jamk.fi

SECURITY TESTING SPECIFICATION

Case JYVSECTEC

Päivi Hirvonen

Master's Thesis

December 2015

Master's Degree Programme in Information Technology

JYVÄSKYLÄN
AMMATTIKORKEAKOULU

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

DESCRIPTION

Author(s)	Type of publication	Date
Hirvonen, Päivi	Master's Thesis	14.12.2015
	Pages 65	Language English
		Permission for web publication (X)

Title

Security Testing Specification, Case JYVSECTEC

Degree Programme

Master's Degree Programme in Information Technology

Tutor(s)

Karo Saharinen, Mika Rantonen

Assigned by

JYVSECTEC, Antti Niemelä

Abstract

The purpose for thesis was to examine for JYVSECTEC how security testing could be brought to a stabilized level in quality despite of the test user. The research was conducted by using the OWASP (Open Web Application Project) testing checklists for guidance, using which the testing group executed the testing. The comparison group did not use any guidance for testing, instead, it rather executed the testing and chose the test cases freely.

The tested target was a WordPress website, to which the testers had access through virtual machine network. Kali Linux distribution security testing tools were used for testing.

The research revealed that the use of checklists for testing is valuable and gives a more versatile and comprehensive result of security of the tested target; however, there was a great deal of dispersion in the results which was mainly due to the lack of test users' experience or knowledge.

Using the checklists enables to achieve stability in testing quality if the test users are trained and prepared for the same level in the use of the testing tools, tested target, test environment, test case analysis and know-how.

Keywords

Information security, testing methods, quality assessment, standards

Miscellaneous

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

KUVAILULEHTI

Tekijä(t)	Julkaisun laji	Päivämäärä
Hirvonen, Päivi	Opinnäytetyö	14.12.2015
	Sivumäärä 65	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty (X)

Työn nimi

Security Testing Specification, Case JYVSECTEC

Koulutusohjelma

Master's Degree Programme in Information Technology

Työn ohjaaja(t)

Karo Saharinen, Mika Rantonen

Toimeksiantaja(t)

JYVSECTEC, Antti Niemelä

Tiivistelmä

Opinnäytetyössä oli tarkoituksena tutkia JYVSECTECille, miten tietoturvan testaamisesta saataisiin tasalaatuista riippumatta testaavasta henkilöstä. Tutkimuksessa käytettiin testauksen perustana OWASP (Open Web Application Security Project) -testauslistoja, joiden mukaan testiryhmä suoritti testauksen. Vertailuryhmä ei käyttänyt mitään ohjeistusta testien tekemiseen, vaan suoritti testauksen ja valitsi sopivat testitapaukset vapaasti.

Testattavana kohteena oli WordPressin websivusto, johon testaajilla oli pääsy virtuaaliverkon kautta. Testaustyökaluina käytettiin Kali Linuxissa saatavilla olevia tietoturvan testaamiseen tarkoitettuja applikaatioita.

Tutkimuksessa ilmeni, että testauslistojen käyttäminen on hyödyllistä ja antaa monipuolisemman ja kattavamman tuloksen testattavan kohteen tietoturvan tilanteesta. Kuitenkin tuloksissa oli paljon hajontaa, joka johtui pääosin puutteista testaajien kokemuksessa ja taidoissa.

Testauslistojen käyttäminen edesauttaa testauksen tasalaatuisuuden saavuttamisessa, jos myös testaajat on valmennettu samalle tasolle työkalujen käytön, testattavan kohteen, testiympäristön tuntemisen, testitapausten analysoinnin ja osaamisen suhteen.

Avainsanat (asiasanat)

Tietoturva, testausmenetelmät, laadunarviointi, standardit

Muut tiedot

ACRONYMS

Term	Explanation
Armitage	Graphical cyber attack tool
AV	Anti-Virus
CA	Certificate Authority
CIA	Security triad of confidentiality, integrity and availability
CobIT	Control Objectives for Information and Related Technology
CSRF	Cross-Site Request Forgery
Dirb	Web content scanner
DNS	Domain Name System
EICAR	European Institute for Computer Antivirus Research
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IFRAME	Used to display a web page within another web page
IoT	Internet of Things
ISO	International Organization for Standardization
John the Ripper	Password cracker
KATAKRI	National Security Auditing Criteria
LDAP	Lightweight Directory Access Protocol
MAC	Media Access Control
MERIT	Management and Education of the Risk of Insider Threat, threat modeling system
MITM	Man in the middle attack
Nessus	Web vulnerability scanner
Nikto	Web server scanner
NIST	The National Institute of Standards and Technology
NMAP	Security and port scanner
OCTAVE Allegro	Operationally Critical Threat, Asset and Vulnerability Evaluation, threat modeling method
Open VAS	Open Vulnerability Assessment System
OS	Operating System
OWASP	Open Web Application Security Project
OWASP ZAP	OWASP Zed Attack Proxy penetration testing tool
PHP	Hypertext Preprocessor
phpMyAdmin	Managing MySQL administration
RFID	Radio Frequency Identification
SAMATE	Software Assurance Metrics And Tool Evaluation project

SQL	Structured Query Language
SQLmap	SQL injection and penetration testing tool
SSH	Secure Shell
STRIDE	A threat modeling method developed by Microsoft
SYN flood	Denial of Service attack, synchronize message
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
URL	Uniform Resource Locator
VAHTI	Government Information Security Management Board
Wireshark	Network protocol analyzer
VMWare	Virtualization software
VPN	Virtual Private Network
XML	Extensible Markup Language
XSS	Cross-Site Scripting

CONTENTS

Tables

Figures

1 Introduction

The thesis describes different areas of non-automated security testing, methods and processes, and how they are all interconnected and what kind of connections these items have to each other. Testing is not just focusing only on the tested target, whether it passes or fails the individual testing case.The overall situation is important in tested environment; it varies and lives constantly due to new features or system patches. Newer attacking methods, malware or exploits are released in the cyber world that are easily available for attackers.

The thesis work focuses to a particular development area: on how the testing could be uniformed and maintained on the same level in the testing quality, despite of the test users. Using of testing checklists is studied, how the testing results could be improved and unified, and the dependencies of the test user's competence or knowledge of the tested areas minimized.

Research is conducted by using two groups of people, first group is using checklists in testing, and the second group is testing without the checklists. Results and findings of these groups are compared and analyzed for further development, whether using the checklists gives more better results in unifying the overall testing quality.

Today many physical services of the modern society are replaced by an electronic or internet based service, which gives more surface for attackers. Security testing also must be updated, innovative, thorough, versatile and continuous to meet these challenges in today's internet usage in every instrument.

2 Security testing methods

2.1 Black box

Black box testing, or functional testing is executed without any knowledge of the internal software or program structure. It is based only on the input and the output of the system. The testing criterion is that the program or software functionality meets the given specification. It can be compared to driving a car, where the driver does not have to know how the car works internally when he is driving it.

Black box testing is suitable for testing anything: software or a system that is working as it is designed with the given input. Disadvantage of this functional testing method is that the possible errors or malfunctions of the internal structure remain unknown. (Lewis 2009, 39-40.)

Other disadvantages are also that no-knowledge based testing is more expensive, it takes more time to perform and gives experience of an external attacker, whereas in real life insiders with more knowledge carry out the attacks. (Gregg 2013, 11)

2.2 White box

White box or structural testing, is the opposite testing method to black box testing. In white box testing, the focus is on the internal logic of the system, software or code. In this testing method, the output or the actual behavior is not checked against the specification.

According to Lewis (2009, 40), the benefit of this structural testing method is, that the code or system is inspected very thoroughly. Should the code include any errors, they would be found by this testing method. As in the example of a car not working, a car mechanic would find the reason for it easily, by checking the car engine. However, the white box testing forgets to pay attention to the outcome, how it meets the specifications, though the internal structure would be accurate.

2.3 Grey box

Grey box testing uses both black box and the white box testing methods. The tester is familiar with the functionality of the system (black box) and also knows how the internal system or the code should work (white box). In grey box testing the best parts of these two testing methods can be joined, which can save testing time or reduce the amount of the needed test cases. (Lewis 2009, 39-40.)

2.4 Security testing techniques

Different security testing techniques are described in the following chapters. Which technique is the most suitable for testing purposes, depends on the testing target and what kind of testing is required, for example looking for vulnerabilities of system or trying to penetrate and break in to the system.

2.4.1 Source code analysis (white box)

Source code security analysis is executed for searching weaknesses and vulnerabilities from the actual source code of the software. Examining the source code is beneficial, since the source code contains more info than the reverse-engineered one from byte code to binary, also any security vulnerabilities are easy to fix in the original form of the source code. (Ransome, Anmol & Schoenfield, 2014)

2.4.2 Property-based (white box)

Property-based testing ensures that the software's implemented functionality meets the specifications. Testing is executed by comparing the security-relevant properties (e.g. absence of insecure state changes) to software's specifications and validating the requirements. (Ransome, Anmol & Schoenfield, 2014)

2.4.3 Source code fault injection (white box, gray box)

In fault injection testing, errors are inserted to the software in order to simulate the unintentional attacks against the software and the environment. Tester can monitor how the software state changes from secure to non secure, while the injected fault goes through the whole of the source code. (Ransome, Anmol & Schoenfield, 2014)

2.4.4 Dynamic code analysis (gray box)

Dynamic code analysis occurs in a running application of the software. All the vulnerabilities detected from the application interfaces, are located and fixed from the source code at the same time during the analysis. (Ransome, Anmol & Schoenfield, 2014)

2.4.5 Binary fault injection (gray box, black box)

In binary fault injection technique, the fault injections are executed while system call traces in the application are controlled. Examining the traces, it can give information about the names of the system calls, the parameters to particular call and also how the resources are used. (Ransome, Anmol & Schoenfield, 2014)

2.4.6 Fuzzy testing (black box)

In the fuzzing technique, testing is executed by feeding random inputs, faulty and non-faulty, to the software. Then the results are analyzed in terms how the software has survived from the wild and not proper inputs, or if it has crashed totally.(Ransome, Anmol & Schoenfield, 2014)

2.4.7 Binary code analysis (black box)

Binary code analysis is done by machine code scanners, which analyze the program's behavior. Then the result of the model is analyzed by a vulnerability

scanner, which will detect for coding errors or back doors from the model.(Ransome, Anmol & Schoenfield, 2014)

2.4.8 Byte code analysis (black box)

Byte code scanners are useful for monitoring, when the source code is not available, or if it needs to be clarified what impact another software component might have in the security or vulnerability of the application.(Ransome, Anmol & Schoenfield, 2014)

2.4.9 Black box debugging (black box)

Black box debugging is examined using the source code of the application. Program execution is controlled, and its values can be modified for testing. Debugging can be prevented from commercial software by code obfuscation.(Ransome, Anmol & Schoenfield, 2014)

2.4.10 Vulnerability scanning (black box)

Vulnerability scanning is executed by tools, which look for recognizable, specific and known vulnerabilities from the software. Also, attack patterns used by hackers are being simulated for inquiring the software or application for exploitable vulnerabilities.(Ransome, Anmol & Schoenfield, 2014)

2.4.11 Penetration testing (black box)

Penetration testing focuses on breaking the software, application or system security. Penetration testing can be done automatically with software applications or they can be executed manually. The aim is to find a security weakness, and get access to the data, or system by this hole in the security. The testing environment has same the conditions as ordinary end users.(Ransome, Anmol & Schoenfield, 2014)

2.4.12 Threat-model based testing

Threat modeling is a testing method, which is used to find the possible threats against the system or application, and create appropriate mitigations for those threats. Threat modeling focuses on the attacker's point of view, which of the assets might have some value to the attacker, and might need mitigation and more protection. Threats live forever, and that is why threat modeling is also a continuous task. (Linden, 2007)

Threat modeling includes a description of the possible attacker, the asset or the target that might be attacked, and a communication way for the information transferring between the data and the system. Any vulnerabilities, known but also yet undiscovered, which can be exploited, are detailed with mitigated actions. (Linden, 2007)

STRIDE is a threat modeling method developed by Microsoft, which classifies the vulnerabilities in six different categories. **S=** Spoofing of user identity, **T=** Tampering with data, **R=**Repudiation, **I=**Information disclosure, **D=**Denial of service, **E=**Elevation of privilege. (Linden, 2007)

MERIT (Management and Education of the Risk of Insider Threat) is a threat modeling system, which concentrates on the attack coming from inside the organization. MERIT is developed by the Carnegie Mellon University's Software Engineering Institute. (Linden, 2007)

OCTAVE Allegro (Operationally Critical Threat, Asset and Vulnerability Evaluation) is a method that focuses on the threats in large organizations and on securing the information assets from the organizational point of view. (Cert.org, 2015)

2.4.13 Risk-based security testing (white box)

Risk-based security testing method focuses on already previously identified risks and verifying the mitigation efforts if they meet the requirements thoroughly. Risk analysis reveals the potential risks, e.g. system architecture, attack patterns, security breaches, violations or vulnerabilities. Risk-based testing simulates the possible attack or security breach and at the same time

measures the mitigation activities, if they are as efficient and protective enough as designed. (Secappdev.org, 2015)

Before the testing can be started, the risk in security testing must be defined and identified. The risk might not ever occur, however, the worst case must be still imagined, and how it will be prevented. Risk analysis consists of clarification of all possible risks and their impacts on the business. Risk based security testing uses the best practices and prioritizes the most important assets to be protected. Test cases and tested features or functions can be selected by the prioritization and risk definitions in the risk management system. It is useful to test the feature, application or functionality that has the highest impact in the organization's core area and also the most probability of failure. Risk based testing is useful in a situation of restricted resources, e.g limited amount of testing time, effort, testing staff or budget. By testing the most critical and vital features, it also develops the quality of security testing. (Softwaretestingclass.com, 2015)

3 Testing process

3.1 Goal

The goal can be determined by what is wanted from the security testing and what purpose the testing is executed for. According to Gregg, (2013, 7-8) security triad of confidentiality, integrity and availability (CIA), gives three aspects of security testing.

Confidentiality means that secrecy and privacy of information on, e.g. passwords, encryption and data during storage or transmission is secured. Integrity can be seen as information or data is being accurate, and that it has not been changed in any way during storage or transmission. Availability means that when the legitimate user wants to use the service or information, e.g. website or cloud service, it should be available. (Gregg 2013, 7-8.)

3.2 Approach

What kind of testing is executed to analyze the system infrastructure? Is there a problem in the security management, is the protection designed to cover all the organization functionality, and is it up to date? These questions may arise, when the best possible or useful testing type is considered. At first, information gathering from the target company is useful to find any information that could be used in an attack against it.

Gregg (2013,19-20) describes various possible ways for taking the most suitable approach for testing. For example, testing could be targeted at securing organization's employees from the social engineering attacks and ensure that the company's sensitive information is secured.

Another approachable way is penetration testing for simulating the attack coming from outside via internet or inside the organization, if that is the weakest area in the company security management or poorly designed, out of date protection worries.

Selecting the most usable testing approach depends mostly on the target (e.g. organization) under testing, its features, size and the used network solutions. Gregg also points out that testing the network devices, like firewalls, router, switches and also the wireless network devices like Radio Frequency Identification (RFID), is another point of view of security testing, when the correct perspective is under examination. (Gregg 2013, 19-20.)

3.3 Resources and restrictions

It is important to think about the available resources and possible restrictions before the actual testing is started. Resources can considered as employees, testing equipment, testing environment, testing facilities, usable time for testing, or budgeted money for testing.

Also, these same elements can be taken as restrictions as well. For instance, if the employees do not have the needed skills for executing the testing session and they must have special training for that, it will be a restrictive feature. Furthermore, testing schedule, time scale and money (testing expenses or available budget) have influences on both sides. (Watson & Jones, 2013).

3.4 Timetable

Timetable will cover the whole testing session from the beginning to the final end and includes also the time used for the individual testing case. Timetable is dependent on the scale and amount of the testing tasks (e.g. how time consuming test cases, how many repeats, how many cases and features) and accessible employees and their workload. (Watson & Jones, 2013)

3.5 Testing targets

It should be decided what exactly is tested, whether to focus on applications and features that they meet the requirements or hunt for vulnerabilities, bugs or exploits? According to Peter Farrell-Vinay (2008) ”Testing cannot prove the absence of bugs, only their presence.”

3.6 Risks

According to (Techrepublic, 2015), Parker (2007) in his article ”Risks of risk based security” defines a security risk as follows:

> *Security risk is not measurable, because the frequencies and impacts of future incidents are mutually dependent on variables with unknown mutual dependency under control of unknown and often irrational enemies with unknown skills, knowledge, resources, authority, motives, and objectives—operating from unknown locations at unknown future times*.

Risks in security testing are versatile, anything could go wrong, from tiny human error to a natural disaster. Risk management systems are designed to control the risks and being prepared for the possible setback.

Risks can affect many parts during testing session, which needs to be evaluated in the risk management plan. Security controls are important when thinking of mechanisms to prevent the possible risk or damage to the protective assets. ISO 27001, CobIT and NIST 800-53 have described controls for optimizing the security.

- Assets and asset values vital for the testing project: e.g. people, knowledge, software, hardware, firmware, network, physical equipment, image, or reputation
- Project risk: e.g. not enough competent employees allocated for the project, lack of employee training or schedule delays
- Threats: Anything harmful that could happen, e.g. natural phenomenon or human actions, accidental or deliberate, like power loss or breaking in the premises
- Vulnerabilities: Any weakness that a threat could exploit and endanger an asset, e.g. sensitive traffic not encrypted properly or lack of educated testing experts

4 Security testing cycle

The next figure demonstrates how security testing and auditing can be divided in four different categories, which perform a continuous cycle.

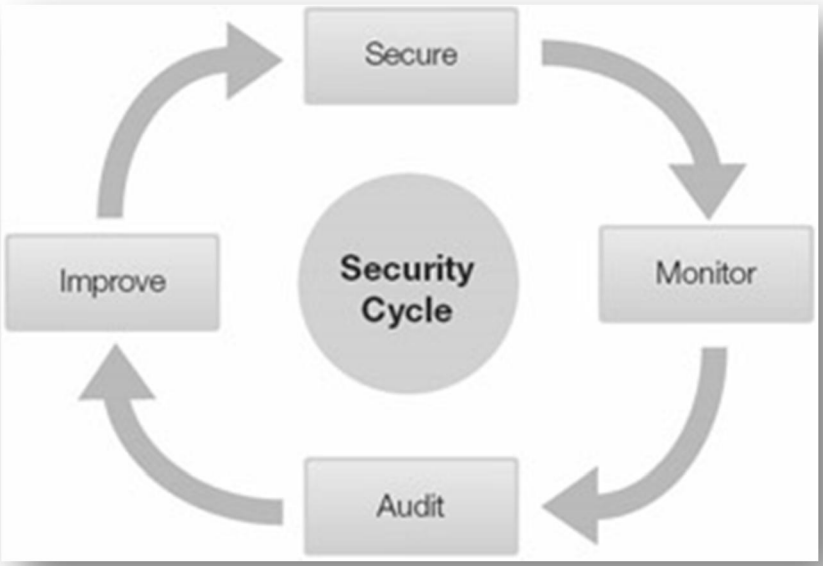

Figure 1. Security testing cycle (Kim & Solomon, 2014)

4.1 Monitor

Security testing cycle starts from the monitoring. In the monitoring phase is important, that all the used controls are measured and reviewed, to detect and capture all actions and changes in the system. Monitoring is the first phase of detecting the abnormal behavior that reveals the malicious actions.(Kim & Solomon, 2014)

4.2 Audit

In the auditing part, used security policies are reviewed by analyzing the logs and overall environment. Security audit area can be very large, containing entire department or business function, or just a one specific system can be audited. Auditing can focus on security policies, networks, or devices.

Purpose of audit is to make sure that the used security controls are appropriate and are suitable against the particular risk they were originally

designed. Audit checking can determine if the security control is installed correctly and located in the correct place for meeting the purpose. (Kim & Solomon, 2014)

4.3 Improve

Security is improved by gathering new ideas and suggestions of those areas that are detected to be insecure. All the cases found during the audit that needs improvements should be corrected. It is also possible that the software will become immune to the same test cases in time just like germs will get resistant to antibiotics when they are used long enough. Making testing versatile and innovative, making surprising combinations in the testing methods will reveal new errors or yet undiscovered security holes.(Kim & Solomon, 2014)

4.4 Secure

Used security controls are designed to protect an asset from a certain threat. Testing the controls means that they are working properly, and they are really necessary. A security control which is without a certain threat, just overall or in case, does not increase the security level. Security controls must be tested for vulnerabilities in the system, during certain changes, e.g. after technology upgrades, application changes, or after revealing new threats or vulnerabilities in public.(Kim & Solomon, 2014)

5 Types of security testing

5.1 Network scanning

The purpose of the network scanning is to find all active hosts and open e.g. TCP/UDP (Transmission Control Protocol/User Datagram Protocol) ports of the network. Also, information of the used applications is gathered from the identified ports in order to find any vulnerable services or unauthorized applications. Network scanning can be useful when IDS (Intrusion Detection System) are installed, firewall restrictions are defined or penetration testing is being planned. Other benefits for network scanning are using it for verifying the functionality of the security controls and gathering information for penetration testing analysis. (Lewis 2009, 353-354.)

According to Orebaugh & Pinkard (2008), network scanning has four different categories:

- Network mapping: Sending messages to a host, to see if it is active and responses or not active
- Port scanning: Sending messages to a port, to see if it is active or not
- Service and Version detection: Sending tailored messages to an open port, to analyze the type and version of the ongoing services from the port response
- OS detection: Sending specially designed messages to an open host, which responses will reveal the used operating system on the host

Nmap (Network Mapper) is a common tool for network scanning. Nmap website http://insecure.org (Orebaugh & Pinkard, 2008).

5.2 Vulnerability scanning

Vulnerability scanning or management, means finding and identifying the vulnerable system and resolving the possible unsecure feature from it before it can be exploited. Vulnerability can be anything; e.g. error in the software program, misconfiguration, weak password policy that could enable the attacker getting unauthorized access to the system.(Rogers, 2008)

Vulnerability assessment is proactive security method, which gives valuable information before the intrusion will come true. With this information the problem can be solved and corrected and the attack totally prevented.(Rogers, 2008). Critical vulnerabilities with explanations are listed in the Table 1.

Table 1. Critical vulnerabilities (Rogers, 2008)

Critical vulnerability	**Explanation**
Buffer overflow	Input data is greater than the size of the destination buffer
Directory traversal	Files are accessed outside a restricted directory structure
Format string attacks	Format string vulnerabilities are often used to overwrite a function pointer with the memory address of user-supplied data (usually shell code). After the **printf()*in C language function is called, any calls to the overwritten function will result in the attacker's code

	being executed.
Default passwords	Default passwords in devices, applications or services
Misconfigurations	E.g. Poorly configured system settings enabling anonymous access
Known backdoors	Known backdoor programs for crackers
Zero-day attack	Yet unknown vulnerability or not in publicly fixed vulnerability

5.3 Password cracking

Password cracking can be made in various ways, from guessing the too easy passwords or using brute-force method to crack the password eventually. In this context attacking against the password in order to crack it means that is performed only in the testing purposes for evaluating the password security.(Graves, 2010)

Passive online attacks

Passive online or sniffing is one of the password cracking methods.It can be done in wired or wireless network by capturing the password during authentication process which user can not detect. (Graves, 2010)

In the Man in the middle (MITM) attack, a sniffer is inserted between the server and the client to intercept both connections and capture a password. In a replay attack, the authentication packets containing password are captured on the way to the server and used later when authenticating as the client.(Graves, 2010)

Active online attacks

In the active online password cracking, the easiest method is to guess the used password. If the password is weak, or default password is used cracking is simple.Automated password guessing is done by tools or scripts, which reduce the cracking time.(Graves, 2010). In the following Table 2 some examples of online attacks are described.

Table 2. Online attacks (Graves, 2010)

Type of attack	**Characteristics**	**Example password**
Dictionary attack	Attempts to use passwords from a list of dictionary words	Administrator
Hybrid attack	Substitutes numbers of symbols for password characters	Adm1n1strator
Brute-force attack	Tries all possible combinations of letters, numbers, and special characters	Ms!tr245@F5a

In the online attack, the captured password file is copied from the actual computer to the hacker's system, where the password can be cracked. In the dictionary attack, the password is an actual word from a dictionary. It cannot contain any numbers or special symbols. A dictionary file of possible words is hashed with the same algorithm than the authentication process. Then the hashed dictionary file with words is compared to the passwords when a legitimate user logs on to the system. (Graves, 2010)

The hybrid attack is used if the dictionary attack method cannot find the actual passwords. If the password has anomalies, e.g. numbers or symbols substituting letters, the hybrid attack is focused on finding those types

passwords. A brute-force attack will check every possible combination of letters, lowercase and uppercase, numbers and symbols in the passwords. It is very time consuming but effective. Eventually all passwords can be cracked by brute-force method. Using rainbow tables or already hashed dictionary words it is possible to speed up the discovery and cracking time of passwords. (Graves, 2010)

Nontechnical attacks

Nontechnical attacks are done without any technical devices. This is enabled by social engineering, shoulder surfing, keyboard sniffing or dumpster diving. (Graves, 2010)

5.4 Log management

Log management can be hacked in various point of system. Log files can be e.g. audit records or event-logs, meaning large amount of computer gathered log messages.

- The source: where the log files are generated
- During transmission: transit between the source and the log host
- Logging data storage: Database or other system for storage and archiving the log data files (Chuvakin & Schmidt & Phillips, 2013)

Log information important in security testing is presented below as follows:

- Authentication server or system logs which contains information of successful and failed authentication attempts
- System logs may containing traces of system and service start up and shutdown information, installation of unauthorized software, file accesses, security policy changes and account creation or deletion
- Intrusion detection and prevention system logs revealing malicious activity and inappropriate use

- Firewall and router logs indicating outgoing connections from compromised internal devices (e.g., root kits, bots, Trojan horses, spyware).
- Application logs containing account changes, use of privileges, application or database usage information, and unauthorized connection attempts with firewall logs
- Anti virus logs have valuable data of update failures and other indications of outdated signatures and software
- Security logs, Intrusion Detection System (IDS) and Intrusion Prevention System (IPS) products, gather information of known vulnerable services or applications (National Institute of Standards and Technology 800-115, 2008.)

Log management attacks are categorized by threat analysis into three parts, concerning confidentiality, integrity and availability. In Table 3, the attacks against confidentiality with explanations are listed.

Table 3. Attacks on Confidentiality (Chuvakin & Schmidt & Phillips, 2013)

Log management attack	**Explanation**
Confidentiality at the Source	Unauthorized access to directories and log files
Confidentiality in Transit	Intercepting network traffic between source and log host
Confidentiality at the Log host	Unauthorized access to the log host
Confidentiality at the Log Store	Unauthorized access to the log storage
Confidentiality at Analysis	Unauthorized access to the analysis location

In the following Table 4, the attacks against integrity with explanations are listed.

Table 4. Attacks on Integrity (Chuvakin & Schmidt & Phillips, 2013)

Log management attack	**Explanation**
Integrity at the Source	Unauthorized modifying of log data
Integrity in Transit	Intercepting network traffic between source and log host and modifying log data files
Integrity at the Log host	Unauthorized modifying of log files
Integrity at the Database	Unauthorized modifying of arriving or stored log files in database
Integrity at Analysis	Unauthorized modifying of input data, analyzing tools or output data e.g. report

In Table 5, the attacks against availability with explanation are listed.

Table 5. Attacks on Availability (Chuvakin & Schmidt & Phillips, 2013)

Log management attack	**Explanation**
Availability at the Source	Deleting of log data at the source, zeroing and overwriting log data
Availability in Transit	Flooding the legitimate traffic with bogus traffic, e.g. Ping flood
Availability at the Log host	Network and SYN flooding, exploits to crash the host, deletion of stored

	data files
Availability at Analysis	Flood analysis system with false alarms

5.5. Web security

Almost everything can be done on the internet by using web services. Users can buy food or fashion from online stores around the world. Booking hotel rooms, or buying flight tickets or holiday trips is easy.

Making doctor's appointment and updating medical electronic prescriptions from the web is handy. Most people pay their bills using web bank services or look for information on web search such as Google.

Many people use social networking to keep in touch with others by Facebook or web blogs. Also, gambling or betting on the net is possible, but also attending to auctions for example eBay, where almost anything from electronics to antiques can be bought. (Stuttard & Pinto, 2012)

Internet of Things (IoT) is growing from internet connected machines in the industry to cars and television or even the refrigerator at home. How fast the IoT will develop in the future and is it all really essential or safe for us as consumers? It is probable anyway that when the IoT expands to different devices in people’s lives, the more vulnerable using the internet will be with more attacking surface and security faults.

5.5.1 Web server hacking

Sample files

Sample files, scripts or codes offered by vendors must not be installed without proper reviewing or auditing before packing. Misconfigured sample files in servers can be exploited without patching. (McClure & Scambray & Kurtz, 2012)

Source code disclosure

In the source code disclosure, malicious user is able to view the source code of confidential files, e.g. passwords, on vulnerable web server. That is why application source code should not contain any protect able data, e.g.database passwords or encryption keys. (McClure & Scambray & Kurtz, 2012)

Canonicalization attacks

Canonicalization attack refers to addressing the server resources by another representation. As an example C:\file.txt can be accessed by another address like ..\file.txt, which is pointing to the same path in the root. (McClure & Scambray & Kurtz, 2012)

Denial of service

Denial of service may be seen as unavailability of resource, inaccessibility of a website or slow activity of the website. Denial of service attacks can be performed in different ways. The server can be overwhelmed with requests, until all the resources are used up. Service request floods, SYN flood attacks or ICMP (Internet Control Message Protocol) flood attacks make use of this by making the server unable to perform its normal duties. (Oriayano, 2014)

5.5.2 Web application hacking

Web application attacks are focused on the user application code and its possible vulnerabilities.Common targets for web application attacks are authentication, session management, database interaction or generic input validation. (McClure & Scambray & Kurtz, 2012)

In the following Table 6 the most critical web application security risks by Open Web Application Security Project (OWASP) organization are listed. The list is gathered by security experts around the world using their expertise for the most critical flaws in the web.

Table 6. OWASP Top 10 Web Application Security Risks 2013

(Owasp.org, 2015)

OWASP TOP 10-2013	**Description**
A1- Injection	SQL, OS, LDAP injection with untrusted hostile data sent to an interpreter as part of a query
A2- Broken Authentication and Session Management	Passwords, keys or session tokens compromised by incorrect implementation in authentication and session management functions
A3- Cross-Site Scripting (XSS)	Application receives untrusted data and sends it to a web browser without proper validation
A4-Insecure Direct Object References	Files or directories can be accessed by manipulating references
A5-Security Misconfiguration	Security settings and software should be updated and properly configured
A6-Sensitive Data Exposure	Authentication credentials or credit card numbers poorly protected
A7-Missing Function Level	Function level access rights forged, if

Access Control	access control is not properly checked
A8-Cross-Site Request Forgery (CSRF)	Attacker has forced the victim's browser to a malicious site
A9-Using Known Vulnerable Components	Libraries or frameworks with full privileged are dangerous, if they are exploited
A10-Unvalidated Redirects and Forwards	Validation when redirecting or forwarding the web pages, to avoid entering malware sites

5.5.3 Database hacking

The most important, valuable and sensitive information is stored in the database of the company. It may contain passwords, credit card numbers or other customer related data. SQL (Structured Query Language) injection is commonly used attack method against the database. There are different kind of SQL injection methods, explained in the following Table 7. (Gregg 2013,329-331)

Table 7. SQL injection techniques (Gregg 2013, 331)

SQL injection techniques	**Explanation**
Simple SQL injection	Invalidated input is exploited
Union SQL injection	Union select-command used to steal data
Error-based SQL injection	Try to get the database response with error messages
Blind SQL injection	Attacked database error messages are hidden

Other database breaches are related to the user privileges. Attack can be performed by privilege abuse; a legitimate user is accessing unauthorized data. Access rights may be oversized for the job level or position in the enterprise, which can be susceptible for misuse. Access rights can be convert from low-level to high-level privileges, in order to get for example administration level access rights by exploiting vulnerabilities. Weak authentication and weak password policy predispose the database to forged identity of users. Legitimate user credentials are gained by social engineering or brute force attacks. (The British Computer Society, 2015)

5.6 Malware

Malware in the meaning of malicious software has several different types. Different kind of malwares are presented as an example in the following chapters. Malwares can be divided by how they are spread, executed or what they do when activated. Generally everyone is avoiding contamination of malwares but they can be used in security testing purposes, e.g cyber security exercise where attacking team is getting access to defensive team email system by using suitable malware.(Messier, 2014.)

Viruses

Virus can be embedded to another software, be inside an attachment file in an email, or it can be masqueraded as a game, picture or other application that the user wants to run or open.

Virus in email attachments can copy itself to all the recipients in the user's contact list and spread all over. Virus can modify or delete data, or in the worst case, steal information from the victim's computer system, or used as an instrument for cyber espionage. (Messier, 2014.)

Worm

Worm does not need help from the user to get spread, it can find its own way to other systems and get them infected. Worms are very rapid to spread all over and pollute an enormous amount of systems in a short time. It can exploit a vulnerability in a system to harm the host computer. (Messier, 2014.)

Worm is able to update itself and download and install extra additional malware. It can protect itself from detection or removal. On the other hand, it may ensure that the used vulnerability gets patched in order to remain the one and only ma lware making use of that particular vulnerability. (Messier, 2014.)

Trojan horse

A malware that is quite different from what it seems is called a Trojan horse. A Trojan is usually a virus needing the user to open, run or click it for activation. A Trojan can be attached to another program, or it can have a transportation software, the purpose of which is to carry the Trojan from system to another. (Messier, 2014.)

A Trojan can provide unauthorized access to the system without the user knowing it. It can put key logger capturing software to the computer and gather information of passwords, user names and credit card numbers. (Messier, 2014.)

Spyware/aAdware

Spyware is used to advertising, collecting user's personal information or changing the computer's configuration. Spyware does not ask the user's approval and it can be difficult to remove from the computer. (Spyware-whatis. n.d.)

Adware is a program, which displays advertisements on the visited websites. It can also forward a user's search request to advertising websites or collect

marketing information about users, which websites they usually visit and focus customized advertisements on them. Popup advertisements are annoying and they are easily detected, however, adware can be difficult to find. Using of free ware or share are can install a legitimate adware on computer. Another way of getting an unauthorized adware is a visit to an infected website. Through the vulnerable website a hacker is able to attack a victim computer and install a browser hijacker, a certain Trojan, which is used to steal sensitive information from the target computer (Internet Security Center. n.d.)

5.7 Penetration testing

Penetration testing focuses on finding the possible exploitable vulnerabilities on the organization's network, systems or devices. It helps to see what kind of damage a possible attack might do. Penetration testing cannot be compared to hacking, because hackers do not have to play by the rules, however, the penetration tester must follow the regulations. Also, hackers can spend plenty of time for hacking activities, however, penetration testing session is executed in a certain time frame. (Vacca, 2013)

Penetration testing starts by gathering information about the organization from various sources, the internet, social media or newspapers. In the vulnerability analysis any weak points are search from the system, network, policies, procedures or practices in the organization. (Vacca, 2013)

External penetration testing finds all the vulnerabilities that can be attacked from the internet, outside the network. Internal penetration testing focuses on finding all the exploitable vulnerabilities from inside the organization. Penetration tests includes also testing routers or firewalls, stolen devices e.g. laptops, mobile phones, or tablets. (Vacca, 2013)

6 Quality in security testing

6.1 Definition of quality

The International Organization for Standardization (ISO) defines quality ”as the degree to which a set of inherent characteristics fulfills requirements.” When talking about the quality, it can be bad or good quality. Quality has requirements that it should meet, in order to reach the level of good quality. Requirements can be stated from outside due to a legislation, or from inside, from a product or testing specifications. Characteristics mean how the quality is performed in the robustness and fitness of the product or testing. Degree in quality means it is a continuous, never ending effort and task, which must be taken care of on a daily basis. (Chemuturi, 2011)

Software reliability depends on how stable the configuration of the software and the hardware are. There are many factors to affect this stability and also give indications to the test planning also. For example, new versions of an operating system are available and the older ones should be updated. Updates for web browsers are released frequently. New viruses, spyware or other malware spread on the Internet all the time. Computers suffers from heavy loading of new tools, applications or down loadable data. Software products may use the shared libraries supplied by the operating system, eventually these libraries will be updated or modified. Also installing or uninstalling the utilities on the computer system may cause the changes or removal of the shared libraries. (Chemuturi, 2011)

6.2 Ensuring quality

Testing is following a certain specification or plan with designed and detailed test cases. To ensure quality in using the specification of testing is to process the documentation, take advantage of standards, guidelines, formats and templates, and make use of checklists to verify everything is included as planned. (Chemuturi, 2011)

6.3 Quality control

Quality control means the processes and methods used to finding out whether the requirements for quality are met. Controls can be carried out by doing reviews on testing specifications, processes or plans including executing the actual test cases to dig out the faulty behavior or reveal vulnerabilities from the complete tested version. Inspections, walk troughs and technical reviews are functions of quality control. Inspections can consist of detailed checklists for examining that the stated criteria are fulfilled and aim to prevent any faults before the actual testing begins. One important goal of quality control is to detect and correct faults, as quality assurance is more designed to prevent the faults in the first place. (Lewis, 2009)

6.4 Problems in quality assurance

According to Chess (2006) in his article of National Institute of Standards and Technology (NIST) Software Assurance Metrics And Tool Evaluation (SAMATE) project the use of penetration testing for assuring quality still does not give a maximum proof for security. Penetration testing focuses on finding any flaws before the hackers and getting them fixed. However, penetration testing cannot reveal all the problems in the application. Also, the software developers get very little feedback on the discovered flaws during penetration testing for fixing and preventing them in the first place in the source code. Thus, it is probable that the same flaws or vulnerabilities are found again in the future.

Another challenge in providing security is to think that software security is just another aspect of software quality, meaning a set of features compared to a specification or requirements. Software security in overall requires much more than just implemented security features. The ordinary quality assurance process that is adequate in achieving good quality in a traditional product or manufacturing industry is not enough in security issues. In order to get good results in quality, security matters must be focused and improved specially. Chess (2006) states, that ”good security is not a byproduct of a good quality.”

No attacks to the organization may result in an untrue feeling of security. A wrong way of thinking is that if they have not been a target for security attack yet, they are probably safe. Actually, as more time has gone by the security level degrades due to the new types of attacks or vulnerabilities continuously discovered by the hackers worldwide. (Chess, 2006)

Furthermore, if an application, software, device or program is not very widely used by a large volume of end users, it is assumed that the hackers are not interested in this kind of a minor application. With a new application of a large scale of users, people think that it must be secure after it is released in public. Reality shows that security does not depend on how long the software has been on the market or how many end users the software has. The following Figure 2 illustrates the situation where only the common behavior of users is noticed in testing and focusing only those features gives many other opportunities for hackers. (Chess, 2006)

As the Figure 2 shows below the seldom used cases in the corners are easily exploitable for attackers.It is important to execute negative test cases where is ensured that the software can handle an abnormal situation also. For ensuring the overall quality it needs to be clarified that the tested target e.g. software can not crash, reveal sensitive information or endanger the security by an uncontrolled behavior in a negative testing.

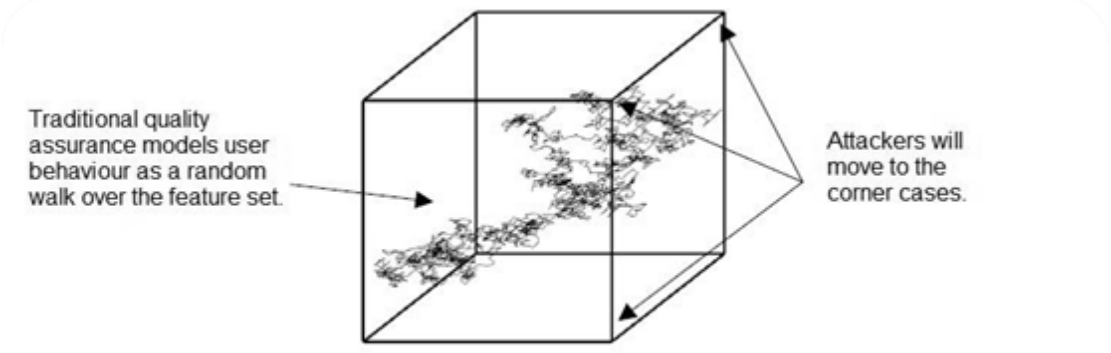

Figure 2. A quality-oriented approach to security leaves many opportunities for attackers (Chess, 2006)

7 Standards

7.1 Requirements in different security testing levels

Security testing and auditing may have to pay attention to the existing standards that are being applied. Before the testing is started it is vital to review is there a mandatory or recommended standard which should be followed. Standards or official criteria may set various demands for security testing by the different levels of security. These special requirements or level of security demands must be followed in security testing in order to meet the challenges.

7.2 KATAKRI

National Security Auditing Criteria (KATAKRI) determine the criteria for national official authorities and enterprises for conducting auditing. KATAKRI has four areas, administrative security (security management), personnel security, physical security, and information assurance.

All categories are divided into different classification levels, as restricted IV, confidential III and secret II. The purpose of using KATAKRI is to verify that after accomplishing auditing there would be no unidentified risks left. The information assurance area focuses on ”providing the minimum requirements for information whose confidentiality, integrity and usability shall be protected.” When executing security testing for web applications or filtering traffic there are special requirements for all security levels that have to be followed. (Ministry of Defence, Defmin.fi, 2015)

7.3 VAHTI

Government Information Security Management Board (VAHTI) consists of information security instructions to public administration, business, information

security co-operation, education and civil activities. Information security evaluation instruction 2/2014 (Tietoturvallisuuden arviointiohje 2/2014) gives detailed instructions for making information security evaluations for administration and the service provider stakeholders.

Authorities must secure that the organization, services and information are protected properly. In order to keep that secured situation up to date, they must evaluate their functions regularly and measure that the information security controls are adequate. Governmental information security is also regulated by law, giving requirements of maintaining information integrity, availability and usability. These special requirements, information classification levels and information security levels have to be observed, when the security testing is planned. (Valtiovarainministeriö, 2015)

7.4 ISO 27001/ISO 27002

ISO 27001/27002 standards provides instructions of different areas concerning information security management system (ISMS) and how these requirements are fulfilled. Also, an accredited certificate ISO/IEC 27001 is an achievement to an organization, indicating that the information security policies and strategies have been processed and executed in the corporation.

ISO/IEC 27002:2013 a code of practice provides guidance and best practices of information security management systems, what is important in various areas of assuring security. Technology is developing fast, it is useful to follow the latest and updated version of the standard. Areas beginning from the security policies through information security aspects of business continuity are handled with detailed knowledge in order to secure the information in different states or modes and during transmission or storage. (Calder & Watkins, 2012)

8. Human factor

Human factors can affect the security testing, both in good or bad. People are much more creative and innovative in testing than an automated test machine. People have their own weaknesses as well lack of knowledge or motivation. Ethical hackers intend to do penetration testing by the same methods than an evil hacker would do, however, without the malicious activity or goal. Security tester might end up a hacker's victim himself, either by mistake or intentionally. (Tipton & Krause, 2007)

8.1 Benefits

Human beings can be more flexible and innovative in testing than a machine. Computers do their tasks by the rules and policies as they are told. People can be more creative in testing, combine new information with their experiences from the past or trying to do something quite unexpected or forbidden. (Tipton & Krause, 2007)

For instance, when testing inputs for a web form, (write your user name to box), a user might wish to write longer names than allowed, use special marks or letters, writing numbers instead or letters, or leave the box empty. Highly trained, professional employees will find much more different solutions and situations in security testing that a machine cannot imagine.

A good attitude for testing is to think that there are undiscovered errors or vulnerabilities just waiting to be found. Human beings are goal-oriented creatures; if the goal is to find the errors, bugs or vulnerabilities we are focusing on that goal subconsciously. Also, when finding an error, it is often considered as failure or unsuccesful event in the tested target. This definition should be quite opposite, any found errors during the testing are succesful

events, since developing a perfect error free software or application is not realistic. (Mayers, Sandler & Badgett, 2012)

8.2 Drawbacks

It is said that the chain is only as strong as its weakest link. Also, in security testing in the worst cases, a human could compromise the system. The testing persons themselves could be a victim of a social engineering, phishing, or identity theft. (Tipton & Krause, 2007)

User's equipment and devices could be stolen from the office, car or at home and critical testing information could be lost. Frustrated, dissatisfied or employees with rage under getting sacked perform a high risk because they are inside the system, legitimate users with access rights to the infrastructure, network and facilities. (Gregg, 2013)

Recruiting the right kind of people for the job, employees' continuous education, proper training for the tasks, and increasing security awareness for everyone will improve the overall result. (Gregg, 2013)

Testing is not just about ensuring that the tested target works as it is designed to do. Or if no errors have been found, it does not prove that there are not any. It may be a due to wrong kind of testing methods, cases or testing wrong areas or features. (Mayers, Sandler & Badgett, 2012)

If the tester feel that the given testing task is impossible to execute, for example in too short a time or with inadequate tools,a psychological problem is that the person will give up in an early state and his performance will be poor. (Mayers, Sandler & Badgett, 2012)

If the expectations and circumstances to conduct the testing efforts are reasonable, and testing can be considered as a process to find errors in the target, it will improve the results. (Mayers, Sandler & Badgett, 2012)

Testing tasks can be an extremely creative and intellectually challenging to execute. Designing a software program or an application requires creativity, however, testing it sufficiently as a whole exceeds these requirements.

(Mayers, Sandler & Badgett, 2012)

In other words, a testing person has to find all the possible errors or vulnerabilities in the target; however, the hacker needs to find just one to attack. (Mayers, Sandler & Badgett, 2012)

9. Strategy of the research

The research part of this thesis work is carried out by using OWASP Web Application Security Verification Standard (ASVS) testing checklists (Appendix 1-9) in two study groups of test users. Group A has four test users and group B has five test users, with the group B is not using any lists of testing tasks and considered as a control group for comparing the final results. Also, one aspect is to study, how a test user with no or less experience could make use of following the testing by detailed lists, which would cover all the important features of the tested target, than not using a listing at all.

Both groups had two weeks for testing time and they performed the testing individually. Group members selected and used those testing tools as they wished. Group A consisted of testers from Master of Engineering students and Group B from Bachelor of Engineering students. Since the group members were not studying on the same degree level it was possible that there might be some differencies in the individual experience and know-how.

It is possible, at least in theory that the tested target has no errors or faulty behavior at all. Or there could be just one very fatal error with an impact on wide areas. Considering the testing method between qualitative or quantitative, when the focus is stabling the quality of security testing, all the findings are valuable. The amount of found abnormalities or deviations is important for indicating that there are areas in the tested target in need to be fixed. On the other hand, the nature or severity of the found vulnerabilities or errors depends of the organization's assets and it is vital to cover thoroughly all the parts that are crucial to the information security.

Open Web Application Security Project (OWASP) Application Security

Verification Standard (ASVS) checklists are used by testing the Level 1 (Opportunistic) category test cases; of all the test areas only the Mobile part is excluded. Level 1 verification covers vulnerabilities that are found with low effort, and the detected exploits are simple and easy to find. Level 2 (Standard) and Level 3 (Advanced) are out of the scope of this thesis due to the demanding requirements and much larger work load.

Testing environment is established in JYVSECTEC virtual network, platform is VMWare vCloud requiring Firefox web browser, VMWare integration plugin and Flash player. Remote testing is enabled by using Virtual Private Network (VPN) connection to the testing network. Testing target is WordPress hosting platform for creating blogs or websites.

Used testing tools are included in the Kali Linux penetration and security testing tool distribution which is available in the virtual environment testing machines.

10. Results

10.1 Group A final test results

Group A consist of four test users conducting the tests by the Open Web Application Security Project (OWASP) Security testing checklists. Only Level 1 cases are selected for testing. Master's of Engineering students executed the testing individually with the tools they have chosen to use. They gave the final results for test cases by the following category where pass is considered as everything is working fine; fail meaning there is a fault detected and not applicable as the test case is not possible to execute or the test user do not know how to verify the case is practice. Group A results are presented in the following Tables 8-16 with test case description and final verdict; pass,fail or not applicable.

Web Application Security Testing checklists for Group A

Authentication

Table 8 (Owasp.org, 2015)

Testing task	**Pass**	**Fail**	**Not applicable**
Test all pages that require authentication	3 With the exception of those specifically intended to be public	1 Some pages under http://magazine.yiip.local/wp-admin are viewable without user authentication, for example http://magazine.yiip.local/wp-admin/css/media.css	
Test that user's password is not echoed after entering	3 Password is not echoed		1
Test also authentication controls fail securely, that attacker can't log in	4		
Test that credentials and other identification information handled by the applications, do not traverse unencrypted		3 Login is in plain HTTP	1
Test that forgotten password and other recovery paths do not reveal the current password and the new password is not sent in clear text	3 Password recovery is disabled		1
Test that user name enumeration is not possible via log in, password reset or forgot account functionality	1	2 Usernames can be enumerated on the wp-admin login prompt which indicates whether the user exists or not	1
Test that default user names or passwords are not used (Admin/password)	3 Wordpress does not have a default	1 Wordpress default user	

	password it is auto-generated	name admin used	

Session Management
Table 9 (Owasp.org, 2015)

Testing task	**Pass**	**Fail**	**Not applicable**
Test that frameworks default session managament control is used by the application	2 Wordpress default management control found and no other		2
Test that sessions are invalidated when the user logs out	3		1 Application credentials not provided
Test that session is time outed after a specified period of inactivity		2 Log in and leaving the web browser open, possible to operate after several hours	2 Application credentials not provided
Test that all pages requiring authentication, has log out links	3 Every authenticated page has a top banner with logout link		1 Application credentials not provided
Test that session id is not disclosed other than in cookie headers, in URLs, error messages, or logs. Verify that the application does not support URL rewriting of session cookies.	1 Session ID is only in cookies		3 Application credentials not provided
Test that authenticated session tokens using cookies sent via HTTP, are protected by the use of ”Http Only”		2 Wordpress is using plain cook-ies without HttpOnly or Secure attribute	2
Test that authenticated session tokens using cookies are protected with the secure attribute and a strict transport security header are present		1 Security attribute is not present	3 Application credentials not provided

Access Control

Table 10 (Owasp.org, 2015)

Testing task	**Pass**	**Fail**	**Not applicable**
Test that users can access only to the services they are authorized	2 No bypasses found		2 Application credentials not provided
Test that users can access only to the URLs they are authorized	2 No bypasses found		2 Application credentials not provided
Test that users can access only to the data files they are authorized	1 No bypasses found		3 Application credentials not provided
Test that direct object references are protected(for example file names, paths, database record) for direct object reference tampering	2 No bypasses found		2
Test that directory browsing is disabled	1	3 Directory browsing is possible e.g …/wp-content/uploads	
Test that access control fails in a secure manner	2		2
Test that all access controls are enforced on the server side	2 No user side access control checks found		2
Test that the application generates strong random anti-CSRF tokens unique to the user as part of all high value transactions or accessing sensitive data, and the application verifies the presence of this token with the proper value for the current user when processing these requests	2 Anti-CSRF field called ”_nonce” is present and it could not be broken. Note that earlier versions of wordpress had CSRF bypass vulnerabilities		2

Malicious input handling
Table 11 (Owasp.org, 2015)

Testing task	**Pass**	**Fail**	**Not applicable**
Test that buffer overflows are prevented in the run time environment	1	1 Wordpress is running on PHP 5.3.3 which has multiple buffer overflow vulnerabilities	2
Test that all input validation failures result in input rejection	3		1
Test that all input validation or encoding routines are performed and enforced on the server side	3 No client side validation code found		1
Test that SQL injection is prevented in security controls	2 Could not find SQL injections	1 Based on the detected Wordpress version number 4.0, SQL injection might be possible, e.g. https://wpvulndb.com/vulnerabilities/7929	1
Test that LDAP injection is prevented in security controls	3 None detected. Test scope limited to web interface, source code review out of scope		1
Test that OS Command injection is prevented in security controls	3 Could not find command injections		1
Test that XML External Entity attack is prevented in security controls	2 No XML found		2
Test that XML Injection is prevented in security controls	3 No XML found		1
Test that all untrusted data that are output to HTML are properly escaped for the applicable context	1	1 Wordpress 4.0 has multiple cross site scripting vulnerabilities	2

Error handling and logging
Table 12 (Owasp.org, 2015)

Testing task	Pass	Fail	Not applicable
Test that application does not reveal in error messages or stack traces containing sensitive data that could help attacker, including session id or personal information	3 No stack traces or other useful error messages found		1

Data Protection
Table 13 (Owasp.org, 2015)

Testing task	Pass	Fail	Not applicable
Test that all forms containing sensitive information have disabled client side caching, including auto complete features	1 If user name/email fields are not considered sensitive	1 Autocomplete is not disabled	2
Test that all sensitive data is sent to the server in the HTTP message body, for example URL parameters are never used to send sensitive data	2 No sensitive data in URL parameters		2

Communications Security
Table 14 (Owasp.org, 2015)

Testing task	Pass	Fail	Not applicable
Test that a path can be built from a trusted CA to each Transport Layer Security (TLS) server certificate, and every server certificate is valid		3 A certificate exists, but is only valid for CN 'wp-test', is not signed by a trusted CA and has already expired as well.	1

HTTP Security
Table 15 (Owasp.org, 2015)

Testing task	**Pass**	**Fail**	**Not applicable**
Test that application accepts only agreed HTTP request methods, e.g. GET/POST, all other methods are blocked	1	1 The application accepts HTTP TRACE, OPTIONS and HEAD methods.	2
Test that every HTTP response contains a content type header specifying a safe character set, for example UTF-8	2 UTF-8 defined	1 Content type header value not seen in responses	1
Test that HTTP headers for older browsers are protected against click jacking attacks		2 Security header options seem to be missing	2

Files and resources
Table 16 (Owasp.org, 2015)

Testing task	**Pass**	**Fail**	**Not Applicable**
Test that URL redirecting or forwarding does not include unvalidated data	2 Not added unvalidated		2
Test that file names and path data from untrusted sources is canonicalized, to prevent path traversal attacks	2 Could not force directory traversal		2
Test that files from untrusted sources are scanned by AV scan to eliminated uploading of known malicious content		1 It was able to upload EICAR antimalware test file	3 No uploading functionality detected
Test that parameters obtained from untrusted sources are not used in manipulating file names, path names or any file system object without first being canonicalized and input validated to prevent local file inclusion attacks	2 Was not able to do a file inclusion attack		2

Test that parameters obtained from untrusted sources are canonicalized, input validated, and output encoded to prevent remote file inclusion attacks, particularly where input could be executed, such as header, source or template inclusion	2 Was not able to do a file inclusion attack		2
Test that remote IFRAMEs and HTML5 cross-domain resource sharing does not allow inclusion of arbitrary remote content	2		2

10.2 Group B final results

Group B consist of five test users conducting the tests without any guidance or organized testing structure. All the detected vulnerabilities are listed in the following Table 17, with corrective actions if it is recommended.

Table 17. Group B final test results

Found vulnerabilities	**Corrective actions**
Old version of WordPress	Update to the latest version
Old version of Apache server	Update to the latest version
Apache server has debug features enabled	Disable unnecessary debug features
Ping enabled (port sniffing possible)	Disable ping
Default user name and password	Change default user name and password
Too short passwords	Use longer and safe passwords
Open ports found 22(SSH), 80(HTTP)and 443(HTTPS)	Close SSH port for preventing accessing the root log in

SSH allows weak MD5 and 96-bit MAC algorythms	
DNS allows listing of used names	
HTTP TRACE method is active	Deactivate TRACE method for cookie capture
PHP reveals potentially sensitive information via certain HTTP request that contain specific query strings	
/icons/: Directory indexing found	
/xmlrpc.php was found	
Apache default file found	
/license.txt: License file found may identify site software	
/wp-content/uploads/: Directory indexing found (browsable)	

Group B reported what testing used they had chosen to use during the testing session. In the following table the security tools are listed with describtion of usage.

Table 18. Used security testing tools

Security testing tool	**Purpose**
Armitage	Graphical cyber attack tool
Dirb	Web content scanner
John the Ripper	Password cracker
Nessus	Web vulnerability scanner

Nikto	Web server scanner
NMAP	Security and port scanner
Open VAS	Open Vulnerability Assessment System
OWASP ZAP	OWASP Zed Attack Proxy penetration testing tool
phpMyAdmin	Managing MySQL administration over the internet
SQLmap	SQL injection and penetration testing tool
Wireshark	Network protocol analyzer

11. Conclusions

According to the final results, quantity and quality in the detected testing areas where better in the group A executing testing by the listing. Group A testing by the OWASP checklists produced much versatile and detailed information of the tested target, its overall situation as well as pointing out faulty or vulnerable areas in the security. The control group executing testing by their own knowledge and talent, seeking information of possible vulnerabilities and suitable exploits from the internet did not perform very thorough examination of the situation. Control group checked out the first steps which are interesting and easiest to begin with the possible attack, including all the open ports, default user names and weak passwords, found vulnerabilities and old versions on the system. That gave valuable information of which corrective actions should be done to improve the security at the lowest level.

Despite the better final results of using checklists in the testing, there was some disagreements in the results among the test users, same test case

could have all the three results; pass, fail or not applicable which rise a question of reasons for such a dispersion in the conclusions. Possible reasons for this disagreement would be in the experience and the knowledge of the test users, familiarity of the testing tools or the testing environment and target. Many cases were undone and got the not applicable verdict for the test user not understanding the actual case or how the testing could be executed and verified in practise. Also the two testing groups consisted of master and bachelor level students which are not equal in experience and talent. The master students of group A might have more know-how and more working years behind them which shows in the results.

Those differencies in results could be efficiently be decreased by users training in tools, testing target, environment and going through the checklist cases, so that the lack of talent or knowledge could be updated to the required level before the testing session.

One reason for conducting the thesis research was to find a meaningful way to achieve an equal and stabile quality in testing by examinating if using the checklists as a guidance for testing would be useful. Evaluating the results within the group A pointed out that there was variation in the test case verdicts, but it revealed important issues of how this goal could be reached by proper training before the testing session is started. I think it is important to take care of all the aspects that can have influence to the testing quality as mentioned earlier in the theory section; human factors, limitations to testing resources (e.g. time, money, equipment) and possible standards that must be followed. In my opinion using the checklists have benefits but it can not resolve the quality problem in testing alone.

References

Bath, G. & McKay J., 2008. The Software Test Engineer's Handbook. A Study Guide for the ISTQB Test Analyst and Technical Test Analyst Advanced Level Certificates. 1St

Edition. Rocky Nook Inc.

The British Computer Society. The Chartered Institute for IT. Top 10 Database Attacks. Accessed 27.01.2015. Retrieved from

http://www.bcs.org/content/ConWebDoc/8852

Calder, A., & Watkins, S. 2012. IT Governance: An International Guide to Data Security and ISO27001/ ISO27002, Fifth Edition. Kogan Page. Accessed 12.03.2015. Retrieved from Books24x7.

http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=45978

CERT Division of The Software Engineering Institute (SEI). Carnegie Mellon University. Accessed 10.02.2015. Retrieved from

http://www.cert.org/resilience/products-services/octave/

Chemuturi, M., 2011. Mastering Software Quality Assurance: Best Practices, Tools and Techniques for Software Developers.J. Ross Publishing. Accessed 19.03.2015. Retrieved from Books24x7.

http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=37815

Chess, B., 2006. Quantifying Software Security Risk. Software Assurance Metrics and Tool Evaluation. Accessed 20.03.2015. Retrieved from http://samate.nist.gov/SSATTM_Content/papers/Metrics%20That%20Matter%20-%20Chess.pdf

Chuvakin, A., Schmidt, K., Phillips, C., 2013. Logging and Log Management: The Authoritative Guide to Understanding the Concepts Surrounding Logging and Log Management. Syngress Publishing. Books24x7. Accessed 21.01.2015. Retrieved from http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=51022

Farrell-Vinay, P. 2008. Manage Software Testing.Auerbach Publications. Books24x7. Accessed 15.01.2015. Retrieved from http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=26451

Gregg, M. 2013. Certified Ethical Hacker (CEH) Cert Guide. Pearson Education, Inc.

Graves, K. 2010. CEH: Certified Ethical Hacker: Study Guide. Sybex. Books24x7. Accessed 20.01.2015. Retrieved from http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=31967

IEEE Standard for Software and System Test Documentation. 829-2008. Published 18.07.2008 by IEEE.

Kaspersky Lab, USA. Information Security Company. Accessed 27.01.2015. Retrieved from

http://usa.kaspersky.com/internet-security-center/threats/adware

Lewis, W. E. 2009. Software Testing and Continuous Quality Improvement. Third Edition. Auerbach Publications. Taylor & Francis Group.

Linden, M., 2007. Testing Code Security. Auerbach Publications. Accessed 10.02.2015. Retrieved from Books24x7. http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=26476

McClure, S, Scambray,J.,Kurtz, G., 2012. Hacking Exposed 7: Network Security Secrets & Solutions. McGraw-Hill/Osborne. Accessed 23.01.2015. Retreived from Books24x7. http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=47609

Microsoft. Safety & Security Center. What is spyware? Accessed 27.01.2015. Retrieved from

http://www.microsoft.com/security/pc-security/spyware-whatis.aspx

Ministry of Defence. National Security Auditing Criteria.

Accessed 11.03.2015. Retrieved from
http://www.defmin.fi/files/1870/KATAKRI_versio_II.pdf

Ministry of Finance. Information security and cybersecurity.
Accessed 11.03.2015. Retrieved from

https://www.vahtiohje.fi/web/guest/729

Messier, R. 2014.GSEC GIAC Security Essentials Certification All-in-One Exam Guide. McGraw-Hill/Osborne. Accessed 27.01.2015. Retrieved from Books24x7.
http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=59129

Myers, G., Sandler, C.,& Badgett., T., 2012. The Art of Software Testing, Third Edition. John Wiley & Sons. Accessed 13.03.2015. Retrieved from Books24x7.

http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=45143

Open Web Application Security Project. OWASP Top 10-2013. The Ten Most Critical Web Application Security Risks. Accessed 23.01.2015. Retrieved from https://www.owasp.org/

Open Web Application Security Project. Application Security Verification Standard. Accessed 10.03.2015. Retrieved from
https://www.owasp.org/images/5/58/OWASP_ASVS_Version_2.pdf

Orebaugh, A & Pinkard,B. 2008.Nmap In the Enterprise: Your Guide to Network Scanning. Syngress Publishing. Books24x7. Accessed 16.01.2015. Retrieved from
http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=32227

Oriyano, S. 2014. CEHv8: Certified Ethical Hacker Version 8 Study Guide.Sybex. Books24x7. Accessed 23.01.2015. Retrieved from
http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=63487

Ransome, J., Anmol M., & Schoenfield, B. 2014. Core Software Security: Security at the Source. Auerbach Publications. Books24x7. Accessed 09.02.2015. Retrieved from

http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=51951

Rogers, R. 2008. Nessus Network Auditing, Second edition.Syngress Publishing. Books24x7. Accessed 16.01.2015. Retreived from http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=32366

Kim, D. & Solomon, G., 2014. Fundamentals of Information Systems Security, Second Edition. Jones and Bartlett Publishers. Books24x7. Accessed 04.02.2015. Retrieved from http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=69815

Secure Application Development. Accessed 10.02.2015. Retrieved from

http://secappdev.org/handouts/2012/Paco%20Hope/RBST-SecAppDev-Hope-2012.pdf

Software Testing Class. Complete Website for Software Testing Folks. Accessed 13.02.2015. Retrieved from http://www.softwaretestingclass.com/what-is-risk-based-testing-in-software-testing/

Stuttard, D. & Pinto, M., 2012. The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws, Second Edition. John Wiley & Sons. Accessed 22.01.2015. Retrieved from Books24x7.

http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=44458

Technical Guide to Information Security Testing and Assessment. National Institute of Standards and Technology. NIST Special Publication 800-115. Accessed on 22.01.2015. Retrieved from

http://csrc.nist.gov/publications/nistpubs/800-115/SP800-115.pdf

TechRepublic. Tech Pro Research. Accessed 13.02.2015. Retrieved from

http://www.techrepublic.com/blog/it-security/the-pros-and-cons-of-security-risk-management/

Tipton, H & Krause, M. 2007. Information Security Management Handbook, Sixth Edition, Volume 1.Auerbach Publications.Books24x7. Accessed 15.01.2015. Retrieved from http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=26438

Watson, D. & Jones, A., 2013. Digital Forensics Processing and Procedures: Meeting the Requirements of ISO 17020, ISO 17025, ISO 27001 and Best Practice Requirements. Syngress Publishing. Books24x7. Accessed 13.01.2015. Retrieved from http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=56537

Vacca, J. 2013.Computer and Information Security Handbook, Second Edition. Morgan Kaufmann Publishers. Accessed 16.01.2015. Retrieved from Books24x7. http://common.books24x7.com.ezproxy.jamk.fi:2048/toc.aspx?bookid=54052

Appendices

Web Application Security Testing checklists

The research using of Open Web Application Security Project (OWASP) Security testing checklists in the testing process by other group of testers and the other group is testing without any guidance. Only Level 1 cases are selected for testing.

Appendix: Authentication

Table 8 (Owasp.org, 2015)

Testing task	**Pass/Fail/Not applicable**
Test all pages that require authentication	
Test that user's password is not echoed after entering	
Test also authentication controls fail securely, that attacker can't log in	
Test that credentials and other identification information handled by the applications, do not traverse unencrypted	
Test that forgotten password and other recovery paths do not reveal the current password and the new password is not sent in clear text	
Test that user name enumeration is not possible via log in, password reset or forgot account functionality	
Test that default user names or passwords are not used	

(Admin/password)	

Appendix: Session Managment

Table 9 (Owasp.org, 2015)

Testing task	**Pass/Fail/Not applicable**
Test that frameworks default session managament control is used by the application	
Test that sessions are invalidated when the user logs out	
Test that session is time outed after a specified period of inactivity	
Test that all pages requiring authentication, has log out links	
Test that session id is not disclosed other than in cookie headers, in URLs, error messages, or logs. Verify that the application does not support URL rewriting of session cookies.	
Test that authenticated session tokens using cookies sent via HTTP, are protected by the use of ”Http Only”	
Test that authenticated session tokens using cookies are protected with the secure attribute and a strict transport security header are present	

Appendix: Access Control

Table 10 (Owasp.org, 2015)

Testing task	**Pass/Fail/Not applicable**
Test that users can access only to the services they are authorized	
Test that users can access only to the URLs they are authorized	
Test that users can access only to the data files they are authorized	
Test that direct object references are protected(for example file names, paths, database record) for direct object reference tampering	
Test that directory browsing is disabled	
Test that access control fails in a secure manner	
Test that all access controls are enforced on the server side	
Test that the application generates strong random anti-CSRF tokens unique to the user as part of all high value transactions or accessing sensitive data, and the application verifies the presence of this token with the proper value for the current user when processing these requests	

Appendix: Malicious input handling

Table 11 (Owasp.org, 2015)

Testing task	**Pass/Fail/Not applicable**
Test that buffer overflows are prevented in the run time environment	
Test that all input validation failures result in input rejection	
Test that all input validation or encoding routines are performed and enforced on the server side	
Test that SQL injection is prevented in security controls	
Test that LDAP injection is prevented in security controls	
Test that OS Command injection is prevented in security controls	
Test that XML External Entity attack is prevented in security controls	
Test that XML Injection is prevented in security controls	
Test that all untrusted data that are output to HTML are properly escaped for the applicable context	

Appendix: Error handling and logging

Table 12 (Owasp.org, 2015)

Testing task	**Pass/Fail/Not applicable**
Test that application does not reveal in error messages or stack traces containing sensitive data that could	

help attacker, including session id or personal information	

Appendix: Data Protection

Table 13 (Owasp.org, 2015)

Testing task	**Pass/Fail/Not applicable**
Test that all forms containing sensitive information have disabled client side caching, including auto complete features	
Test that all sensitive data is sent to the server in the HTTP message body, for example URL parameters are never used to send sensitive data	

Appendix: Communications Security

Table 14 (Owasp.org, 2015)

Testing task	**Pass/Fail/Not applicable**
Test that a path can be built from a trusted CA to each Transport Layer Security (TLS) server certificate, and every server certificate is valid	

Appendix: HTTP Security

Table 15 (Owasp.org, 2015)

Testing task	**Pass/Fail/Not applicable**
Test that application accepts only agreed HTTP request methods, e.g. GET/POST, all other methods are blocked	

Test that every HTTP response contains a content type header specifying a safe character set, for example UTF-8	
Test that HTTP headers for older browsers are protected against click jacking attacks	

Appendix: Files and resources

Table 16 (Owasp.org, 2015)

Testing task	**Pass/Fail/Not applicable**
Test that URL redirecting or forwarding does not include unvalidated data	
Test that file names and path data from untrusted sources is canonicalized, to prevent path traversal attacks	
Test that files from untrusted sources are scanned by AV scan to eliminated uploading of known malicious content	
Test that parameters obtained from untrusted sources are not used in manipulating file names, path names or any file system object without first being canonicalized and input validated to prevent local file inclusion attacks	
Test that parameters obtained from untrusted sources are canonicalized, input validated, and output encoded to prevent remote file inclusion attacks, particularly where input could	

be executed, such as header, source or template inclusion	
Test that remote IFRAMEs and HTML5 cross-domain resource sharing does not allow inclusion of arbitrary remote content	