standards Australia, 2013) extends upon this idea and considers security a characteristic in its own right. ISO/IEC 25010 is therefore a replacement for ISO 9126.

#### **2.4.6 Summary**

The McCall model and Boehm model appear similar, however the Boehm model attempts to improve upon McCall with its structure by means of partitioning the top most level more broadly. For instance rather than focusing on software operation, review and transition as a basis, it prioritises portability, maintainability, and as-is utility as primary factors, therefore a departure from McCall's original approach. Turning to the Dromey model, the approach is more dynamic, suggesting that each product should be evaluated differently, whereby product properties link directly to quality characteristics. ISO/IEC 25010 is a replacement for ISO 9126, therefore could be

considered the more relevant ISO quality standard in this context. ISO/IEC 25010 considers “user experience”, thus is a two-part model, as compared to the aforementioned models whose focus is primarily software development.

Table 2.1 summarises the most common quality characteristics displayed across the five models reviewed. Table 2.1 shows that maintainability, portability, and reliability are used across all models, therefore may be considered significant quality metrics overall, followed closely by reusability and testability.

Table 2.1: Quality Characteristics Comparison

	McCall	Boehm	Dromey	ISO 9126	ISO/IEC 25010	Total
Correctness	*		*			2
Efficiency	*	*	*			3
Functional			*	*	*	3
Integrity	*	*				2
Maintainability	*	*	*	*	*	5
Modifiability		*		*	*	3
Operability	*			*	*	3
Portability	*	*	*	*	*	5
Reliability	*	*	*	*	*	5
Reusability	*		*	*	*	4
Security				*	*	2
Testability	*	*		*	*	4
Usability	*			*	*	3

## 2.5 Conclusion

Business objectives and project scope will likely influence the selection of a particular framework or methodology when considering undertaking a penetration-testing project. When determining an approach, clearly defined terms are one significant factor for consideration, primarily because penetration testing encompasses repeatable phases and tasks similarly to that of a project. Therefore, adapting a methodology or framework that is well-defined within a penetration testing context would be an advantage. The literature, however, suggests confounding terms within this context. Particular frameworks cited in section 2.3 take a distinct approach to penetration testing.



Some are aimed at the entire security posture of an organisation, while others are more specialised in their approach, for example, OWASP Testing Guide illustrates a strong focus on web applications. Commonalities among them do however, exist. Phases identified predominately include: planning and preparation, information gathering, vulnerability identification, exploitation, maintaining access, and reporting. Although not limited to these phases, generally a framework will recognise one or more of these phases in one form or another.

Quality of a particular framework or methodology is of further interest. This research intends to evaluate penetration testing frameworks for quality, therefore a reliable underlying model to draw from is of significance. Quality relating to documented frameworks is scarce, however that is not to say non-existent, moreover difficult to obtain. Before considering the practical applicability of an underlying quality model for evaluating penetration testing frameworks, criteria for devising a proposed quality model must be selected. Therefore, the quality models reviewed in section 2.4 were considered for two reasons. First, they were sourced from a related field of study (that of software engineering), and second, they were developed from an authoritative source, in other words widely recognised and accepted in industry. The existing body of knowledge offers little evidence for comparing documented frameworks for quality and functionality. Certainly, literature exists; however, it tends to generalise penetration testing with a narrow focus (i.e., one or two frameworks). Furthermore, comparative or evaluative literature that is publically available is largely unscholarly, thus this review omitted particular material due to the absence of authoritative, peer-reviewed sources.

Penetration testing as a mitigation strategy is growing rapidly as cyber security concerns grow. Penetration testing courses, certification, and training are now widely available. Consequently, specialised firms are gaining popularity. Although acknowledged as not a new phenomenon, it is still in its infancy in terms of a formal approach. Therefore to conclude, distinctively differentiating between framework and methodology would be of benefit. In addition, an evaluation of currently available penetration testing methodologies and frameworks that take a unified approach acknowledging quality characteristics based on an underlying model would be a significant contribution to the industry as a whole.

### Chapter 3 : Research Methods and Design:

This chapter will discuss the research approach, research method, and research design adopted to facilitate the research. The chapter begins with a discussion on the research approach. Particular attention is paid to the field of information systems predominantly because it is considered a closely related field of study that can be conveniently adapted, if not directly applied, to computer science. Through a process of elimination a suitable research approach is identified. Following the same principal, pertinent methods for the research approach are explored by means of identifying the most appropriate topic area (object of interest), thus, further narrowing the candidates to one research method applicable to this research. Next, the research design is presented detailing various phases employed to conduct data collection and analysis. The chapter then proceeds to discuss limitations of the research, and finally conclude with a summary of expected outcomes.

#### 3.1 Research Approach:

There exists a range of approaches advocated as suitable for research. Galliers (1990) reviewed an array of approaches applicable to the field of information systems and presented a revised taxonomy that assesses strengths and weaknesses of various research approaches. The approaches reviewed by Galliers are categorised as, empirical (quantitative) and interpretive (qualitative). One advantage of Galliers’ model is its direct correlation between topic area (referred to as object of interest) and research methods. The revised taxonomy proposed by Galliers (see table 3.1) is used to identify and guide this research through the process of elimination, thus adopt a suitable research approach.

Table 3.1: Information Systems Research Approaches (Galliers, 1990)

Object	Modes for traditional empirical approaches (observation)					Modes for newer approaches (interpretation)				
	←	→								
	Theorem Proof	Laboratory Experiment	Field Experiment	Case Study	Survey	Forecasting and Futures Research	Simulation and Game/Role playing	Subjective/ Argumentative	Descriptive/ Interpretive (reviews)	Action Research
Society	No	No	Possibly	Possibly	Yes	Yes	Possibly	Yes	Yes	Possibly
Organisation group	No	Possibly	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Individual	No	Yes	Yes	Possibly	Possibly	Possibly	Yes	Yes	Yes	Possibly
Technology	Yes	Yes	Yes	No	Possibly	Yes	Yes	Possibly	Possibly	No
Methodology	No	No	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes
Theory Building	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Theory Testing	Yes	Yes	Yes	Possibly	Possibly	No	Possibly	No	Possibly	Possibly
Theory Extension	Possibly	Possibly	Possibly	Possibly	Possibly	No	No	No	Possibly	Possibly

### 3.1.1 Object of Interest

Table 3.1 presents the suitability of each approach in the context of a particular topic area (object) of study. To recall, the research questions are concerned primarily with methodologies and frameworks that exist in documented form, therefore aims to identify characteristics of frameworks and methodologies to differentiate between them. In addition, the questions aim to evaluate particular frameworks for quality, thus:

RQ 1: How to differentiate between methodologies and frameworks for the purposes of penetration testing

- a) What characteristics define a penetration testing framework?
- b) What characteristics define a penetration testing methodology?

RQ 2: What quality metrics can be defined for penetration testing frameworks?

RQ 3: How can penetration testing frameworks be evaluated using quality metrics?

Clearly, the research is not concerned with opinions, people or places, therefore society, organisational groups, and individual are not appropriate. Likewise, technology does not present as the most appropriate object of interest. Technology is more concerned with the “development of” technology as opposed to evaluation of documented methodologies and frameworks. Turning to the theory categories, theory building is concerned with the extension of theories within a limited knowledge area. Similarly, theory extension is concerned with the improvement on existing theories. This research is not intended to build or extend on any existing theory, moreover aims to evaluate documented frameworks from a practical perspective facilitated by the existing body of knowledge. Comparatively, theory testing is concerned with the development of new knowledge by means of testing existing theories. As a consequence, the theory category of objects is not suitable, therefore eliminated.

One of the key objectives of the research is to explore methodologies and frameworks within the boundaries of penetration testing. Fortunately, Galliers’ taxonomy provides methodology as an object of interest; therefore, given that the research questions are about evaluating methodologies and frameworks, methodology is preferred.

### 3.2 Research Method:

This research will rely on quantitative (empirical) measures of quality for evaluating methodologies and frameworks, as a result all modes interpretivist in nature are eliminated. Therefore, the available approaches (shown in table 3.1) for methodology are limited to field experiment, case study, and survey. Each approach is discussed for applicability.

The objective of field experiment is to use real-world organisations or societies in an attempt to construct an experiment in a more realistic manner, minus the restrictions of a controlled laboratory environment (Galliers, 1990). The aim of this research is to evaluate and test various methodologies and frameworks for quality as opposed to penetration testing for an organisation, therefore field experiment is ruled out.

Surveys aim to capture a snap shot in time using aids such as questionnaires or interviews. The research questions presented are oriented towards testing quality criteria within a real-world context and cannot be answered by opinions of people; therefore they require a more practical approach. Consequently, survey is ruled out leaving the remaining approach, i.e., case study.

The purpose of a case study is to use a real “case” to be studied. One benefit of the case study approach is its ability to capture, in considerably more detail, a complex object (case) in its natural context (Galliers, 1990), therefore case study is more appropriate and closely aligned to answer the research questions presented. The “case” of interest for this research is an evaluation of penetration testing frameworks by means of testing an eAuthorisation system (see section 3.3.1). This research will adopt a real-world project that is suitable and available for penetration testing.

### **3.3 Research Design**

The research design is presented in five phases, as illustrated in figure 3.1. To begin, phase 1 will attempt to answer RQ 1, whereby the terms “methodology “and “framework” appear to be poorly defined. Despite the confusion of terms there are many pen testing methodologies/frameworks available, therefore phase 1 will discuss nomenclature resolution. Phase 1 is a pre-requisite to facilitate the process of performing a gap analysis in phase 2.

Phase 2 will perform a gap analysis on a selection of nine frameworks and methodologies that identify as security testing or penetration testing specific. The gap analysis will use a set of well-defined properties to evaluate characteristics of each framework. The analysis will show a pre-evaluation and post-evaluation categorisation. The analysis will determine if a pen testing framework is actually a framework relying upon nomenclature resolution (phase 1) as the foundation.

Phase 3 will extend upon phase 2 whereby the two most appropriate frameworks are selected to undergo quality evaluation. The analysis of the candidate frameworks will take the form of a

suitability matrix. Frameworks are eliminated by two criteria, suitability and classification. First a candidates' post-evaluation category (see phase 2, gap analysis) should identify as framework or methodology; second, the scope of a particular framework. In other words, a candidate should first identify as a framework or methodology post evaluation and identify as penetration testing specific as opposed to an overall security assessment.

Phase 4 is divided into two parts with the objective of addressing RQ 2 (see table 3.2). First, to identify and define a set of quality metrics, and second, develop an underlying quality model. Prior to undertaking quality evaluation a set of quality metrics need to be established, thus phase 4 defines and details a candidate set of metrics to form the foundation for an underlying quality model. The initial candidate metrics are largely drawn from a review of four existing quality models in the literature, combined with ISO/IEC 25010. The candidate metric set are namely; extensibility, maintainability, domain coverage, usability, availability, and reliability. Taking into consideration the differences that exist among frameworks and methodologies, it is expected that re-examination of the candidate set could occur; consequently, repeatedly refining the metric set. This phase concludes with a proposed underlying quality model suitable for evaluating the two selected frameworks.

The final phase (phase 5) entails multiple penetration tests conducted against a target system using two frameworks selected from phase 3. The objective of the penetration tests is to identify potential vulnerabilities that exist, in addition, attempt to measure quality of each framework for their real-world application in research or practice. Fortunately, Edith Cowan University (ECU) has available a suitable project, Authentication and Authorisation for Entrusted Unions (AU2EU). AU2EU consists of twelve partners across Europe and Australia and is funded under the European Seventh Framework Programme (FP7). The aim of the project is to design and implement an eAuthorisation and corresponding eAuthentication platform to allow delivery of services across different organisational or governmental jurisdictions (Dhouha et al., 2014). The AU2EU project consists of two pilot systems, the German Red Cross (DRK), and CSIRO. Each consists of differing security domains and associated access policies. From the outset, it is intended that the two case studies undergo penetration testing in order to provide cross-validation. Under these circumstances, and in combination with the case study approach, the research questions presented are addressed.

### **3.3.1 Case Study Description**

Case Study 1: The DRK case study is an e-Health collaborative assistant living pilot. The German Red Cross provide regional emergency and social services to patients. New clients are entered into

the system via a web application interface followed by a hardware installation of the emergency-buttons, sensors, and communication device in the patients’ home. A web interface is available for on-site care providers to login and register new clients. The web interface is accessible via mobile devices and/or laptop computers.

Case Study 2: The CSIRO case study is a bio-security incident management system and considered an important mechanism to sustain Australia’s agricultural productivity and export trade that relies heavily on a disease free status (Dhouha et al., 2014). When a disease outbreak occurs the Consultative Committee of Emergency Animal Disease (CCEAD) collaborate to determine research and diagnostic information strategies for dealing with the emergency, thus the collaboration involves multiple groups from various locations and sectors whereby sensitive information must be shared in a timely manner.

Table 3.2: Research Question Phase Mapping

<b>Research Questions</b>	<b>Related Phase</b>
RQ 1:	Phase 1
RQ 1a)	Phase 2
RQ 1b)	Phase 2
RQ 2:	Phase 4
RQ 3:	Phase 5

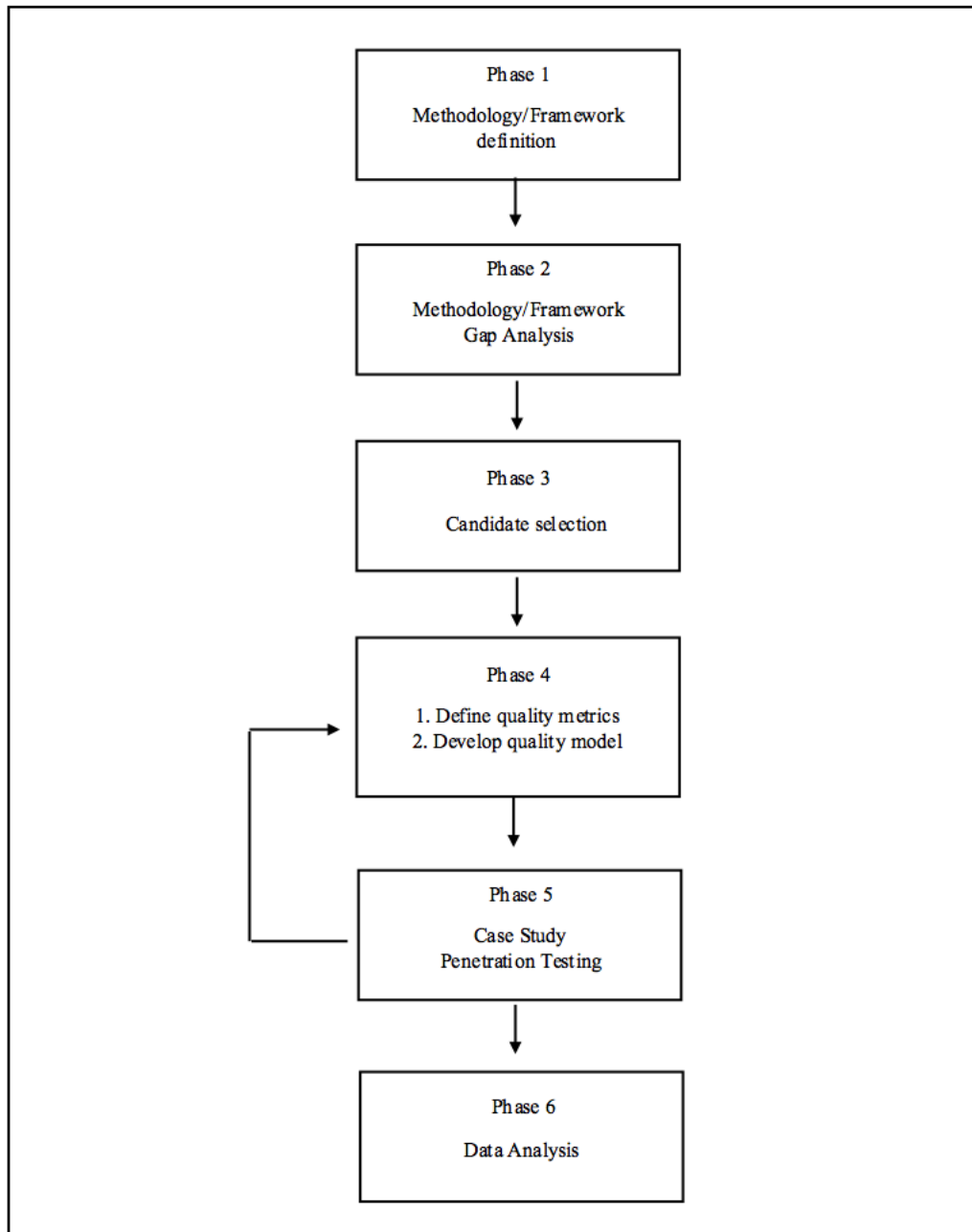


Figure 3.1: Research Design

### 3.4 Materials:

Materials used for the research will consist of a computer capable of running Kali Linux, a known Linux distribution pre-packaged with penetration testing software. In addition, documentation sourced from the originating providers of the selected methodologies and frameworks.

### 3.5 Data Analysis:

Data analysis will entail three primary areas of discussion. The first area does not adopt quantitative measures, moreover an analysis of existing definitions from authoritative sources concerning definition of terms, that of “methodology” and “framework”. The discussion will centre on

methodology and framework characteristics that closely align with documented processes, primarily focused on penetration testing drawn from the existing body knowledge. The discussion is included as part of the initial data analysis to show how framework and methodology characteristics are identified, thus, subsequently used in the proceeding gap analysis whereby framework and methodology characteristics are categorically defined. The objective is to come to a defensible conclusion as to what could effectively be referred to as a penetration testing framework or penetration testing methodology by means of a gap analysis. The second area of discussion is concerned with defining a set of suitable quality metrics. Subsequently, a penetration testing quality model is proposed whereby measurements of quality can be made and analysed. Finally, data analysis will involve assessing two nominated frameworks for quality based on the proposed quality model (see section 4.3.2) during a penetration test. Quality metrics and associated sub characteristics will be explicitly defined coupled with a quantitative approach to measurements that can be directly applied. This entails measuring certain quality characteristics against each framework respectively.

Analysis of the data will involve the interpretation of results derived from penetration tests performed against the AU2EU project (eAuthorisation system). The two frameworks nominated for testing are evaluated for quality based on an underlying model coupled with appropriate measures of quality characteristics. In addition, the steps performed to reach a conclusion on a particular framework are detailed accordingly. The analysis aims to assess the suitability of each framework for its application to penetration testing from conception to completion in practice or research while maintaining a primary focus on quality attributes that can potentially be measured using quantitative means.

### **3.6 Limitations:**

This research will make use of the AU2EU project (see section 3.3). The AU2EU project has available two eAuthorisation and eAuthentication pilot systems (DRK and CSIRO), suitable for penetration testing. The research is limited to two case studies, nevertheless it is intended the case studies will provide cross-validation and provide significant insight into vulnerability assessments and/or exploitation due to the real-world application of an eAuthorisation system. Furthermore, the data analysis will present a valuable resource that can be adapted within other penetration testing domains. Additional limitations identified are; the number of quality metrics selected for theoretical analysis, and, the number of frameworks selected to undergo penetration testing. With reference to quality variables, a limitation of six is considered sufficient for theoretical analysis, however a revision of these variables is possible. Next, frameworks nominated to undergo penetration testing



are limited to two, taking into consideration the complexity of performing a penetration test in its entirety.

### **3.7 Summary**

This chapter presented research methods and design suitable for answering the research questions articulated. Several research approaches were considered based on Galliers' taxonomy. Through a process of elimination an object of interest (that of Methodology), was identified. Following this, the Case Study method was selected as appropriate for this research. Combining the Methodology object and Case Study method, a quantitative approach was defined allowing the research design to be developed. The research design was partitioned into a five-phase process whereby each phase was mapped directly to the research questions. Finally, potential limitations of the research design are noted.

## **Chapter 4 : Analysis and Discussion**

This chapter consists of six sections that discuss in detail the research topic of this thesis, as a consequence answering the research questions presented. The chapter begins with nomenclature resolution (section 4.1). Penetration testing methodologies and frameworks appear to be poorly defined; despite this confusion of terms there are many penetration testing methodologies/frameworks available. Nomenclature resolution attempts to answer research question one (RQ1), which asks how to differentiate between framework and methodology for the purpose of penetration testing. Section 4.1.1 and section 4.1.2 extend upon nomenclature resolution by categorising penetration testing artefacts based on characteristic analysis. The characteristics analysis proposes a framework vs. methodology matrix offering remedy for RQ1-a and RQ1-b accordingly. Following this, a gap analysis is performed in section 4.2 on nine penetration testing methodologies and frameworks to determine if a penetration testing framework is actually a framework, i.e., it has a sound underlying ontology. One primary objective of the research is to evaluate two chosen frameworks for quality, however prior to any worthwhile evaluation, quality metrics must be established, therefore research question two (RQ2) seeks to discover what quality metrics can be defined, thus quality model analysis is performed in section 4.3.1 and consecutive sub-sections. Section 4.3.2 builds upon this and proposes six quality metrics that subsequently form part of an underlying quality model in preparation for the two selected frameworks to undergo evaluation. Next, section 4.4 evaluates the two selected frameworks adopting the proposed quality model that ultimately will determine their suitability for real world applications, offering remedy for the final question (RQ3); can penetration testing frameworks be evaluated using quality criteria. Finally, a discussion of results in section 4.5 is presented. The chapter concludes with section 4.6 whereby amendments to the research environment are discussed.

### **Case Study Amendments**

Initially it was intended that two implementations of an e-Authorisation and eAuthentication system undergo penetration testing. Regrettably, due to circumstances outside the control of the research, testing was not completed as originally planned, which appears to be a risk when dealing with live, real-world cases. To reintroduce the AU2EU project, the two e-Authorisation implementations (case studies) consisted of the German Red Cross (DRK), and CSIRO. As will be elaborated on further in section 4.6, the CSIRO implementation was not tested, therefore, it was substituted with a virtual implementation provided by IBM (another partner organisation in the aforementioned AU2EU project). IBM was the original source for the two case studies, therefore a valid substitute for this research. This chapter frequently refers to the IBM implementation and the DRK implementation; furthermore, results obtained throughout this chapter are drawn from DRK and IBM, omitting the

CSIRO implementation originally proposed.

## **4.1 Nomenclature Resolution**

Nomenclature resolution attempts to answer RQ1, to recall, RQ1 asks how to differentiate between methodologies and frameworks for the purposes of penetration testing. Very little research has been undertaken to determine the difference between the two terms relating specifically to penetration testing, however discussions and definitions do exist on each term in isolation. Avison and Fitzgerald (2006, p. 418) discuss in detail the loose but extensive use of the term “methodology” and argue that there is little agreement as to what it means other than at a very general level. Consequently, the difference between penetration testing methodologies and frameworks appears to be poorly defined. Despite this confusion of terms many penetration testing methodologies/frameworks are available. The approach taken was to first review literature and definitions already available then, subsequently extract characteristics of each term in isolation, thus forming a baseline of common properties for each term.

### **4.1.1 Defining Methodologies and Frameworks for the purpose of pen testing**

Although the literature provides definitions for methodologies and frameworks in various contexts, primarily information systems, there appears to be a disparity when relating these terms to penetration testing. For instance, Avison and Fitzgerald (2006) define a methodology as "a collection of procedures, techniques, tools and documentation aids, which will help the systems developers in their efforts to implement a new information system. A methodology consists of phases, themselves consisting of sub phases, which will guide the systems developers in their choice of the techniques that might be appropriate at each stage of the project and also help them plan, manage, control and evaluate information systems projects". The Oxford Dictionary defines a framework as “An essential supporting structure of a vehicle or object”, or more software focussed; “a basic structure underlying a system, concept, or text” ("Framework", 2015). Comparatively, Johnson (1997) states that a framework is the skeleton of an application that can be customised by an application developer. In addition Johnson further states that, “a framework is a reusable design of all or part of a system that is represented by a set of abstract classes and the way their instances interact”. From an application development context, Land (2002) defines frameworks as a skeleton from which software applications can be built. For penetration testing however, a software application is typically not the final product; moreover the final product is generally a report detailing results of a series of tests that subsequently is presented to the client. In contrast, Wilhelm (2009) asserts that frameworks focus primarily on processes necessary to achieve results and methodology encapsulates processes, activities, and tasks.

Considering common properties of the above definitions, a penetration testing methodology can be defined as; a collection of procedures, techniques, tools, and documentation aids which assist the penetration tester in their effort to perform penetration testing in its entirety. A penetration testing methodology consists of phases themselves encapsulating sub-phases that will guide security practitioners in their choice of techniques that might be appropriate at each stage of a project. It may further detail how to perform a particular task providing tools and techniques, additionally, assist with planning and management control as part of the methodology. Methodology is concerned with the “how” and “why” things are done. Similarly, a penetration testing framework can be defined as, a reusable shell or skeleton whereby new methodologies can be added in order to tailor them for new requirements. A framework can be considered a reusable template or outline for penetration testing that may take the form of documentation, software or, a combination of both. Framework is concerned with “what” needs to be done and allows the security practitioner to decide “how”. Based on the aforementioned definitions, the following factors outlined in table 4.1 are considered when classifying a particular methodology or framework.

Table 4.1: Factors for Classification

Factors for frameworks	Factors for methodologies
Does it describe what to do opposed to how to do it?	Does it describe how processes are done?
Is it a template or skeleton that can be reused to cater for new requirements?	Is it based on underlying principles or philosophy?
Can new processes be added?	Does it contain a collection of procedures, tools and techniques? Does it contain phases and/or sub phases?
Is it a written document or software based?	Does it aid in reporting, planning, management and control of a penetration test?
Is it flexible?	Is it repeatable?
Is it reusable?	

#### 4.1.2 Nomenclature Definition

In combination with factors outlined in table 4.1, framework and methodology characteristics (see table 4.2) show a summary of differences between a framework and methodology, thus each framework or methodology discussed in the following sections is classified according to these characteristics.

Table 4.2: Framework vs. Methodology Characteristics

Characteristics	Framework	Methodology
Flexible / adapt to new circumstances	Yes - descriptive	Possibly - prescriptive
Tools and Techniques	possibly	yes
Describes what to do	yes	possibly
Describes how and why	No	yes
Based on an underlying philosophy or principles (objectives)	Yes	yes
Can embed other methodologies	yes	no
Provides planning, management and control	yes	yes
Provides phases and/or sub phases	yes	yes
Provides outputs/reports	possibly	yes
Repeatable processes and procedures	possibly	yes

## 4.2 Gap Analysis

This section will describe a theoretical review of nine penetration testing methodologies and/or frameworks with the aim of classifying each respectively into an appropriate category, in particular, that of methodology or framework. The review will form of an overview whereby three primary areas are discussed. First, the classification of each framework and/or methodology provided by the authors; second, an overview of the framework and its purpose. Finally, highlight characteristics that either agree or disagree with its current classification. The outcome of the review will present a gap analysis matrix (see table 4.3) of the nine frameworks and methodologies reviewed showing pre-evaluation and post-evaluation classification. The gap analysis matrix consists of six categories, namely: standard or guide, framework, methodology, application suite, manual or resource. Following the results of the gap analyses, two suitable frameworks are selected to undergo further evaluation in section 4.2.3.

## **4.2.1 Theoretical Review**

### **4.2.1.1 ISSAF**

The Open Information Systems Security Group (OISSG) describes the ISSAF as a framework. The penetration testing methodology embedded within ISSAF is partitioned into three phases that include: planning and preparation, assessment, reporting and clean up (OISSG, 2005). The remaining sections focus on a multitude of domain areas. Domain coverage is comprehensive, including, but not limited to: password security testing, switch and router security, firewall security, intrusion detection systems, assessments, VPN security, web application security, and windows security. Although not all domains are listed here, it is suffice to say that the ISSAF attempts to cover an exhaustive list of domains to cater for all possible scenarios. It is worthy to reiterate however, that the documentation has not undergone review since 2006, thus particular information is considered out-dated and not directly applicable to certain technologies and/or threats emerging today. Embedded within these domains are methodologies themselves, whereby, processes, tools and techniques are discussed in detail. For example, the planning and preparation phase embeds a passive information gathering methodology that consists of its own phases, tools and techniques. The information gathering methodology (planning and preparation phase), explains how and why certain processes need to occur coupled with screen shots and examples of how to use certain tools. This structure is repeated throughout the ISSAF. In summary the ISSAF is classified as a framework that encapsulates various methodologies. The document portrays framework characteristics whereby it displays a well-structured template that can be adapted to cater for a variety of environments. In addition, ISSAF provides phases and sub phases that encapsulate methodologies that aid in the planning and management of a penetration testing project. ISSAF covers the “what” and “why” within its embedded methodologies, furthermore, the framework that encapsulates these methodologies presents as flexible and proficient.

### **4.2.1.2 Open Source Security Testing Methodology Manual (OSSTMM)**

The OSSTMM is described as a manual that contains a methodology. The OSSTMM is foremost a security auditing methodology as opposed to a penetration testing methodology. The document purpose states; “The primary purpose of this manual is to provide a scientific methodology for the accurate characterisation of operational security (OpSec) through examination and correlation of test results in a consistent and reliable way” (ISECOM, 2000). Initially, the characteristics appear to be more closely aligned to that of a framework. Prior to 2005 the OSSTMM was considered a framework as claimed by the authors, thus “Since its start at the end of 2000, the OSSTMM quickly grew to encompass all security channels with the applied experience of thousands of reviewers. By 2005, the OSSTMM was no longer considered just a best practices framework. It had become a

methodology to assure security was being done right at the operational level. As a methodology it is designed to be consistent and repeatable” (ISECOM, 2000). It could be argued that the OSSTMM is a framework by means of providing the “what” processes and tasks need to be followed rather than “how” processes and tasks are completed, however, this could be the case if it were considered a penetration testing manual foremost. In the context of security auditing it closely aligns with methodology. The OSSTMM does not describe how to perform a task but it does provide enough detail for advanced security professionals to determine appropriate tools to use during a security audit. Additionally, each phases consists of sub-phases that guide the security practitioner in their efforts to complete a particular phase, aiding in the planning, management and reporting, therefore exhibit more methodological characteristics.

#### ***4.2.1.3 NIST 800-115***

NIST 800-115 is defined as a technical guide for information security testing and assessment. The purpose of the guide is to provide information on planning and conducting information security assessments The guide further outlines how to analyse findings and develop mitigation strategies for organisations (NIST, 2008). At a minimum, NIST 800-115 recommends that organisations should: establish an information security assessment policy, implement a repeatable and documented methodology, determine objectives for each assessment, and analyse findings to develop risk mitigation techniques. It contains eight sections, each of which contain sub-sections that follow a well-defined structure that present technical testing and examination methods. The format for each section is consistent offering the “what” and “why”, however, the “how” is determined by the security practitioner. Each section outlines a table listing of techniques coupled with the skill set required to implement certain tasks. Penetration testing is not covered extensively, thus NIST 800-115 could not be used primarily for penetration testing. In summary NIST 800-115 is as the name suggests, a Technical Guide for assessing the security posture of an organisation. It does not offer enough detail to be considered a penetration testing framework or methodology, moreover, an overview of an information security assessment. The document could be considered a framework for security assessments mainly because it portrays various framework characteristics, primarily its structure and skeleton properties. In reference to penetration testing, it could only be used as a basis to create a framework with the option of embedding a methodology; therefore it is classified as a standard or guide.

#### ***4.2.1.4 OWASP Testing Guide (OTG)***

OTG is defined as a guide that aims to create a standard for securing web applications, i.e., create web applications with security in mind rather than address security concerns after the application is in production. In addition, the testing guide attempts to provide a consistent and repeatable

approach to testing web applications (OWASP, 2014a), in other words, penetration testing web applications. OTG consists of five sections. The first two sections cover the OWASP project and an introduction to security testing. Section three, OWASP Testing Framework, provides a basic framework for web application development addressing web application security. It does not discuss the “how” rather, what needs to be completed throughout the design, development, and deployment process of web applications. Section four, Web Application Security Testing, details a web application testing methodology whereby processes, tools and tasks are documented that relate to penetration testing web applications. Section four is detailed enough to be considered a methodology inasmuch as it discusses “how” and “why”, thus sets out to solve a particular problem providing necessary tools and techniques. In addition, section five (Reporting) provides a framework for reporting processes. In summary, the OWASP Testing Guide displays framework characteristics primarily because it encapsulates two frameworks that present a skeleton or template coupled with explanations of requirements, that of section three (OWASP Testing Framework), and section five (Reporting). Additionally, it encapsulates a web application testing methodology (section four) that details tools, techniques, and processes to solve a specific problem. OWASP Testing Guide is therefore classified as a framework with an embedded methodology.

#### ***4.2.1.5 Building Security in Maturity Model (BSIMM)***

BSIMM is a software security framework (SSF) that originated by studying and quantifying security initiatives of sixty-seven successful firms with the aim of helping organisations plan, carry out, and measure security initiatives independently (McGraw et al., 2009). BSIMM is concerned with what organisations actually “do” in the real world compared to the more generic approach of what organisations “should” do. The SSF embedded within BSIMM consists of 112 activities divided into twelve practices that support four domains, mainly: governance, intelligence, SSDL touch points, and deployment. BSIMM displays framework characteristics by means of its flexibility. The practices recommended allow the security practitioners or business executives to decide on how activities will be carried out and by whom. The framework is designed to enable organisations the freedom to adapt it to business requirements. The SSF is not a penetration testing framework, moreover, it is a security assessment framework that includes penetration testing as one activity of the overall security initiative. Techniques and tools are not detailed, thus the SSF is a skeleton (framework) that organisations can implement.

#### ***4.2.1.6 Penetration Testing Framework (PTF) 0.59***

PTF is defined as a penetration testing framework. The framework is concerned with network foot printing, enumeration, password cracking, and vulnerability assessment (Lawson et al., n.d). It also covers specific domain areas, for instance; wireless penetration testing, physical security, blue tooth



specific testing, and VoIP services. Each section details the recommended tools and techniques, however, the security practitioner is expected to possess the knowledge and skill for each test. In addition templates and checklists are provided for documentation and/or reporting. Information outlined in the PTF is out-dated, for example; out-dated tools and broken links. Therefore, the framework could not be considered complete or up-to-date. The structure lacks consistency and does not appear to show framework or methodology characteristics, i.e., processes and phases are not clear. Flexibility is questionable due to its lack of cohesion and loose structure; therefore it is classified as a useful resource that has the potential to be further developed into a framework.

#### ***4.2.1.7 Penetration Testing Execution Standard (PTES)***

PTES is defined as a penetration testing standard. It was originally created by a group of information security professionals who felt there was a lack of quality and guidance within the cyber security testing industry (Nickerson et al., n.d). The goal was to provide both business and security providers a common language and scope for performing penetration tests. The primary goal was to establishing a baseline that would define the boundaries of a penetration test. In other words, what is the minimum set of tests, processes and outputs that should be employed before a penetration test could be considered professional and complete. PTES includes: pre-engagement interactions, intelligence gathering, threat modelling, vulnerability analysis, exploitation, post exploitation, and reporting. An extensive list of tools and techniques are set out in each of the seven sections accompanied by a description. Screen shots and links to resources are detailed throughout the document. PTES is not complete as of this evaluation and could not be implemented in its current state. Methodology is excluded due its lack of key characteristics, primarily the “how” and “why”, and incomplete documentation. Framework characteristics do exist, for example; skeletal structure, flexibility, and additionally describes “what” is required for each phase. Overall PTES presents as reusable and well structured, however, the absence of primary framework characteristics suggest it more closely aligns with a resource, therefore could not be considered a framework or methodology without undergoing further development.

#### ***4.2.1.8 Metasploit Framework (MSF)***

MSF is one of four Metasploit editions now supported and maintained by Rapid7, a commercial company offering cyber security products. MSF is a platform for writing, testing, and using exploit code to deliver payloads to target systems for the purposes of penetration testing. MSF is a command line interface with an underlying module-based architecture. It is built with extensibility in mind by permitting different modules (software based) to perform different tasks, thus new modules can be added (Holik et al., 2014). The basic process for security practitioners is to first choose an exploit that exists on the target system. Next, determine if the target is vulnerable to the

exploit, followed by selection and configuration of a payload before attempting to exploit the target. Specific information about the target is required; therefore third-party tools are often used in combination to obtain certain information, i.e., the target operating system, open ports, and/or potential vulnerabilities. MSF is considered a suite of tools in relation to penetration testing. It could be argued that MSF is a framework from a software application perspective, however, the focus of this research is penetration testing frameworks, thus the evaluation is viewed from this context. MSF is classified as an application suite, a practical solution that provides an array of tools to facilitate a penetration test rather than a documented outline of process and methods to follow. MSF is excluded from the methodology category for obvious reasons, primarily because it is not a document containing processes and phases with a set of underlying principles. Furthermore, it is void of the “how” and “why”. Exclusion from the framework category is somewhat more difficult due to its flexibility and extensibility from an application framework perspective as mentioned previously. In reference to penetration testing framework characteristics, MSF cannot be relied upon alone to complete a penetration test, moreover utilised as a recommended tool. The project aspect of penetration testing is not addressed, thus implementation of some framework or methodology prior to using MSF would be an advantage; therefore classified as an application suite that can facilitate a penetration test.

#### ***4.2.1.9 Browser Exploitation Framework (BeEF)***

BeEF is an open source browser exploitation framework used primarily as a tool to detect and exploit browser based vulnerabilities. BeEF comes packaged with a web interface, console application interface, and metasploit integration (Alcorn et al., 2012). Similarly to other frameworks, BeEF encompasses information gathering, social engineering, and network discovery as packaged software modules. BeEF provides the experienced security practitioner the means of assessing the security posture of an environment by using client side attack vectors. BeEF is excluded from the methodology category naturally because it does not document processes and methods to follow or provide an underlying set of principles, of which are considered two key characteristics of methodology. Similarly to MSF, it could be argued that BeEF is a framework from a software standpoint, however, to reiterate, software frameworks are not the focus. For the purposes of penetration testing, framework is ruled out primarily because it is an off the shelf ready to use product, thus will facilitate a penetration test for one domain area, that of web application using web based browsers. Furthermore, BeEF does not suggest the “what” or provide guidelines to follow although documentation is available by means of a wiki. Nonetheless, not considered a substitute for processes and procedures. In addition, flexibility is questionable of which are considered key characteristics of a framework.

### 4.2.2 Gap Analysis Matrix

Table 4.3 shows a summary of the nine penetration testing methodologies and/or frameworks discussed in the theoretical review (section 4.2.1). Original classification (pre-evaluation) and post evaluation classifications are shown. The classifications illustrated in table 4.3 are ordered according to definitions, for example; a framework encapsulates a methodology and methodology encapsulates tools, techniques, and resources. In addition, frameworks that embed methodology are identified.

Table 4.3: Gap Analysis Matrix

	Standard or Guide	Framework	Framework encapsulates Methodology	Methodology	Application Suite	Manual or Resource
ISSAF		* +	yes			
OSSTMM	*			* +		*
NIST 800-115	* +					
OWASP Testing Framework	*	+	yes			
BSIMM		* +				
PTF 0.59		*				+
PTES	*					+
Metasploit Framework		*			+	
BeEF		*			+	

Legend	
Pre-evaluation Classification	*
Post-evaluation Classification	+

### 4.2.3 Candidate Selection

From the gap analysis it was possible to develop a taxonomy of penetration testing specific frameworks or methodologies whereby suitable candidates can be identified to facilitate their use in research or practice. It was found that some frameworks were indeed frameworks in the accepted sense of the word, whereas others were simple collections of tools without a discernible underlying ontology. This is considered an important distinction because novice penetration testers would benefit from the additional support provided by a mature framework.

Table 4.4: Classification of Penetration Testing Frameworks and Methodologies

	Classification			Suitability	
	←				→
Candidate	Framework	Methodology	Other	Penetration Testing Specific	Security Assessment
ISSAF	✓			✓	
OSSTMM		✓	✓		✓
NIST 800-115			✓		✓
OWASP Testing Framework	✓			✓	
BSIMM	✓				✓
PTF 0.59			✓	✓	
PTES			✓	✓	
Metasploit Framework			✓	✓	
BeEF			✓	✓	

The next step, selection of two candidates suitable for quality evaluation, is determined by two criteria. First, whether or not a particular candidate classified as either framework or methodology. Candidates that fall under a methodology or framework were deemed in-scope, thus eliminating all other candidates. The second criterion was to examine the boundaries of a particular candidate for its scope, in other words, whether or not a particular candidate is focused entirely on penetration testing as opposed to an overall security assessment. The research being undertaken has a primary goal of evaluating penetration testing methodologies and frameworks explicitly rather than assessing the security posture of an organisation in its entirety; therefore candidates that are categorised as penetration testing explicitly are preferred over security assessment specific candidates. As a result, the two remaining candidates, ISSAF and OTG, are selected to undergo further evaluation.

### 4.3 Quality Metrics and Evaluation

Research question two (RQ2) asks, what quality metrics can be defined for penetration testing frameworks? To adequately answer the research question presented, it is first necessary to define a suitable set of quality metrics applicable to penetration testing frameworks. To address the question, this section is divided into two themes. Section 4.3.1 defines quality metrics applicable to penetration testing and section 4.3.2 adopts the ISO/IEC 25010:2013 product quality model (Standards Australia, 2013) as a foundation whereby an amended model is proposed.

The Institute of Electrical and Electronics Engineers (1990) defines quality as; the degree to which a system, component, or process meets specified requirements, and, the degree to which a system, component, or process meets customer or user needs or expectations. Similarly Standards Australia (2000) defines quality as, “the totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs”. It is postulated that penetration testing frameworks are a product, thus should exhibit quality characteristics to some degree, however, measures of quality differ considerably (Avison & Fitzgerald, 2006). To recall, section 2.4.6 showed the most common quality characteristics across a range of models, for convenience table 4.5 reintroduces the models reviewed.

Table 4.5: Quality Metric Comparison

	Mc Call	Boehm	Dromey	ISO9126	ISO25010	Total
Contextual			*			1
Correctness	*		*			2
Descriptive			*			1
Efficiency	*	*	*			3
Flexibility	*					1
Functional			*	*	*	3
Integrity	*	*				2
Internal			*			1
Interoperability	*					1
Maintainability	*	*	*	*	*	5
Modifiability		*		*	*	3
Operability	*			*	*	3
Portability	*	*	*	*	*	5
Reliability	*	*	*	*	*	5
Reusability	*		*	*	*	4
Security				*	*	2
Testability	*	*		*	*	4
Understandability		*				1
Usability	*			*	*	3

As previously stated in chapter 2, table 4.5 shows that maintainability, portability, and reliability are common among the five models reviewed. Therefore they are considered significant quality metrics, followed closely by reusability and testability. While it can be postulated that these metrics are deemed most suitable, the models reviewed are primarily used to evaluate software quality. The relative importance of quality characteristics will depend on the high-level goals and objectives for the project. Standards Australia (2013) asserts that not all quality characteristics will be relevant for every scenario. Relevance (to penetration testing) should be considered before finalising a set of quality characteristics to establish evaluation criteria. Therefore the quality characteristics outlined have been adapted to align more closely with penetration testing. The candidate set of quality metrics used to form the basis of the underlying quality model are: extensibility, maintainability, domain coverage, usability, availability, and reliability. Extensibility, although not outlined in table 4.5 explicitly, is considered of particular relevance within the penetration testing domain for reasons of updateability. Extensibility draws on quality metrics outlined in table 4.5 to form sub-characteristics that further support its role in quality evaluation. Extensibility could be considered closely related to maintainability, however, it deserves attention in isolation due to the ongoing changing nature of cyber threats. Ideally frameworks should have the capacity for extension due to the increased frequency of emerging threats (Australian Crime Commission, 2013), therefore provide the flexibility to respond to new threats or vulnerabilities as they emerge. Similarly, domain coverage and availability are included as primary characteristics within the metrics set. Domain coverage within penetration testing frameworks is one primary reason a framework may or may not be selected for use, i.e., OWASP Testing guide is explicitly focused on the domain area of web applications, for this reason, could include or exclude its suitability for a particular project. For this reason, domain coverage is relevant and of high significance in this context. Likewise, availability is concerned with the accessibility of documentation, and in some cases, third party tools required for project completion, thus it encapsulates four sub-characteristics (discussed in section 4.3.2) that draw from table 4.5 to support its function. Usability on the other hand, is not considered of high importance among the models reviewed in table 4.5, however, for documented frameworks usability is significant from a security practitioner perspective. Usability encapsulates four sub-characteristics appropriate for documented frameworks, among them, readability and learnability, specifically aimed at measuring documented frameworks. Consequently, its level of importance is elevated and therefore included as a primary level metric. Finally, in agreement with table 4.5, maintainability and reliability are included as primary characteristics in the metrics set.

#### **4.3.1 Quality Metrics Defined**

This section defines top-level quality metrics that form the basis for the Penetration Testing

Framework (PTF) Quality Model detailed in section 4.3.2. To reiterate, the selected metrics are: extensibility, maintainability, domain coverage, usability, availability, and reliability. Each metric encapsulates one or more sub-characteristics that are discussed in further detail in section 4.4, Framework Quality Evaluation.

#### ***4.3.1.1 Extensibility***

Laplante (2001, p. 173) defines extensibility as “the capacity of a system, component, or class to be readily and safely extended in its behavioral specification/operational and/or structural capabilities”. Likewise, the IEEE describes expendability or extensibility, as the ease with which a system or component can be modified to increase its functional capacity (Institute of Electrical and Electronics Engineers, 2011). Within a framework context, extensible pertains to the capacity of the framework to be readily and safely extended with minimal effects on its structure and have the ability to accommodate changes to meet new requirements with minimal impact.

#### ***4.3.1.2 Maintainability***

From a software context, to maintain is the ease in which it can be understood, corrected, adapted and/or enhanced (Laplante, 2001, p. 293). Sommerville (2007) defines maintainability as “software that can be adapted economically to cope with new requirements and where there is a low probability that making changes will introduce new errors into the system”. The Institute of Electrical and Electronics Engineers (2011) asserts that maintainability is the ease in which a software component can be modified to change or add capability. With reference to maintainable penetration testing frameworks, maintainability is primarily concerned with the amount of effort required to keep the system running satisfactorily and continuing to meet changing requirements over its lifetime (Avison & Fitzgerald, 2006). For example, new and emerging technologies can be added and out-of-date practices removed. In other words, the document or software undergoes regular updates and revisions.

#### ***4.3.1.3 Domain Coverage***

Generically, a domain can be defined as “a sphere of control, influence or concern” (Oxford, 2008). Within a penetration testing context, domain coverage is concerned with a particular area of interest for the purposes of a security assessment, i.e., wireless, mobile phone, internal network, or web application. The scope and business objectives will likely determine the domain areas that will undergo a penetration test. Particular domain areas require the ability to be included or excluded depending on business requirements without impacting the overall outcome of a penetration test. Domain coverage as a quality metric is concerned with the coverage of domain areas the framework supports.

#### **4.3.1.4 Usability**

Usability refers to the ease of use and efficiency with which a user can learn to operate, prepare inputs, and interpret outputs of a system component (Institute of Electrical and Electronics Engineers, 2011). Usability primarily relates to the initial effort to learn and reoccurring effort to use and interpret the functionality of software (Deutsch & Willis, 1988). When relating usability to penetration testing frameworks, the documentation or software should be easy to use and interpret, thus require minimal learning time and increase productivity.

#### **4.3.1.5 Availability**

Availability is concerned with the degree to which a system or component is operational and accessible when required for use (Institute of Electrical and Electronics Engineers, 2011). The frame of reference aligns closely with software engineering whereby uptime and downtime are considered. In the case of documented frameworks availability is not about uptime, moreover, the effort required to obtain a particular framework and/or the software tools that it recommends. Put more simply, the framework of choice used to facilitate a penetration test should be obtainable throughout the life of the project. In addition, the recommended software should also be accessible to perform its required function over a stated period of time.

#### **4.3.1.6 Reliability**

Reliability deals primarily with the rate of failures in software that render it useable (Deutsch & Willis, 1988). The Institute of Electrical and Electronics Engineers (2011) defines reliability as the ability of a system or component to perform its required functions under stated conditions for a specified period of time. Avison and Fitzgerald (2006) discuss reliability with a focus on methodologies in particular, thus assert that error rates should be minimised and outputs are consistent and correct. Similarly, a direct measure of reliability in relation to penetration testing frameworks, considers the error rate whereby false positives are minimal and the outputs are consistent enough to render it useable and consequently, repeatable.

### **4.3.2 Quality Model**

Figure 4.1 shows a generic quality model. Standards Australia (2013) clearly state that “It is not practically possible to specify or measure all sub-characteristics for all parts of a large computer system or software product. Similarly, it is not usually practical to specify or measure quality in use for all possible user-task scenarios. The model should therefore be tailored before use as part of the decomposition of requirements to identify those characteristics and sub-characteristics that are most important, and resources allocated between the different types of measure depending on the stakeholder goals and objectives for the product.” Therefore, the ISO model has been amended to



focus less on software quality evaluation and more on aspects of penetration testing framework evaluation (see figure 4.2).



Figure 4.1: Abstract Quality Model (adapted from ISO/IEC 25010:2013)

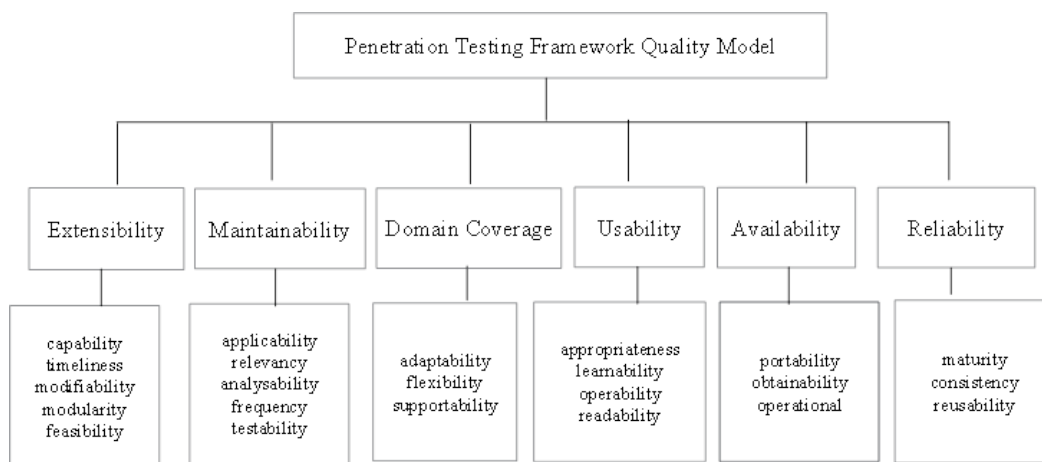


Figure 4.2: Penetration Testing Quality Model adapted from (ISO/IEC 25010:2013)

#### 4.4 Framework Quality Evaluation

The Penetration Testing Quality Model (PTF Quality Model) proposed in figure 4.2 is composed of six top-level characteristics and supporting sub-characteristics that are relevant to aspects of penetration testing frameworks. The scope of the model is designed to provide a foundation whereby quality can be evaluated. It is not practically possible to measure all characteristics outlined in the model, thus certain characteristics are quantified while others are simply observed. This section will examine each top-level characteristic and supporting sub-characteristics detailed within the PTF Quality Model with the aim of evaluating the two selected frameworks, ISSAF and OTG, for quality.

#### 4.4.1 Measuring Extensibility

To recall, the definition of extensibility can be described as, the ease with which a system or component can be modified to increase its functional capacity (Institute of Electrical and Electronics Engineers, 2011). When referring to penetration testing frameworks, an extension can be in the form of additional tests to cater for new vulnerabilities as they are discovered, or, a request for change to reflect new and emerging vulnerabilities. Measuring extensibility depends on multiple factors. First, whether or not the framework can be extended in any capacity. For instance, it is possible that a framework may not accommodate modifications in any form; under these circumstances the framework is not considered extensible. The second factor of significance is the process involved for a potential extension to be implemented, put another way; do the authors cater for extensibility in a timely manner? If an extension request impacts the project timeline then the frameworks efficiency is reduced. Third, relevancy relates to document content detailed within the framework and can be evaluated analytically. For instance, it would be of no benefit for security practitioners employing a particular framework to test a system with outdated information, as is the case with ISSAF. ISSAF documents testing of outdated servers such as Novell Netware, although support ceased in May 2009 (Novell, 2015). There are numerous security tests contained within the ISSAF considered outdated, and in some cases, no longer pertinent, consequently affecting the framework's relevancy. To reiterate, ISSAF has not undergone document updates since 2006, thus it is no surprise that particular information is not relevant. Although relevancy is not quantified statistically, its existence should be obvious. The final factor taken into consideration is whether the extension impacts the overall structure of the framework negatively, in other words, modularity facilitates ease of updates without the requirement for document restructure. Ultimately it is up to the discretion of the authors and/or technical team to discern whether an update or extension is feasible and determine the overall impact on the framework in question. In practice security practitioners do not have control in this regard, however, it is still of significance to mention given extension requests may or may not be approved if required throughout the duration of a penetration testing project. In the case of ISSAF, the document is no longer maintained; as a result it is not considered extensible. OWASP Testing Guide (OTG) on the other hand, provides extensibility by virtue of an approval process for a change request and encourages security professionals to put forward extensions for benefit of the frameworks capability. According to the project leader Andrew Muller, there is no formal process for change requests. Updates for OTG are raised on the OWASP mailing list allowing permission for anyone to add content. In the future this process is expected to improve and become more formalised. The turn around time for changes from suggestion to acceptance submitted via the wiki (online version), is approximately one day. Updates to the official released document (downloadable version) can range from one to two years (Muller, 2015). It is

important to note that it is not uncommon for organisations or security practitioners to implement local extensions by means of manual additions, particularly if the framework is open source. Depending on the framework in question, manual additions in some cases can lead to uncontrolled versions, thus causing confusion about the content if manual additions are not formalised. For the purposes of measuring extensibility, manual changes are not considered a formal process for extensibility; therefore, manual extension in this way is not considered a measure of extensibility.

**Extensibility summary:**

Based on evaluation of both OTG and ISSAF with respect to quality metrics, table 4.6 indicates that OTG is more extensible than ISSAF. It is important to note that the scale used for each property is nominal, not ordinal, therefore a framework either exhibits a property or it does not. All properties contribute equally to the quality metric i.e. there are no weighting factors assigned to each of the properties. It is acknowledged that this is a simple model, but it is sufficient for this purpose and will be used until proven deficient.

Table 4.6: Extensibility

<b>Sub-characteristics</b>	<b>ISSAF</b>	<b>OTG</b>
Capability	No	Yes
Timeliness	No **	Yes *
Modifiability	No	Yes
Modularity	Yes	Yes
Feasibility	No	Yes

Legend

\* Wiki updates are instantaneous, however the cycle time for the formally ratified version is two years.

\*\* No longer supported

**4.4.2 Measuring Maintainability**

Measures of maintainability are well established in the software engineering literature. The Maintainability Index (MI) is one such method that relies on techniques for counting lines of code, among others (Heitlager, Kuipers, & Visser, 2007). Clearly software methods for measuring maintainability are not suitable for penetration testing frameworks, however, that is not to say the concept is not relevant. Furthermore, penetration testing in its essence is testing software and/or hardware employing software of some type in most cases. Maintainability as defined in section

4.3.1, is the ease in which a framework can be understood, adapted, enhanced or modified by the intended maintainers, consequently undergo regular updates and revisions. It is of significance within the Computer Security discipline that new vulnerabilities are identified sooner rather than later; thus security professionals require up-to-date information as soon as it is publically available for penetration tests to be effective (Li & Rao, 2007). To measure this particular quality characteristic, consideration is given particularly to the number of revisions (frequency) a framework has undergone since inception. In addition to frequency of revisions, attention is paid to sub-characteristics as outlined in the Penetration Testing Framework Quality Model (see figure 4.2), namely: analysability, testability and modifiability. As previously stated, it is not practically possible to measure all characteristics or sub-characteristics of quality metrics, however, in the case of maintainability, sub-characteristics outlined in figure 4.2 should be present although not necessarily quantitative in nature. Each sub-characteristic is examined respectively.

Analysability, as defined by Standards Australia (2013), is the degree of effectiveness and efficiency with which it is possible to assess the impact on a product of an intended change to one or more of its parts. Analysability is concerned with diagnosing a product for deficiencies to identify parts to be modified. Though not the primary concern of security practitioners, it should be obvious from analysis whether or not the product is maintained. Similarly, modifiability is the degree to which a product can efficiently be modified without introducing defects or degrading system quality (Standards Australia, 2013), thus analysability has a direct influence on modifiability. Finally, testability establishes certain test criteria suitable to test a product. Subsequently, tests are performed to determine whether these criteria have been met. Measuring testability is not easy to quantify and will largely depend on the circumstances and environment of which a particular framework is used, therefore discernment is necessary. Returning to frequency, the primary characteristics preferred for measuring maintainability, is to quantify the number of revisions a framework has undergone in its lifetime. For instance, additional features implemented since inception are characteristics that contribute to revisions. The underlying assumption is revisions reflect a measure of activity. The type of activity, i.e., bug fixes, documentation edits, addition of new features, or any other activity, is not considered in isolation in this instance. To elaborate further, ISSAF restricts revision information. This means ISSAF omits the type of revision comparative to OTG whereby revision activity is provided in greater detail. Moreover, not enough comparative information is available between the two frameworks; therefore that level of detail is restricted. It is acknowledged that measuring frequency is not the only way to determine maintainability, however, it is the preferred method given the “product” is presented in documented form. Data analysis is achieved by calculating the average number of revisions annually for each

framework.

ISSAF underwent five revisions in its lifetime while OTG underwent five major revisions and numerous sub-revisions from OTG version 1.0 through OTG version 4.0. OTG originated in 2003 and subsequently transformed into an online wiki (online collaborative database), thus revision history became abundantly available (OWASP, 2014a). For the purposes of calculating revisions, OTG versions 1.0, 1.1, and 2.0 are omitted for reasons relating to the availability of historical data. Furthermore, earlier revisions of OTG did not include web application security testing which is the underlying factor for selection of this particular framework. Consequently, OTG versions 3.0 and 4.0 are quantified in the analysis.

Table 4.7: Revision Frequency

	Total Revisions (r)	Total Years (t)	Date of Inception
ISSAF	5	2	December, 2004
OTG 3.0 + 4.0	245 + 223 468	7	May, 2008

### Data Collection

The Open Information Systems Security Group (OISSG) no longer provides a web presence or statistical information relating to the ISSAF release cycle. As a result, information was difficult to obtain. To retrieve information regarding ISSAF revisions and updates, an alternate information resource was sourced from Internet archives (archive.org), an information resource that compiles snapshots of websites over time. Fortunately, data was retrieved from a specific year and/or month. The final update for ISSAF was 2006; therefore the dates entered for data retrieval are dated March 2007. Information regarding OTG revisions and updates is publically available from OWASP, therefore sourced directly from the OWASP website.

Table 4.7 shows OTG underwent 468 revisions over 7 years (OWASP, 2014b) in comparison to ISSAF of which underwent 5 revisions over 2 years (Internet Archieve, 2007). While it is obvious OTG had far more revisions than its counterpart, it does not necessarily prove that ISSAF lacks quality. Moreover, other factors come to the fore, for example; does OTG have more contributors

than ISSAF, were revisions for OTG related to bad design, or does it simply suggest the product is better overall? These latter questions are interesting, but beyond the scope of this research. While it is acknowledged that frequency of change (a sub-characteristic of maintainability) is one of various means to measure maintainability, the number of revisions a framework has undergone since inception can quantify frequency, thus reflects a measure of activity. Therefore, frequency of revisions is considered a sufficient method for measuring frameworks that present in documented form. To conclude, ISSAF is no longer supported by OISSG, and cannot be considered maintainable, however, maintainability was achieved in its lifetime averaging 2.5 revisions per year. Comparatively, OTG presently undergoes frequent revisions showing on average 66 updates per year, consequently, show greater maintainability than its counter part due to higher revision activity.

#### **4.4.3 Measuring Domain Coverage**

Penetration testing projects will often detail a well-defined timeframe and scope. To elaborate further, the AU2EU project used in this research had at its disposal two instances (cases) of an e-Authisation system. First, the German Red Cross (DRK) whereby the access point of the system was a browser based web interface; second, the CSIRO implementation whereby Near Field Communication (NFC) Cards are used for system access. In addition to access points, users of the system are granted access levels to authenticate, granting permissions to system functions appropriate for each access level. In the case of DRK, the primary access point is a web interface accessible via a web browser; therefore the appropriate domain coverage for this particular implementation is web application. Regrettably, the CSIRO implementation did not undergo testing due to project limitations not identified prior to this research. As a result, the IBM and DRK implementations are evaluated for domain coverage respectively for both frameworks. With this in mind, the framework of choice should include appropriate domain coverage to align with the agreed project scope, i.e., web application as the primary domain coupled with database and password assessment as sub-domain coverage. To determine if domain coverage is adequate, several factors are considered. First the application(s) or hardware devices that require assessment (primary domains). Second, corresponding sub-domain coverage of each primary domain identified. This research was performed on a web application, thus only one primary domain area was required for coverage, that of web application. Sub-domain coverage for web application includes, but not limited to: cross-site scripting tests, database assessment, and password attacks among others. Additionally, it is worthy to note that information gathering is a fundamental phase of all penetration testing frameworks and should be included in a given project. In the case of OTG, only one domain area is covered, however, if the focus of the penetration test is web application then it is suffice to say that the domain coverage is adequate provided sub-domain coverage of the web

application domain is addressed. The structure of OTG is purely focused on web application testing compared to ISSAF of which adopts a much broader approach, thus the structure of ISSAF supports far more primary domain areas, for instance; VPN assessments, firewalls, and network storage. The OISSG, responsible for the creation of ISSAF, asserts that it is easier to remove information rather than develop it (OISSG, 2005). For this reason, it attempts to cover every scenario and prefers a one size fits all approach as compared to OTG whereby its narrow focus is designed specifically for one primary domain. The question that arises from unitary coverage is whether it is detailed enough, in other words, are there sufficient sub-domains covered within the framework to adequately test the web application interface and infrastructure in its entirety. With reference to ISSAF, domain coverage is far more extensive in comparison to OTG. As will be demonstrated, this does not mean that ISSAF was the better choice. Moreover, the context and scope of the research was primarily based around the web application domain, therefore OTG provided adequate coverage. Literature investigating domain coverage relating to penetration testing is scarce. Even less discussion exists regarding an approach as to how it could potentially be measured. This research proposes a simple model for assessing frameworks for domain coverage taking into consideration corresponding sub-characteristics outlined in the PTF Quality Model (see figure 4.2). Sub-characteristics are either present or they are not; therefore the scale is categorical in nature. To recall, the supporting sub-characteristics are: adaptability, flexibility, and supportability.

Standards Australia (2013) define adaptability as “the degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage requirements”. In relation to domain coverage for penetration testing purposes, adaptability takes on similar meaning by way of accommodating domain coverage for different circumstances that might not be addressed explicitly, thus deemed flexible without requiring a modification or change request. This is not to be confused with extensibility whereby an extension would be considered more appropriate. One example of adaptability encountered while implementing ISSAF was password attacks. Password attacks against login authentication systems are not covered within the web application domain area but are addressed under an additional domain area “password attack”, therefore, by means of cross domain coverage, adaptable. Similarly, flexibility pertains to the flexible nature of a specific domain area. This essentially means that although a domain area may not be covered in its entirety, it is still applicable. To demonstrate, SQL injection is recommended as a sub-domain of web application testing within ISSAF. The recommended technique shows only how to inject MySQL and ignores PostgreSQL injection. This is not to say that PostgreSQL injections are not applicable for a different project, moreover, the same technique can be adapted to cater for PostgreSQL injection simply by changing the input data

(injection strings). Finally, supportability is relatively simple, the framework either supports a particular primary domain area for security assessment or it does not. Naturally, the security practitioner will examine the framework for relevant domain coverage and predetermine suitability before employing a framework for a particular project.

Table 4.8: Domain Coverage

<b>Domain Coverage</b>		
	ISSAF	OTG
Supports Primary Domain	Yes	Yes
Sub-domain coverage adequate	Yes	Yes
Adaptability	Yes	Yes
Flexibility	Yes	Yes

### **Domain Coverage Summary**

Provided project scope is well defined, analysing a framework for appropriate coverage is possible. The sub-characteristics coupled with framework analysis present a proposed model for assessing framework domain coverage. Table 4.8 shows domain coverage for ISSAF and OTG is adequate for this research.

#### **4.4.4 Measuring Usability**

Prior to any meaningful evaluation of usability, it is necessary to consider the assumed knowledge and experience of the intended user(s). Naturally, users referred to here are security practitioners who specialise in penetration testing; therefore, basic knowledge of penetration testing terms, tools and techniques is the underlying assumption. At a minimum, familiarity of common phases, such as, information gathering and associated techniques/tools, (i.e., nmap) is expected. While it is acknowledged other penetration testing distributions exist, familiarity with Kali Linux is assumed. It is outside the scope to discuss tools and techniques at length, however, suffice to say that security practitioners performing penetration testing should be knowledgeable and capable of using common tools and techniques. Security practitioners are not expected to be experienced in every aspect of penetration testing.

Usability is described as the ease of use and efficiency that security practitioners can learn to operate, prepare inputs, and interpret outputs. Penetration testing frameworks ideally should present as easy to use, thus require minimal learning time. Measuring usability in this research will mostly



rely on practical use and assessment manually, however, usability does exhibit sufficient sub-characteristics to aid in measuring frameworks for usability. To recall, sub-characteristics outlined in the PTF Quality Model (see figure 4.2) are: appropriateness, learnability, operability, and readability.

Standards Australia (2013) define appropriateness as; the degree to which users, in this case security practitioners, can recognise whether a product is appropriate for their needs. This can be assessed simply by means of examination of the framework. First, determine the framework is aimed at penetration testing explicitly as opposed to assessing the entire security posture of an organisation. Next, ensure the framework details the domain of the problem space. The problem space can be examined by exploring the introduction and table of contents for domain coverage and scope.

To demonstrate one aspect of appropriateness, the case study selected for this research is web application. ISSAF and OTG were appropriate inasmuch as domain coverage of web applications within both frameworks is obvious by examination. The introduction adequately discussed the scope of the framework and the table of contents suggests the framework provides adequate scope for the project. In addition, a quality framework may provide examples of tests that a reader can identify as relevant for a particular project, or include appendices that further support the security practitioner in completing the project. Adopting a qualitative approach, table 4.9 is proposed as a means of assessing appropriateness for ISSAF and OTG.

Table 4.9: Framework Appropriateness

	<b>ISSAF</b>	<b>OTG</b>
Penetration Testing explicitly	Yes	Yes
Introduction relevant to project	Yes	Yes
Domain areas outlined in the table of contents	Yes	Yes
Examples provided (i.e. input and output data)	Yes	Yes
Appendices provided	Yes	Yes

Learnability is described as the degree to which a product can be used by specific users to achieve goals of learning. Additionally, use the product with effectiveness, efficiency, freedom of risk, and satisfaction in a specific context (Standards Australia, 2013). The “product” is a framework and user

refers to security practitioner in this context. Learnability can be expanded to include recommended tools, in other words, if recommended tools to perform specific security tests cannot be applied practically then learnability is questionable. To elaborate further, there should be adequate instruction or reference to additional documentation to aid learnability for a specific phase or test for learnability to be met. To address learnability, ISO and IEC (2002) provide a learnability metrics table that can be adapted for penetration testing frameworks (see table 4.10).

Table 4.10: Learnability Metrics (ISO & IEC, 2002)

<b>Internal learnability metrics</b>									
Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Target audience
<b>Completeness of user documentation and/or help facility</b>	What proportion of functions are described in the user documentation and/or help facility?	Count the number of functions implemented with help facility and/or documentation and compare with the total number of functions in product.	$X = A/B$ A= Number of functions described B= Total of number of functions provided	$0 \leq X \leq 1$ The closer to 1, the more complete.	absolute	X=count/count A=count B=count	Req spec Design Review report	Verification Joint review	Requirers Developers

**NOTE:** : Three metrics are possible: completeness of the documentation, completeness of the help facility or completeness of the help and documentation used in combination.

Adopting table 4.10 as an underlying model, the following amendments (see table 4.11) are proposed as a means of measuring learnability for ISSAF and OTG.

Table 4.11: Amended Learnability Metrics

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Input to measurement	Target audience
<b>Completeness of user documentation</b>	What proportion of tests are in scope in the framework	Count the number of tests implemented with help facility and/or documentation and compare with the total number of tests in-scope.	$X = B/A$ A= Number of tests in-scope B= Total of number of tests implemented	$X * 100$ The closer to 100%, the more complete.	absolute	framework	Security practitioner

Naturally, the amendments proposed in table 4.11 are designed to align with penetration testing. It is likely that not all tests in a particular framework will be in-scope for a project, as discovered throughout this research. To illustrate, ISSAF details a comprehensive range of domain areas that were not relevant for this research, therefore certain tests were not applicable, i.e., testing of firewalls and routers. In comparison, OTG suggests three levels of testing, namely: black-box, white-box, and grey-box testing whereby only black-box testing was undertaken. As a consequence, only tests directly applicable to this research are considered for both frameworks. The number of tests implemented represents tests undertaken without further reading; in other words, the framework provided sufficient instruction to execute a particular test within the bounds of the framework.

Table 4.12: ISSAF and OTG Learnability Result

	Proportion of tests in scope (n)	Number of tests implemented (t)	Measurement formula $x = (t/n)$	Interpretation of measurement value	Metric scale type
ISSAF	31	21	0.677	67.7%	absolute
OTG	52	40	0.769	76.9%	absolute

Table 4.12 shows that OTG demonstrates 76.9% learnability as compared to ISSAF which shows 67.7%. While adequate instruction is provided for ISSAF, it is not nearly as comprehensive as OTG at the testing level. This suggests that execution of certain penetration tests are considered learnable inasmuch as executing a command, however, interpretation of results requires explicit knowledge on behalf of the security practitioner to be beneficial. Moreover, analysis of results is of more concern than rote-learning a command.

Operability is concerned with the attributes that make a product easy to operate and control (Standards Australia, 2013). Operability ensures that inputs and outputs conform to user expectations. Consider a common task performed in the information gathering phase, for example, banner grabbing. Banner grabbing involves the use of a specific tool (i.e. netcat) to obtain information about a web application that requires certain input parameters such as IP address and port number. Once executed a particular output is expected. If the command fails for reasons not outside the pen testers control, i.e., network connectivity, it is not considered operable. In other words, the command does not conform to user expectations and execution fails. Measuring operability for penetration testing frameworks can become somewhat difficult due to the large number of tests performed; therefore evaluation can be based on the overall success of the

penetration test. The tools and techniques recommended should provide adequate instruction and execute as expected. As previously discussed, the expected output is open to interpretation by the security practitioner and deserves further elaboration. For the purpose of this research, discussion on interpretation of results is out of scope in terms of operability. Similarly to learnability, measuring operability is achieved by examination of a similar model provided by ISO and IEC (2002), Operability Metrics (see Appendix A). Likewise, the model is adapted to align with penetration testing.

Table 4.13: Operability Metrics

	Total number of tests performed (t)	Number of tests that provide example input data (d)	Number of tests that can be customised during operation (o)	Number of tests that provide messages that are self-explanatory (m)	Test interfaces that are self-explanatory (s)	Number of tests that tolerate user error (e)
ISSAF	21	12	12	Not assessed	Not assessed	2
OTG	52	50	50	Not assessed	Not assessed	13

### Interpretation of Results

The results detailed in table 4.13 refer to penetration testing undertaken for the IBM implementation. Due to the unexpected unavailability of the DRK implementation (detailed in section 4.6), ISSAF testing was terminated. OTG testing did complete in its entirety. For this reason, testing continued with both frameworks using the equivalent IBM implementation. For comparability, only IBM results are reported for both OTG and ISSAF.

The total number of tests (t) performed (see table 4.13) show that input data (d) provided for each test respectively is 57.14% for ISSAF compared to 96.15% for OTG. This suggests OTG is more operable opposed to its counter part. That is not to say that the remaining tests omitting example input data were not successful, moreover, the input data entered was known. All tests performed were customisable during operation (o) inasmuch as repeatability and variation of parameters. To elaborate further, repeatability was achieved with different data sets and additional parameters. With regard to the number of self-explanatory messages (m) and self-explanatory interfaces (s), evaluation was not performed for reasons of interpretation, in other words, it is difficult to define “self-explanatory” within a penetration testing context, thus open to interpretation and solely based

on expertise. The metrics remain in the proposed model for penetration testers who wish to explore this metric further. User error metrics are based on the number of tests that allow invalid input data (e), but not necessarily obvious based on the output. For example, SQL injection performed using both frameworks did not discriminate between valid SQL injection parameters and invalid SQL injection parameters. This means the output was the same provided the web application was not vulnerable to SQL injection. User error for ISSAF is 9.52% compared to 25% for OTG. While it is obvious that tolerance of user error is greater for OTG, it is worthy to note that tests performed using OTG are vastly different in comparison considering OTG offers far more detail for web application testing. To elaborate further, it is not uncommon for OTG to dedicate one or more pages that suggest an array of tools and techniques for a particular test.

Readability, introduced as a sub-characteristic of usability, is nominated as one possible means of measuring usability for frameworks that present in documented form. Readability has a direct relationship to usability, thus if the document under evaluation is not readable usability is affected. Foremost, readability is concerned with how easy or difficult it is to read or comprehend written text (Ludger & Gottron, 2012). Primary factors considered for readability include syntax and complexity of vocabulary. Educators have developed techniques to measure vocabulary difficulty and sentence length, thus difficulty level of text can be predicted. Consequently, a variety of readability formulas are in use today (DuBay, 2004). One such formula known as Gunning Fog Index (GFI), commonly referred to as Fog index (Gunning, 1952), was developed specifically for adults. Fog index attempts to predict the school grade-level required by a reader to understand a given text at first reading. For example, a fog index score of 10 would indicate that a minimum of 10 years formal schooling (U.S) is required to understand and correctly interpret a given text. The formula works with two variables and produces the formula  $(GL) = 0.4 (ASL + PHW)$ . The average sentence length (ASL) is added to the number of words with more than three syllables for each one hundred words (PHW). The result is then multiplied by 0.4 (DuBay, 2004) and produces the grade level (GL). Whilst there are various formulae for measuring readability the Fog Index is preferred for this research based on four characteristics: suitability for adult readers (intended audience), simplicity, accuracy, and speed (Klare, 2000).

### **Fog Index Analysis and Results**

Results were obtained using an automated open-source tool that calculates Fog index scores ("Readability-Score," 2015). OTG contains 220 pages, therefore evaluated in its entirety. Due to file size limitations within the tool, the 845 pages of ISSAF were split into four separate files consisting of 220 pages each, except the last file that consisted of 185 pages. The average Fog Index of all four

parts for ISSAF is calculated.

Table 4.14: Fog Index Scores

	ISSAF	OTG
GFI Score	7.7	11

From table 4.14 it is obvious ISSAF requires less formal education compared to OTG. ISSAF requires a minimum grade-level of 7.7 comparative to OTG whereby a grade-level of 11 is recommended. Taking into consideration the intended audience, it is theorised that security practitioners would at a minimum, complete 11 years formal schooling based on the underlying assumption that Computer Security extends upon formal education. Put another way, even if 11 years of formal schooling is not achieved, an equivalent form of study or post-schooling is likely to be undertaken. Therefore, readability scores for both frameworks are considered sufficient for the intended audience. Comparatively however, ease of reading and comprehension of written text within ISSAF is easier to interpret in contrast to its counterpart, OTG.

### **Usability Summary**

Usability encapsulates four sub-characteristics, namely: appropriateness, learnability, operability, and readability. The appropriateness of each framework was analysed by qualitatively evaluating each characteristic. ISSAF and OTG exhibit the four characteristics nominated to examine appropriateness, outlined in table 4.9. A different approach was fostered for measuring the learnability sub-characteristic by means of amending the ISO Learnability Model to align with penetration testing frameworks. The amended model was successful inasmuch as demonstrating the comparator used could distinguish between the two frameworks. Similar to learnability, operability was examined by employing an underlying measurement model utilising six metrics as a means of quantifying operability. OTG provided comprehensive input data compared to ISSAF, however, the number of tests that tolerate user error are greater for OTG. GFI was selected as a suitable method to evaluate readability for reasons of intended audience, simplicity, accuracy and speed. Results show ISSAF requires less formal education compared to OTG. While it is acknowledged that an array of usability models exist in the literature, the metrics proposed for this research are nominated for their ability to adapt to penetration testing frameworks, demonstrating that quantifying usability for frameworks can be achieved.

### **4.4.5 Measuring Availability**

Availability pertains to the degree of which a system, product or component is operational and accessible when required for use (Standards Australia, 2013). In software, availability is more

concerned with operational availability; in other words, system users can use the software without experiencing frequent outages (Piedad & Hawkins, 2001). In the case of penetration testing, the framework is the product and the components consist of modules or phases divided into domain testing areas. To measure availability for frameworks, two variables are considered. First the framework must be available for use, i.e., downloadable or obtainable in electronic or printable form. Although it may appear obvious that a framework is available, this might not always be the case for every scenario as will be discussed in more detail later in the chapter. Second, recommended tools or third party software employed for a particular test should also be obtainable without incurring additional cost to the project that has not been planned for. Put another way, security practitioners will generally plan and scope enough resources for a given project, however, if tools recommended for use are present as open source at the planning stage but require purchase at the operational stage then availability is impacted. In addition to the abovementioned factors, sub-characteristics are considered. To recall, sub characteristics of availability are, portability, obtainability, and operability.

Standards Australia (2013) defines portability as “ the degree of effectiveness and efficiency with which a system, product or component can be transferred from one hardware, software or other operational or usage environment to another”. In software, the concern is more about running software on different platforms, for instance, Windows or Macintosh operating systems. Penetration testing frameworks are more about adapting to different circumstances. This may involve employing the same framework for various projects, thus the discussion is directed more towards “usage environment”. Portability is met if the framework can be tailored for different projects or adapted under different circumstances. On the other hand, obtainability is concerned with the effort required to obtain a particular framework and/or the required tools. For instance, a framework may be publically available; nonetheless, the cost or license of certain tools might restrict obtainability to some degree. If documentation or software tools are not available, by default it implies that portability and obtainability are directly affected, consequently, impacts operational use.

To address availability one factor of concern is whether or not a certain framework is obtainable in some form, for example, a printed copy or electronic version. In the case of ISSAF, availability is questionable, not because it cannot be obtained; moreover, it is no longer supported. Therefore availability is impacted. To illustrate, ISSAF was previously available from the official OISSG website, however, the website is no longer accessible. As a consequence, security practitioners are required to search for an alternative source to obtain the framework. As of this research ISSAF is easily accessible from a collection of sources, primarily SourceForge. It should be noted that future

availability might be limited. Similarly, consideration is given to the availability of software tools recommended throughout the framework for particular tests. As previously discussed, software referenced within frameworks are generically referred to as tools, in some a cases, software suites (i.e. Metasploit), or even an entire operating system such as Kali Linux. Ideally, security practitioners would verify availability of tools before settling on a particular framework. In addition, software can and does change ownership and license requirements over time; therefore cost can also be a factor. To illustrate, consider Nessus, a vulnerability scanner often recommended for penetration testing projects. Originally Nessus was free and open source, however, Tenable Network Security purchased the product in 2005, as a result, Nessus is now a commercial product (LeMay, 2005). It is suffice to say that availability of the product (ISSAF) is satisfied, however, not necessarily obtainable if assumed to be an open source tool, thus obtainability would impact the project budget.

Table 4.15 shows a taxonomy of third party tools recommended by ISSAF and OTG that fit into one of three categories. Third party tools in table 4.15 refer to any software tool not packaged by default with Kali Linux (Offensive Security, 2015). Only tools that are required for web application testing (in-scope) are included.

Table 4.15: Third-party Tool Availability

	Total number of third party tools (not included in Kali Linux)	Number of tools not obtainable	Number of tools requiring purchase
ISSAF	14	5	5
OTG	32	5	4

Availability is satisfied for both frameworks inasmuch as OTG and ISSAF present as portable, obtainable, and operational post evaluation. As discussed previously, future availability of ISSAF is questionable. Turning to third-party software tools, OTG recommends 32 as compared to 3 recommended by ISSAF. This could be interpreted negatively by means of assumption that OTG poses greater risk of impacting availability, however, on the contrary, OTG recommends an array of tools for one task; in other words, ISSAF recommends one particular tool for one test in most cases opposed to OTG whereby multiple tools are recommended for the same test. Therefore it could be postulated that OTG offers a higher degree of operability. 28% of tools within OTG were either unobtainable or required purchase compared to 71% for ISSAF, unsurprisingly given ISSAF is no



longer supported. While ISSAF has a higher rate of unobtainable tools, it relies heavily on the web browser, thus a majority of the tests were executed; this demonstrates the diverse approach that each framework adopts.

#### 4.4.6 Measuring Reliability

Reliability relates to the degree a system, product or component performs specific functions under specific conditions over a specified period of time (Standards Australia, 2013). Referring to frameworks specifically, conditions might relate to the agreed project scope. Similarly, the specified period of time refers to the time allocated to complete the project in its entirety. To recall, the PTF Quality Model proposes three sub-characteristics in support of reliability (see figure 4.2), namely: maturity, consistency, and reusability.

Maturity pertains to the frameworks ability to meet requirement needs under normal operation (Standards Australia, 2013). There exists a range of software maturity models published within the existing body of knowledge (Al-Qutaish & Abran, 2011), however, as commonly experienced throughout this research, little exists in the literature addressing maturity relating to aspects of penetration testing or frameworks in documented form. The Open Source Maturity Model (OSMM) is one model of particular interest due to its usability and reference to open source products, for this reason, OSMM is considered suitable for an underlying maturity model for this research. OSMM measures maturity by means of allocating a maturity score in three phases. (see figure 4.3).

	Phase 1			Phase 2	Phase 3
	Define Requirements	Locate Resources	Assess Element Maturity	Assign Weighting Factor	Calculate Product Maturity Score
Product Software					
Support					
Documentation					
Training					
Product Integrations					
Professional Services					

Figure 4.3: OSMM Maturity Model (Navica, 2004)

The first phase assesses vital product elements for maturity and assigns a score. Second, a

weighting score is defined for each element based on organisational requirements. Finally, phase three calculates the products overall maturity score (Golden, 2008).

The primary concern of phase one is to assess vital elements. In this case the elements are properties critical to successful completion of a penetration testing project, thus the model has been adapted to reflect elements applicable to penetration testing frameworks. OSMM advocates a four-step process for evaluating product elements. First, requirements are defined to determine business objectives. In most cases the ultimate goal is to identify vulnerabilities that may exist within the security posture of a system, consequently, take appropriate action based on the results. Second, resources are located to support the requirements. For instance, ISSAF does not have a strong community base comparative to OTG whereby forums, mailing lists, among other resources, are widely available due to its collaborative support. Third, maturity is assessed to determine how the element ranks on the maturity continuum. Finally, a maturity score is allocated. Table 4.16 shows the Maturity Evaluation Template used for maturity assessment prior to any modification.

Table 4.16: Maturity Evaluation Template (Navica, 2004)

Element	Potential Score	Actual Score	Weighting Factor	Element Weighted Score
Software	10		4	40
Technical Support	10		2	20
Documentation	10		1	10
Training	10		1	10
Integration	10		1	10
Professional Services	10		1	10
<b>Total Product Maturity Score</b>				100

Table 4.17 and table 4.18 evaluate ISSAF and OTG with the proposed maturity model. Minor adjustments are proposed to align with aspects of penetration testing. Phase one has been amended to combine defining requirements and locating resources that form the elements outlined in table 4.17 and table 4.18.

Table 4.17: OTG Maturity Score

Element	Potential Score	Factors	Actual Score	Weighting Factor	Element Weighted Score
Framework	10	Longevity Revisions downloads availability supporting resources maintainability	9	3	27
Supporting Structure	10	Phases sub phases relevancy	7	1	7
Repeatable processes	10	Non-existent to adequate	8	1	8
Domain Coverage	10	Non-existent to adequate	10	2	20
Instruction – describes “what and how”	10	Non-existent to adequate	7	1	7
Remediation Techniques	10	Non-existent to adequate	8	1	8
Reporting	10	Instruction templates	2	1	2
<b>Total Product Maturity Score</b>					79

Table 4.18: ISSAF Maturity Score

Element	Potential Score	Factors	Actual Score	Weighting Factor	Element Weighted Score
Framework	10	Longevity Revisions downloads availability supporting resources maintainability	4	3	12
Supporting Structure	10	Phases sub phases relevancy	9	1	9
Repeatable processes	10	Non-existent to adequate	8	1	8
Domain Coverage	10	Non-existent to adequate	8	2	16
Instruction – describes “what and how”	10	Non-existent to adequate	7	1	7
Remediation Techniques	10	Non-existent to adequate	6	1	6
Reporting	10	Instruction templates	5	1	5
<b>Total Product Maturity Score</b>					63

Each element has a potential score of 10. The factors are characteristics that support the elements actual score. For example, the element “framework” should exhibit longevity, downloads, maintainability etc., while “domain coverage” considers a range from non-existent to adequate. These factors are taken into consideration before the actual score is assigned. The weighting factor of each element is summed to provide an overall maturity score of the product. Weighting factors can be adjusted to suit the particular product under review providing flexibility for unique circumstances. The only requirement is that the maturity weighting factors must sum to 10 since the final step of OSMM is to create an overall maturity score that normalises to 100 point scale (Golden, 2008). The weighting factors assigned in table 4.17 and table 4.18 are based on the importance of each element specific to penetration testing frameworks. After the actual element score and weighting factors have been assigned, the overall maturity score is calculated. Element

scores are summed on a scale of 1 to 100 where the highest possible score is 100 (see table 4.16).

### **Interpretation of Results**

Adopting the OSSM model, table 4.17 shows OTG has a maturity score of 79 compared to table 4.18 where ISSAF shows a total maturity score of 63. This suggests the overall maturity of OTG is approximately 20% more mature than its counterpart.

Consistency is achieved within a software specification or document when its parts are not in contradiction, thus consistency can be regarded as the degree of uniformity (Laplante, 2001). Deutsch and Willis (1988) assert that consistency is achieved when standards are used throughout, however, this is more inline with software engineering. Consistency, relating to documented frameworks can adapt to the abovementioned definitions by means of following a structured format. The framework itself should exhibit phases and sub-phases consistent throughout by taking a uniform approach. In addition, inputs and outputs should match expected results to a degree acceptable by the security practitioner. To ascertain whether consistency is achieved, a qualitative approach is preferred, therefore this sub-characteristic is categorical, thus the framework exhibits consistency or it does not. OTG demonstrated consistency by means of following a well-structured format. Each testing domain is divided into phases and sub-phases: summary, how to test, input/output, tools, and whitepapers. Comparatively, ISSAF presents as consistent following a similar structure, mainly: description, pre-requisites, examples and results, analysis, counter measures, tools, further reading, and remarks. ISSAF appears to offer a more detailed structure, however, further observation of ISSAF revealed vague or non-existent detail compared to its counterpart whereby detail was far more comprehensive. Nevertheless consistency is achieved.

Reusability as defined by Standards Australia (2013) is the degree to which an asset can be used in more than one system. More generically, the ability to use or easily adapt software developed for a system to build other systems. Reusability can be applied to patterns, frameworks or components (Laplante, 2001). When referring to penetration testing frameworks, reusability is primarily concerned with the ability of the framework to adapt to new circumstances, or, a new project. Similarly to consistency, a qualitative approach is preferred. The two e-Authorisation systems (DRK and IBM) underwent penetration testing using both ISSAF and OTG. Although similarities exist between the implementations (i.e. the underlying architecture), the projects in essence are different, thus highlighting reusability for both frameworks. Therefore, it is concluded that reusability is met for OTG and ISSAF. Table 4.19 shows a summary of reliability for OTG and ISSAF.

Table 4.19: Reliability Summary

	Maturity Score	Consistency	Reusability
ISSAF	63	yes	yes
OTG	79	yes	yes

## 4.5 Discussion of Results

This chapter set out to examine penetration testing frameworks and methodologies in three distinct areas: nomenclature resolution, defining quality metrics, and framework quality evaluation. The primary goal was to determine if penetration testing frameworks could be evaluated for quality. This presented further questions that required remedy before any meaningful quality evaluation could be undertaken. The literature provides abundant information relating to software engineering, however, information relating to penetration testing frameworks appears to be limited, and in some aspects, non-existent. The approach undertaken was to examine the field of penetration testing in isolation and draw upon literature that is authoritative and internationally recognised whereby a foundation could be established.

The first question (RQ1) that arose was how to differentiate between a framework and methodology for the purposes of penetration testing. Section 4.1 (Nomenclature Resolution) was positive inasmuch as a definition was formed. As a result, the Framework vs. Methodology taxonomy (see table 4.2) was devised to determine what characteristics exist, thus addressing sub-questions RQ1:a and RQ1:b, that asked what characteristics define a framework and methodology respectively.

The second question (RQ2) the research presented was what quality metrics could be defined explicitly for penetration testing frameworks. Five quality models were reviewed to determine a suitable set of quality metrics that subsequently formed the quality model proposed in section 4.3.2 The Penetration Testing Quality Model (adapted from ISO/IEC 25010:2013) served as an underlying quality model whereby evaluation was performed combining qualitative and quantitative measures.

The final question (RQ3), investigates whether penetration testing frameworks can be evaluated against the proposed model for quality. The Gap Analysis (see section 4.2) facilitated the selection of two suitable frameworks (ISSAF and OTG) to undergo quality evaluation. Penetration tests against the DRK implementation and the IBM implementation were conducted in an attempt to answer RQ3. The research showed that while no known model currently exists to undertake this

task, quality could indeed be measured in one form or another. To reintroduce; extensibility adopted a qualitative approach resulting in table 4.6. Four properties are nominated and each property contributes equally to a certain quality metric, thus a framework demonstrates extensibility or it does not. Maintainability was addressed by means of using a measure of frequency (a sub-characteristic of maintainability) to determine the level of activity a framework undergoes. The number of revisions and updates are quantified. Domain coverage (section 4.4.3) proposed a simple model (see table 4.8), whereby four properties are examined to determine appropriate domain coverage for a given project. In the case of this research, it was demonstrated that domain coverage was adequate. Availability was quantified by assessment of third-party tools that could potentially impact project time and cost. This was considered important due to planned project budget, thus if a framework recommends open source third-party tools then the tools should be obtainable at the time of testing, however, this was not the case for OTG and ISSAF. It was found that OTG recommended 32 third-party tools not included in a default installation of the penetration testing distribution (Kali Linux) compared to ISSAF whereby 3 are recommended. Finally, reliability was evaluated using one of its sub-characteristics, maturity. Maturity scores were calculated by adapting the Open Source Maturity Model (OSMM), the result being that OTG was more mature than ISSAF.

The quality evaluation does not attempt to argue any one particular approach to be correct; nevertheless, the research shows that quality evaluation of penetration testing frameworks is possible.

#### **4.6 Amendments to the Research Environment**

This research originally intended to use the AU2EU project whereby two pilot implementations of e-Authorisation systems were nominated to undergo penetration testing. To recall, the two pilots included as part of the AU2EU project are DRK (the German Red Cross) hosted by NEC, and CSIRO. DRK provided a web interface as an access point allowing users of the system to authenticate enabling them to manage services provided by the Red Cross to clients. The CSIRO pilot (using the same underlying architecture) enabled authorised system users to authenticate via a near field card. The purpose of the CSIRO pilot was to permit intended users of the system to enter and communicate biological information regarding animal disease. While it is outside the scope to discuss in depth the purposes of these systems, it is worthy to mention for reasons of context. Regrettably, a number of issues were encountered during the research process resulting in amendments to the proposed research design.

Foremost, regarding DRK, four key issues hindered the research process. First, the concrete

implementation was not delivered until November 6, 2015, therefore outside the timeframe for this research. In addition, time allocated for penetration testing (by the client organisation) was one day as opposed to the agreed timeframe of two weeks. Second, source code was not provided as originally intended restricting the inspection of code. Third, interface changes were encountered whilst penetration testing progressed, thus changing the stability of the environment. Finally, system credentials were not obtainable. Testing of user permissions, password resets among an array of other recommended tests, was not possible. Despite these problems, penetration testing did go ahead, however, it was not practically possible to accurately assess the implementation as originally intended. While the system could not be accurately tested, OTG was put into practice in its entirety. Certain tests were successful inasmuch as output was reported (see Appendix B); ISSAF was not utilised.

Turning to the CSIRO pilot, a misunderstanding of the test objectives and the actual system was the primary factor restricting penetration testing. To elaborate further, the environment was secure from external threats due to the access point, i.e., card scanning was the authentication mechanism as opposed to a web interface originally assumed. While penetration testing a card interface is possible, the resources required to support it were not available in the allocated timeframe. Furthermore, the time required to negotiate access was of concern given the pilot system was behind a corporate firewall. Any testing of the pilot may have adversely affected production systems; as a result the CSIRO pilot was not tested in any capacity.

Notwithstanding the limitations encountered, the research continued. Fortunately, IBM had at its disposal a virtual implementation using the same underlying architecture that facilitated the remainder of the research. Similar to DRK, the IBM implementation access point was a web interface suitable for penetration testing, thus utilising both ISSAF and OTG. While it is acknowledged that penetration testing is not restricted to web application testing, for reasons of comparatively evaluating frameworks, it was preferred.



## Chapter 5 : Conclusion

It is no longer questionable that cyber crime is on the rise worldwide. In the first quarter of 2015, more than \$234 million of financial loss was reported via the Australian Cybercrime Online Reporting Network (ACORN) (Australian Crime Commission, 2015). The industry is starting to mature; consequently, the need for penetration testers has grown (Allen, 2012). As a result, this research set out to examine penetration testing methodologies and frameworks to determine whether frameworks and methodologies promoting penetration testing could be evaluated for quality. However, this question is complex inasmuch as first determining what defines a framework and methodology in this context.

Despite there are many frameworks and methodologies available, the terminology is often misleading. The research set out to resolve the difference between the two terms to align more closely with the aspects of penetration testing. It was found that no known definition exists for documented frameworks within the field of penetration testing. Generically, however, and within the information systems discipline, definitions of these terms are attainable. Based on already established authoritative sources, the research was able to define methodologies and frameworks with a specific focus on penetration testing. In addition, identify characteristics that could potentially distinguish between them and provide a foundation of which the concepts can be understood and built upon. Interestingly enough, it was discovered that while some methodologies and frameworks were certainly what they claimed, others were found to be a collection of tools or entire security assessments. The inexperienced pen tester would benefit from the knowledge gained in this regard.

Following nomenclature resolution, quality metrics specifically aimed at penetration testing frameworks was not easy to obtain, therefore quality metrics were drawn from existing quality models where quality metrics are already established. The research was successful inasmuch as devising an underlying quality model based on the ISO product quality model (Standards Australia, 2013), as a result, a penetration testing quality model is proposed. While it could be argued what metrics should form part of the model put forth, it was shown that quality metrics could indeed be defined and measured in some form, in other words, measuring quality is possible. Measuring quality of a particular framework, in this case ISSAF and OTG, did produce further questions. For example, interpretation of results is one concern proposed for future research. Frameworks do not go far enough inasmuch moving away from rote learning commands to understanding and correctly interpreting results. This is considered a key factor in any penetration testing project. Knowledge gained from this research suggests that penetration testing should be treated as a project, therefore

would benefit from a mature underlying framework or methodology that security practitioners can follow. Furthermore, evaluating frameworks for quality is possible when clearly defined terms are established. Approaching security assessments would certainly benefit by moving away from the quick and easy approach of running vulnerability scanners and ad-hoc tests to assess the security posture of a given organisation. A penetration test is merely a snapshot in time, thus regular penetration tests should be undertaken. With a sound underlying framework of which to work, securing systems more effectively can be achieved.

In conclusion, this research reviewed nine penetration testing frameworks and methodologies of which underwent examination using various methods to determine suitability for evaluation. It was found that many frameworks were either mis-named (i.e., were not actually frameworks in reference to characteristics outlined in table 4.2), or lacked domain coverage or a sound ontological foundation and thus restricted in their application. The two frameworks selected for evaluation (ISSAF and OWASP's OTG) underwent quality evaluation based on their focus (penetration testing specific or security general) and ability to act as a framework (rather than a collection of techniques without a unifying theme). The research discovered that many "frameworks" could not be generalised across problem domains (as would be expected for a generic pen testing framework). The quality metrics mapped well to the selected frameworks (ISSAF and OTG), which suggests that they are appropriate candidates to evaluate penetration testing frameworks. A solid understanding of methodologies, frameworks, and quality models particularly focused on penetration testing will help the security profession deliver better services overall.

## **5.1 Research Outcomes**

A number of research questions were formed to determine whether penetration testing frameworks could be measured for quality, the primary focus of this research. Prior to any meaningful quality evaluation, research questions one and two emerged and required remedy (see table 5.1). Both qualitative and quantitative methods are used to address the research questions presented. Table 5.1 shows the research questions and design phase mapped to the solution proposed.

Table 5.1: Research Questions and Proposed Solution

Research Question	Related Design Phase	Proposed Solution
RQ1: How to differentiate between methodologies and frameworks for the purpose of penetration testing	Phase 1 Methodology/ Framework definition	Section 4.1: Nomenclature resolution
RQ1-a): What characteristics define a penetration testing framework		Section 4.1.2 Nomenclature Definition
RQ1-b): What characteristics define a penetrating testing methodology		Section 4.1.2 Nomenclature Definition
RQ2: What quality metrics can be defined for penetration testing frameworks		Section 4.3 Quality Metrics and Evaluation
RQ3: How can penetration testing frameworks be evaluated for quality		Section 4.4 Framework Quality Evaluation

RQ1: How to differentiate between methodologies and frameworks for the purpose of penetration testing?

Section 4.1.1 began with a review of existing definitions focused on information systems. The reviewed formed a foundation from which a definition was drawn. Following nomenclature resolution, table 4.1 was formed; factors for classifying frameworks and methodologies are proposed. Nomenclature resolution (section 4.1.2) is one key outcome from which the remainder of the research is based.

RQ1-a): What characteristics define a penetration testing framework?

RQ1-b): What characteristics define a penetration testing methodology?

Section 4.1.2 extended upon section 4.1.1. Characteristics of framework and methodology are identified respectively, thus table 4.2 presents a Framework vs. Methodology characteristics matrix.

Based on the outcome of table 4.1 (classifying factors) and table 4.2 (frameworks vs. methodology), a gap analysis was possible. A review of nine methodologies and frameworks was performed with the goal of classifying each into either the “framework” or “methodology” category based on their focus and ability to act as a framework. This was successful inasmuch as certain frameworks identified as either incorrectly named or incomplete that consequently, restricted their use. From the gap analysis it was plausible to select two suitable frameworks (ISSAF and OTG) shown in table 4.4 that could be put into practice.

RQ2: What quality metrics can be defined for penetration testing frameworks?

Section 4.3 reviewed five existing quality models with the aim of understanding what quality metrics could potentially be suitable for penetration testing frameworks, and, what metrics could be used to measure quality. Table 4.5 (Quality Metric Comparison) showed that maintainability, portability, reliability, reusability, and testability were the most common across the five models reviewed. After analysis it was determined that not all of these metrics were suitable, thus a proposed metrics set is defined (see table 4.5) based on aspects of penetration testing frameworks. The candidate metrics outlined in table 4.5 formed the basis for a proposed quality model depicted in figure 4.2.

RQ3: How can penetration testing frameworks be evaluated using quality metrics?

Section 4.4 implements the proposed quality model (see table 4.5) against the two frameworks ISSAF and OTG. Each top-level metric (characteristic) and its supporting sub-characteristics mapped well to the two frameworks undergoing evaluation. Qualitative and quantitative methods are used to measure various aspects of quality for the two frameworks successfully which suggests that quality evaluation is possible.

## **5.2 Critical Review of the Research**

The case study approach was preferred for this research due to the invaluable experience that can be gained from real-world projects. The AU2EU project consisted of two third-party providers (DRK and CSIRO). Regrettably, both third-party providers did not deliver the resources necessary for this research to be completed as originally intended. While the research did complete, the impact of late or non-existent project resources (see section 4.6) far outweighed the benefits of the case study approach, in other words, it would have been far more beneficial to use the mitigation strategy (IBM implementation) from the onset.

Further to project limitations, it was discovered that technical expertise within the penetration

testing discipline is a crucial factor prior to undertaking research that involves practical application. Technical ability to interpret results is one primary aspect that has the potential to inhibit a given project as discovered throughout this research. While some tests did produce results as expected, additional research was required to interpret the result. Put another way, it was not enough to execute a command; moreover the output would play a major role on the next set of tests. The level of experience required for penetration testing projects covers not only technical expertise but incorporates other disciplines, for instance, project management, report writing, and communications skills, among others. This suggests that a security team rather than an individual practitioner would be beneficial if penetration testing projects are to be more effective. For inexperienced penetration testers, effort to understand a framework's structure and procedures is a pre-requisite prior to undertaking a particular project. The result of using a mature framework is well worth the effort given the complexity of penetration testing.

In conclusion, the research was successful inasmuch as, the research questions were addressed, however, it would have been an advantage to gain more experience within the cyber security discipline prior to this research. In addition, ensure a mitigation strategy was in place from the onset should third-party providers falter on deliverables.

### **5.3 Future Work**

This research revealed several areas worthy of future research. Defining methodologies and frameworks for a specific purpose proved to be challenging, however, understanding these concepts in the correct context can help clear any confusion for penetration testers not yet experienced with commercial projects. Repeating the quality evaluation in particular, against penetration testing project(s) might prove valuable for the cyber security discipline by means of external validity. Validity of the models proposed could potentially add to the body of knowledge should a peer-reviewed quality model be put forth.

Throughout the literature reviewed for this research it was not uncommon for published works to cite penetration testing frameworks and methodologies that underwent review for this research. For example; CREST (2012) provides a Penetration Testing Services Procurement Guide of which refers to OWASP, PTES, and OSSTMM as sources. NIST 800-115 (a standard reviewed in this research) refers to OSSTMM and OWASP, among others. While it is outside the scope to list all the cross referencing among the publications reviewed, it is suffice to say that a one-size fits all approach to penetration testing projects is not practical. In addition, there appears to be no internationally agreed upon best practice or standard to follow, therefore, a useful resource that

could extend upon this research is a taxonomy of frameworks and/or methodologies that map to a problem domain. For instance, web application testing projects would certainly benefit from a direct mapping to OWASPS's OTG, ISSAF's planning and preparation phase is far more comprehensive compared to the other eight frameworks reviewed, and PTF 0.59 provides a comprehensive reporting resource. In other words, even though a certain framework might not meet all requirements for a particular project, there are certainly aspects of each framework reviewed in this research that should not be shelved. Finally, without correct interpretation of results, penetration testing projects can provide a false sense of security, moreover, the value of penetration testing lies in the results. Furthermore, reporting aspects of penetration testing are vastly overlooked. Future research could extend upon this research by evaluating expected outputs and reporting, thus include these aspects as part of quality criteria.

## References

- Abdel-Aziz, A. (2011). Scoping Security Assessments - A Project Management Approach. Retrieved from <http://www.sans.org/reading-room/whitepapers/>
- Al-Badareen, A. B., Selamat, M. H., Jabar, A. J., Jamilah, D., & Sherzod, T. (2011). Software Quality Models: A Comparative Study *Software Engineering and Computer Systems - Second International Conference, held in Malaysia*: ResearchGate. Retrieved from [http://www.researchgate.net/publication/220868681\\_Software\\_Quality\\_Models\\_A\\_Comparative\\_Study](http://www.researchgate.net/publication/220868681_Software_Quality_Models_A_Comparative_Study)
- Al-Qutaish, R. E., & Abran, A. (2011). A Maturity Model of Software Product Quality. *Journal of Research and Practice in Information Technology*, 43(4), 307-327. Retrieved from <http://ecu.summon.serialssolutions.com/>
- Alcorn, W. (2007). Inter-Protocol Exploitation. Retrieved from [http://www.dcs.co.jp/security/NGS\\_freedomloads/InterProtocolExploitation.pdf](http://www.dcs.co.jp/security/NGS_freedomloads/InterProtocolExploitation.pdf)
- Alcorn, W., Orru, M., Coles, B., Passmore, B., & Pilkington, H. (2012). BeEF in 2012: An Introduction. Retrieved from <https://http://www.owasp.org/images/e/e0/Owasp2012-MarkPiper.pdf>
- Allen, L. (2012). *Advanced Penetration Testing for Highly-Secured Environments: The Ultimate Security Guide*. Birmingham: Packt Publishing, Limited.
- Australian Crime Commission. (2013). *Cyber and Technology Enabled Crime*. Canberra: Commonwealth of Australia.
- Australian Crime Commission. (2015). *Organised Crime in Australia 2015*. C. o. Australia. Retrieved from <https://crimecommission.gov.au/sites/default/files/FINAL-ACC-OCA2015-180515.pdf>
- Australian Signals Directorate. (2014). Strategies to Mitigate Targeted Cyber Intrusions. Retrieved from <http://www.asd.gov.au/infosec/top-mitigations/mitigations-2014-table.htm>
- Avison, D., & Fitzgerald, G. (2006). *Information Systems Development: Methodologies, Techniques and Tools*. London: McGraw-Hill.
- Berander, P., Damm, L.-O., Eriksson, J., Gorschek, T., Henningsson, K., Jönsson, P., . . . Rönkkö, K. (2005). Software Quality Attributes and Trade-offs. *Blekinge Institute of Technology*,
- CERT. (2013). *Cyber Crime and Security Report 2013*. C. Australia.
- CREST. (2012). Penetration Testing Services Procurement Guide.
- Cyber Security Market worth \$170.21 Billion by 2020 (2015). [Press release]. Retrieved from <http://www.marketsandmarkets.com/PressReleases/cyber-security.asp>
- Deutsch, S. M., & Willis, R. R. (1988). *Software Quality Engineering A Total Technical and Management Approach*: Prentice Hall.
- Dhouha, A. F., Johnstone, M. N., Camenisch, J., Ignatenko, T., Koster, P., Lange, B., . . . CSIRO. (2014). Authentication and Authorisation in Entrusted Unions.
- Dromey, G. R. (1995). A Model for Software Product Quality. *Software Engineering, IEEE Transactions on*, 21(2), 146-162.
- DuBay, W. H. (2004). The Principles of Readability. Retrieved from <http://www.impact-information.com/impactinfo/readability02.pdf>
- "Framework". (Ed.) (2015) Oxford Dictionaries.
- Frankland, J. (2009). The importance of standardising methodology in penetration testing. *Database and Network Journal*, 39(3), 13. Retrieved from <http://ecu.summon.serialssolutions.com>
- Galliers, R. D. (1990). Choosing Appropriate Information Systems Research Approaches: A Revised Taxonomy. In H.-E. a. H.-S. Nissen, R (Ed.), *The Information Systems Research Arena of the 90's: Challenges, Perceptions and Alternative Approaches - Proc. IFIP TC8 WG8.2 Conference, held in Copenhagen, Denmark*, (pp. 155-173).
- Gartner. (2016). Gartner Says Worldwide IoT Security Spending to Reach \$348 Million in 2016. Retrieved from <http://www.gartner.com/newsroom/id/3291817>

- Golden, B. (2008). Open Source Maturity Model. *The Open Source Business Resource* 4. Retrieved from <http://ecu.summon.serialssolutions.com/>
- Gunning, R. (1952). *The Technique of Clear Writing*: McGraw-Hill.
- Heitlager, I., Kuipers, T., & Visser, J. (2007). *A Practical Model for Measuring Maintainability*. doi: 10.1109/QUATIC.2007.8
- Holik, F., Horalek, J., Marik, O., Neradova, S., & Zitta, S. (2014, 2014). *Effective penetration testing with Metasploit framework and methodologies*. Paper presented at the 15th IEEE International Symposium on Computational Intelligence and Informatics, Budapest, Hungary. doi: 10.1109/CINTI.2014.7028682
- Institute of Electrical and Electronics Engineers. (1990). *IEEE Std 610.12-1990, Standard Glossary of Software Engineering*
- Institute of Electrical and Electronics Engineers. (2011). *ISO/IEC/IEEE 24765 Systems and Software Engineering Vocabularly*. Switzerland: ISO.
- Internet Archive. (2007). WayBackMachine. Retrieved from <https://web.archive.org/web/20070208091506/http://www.oisssg.org/content/view/71/71/>
- ISECOM, I. o. S. a. O. M. (2000). *Open Source Security Testing Methodology*: ISECOM. Retrieved from <http://www.isecom.org>
- ISO, & IEC. (2002). *ISO/IEC 9126-3: Software engineering - Product quality* ISO / IEC.
- Johnson, R. (1997). Frameworks = (components + patterns) (Vol. 40, pp. 39-42): ACM.
- Kang, B.-H. (2008). About Effective Penetration Testing Methodology. *Journal of Security Engineering*, Retrieved from [http://www.sersc.org/journals/JSE/vol5\\_no5\\_2008/8.pdf](http://www.sersc.org/journals/JSE/vol5_no5_2008/8.pdf)
- Kissel, R. (2013). *Glossary of Key Information Security Terms*. United States: NIST Computer Security Division. doi: <http://dx.doi.org/10.6028/NIST.IR.7298r2>
- Klare, G. R. (2000). The Measurement of Readability: Useful Information for Communicators. *ACM Journal of Computer Documentation*, 24(3), 15.
- Land, R. (2002). How can frameworks facilitate component reuse? *Extended Report for I. Crnkovic and M. Larsson (editors), "Building Reliable Component-Based Systems"*, Artech House,
- Laplante, P. A. (Ed.) (2001) *Computer Science Engineering and Technology*. Florida, USA: CRC Press.
- Lawson, L., Byrne, M., Doraiswamy, A., & Ouchn, N. (n.d). Penetration Testing Framework 0.59. Retrieved from <http://www.vulnerabilityassessment.co.uk>
- LeMay, R. (2005). Nessus Security Tool Closes its Source. Retrieved from <http://www.cnet.com/news/nessus-security-tool-closes-its-source/>
- Li, P., & Rao, H. R. (2007). An examination of private intermediaries' roles in software vulnerabilities disclosure. *Information Systems Frontiers*, 9(5), 531-539. doi: 10.1007/s10796-007-9047-2
- Ludger, M., & Gottron, T. (2012). Readability and the Web. doi: 10.3390/fi4010238
- Manadhata, P. K. (2008). *An Attack Surface Metric*. (Doctor of Philosophy). University of North Carolina, Pittsburgh. Retrieved from <http://reports-archive.adm.cs.cmu.edu/anon/2008/CMU-CS-08-152.pdf>
- "Maintain". (2015). Oxford Dictionaries. *Oxford Dictionaries*, Retrieved from <http://www.oxforddictionaries.com/definition/english/maintain>
- McCall, A. J., Richards, K. P., & Walters, F. G. (1977). *Factors in Software Quality*. New York: McGraw, G., Miguez, S., & West, J. (2009). Building Security in Maturity Model. Retrieved from <https://http://www.bsimm.com>
- Midian, P. (2003). Perspectives on Penetration Testing — Finding the Right Supplier. *Network Security*, 2003(2), 9-11. doi: 10.1016/S1353-4858(03)00210-1
- Miguel, J. P., Mauricio, D., & Rodriguez, G. (2014). A Review of Software Quality Models for the Evaluation of Software Products. doi: 10.5121/ijsea.2014.5603
- Miles, G., Rogers, R., Fuller, E., Hoagberg, M. P., & Dykstra, T. (2004). *Security Assessment: Case Studies for Implementing the NSA IAM*. US: Syngress Media Incorporated.
- Mnkandla, E. (2009). About software engineering frameworks and methodologies *IEEE Africon*



- 2009, held in 2009 (pp. 1-5). doi: 10.1109/AFRCON.2009.5308117
- Muller, A. (2015). OWASP Testing Guide. In A. Shanley (Ed.).
- Navica. (2004). Open Source Maturity Model. In OSMMChart (Ed.): Navica.
- Nickerson, C., Kennedy, D., Riley, C., Smith, E., Amit, I., Rabie, A., . . . Strand, J. (n.d). Penetration Testing Execution Standard Retrieved from [http://www.pentest-standard.org/index.php/PTES\\_Technical\\_Guidelines](http://www.pentest-standard.org/index.php/PTES_Technical_Guidelines)
- NIST. (2008). *Technical Guide to Information Security Testing and Assessment NIST 800-115*: NIST.Retrieved from <http://csrc.nist.gov/publications>
- Northcutt, S. (2011). The Attack Surface Problem. Security Laboratory: Defense In Depth Series. Retrieved from <http://www.sans.edu/research/security-laboratory/article/did-attack-surface>
- Novell. (2015). Products. Retrieved from <https://http://www.novell.com/products/>
- Offensive Security. (2015). Kali Linux Tools Listing. Retrieved from <http://tools.kali.org/tools-listing>
- OISSG, O. I. S. S. G. (2005). *Information Systems Security Assessment Framework* OISSG.Retrieved from <http://sourceforge.net/projects/isstf/>
- Ottow, C., van Vliet, F., de Boer, P.-T., & Pras, A. (2012). The Impact of IPv6 on Penetration Testing. In R. Szabó & A. Vidács (Eds.), *Information and Communication Technologies* (Vol. 7479, pp. 88-99): Springer Berlin Heidelberg. doi: 10.1007/978-3-642-32808-4\_9
- OWASP. (2014a). *OWASP Testing Guide* Retrieved from [https://http://www.owasp.org/index.php/OWASP\\_Testing\\_Guide\\_v4\\_Table\\_of\\_Contents](https://http://www.owasp.org/index.php/OWASP_Testing_Guide_v4_Table_of_Contents)
- OWASP. (2014b). Revision history of "OWASP Testing Guide v4 Table of Contents". Retrieved from [https://http://www.owasp.org/index.php?title=OWASP\\_Testing\\_Guide\\_v4\\_Table\\_of\\_Contents&offset=20140805133530&limit=250&action=history](https://http://www.owasp.org/index.php?title=OWASP_Testing_Guide_v4_Table_of_Contents&offset=20140805133530&limit=250&action=history)
- Oxford. (Ed.) (2008) Oxford Dictionary of Computing (6 ed.). Oxford University Press.
- Patteson, C. (2013). *Cyber Security - the facts*. Australia: CERT Australia.
- Piedad, F., & Hawkins, M. (2001). *High Availability: Design, Techniques, and Processes*: Prentice Hall.
- Readability-Score. (2015). Retrieved from <https://readability-score.com/>
- Security Testing Market Worth \$4.96 Billion by 2019. (2014). *PR Newswire Europe Including UK Disclose*,Retrieved from <http://ezproxy.ecu.edu.au/login?url=http://search.proquest.com/docview/1552894115>
- Shah, S., & Mehtre, B. M. (2014). An Overview of Vulnerability Assessment and Penetration Testing Techniques. doi: 10.1007/s11416-014-0231-x
- Sommerville, I. (2007). **Critical Systems Software Engineering** (8 ed., pp. 49): Addison-Wesley.
- Standards Australia. (2000). *98.1:2000 ISO/IEC 145 Information Technology Software Product Evaluation*
- Standards Australia. (2005). *AS/NZS ISO/IEC 9126.1:2005 Software engineering—Product quality*
- Standards Australia. (2013). *AS/NZS ISO/IEC 25010:2013*
- Tang, A. (2014). A Guide to Penetration Testing. *Network Security*, 2014(8), 8. doi: 10.1016/S1353-4858(14)70079-0
- "Technique". (2015). Oxford Dictionaries. Retrieved from <http://www.oxforddictionaries.com/definition/english/technique>
- Valli, C., Woodward, A., Hannay, P., & Johnstone, M. (2014). Why Penetration Testing Is A Limited Use Choice For Sound Cyber Security Practice. *Proceedings of the Conference on Digital Forensics, Security and Law U6* 35. Retrieved from <http://ecu.summon.serialssolutions.com/>
- Warner, N. (2012). ASIS at 60. Canberra, Australia: ASIS.
- Wilhelm, T. (2009). Professional Penetration Testing : Volume 1: Creating and Learning in a Hacking Lab (Vol. 1, pp. 26). Burlington: Syngress. Retrieved from <http://ecu.summon.serialssolutions.com/>
- Yeo, J. (2013). Using penetration testing to enhance your company's security. *Computer Fraud & Security*, 2013(4), 17-20. doi: [http://dx.doi.org/10.1016/S1361-3723\(13\)70039-3](http://dx.doi.org/10.1016/S1361-3723(13)70039-3)

## Appendix A: Operability Metrics

Internal Operability metrics									
Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metric scale type	Measure type	Input to measurement	ISO/IEC 12207 SLCP Reference	Target audience
<b>Input validity checking</b>	What proportion of input items provide check for valid data	Count the number of input items, which check for valid data and compare with the number of input items, which could check for valid data	X=A/B	0 <= X <= 1	Absolute	X=count/co	Req spec	Verification	Developers
			A=Number of input items which check for valid data	The closer to 1, the better.		unt	Design	Joint review	Requirers
			B=Number of input items which could check for valid data			A=count	Review report		
<b>User operation cancellability</b>	What proportion of functions can be cancelled prior to completion?	Count the number of implemented functions, which can be cancelled by the user prior to completion and compare it with the number of functions requiring the precancellation capability	X=A/B	0 <= X <= 1	absolute	X=count/co	Req spec	Verification	Developers
			A=Number of implemented functions which can be cancelled by the user	The closer to 1, the better cancellability		unt	Design	Joint review	Requirers
			B= Number of functions requiring the precancellation capability			A=count	Review report		
<b>User operation Undoability</b>	What proportion of functions can be undone?	Count the number of implemented functions, which can be undone by the user after completion and compare it with the number of functions	X=A/B	0 <= X <= 1	absolute	X=count/co	Req spec	Verification	Developers
			A=Number of implemented functions which can be undone by the user	The closer to 1, the better undoability		unt	Design	Joint review	Requirers
			B= Number of functions.			A=count	Review report		
<b>NOTE:</b> : Either single undoability or multiple undoability after several subsequent actions can be assessed									
<b>Customisability</b>	What proportion of functions can be customised during operation?	Count the number of implemented functions, which can be customized by the user during operation and compare it with the number of functions requiring the customization capability	X=A/B	0 <= X <= 1	absolute	X=count/co	Req spec	Verification	Developers
			A=Number of functions which can be customised during operation	The closer to 1, the better customisability		unt	Design	Joint review	Requirers
			B=Number of functions requiring the customization capability			A=count	Review report		
<b>Physical accessibility</b>	What proportion of functions can be customised for access by users with physical handicaps	Count the number of implemented functions, which can be customised and compare it with the number of functions	X=A/B	0 <= X <= 1	absolute	X=count/co	Req spec	Verification	Developers
			A=Number of functions which can be customised	The closer to 1, the better physical accessibility		unt	Design	Joint review	Requirers
			B=Number of functions			A=count	Review report		

**NOTE:** Examples of physical accessibility are inability to use a mouse and blindness

## Appendix B: OTG Field Notes

### OTG-INFO-002: Fingerprint Webserver

The objective of this test is to determine the version and type of web server running to discover known vulnerabilities, thus determine potential exploits. The recommended tool for fingerprinting is netcat. In addition to netcat, socat was used given the target interface implements https. Socat was not listed as any of the recommended tools to use in the OWASP Testing Framework.

1 Command: ncat 195.37.154.37 10504

HTTP/1.1 400 Bad Request

Server: nginx/1.9.4

Date: Thu, 24 Sep 2015 12:05:12 GMT

Content-Type: text/html

Content-Length: 270 Connection: close

2 Command ncat 195.37.154.37 10501

HTTP/1.1 404 Not Found

Server: nginx/1.4.6 (Ubuntu)

Date: Thu, 24 Sep 2015 12:12:08 GMT

Content-Type: text/plain; charset=utf-8

Content-Length: 52

Connection: close

X-Backside-Transport: FAIL FAIL

X-Cf-Requestid: 6682e50f-f6b8-42bb-45d2-5467e68fa989

X-Cf-Routererror: unknown\_route X-Client-IP: 195.37.154.37 X-Global-Transaction-ID:  
459356431

3 socat - openssl-connect:195.37.154.37:10504,verify=0

HEAD / HTTP/1.0 HTTP/1.1 302 Moved Temporarily

Server: nginx/1.9.4

Date: Fri, 25 Sep 2015 07:28:15 GMT

Content-Type: text/plain; charset=utf-8

Content-Length: 37

Connection: close

X-Powered-By: Express

Location: /ui

Vary: Accept, Accept-Encoding

Set-Cookie:

session=aQzVGg8p1cEd76OrowdZ2g.ajAQTyYqkwnZqBMulDnv9Vep1K8DZzL44JTBs86u4Bk.1  
443166095343.86400000.p3hsiX1mP3UFUdf6Hx2i9QrGrg0LFtefXC\_irX4mKoE;  
path=/; expires=Sat, 26 Sep 2015 07:28:16 GMT; httponly

4 Netcraft Result [http://toolbar.netcraft.com/site\\_report?url=http://au2eu.nlehd.de](http://toolbar.netcraft.com/site_report?url=http://au2eu.nlehd.de)

### **OTG-INFO-003: Review Webserver Metafiles for Information Leakage**

The purpose of this test is to identify information leakage of the web applications directory structure. The purpose is to interrogate the robots.txt file to find any information about directories to be avoided by web crawlers, consequently to map the web applications directory structure.

Command: `wget au2eu.nlehd.de:10501/robots.txt`

(tried browser access using http and https) on port 10504

400 - bad request on port 10504 404 Not found

No robots.txt file found

### **OTG-INFO-004: Enumerate Web Applications on Webserver**

Enumerate web applications within scope that exist on the web server. The recommended tools are nmap

Command: `nmap -sV -O -p10500 10510`

Nmap scan report for au2eu.nlehd.de (195.37.154.37) Host is up (0.29s latency).

PORT	STATE	SERVICE	VERSION
------	-------	---------	---------

10500/tcp	closed	unknown	
-----------	--------	---------	--

10501/tcp	open	http	nginx 1.4.6 (Ubuntu)
-----------	------	------	----------------------

10502/tcp	open	http	nginx 1.4.6 (Ubuntu)
-----------	------	------	----------------------

10503/tcp	open	unknown	
-----------	------	---------	--

10504/tcp	open	ssl/http	nginx 1.9.4
-----------	------	----------	-------------

10505/tcp	closed	unknown	
-----------	--------	---------	--

10506/tcp closed unknown

10507/tcp closed unknown

10508/tcp closed unknown 10509/tcp closed unknown


10510/tcp open ssh OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.3 (Ubuntu Linux; protocol 2.0)

1 service unrecognized despite returning data.

Nessus was recommended for this task however the application was not available at the time of testing.

To identify name servers OWASP further recommends using online services for, DNS lookups and reverse DNS using online tools, for example searchdns.netcraft.com.

## Search DNS Results

<b>Site</b>	<a href="http://au2eu.nlehd.de">http://au2eu.nlehd.de</a>	<b>Netblock Owner</b>	<a href="#">NEC Europe Ltd.</a>
<b>Domain</b>	<a href="#">nlehd.de</a>	<b>Nameserver</b>	kyoto.neclab.eu
<b>IP address</b>	195.37.154.37	<b>DNS admin</b>	postmaster@neclab.eu
<b>IPv6 address</b>	<i>Not Present</i>	<b>Reverse DNS</b>	au2eu.nlehd.de
<b>Domain registrar</b>	denic.de	<b>Nameserver organisation</b>	<i>unknown</i>
<b>Organisation</b>	<i>unknown</i>	<b>Hosting company</b>	neclab.eu
<b>Top Level Domain</b>	Germany (.de)	<b>DNS Security Extensions</b>	<i>unknown</i>
<b>Hosting country</b>	 DE		

Name Servers:

- [kyoto.neclab.eu](#)
- [ws-lei1.win-ip.dfn.de](#)

Additional Information

[Enrico.Giakas@netlab.nec.de](mailto:Enrico.Giakas@netlab.nec.de)

[Thomoas.dietz@nw.neclab.eu](mailto:Thomoas.dietz@nw.neclab.eu)

## DNS lookup using command the command line

Command: `host -t ns au2eu.nlehd.de`

result: `au2eu.nlehd.de has no NS record`

## Zone transfer attempt

Command 1: `host -l au2eu.nlehd.de kyoto.neclab.eu`

Command 2: `host -l au2eu.nlehd.de ws-lei1.win-ip.dfn.de`

## **Result 1**

Using domain server:

Name: kyoto.neclab.eu

Address: 195.37.70.24#53

Aliases: Host au2eu.nlehd.de not found: 9(NOTAUTH)  
; Transfer failed.

## **Result 2**

Using domain server:

Name: ws-lei1.win-ip.dfn.de

Address: 195.37.70.24#53

Aliases: Host au2eu.nlehd.de not found: 9(NOTAUTH)  
; Transfer failed.

## **OTG-INFO-005: Review Webpage Comments**

A review of HTML source code was conducted however no comments relevant to the web application were found. There is however javascript code embedded into the webpage that appears to control the authentication process. Source code inspection of this size is outside the scope of this research however it is important to note that some directory information was found within the code therefore assist with the directory/path mapping of the web application.

Folder: api/v0/emergency

## **OTG-INFO-006: Identify Application Entry Points**

The objective of this test is to understand how requests are formed and analyse the typical response received from the web application.

Attempted test using tamper data firefox plugin and Burpe suite. Unable to obtain a result

## **OTG-INFO-007: Map Execution Paths through application**

Map the target application and understand the principle workflows

Webscarab: POST request goes to ojsp.pca.dfn.de/

<http://ocsp.pca.dfn.de/OCSP-Server/OCSP>

## **OTG-INFO-008: Fingerprint Web Application Framework**

The aim is to identify the type of Web framework.

The target system is powered by Express, node.js. A standard web application framework for node.js running Nginx 1.9.4

Recommended tools: netcat, BlindElephant

- 1 Command: nc 195.37.154.37 10504  
HTTP/1.1 400 Bad Request  
Server: nginx/1.9.4  
Date: Tue, 29 Sep 2015 09:34:23 GMT  
Content-Type: text/html Content-Length: 270  
Connection: close
- 2 BlindElephant.py https://au2eu.nlehd.de:10504 guess  
No results found

### OTG-INFO-009: Fingerprint Web Application

The following tests attempt to identify the web application. Recommended tools are:

- BlindElephant
- Wappalyser
- Whatweb

- 1 BlindElephant : no result
- 2 Wappalyser  
Web Server: Ngix Web framework: Express node.js
- 3 Whatweb 195.37.154.37:10504

### Result for port 10504

```
https://195.37.154.37:10504 [302] Cookies[session], Country[GERMANY][DE],  
HTTPServer[nginx/1.9.4], HttpOnly[session], IP[195.37.154.37], RedirectLocation[/ui],  
X-Powered-By[Express], nginx[1.9.4] https://195.37.154.37:10504/ui [200]  
Cookies[session], Country[GERMANY][DE], HTTPServer[nginx/1.9.4],  
HttpOnly[session], IP[195.37.154.37], X-Powered-By[Express], nginx[1.9.4]
```

### Result for port 10501

whatweb <http://195.37.154.37:10501>

<http://195.37.154.37:10501>

[404] Country[GERMANY][DE], HTTPServer[Ubuntu Linux][nginx/1.4.6 (Ubuntu)], IBM-WebSphere-DataPower[FAIL], IP[195.37.154.37], UncommonHeaders[x-backside-transport,x-cf-requestid,x-cf-routererror,x-client-ip,x-global-transaction-id]

whatweb -c=session <https://195.37.154.37:10504>

<https://195.37.154.37:10504> [302] Cookies[session], Country[GERMANY][DE], HTTPServer[nginx/1.9.4], HttpOnly[session], IP[195.37.154.37], RedirectLocation[/ui], X-Powered-By[Express], nginx[1.9.4] <https://195.37.154.37:10504/ui> [200] Cookies[session], Country[GERMANY][DE], HTTPServer[nginx/1.9.4], HttpOnly[session], IP[195.37.154.37], X-Powered-By[Express], nginx[1.9.4]

## Webscarab

HTTP/1.1 200 OK

Date: Thu, 01 Oct 2015 12:45:11 GMT

Server: Apache Last-Modified: Thu, 01 Oct 2015 12:45:11 GMT

expires: Sun, 11 Oct 2015 12:45:11 GMT

ETag: b193f875140f88487d1baf9bcce1879bcf85b1ec

cache-control: max-age=864000, public, no-transform, must-revalidate

Content-length: 1541

Keep-Alive: timeout=15, max=100

Connection: Keep-Alive

Content-Type: application/ocsp-response

## NIKTO

root@kali:~# nikto -ssl -host 195.37.154.37 -port 10504 -

Nikto v2.1.6

Target IP: 195.37.154.37 +

Target Hostname: 195.37.154.37 +

Target Port: 10504 +

SSL Info: Subject: /C=DE/ST=Baden-Wuerttemberg/L=Heidelberg/O=NEC Europe Ltd./CN=au2eu.nlehd.de Ciphers: ECDHE-RSA-AES256-GCM-SHA384

Issuer: /C=DE/O=NEC Europe Ltd./OU=NEC Laboratories Europe/CN=NECLAB-CA/emailAddress=zertifizierungsstelle@nw.neclab.eu +



Start Time: 2015-10-02 20:54:45 (GMT-4)

Server: nginx/1.9.4 + Cookie session created without the secure flag + Retrieved x-powered-by header: Express + The anti-clickjacking X-Frame-Options header is not present. + The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS + The site uses SSL and the Strict-Transport-Security HTTP header is not defined. + The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type + Root page / redirects to: /ui + Server leaks inodes via ETags, header found with file /F5lQey1l.Htm, fields: 0xW/16a0xLsvGzNKFbcFGil1LQQectA + No CGI Directories found (use '-C all' to force check all possible dirs) + Hostname '195.37.154.37' does not match certificate's names: au2eu.nlehd.de + Allowed HTTP Methods: GET, HEAD + OSVDB-3093: /uifc/MultFileUploadHandler.php+: This might be interesting... has been seen in web logs from an unknown scanner. + 7671 requests: 0 error(s) and 10 item(s) reported on remote host +

End Time: 2015-10-03 07:58:27 (GMT-4) (39822 seconds) + 1 host(s) tested

nikto -host http://ocsp.pca.dfn.de:80/OCSP-Server/OCSP -

Nikto v2.1.6

+ Target IP: 193.174.13.86 + Target Hostname: ocsp.pca.dfn.de + Target Port: 80 + Start Time: 2015-10-01 08:55:14 (GMT-4)

+ Server: Apache + The anti-clickjacking X-Frame-Options header is not present. + The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS + The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type + No CGI Directories found (use '-C all' to force check all possible dirs) + Allowed HTTP Methods: GET, HEAD, POST, PUT, DELETE, OPTIONS, TRACE + OSVDB-397: HTTP method ('Allow' Header): 'PUT' method could allow clients to save files on the web server. + OSVDB-5646: HTTP method ('Allow' Header): 'DELETE' may allow clients to remove files on the web server.

## Configuration Phase

### OTG-CONFIG-001: Test Network Infrastructure

Testing network infrastructure is outside the scope of this project however; port 10510 is identified as a possible administration entry point but will not be tested. SSH OpenSSH 6.6.1

### **OTG-CONFIG-003: Test File Extension Handling**

The purpose is to ensure file extension handling is adequate for the web application by testing known file extension in the browser.

Files identified and tested

ui.php            produced a blank page  
api.php           prompted for a username and password

### **OTG\_CONFIG-005: Enumerate infrastructure and Application Admin Interfaces**

Discovered port 10510 Open SSH 6.6.1 , Out of scope for testing

Directory and file enumeration for administrator interfaces.

Directories discovered using dirbuster and tamper (FF plugin)

/api  
/api.php  
/api/v0  
/api/v0/staff\_free  
/api/v0/client  
/api/v0/role  
/api/v0/status  
/api/v0/staff  
/api/v0/event  
/api/v0/event\_incl  
/uitleg  
/uitleg/uitleg.php  
/apidocs  
/apidocs.php  
/uic  
/uid.php  
/uid  
/uir  
/uic/uic.php  
/uis/  
/uis.php  
/ui\_images.php  
/ui\_images

### **OTG\_CONFIG-006: Test HTTP Methods**

Test to see what http commands are supported. If the trace command is allowed there is potential for cross-site scripting vulnerabilities

Netcat is the recommended tool, however no significant results were returned, thus socat was used as an alternative to cater for https

```
Command: socat - openssl-connect:195.37.154.37:10504,verify=0
OPTIONS / HTTP/1.1
Host:195.37.154.37 HTTP/1.1 200 OK
Server: nginx/1.9.4
Date: Fri, 25 Sep 2015 08:42:43 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 8
Connection: keep-alive X-
Powered-By: Express
Allow: GET,HEAD
ETag: W/"8-8ww6QOmj5lyGjHVKXelZGQ"
SetCookie:session=H7Go6yfo5eydUNtk91WWIA.n3iBkZ_VVWIVm4knV_OVIyjNl3MTv-
BnsxXePRCIUoQ.1443170563408.86400000.c-Ci4_Nh9H5JliNRUzL5PjfFePLYukOC9Wwq-
4VTmA8; path=/; expires=Sat, 26 Sep 2015 08:42:44 GMT; httponly Vary: Accept-Encoding
GET,HEAD
```

The trace command was still tested even though the option was not permitted.

```
Command TRACE / HTTP/1.1
Host:195.37.154.37
```

### **Result: for port 10501**

```
HTTP/1.1 405 Not Allowed Server: nginx/1.4.6 (Ubuntu) Date: Tue, 29 Sep 2015 12:17:40 GMT
Content-Type: text/html Content-Length: 181 Connection: close
```

### **Result for port 10504**

```
HTTP/1.1 405 Not Allowed Server: nginx/1.9.4 Date: Tue, 29 Sep 2015 12:20:04 GMT Content-
Type: text/html Content-Length: 172 Connection: close
```

## **Check for JEFF Vulnerability using socat**

JEFF / HTTP/1.1

Host:195.37.154.37:10504

response: 502 Bad Gateway

## **OTG-CONFIG-007: Test HTTP Strict Transport Security**

The HTTP Strict Transport Security (HSTS) header is a mechanism that websites have to communicate to the browser that all traffic exchanged with a given domain must always be sent over https, thus will help information from being sent over unencrypted requests

Test for strict transfer security:

```
curl -s -D- https://au2eu.nlehd.de:10504 | grep Strict
```

Response: none

## **Identification**

### **OTG-IDENT-004: Account Enumeration and Guessable User Account**

The purpose of this test is to verify whether it is possible to collect valid usernames by interacting with the authorization system, thus whether it is possible to use brute force to find the corresponding password.

Test 1: Record the server response when a valid username is entered

Test 2: Test for valid user with wrong password

Test 3: test for non-existent username: Result, authentication pop up, no result

### URI Probing

Test whether or not a URI can be accessed directly to detect authorization error messages. Based on the directory fingerprinting various URIs were tested, resulting in an unauthorized error message. The error yielded no significant result.

### Guessing Users

To determine whether a valid user and invalid user produce the same error code

### **OTG-IDENT-005: Testing for weak of unenforced username Policy**

Determine the structure of account names and evaluate the systems response to valid and invalid

account names. Use account dictionaries to enumerate a valid account. Ensure the application returns consistent generic error messages in response to invalid accounts, passwords or other credentials entered during the login process.

Result: No credentials provided to perform these tests

### **OTG-AUTHN-001: Testing for Credentials Transported over an Encrypted Channel**

The scope of this test is to verify whether the user credentials are transferred via an encrypted channel, for instance, https. By using tools such as webscarab and burpe suite, interception of the traffic can verify whether or not the data is encrypted.

#### **Test 1: Webscarab: program failed**

#### **Test 2: Burpe Suite Result: Verified data was encrypted.**

GET /api/v0/status HTTP/1.1

Host: 195.37.154.37:10504

User-Agent: Mozilla/5.0 (X11; Linux i686; rv:31.0) Gecko/20100101 Firefox/31.0 Iceweasel/31.8.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8

Accept-Language: en-US,en;q=0.5

Accept-Encoding: gzip, deflate

Referrer: https://195.37.154.37:10504/ui

Cookie:

session=vu5\_pov5ETF3OaYgsIVzEg.dGWclpRTm9BF7u375sAW\_cPZjxfJbN3OvfZ3AVUGHc8.1444199749990.86400000.Gky0MNXbGkcfNB1r9oB5jsgPZX5XGmCwh8ely9KOSUE

Connection: keep-alive

Cache-Control: max-age=0

Authorization: Basic YWRtaW46cGFzc3dvcmQ=

### **OTG-AUTHN-003: Testing for Weak lock out mechanism**

Used to mitigate against brute force attacks. Testing involves attempting to authenticate with invalid credentials for a particular user, thus determine if any lock out procedures are in place.

#### **Test:**

login with incorrect password 3 to 10 times or more, followed by a successful login to verify lockout functionality is triggered.

Result: 10 attempts were made using admin, no lockout mechanism activated. This needs to be tested with valid credentials

#### **OTG-AUTHN-004: Bypassing Authentication Schema**

Test to verify whether authentication can be bypassed by why of bypassing the login page and accessing pages directly via the browser (forced browsing). Knowledge of the directory structure is required. Parameter modification, session ID prediction and SQL injection are other recommended techniques.

##### **Test 1: Direct browser access**

Pages and directories discovered by dirbuster in the information gathering stage were used (see OTG\_CONFIG-005: Enumerate infrastructure and Application Admin Interfaces)

**Result:** Authentication required was encountered for every test

##### **Test 2: Parameter Modification**

Unable to complete test without knowledge of the POST and GET parameters using an authenticated user.

Determine whether directory traversal is possible

##### **Test**

1

<https://195.37.154.37:10504/%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f/etc/passwd>

**Test 2:** <https://195.37.154.37:10504/../../../../etc/passwd>

No result

#### **OTG-AUTHZ-003: Testing for privilege escalation**

Testing for privilege escalation. The purpose is to verify that a user can not modify his or her privileges or roles inside the application in ways that could allow privilege escalation attacks.

Out of scope, credentials required

#### **OTG-INPVAL-001: Testing for Reflected Cross site scripting**

Test for cross-site scripting attacks.

The following code was entered into both the login input box and password input box

### Test 1

<script>alert(123)</script>

>"<script>alert(123)</script>

**Result:** Not vulnerable, the authentication box was presented for a valid login

Unable to test the entire application due to account access.

### Test 2

Directly into the url

[https://195.37.154.37:10504/ui/ui.php?user=<script>>window.onload=function\(\){var](https://195.37.154.37:10504/ui/ui.php?user=<script>>window.onload=function(){var)

AllLinks=document.getElementsByTagName("a");AllLinks[0]href="http://badexample.com/malicious.exe";}</script>>

**Result:** Not vulnerable

### Test 3

[https://195.37.154.37:10504/ui/ui.php?user="%3cscript%3ealert\(document.cookie\)%3c/script%3e](https://195.37.154.37:10504/ui/ui.php?user=)

**Result:** Not vulnerable

### Test 4

[https://195.37.154.37:10504/ui/?var=<SCRIPT%20a=">"%20SRC="http://attacker/xss.js"></SCRIPT>](https://195.37.154.37:10504/ui/?var=<SCRIPT%20a=)

Result: Not vulnerable

### OTG-INPVAL-003: Testing for HTTP Verb Tampering

Determine if the server will accept other HTTP methods other than GET and POST.

Netcat and socat were used.. Socat was not a recommended tool for this test however it was used to cater for https.

**Test command:** socat - openssl-connect:195.37.154.37:10504,verify=0

OPTION / HTTP/1.1

Host:195.37.154.37

### Result

HTTP/1.1 401 Unauthorized Server: nginx/1.9.4 Date: Sat, 10 Oct 2015 07:27:46 GMT Transfer-

Encoding: chunked Connection: keep-alive X-Powered-By: Express WWW-Authenticate: Basic realm="Users" Set-Cookie: session=AuX3mh60LeVb1JuIK11yZg.J1Uk7c4FZ0TXk-b5ABxmIOZW7q5i45ba6z4eVAQBE3E.1444462066586.86400000.Dy9-BFXy0GwNRCd88041S9VctdRdOZI8pb36k88FWwA; path=/; expires=Sun, 11 Oct 2015 07:27:47 GMT; httponly c Unauthorized 0

**Test 2 – command:** nc 195.37.154.37 10504

### Result

```
HTTP/1.1 400 Bad Request Server: nginx/1.9.4 Date: Sat, 10 Oct 2015 07:29:08 GMT Content-Type: text/html Content-Length: 270 Connection: close <html> <head><title>400 The plain HTTP request was sent to HTTPS port</title></head> <body bgcolor="white"> <center><h1>400 Bad Request</h1></center> <center>The plain HTTP request was sent to HTTPS port</center> <hr><center>nginx/1.9.4</center> </body> </html>
```

Server is not vulnerable to HTTP verb tampering

## OTG-INPVAL-005: Testing for SQL Injection

### Test 1 SQL Injection Strings

```
1'or'1'=1'))/*
```

```
password = foo
```

```
1'or'1'=1')—
```

```
1'or'1'=1')—;
```

```
1' OR '1'=1))/*
```

```
1' OR '1'=1)))))/*
```

```
1' OR '1'=1)/*
```

```
1'or'1'='1'
```

```
1'or'1'='1' AND password='1'OR'1'='1'
```

```
1' or'1'=1'))LIMIT 1/* (Various combinations using — and brackets
```

### Test 2 Concatenation Techniques

Attempt to retrieve a database error message



MySQL: 'test'+ 'ing'  
SQL Server: 'test''ing'  
Oracle: 'test' || 'ing'  
PostgreSQL: 'test' || 'ing'

### Test 3          Union Exploitation Technique

Entered into the username field.

admin UNION ALL SELECT client FROM clients  
admin UNION ALL SELECT user FROM users  
angelika + UNION SELECT 1,null); /\*  
angelika AND UNION SELECT 1,null); /\*

Attempt order by clause to force response

admin ORDER BY 2;

### GET Strings

https://195.37.154.37/10504/ui/ui.php?user=julie UNION SELECT 1,1,null/\*  
https://195.37.154.37/10504/ui/ui.php?user=julie UNION SELECT 1,1,null—

<https://195.37.154.37/10504/ui/ui.php?user=1%27%20AND%20%271%27=%272%27>

Response: Server times out

### Error Base Technique

Variations of

https://au2eu.nlehd.de:10504/ui/ui.php?||UTL\_INADDR.GET\_HOST\_NAME((SELECT          user  
FROM USERS))/\*

Response:      produced blank page

Expected:      server error

https://au2eu.nlehd.de:10504/ui/ui.php?id=1orHTTP.REQUEST('testserver.com:80');

### Automated tests:      SQL MAP

Sqlmap -url ip.address:port/ui/php

Sqlmap -url https:195.37.154.37:10504/ui/ui.php?user=

## RESULT

[08:26:33] [WARNING] using un-escaped version of the test because of zero knowledge of the back-end DBMS. You can try to explicitly set it using option '--dbms' [08:28:22] [INFO] testing 'MySQL UNION query (NULL) - 1 to 10 columns' [08:30:10] [WARNING] GET parameter 'user' is not injectable [08:30:10] [CRITICAL] all tested parameters appear to be not injectable. Try to increase '--level'/'--risk' values to perform more tests. Also, you can try to rerun by providing either a valid value for option '--string' (or '--regexp') If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could retry with an option '--tamper' (e.g. '--tamper=space2comment') [\*] shutting down at 08:30:10

### OTG-INPVAL-013: Testing for Command Injection

Attempt to execute OS commands

Test 1 <https://195.37.154.37:10504/api.php?dir=%3Bcat%20/etc/passwd>

Result: none

### Server Errors

This section is to identify server or application errors that may show additional information about the server and/or web application that can be useful for fingerprinting and understanding the web application further.

### OTG-ERR-001: Testing for Error Code

Determine error codes for pages that are both valid and invalid using telnet and socat.

Existing page ui.php

GET df.php HTTP/1.1 HTTP/1.1 400 Bad Request Server: nginx/1.9.4 Date: Sat, 10 Oct 2015 07:50:53 GMT Content-Type: text/html Content-Length: 172 Connection: close

Invalid page

GET df.php HTTP/1.1 HTTP/1.1 400 Bad Request Server: nginx/1.9.4 Date: Sat, 10 Oct 2015 07:50:53 GMT Content-Type: text/html Content-Length: 172 Connection: close

Socat was not a recommended tool however it was used in conjunction with telnet.

**Result** – no useful error messages. 400 bad request for both tests

## **OTG-CRYPST-001: Testing for Weak SSL/TLS Ciphers, Insufficient Transport Layer Protection**

Command: `Openssl s_client -connect 195.37.154.37:10504`

### **Result**

New, TLSv1/SSLv3, Cipher is DHE-RSA-AES256-SHA

Server public key is 2048 bit

Secure Renegotiation IS supported

Compression: NONE

Expansion: NONE

SSL-Session:

Protocol : TLSv1

Cipher : DHE-RSA-AES256-SHA

Session-ID:

A0C5D1F7DEB3C303507BB4CB7399CC08F5FF22748344AA1D5A5E4FCAC3C04A45

Session-ID-ctx:

Master-Key:

7D8F71C8AC816F197FA231968594BB135C7A289F75844E8FDF177E650A17468D5B649EAE1  
1B5FF5F46A0B2C1151765B2

Key-Arg : None

Start Time: 1444464092

Timeout : 300 (sec)

Verify return code: 0 (ok)

## **OTG-CTYPST-003: Testing for Sensitive information sent via unencrypted channels**

Test whether sensitive data is protected when transmitted

### **Test 1 using Curl**

```
curl -k 'https://195.37.154.37:10504/' -H 'Authorization: Basic YXNkZjphc2Rm' -H 'Accept-  
Encoding: gzip, deflate, sdch' -H 'Accept-Language: en-US,en;q=0.8' -H 'Upgrade-Insecure-  
Requests: 1' -H 'User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_3)  
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/45.0.2454.101 Safari/537.36' -H 'Accept:
```

```
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8' -H 'Cache-Control:
max-age=0' -H 'Cookie:
session=rGe27EvJLqQ48q1t42vQ_A.8C2DaaCg_pueKscSCXENuGo4yP3Da0VK4uAobSRl4J8.1
444357191730.86400000.3EWG3eFCOYZx2UFsfil1PyUu0HVI8A8pUBV0q3s_ktY' -H
'Connection: keep-alive' --compressed -vvvv
```

## Test 2 Using testssl.sh

See file (testssl.docs)

## OTG-CLIENT-004: Testing for Client Side URL Redirect

The purpose is to determine if the client side URL redirection, commonly known as Open Redirection is vulnerable. It is input validation flaw that exists when an application accepts a user controlled URL that specifies a link that leads to an external URL that potentially could be malicious

Test 1: <https://195.37.154.37:10504/?#redirect=google.com>

Test 2 <https://195.37.154.37:10504/#redirect=google.com>

Test 3: <https://195.37.154.37:10504/ui/?user=#redirect=google.com>

Test 4: <https://195.37.154.37:10504/ui/?#google.com>

Test 5: [https://195.37.154.37:10504/?#javascript:alert\(document.cookie\)](https://195.37.154.37:10504/?#javascript:alert(document.cookie))

Result: Site not vulnerable.

## OTG-CLIENT-009: Testing for Click jacking

To test for this vulnerability a HTML page is created using the iframe HTML tag as follows

```
<html>
  <head>
    <title>Click jack page</title>
  </head>
  <body>
    <p>website is vulnerable to click jacking</p>
    <iframe src="https://195.37.154.37:10504" width="600" height="600">
  </iframe>
  <p>Your browser does not support iframes.</p>
</html>
```

```
</body>  
</html>
```

The page was successfully loaded into the browser via the iframe tag, thus the web application is potentially vulnerable to click jacking (page 201 of the OWASP Testing Guide). To verify this login credentials are required to test thoroughly.