Dalalana Bertoglio and Zorzo *Journal of the Brazilian Computer Society* (2017) 23:2
DOI 10.1186/s13173-017-0051-1

Journal of the
Brazilian Computer Society

RESEARCH Open Access

Overview and open issues on penetration test

Daniel Dalalana Bertoglio* and Avelino Francisco Zorzo

Abstract

Several studies regarding security testing for corporate environments, networks, and systems were developed in the past years. Therefore, to understand how methodologies and tools for security testing have evolved is an important task. One of the reasons for this evolution is due to penetration test, also known as Pentest. The main objective of this work is to provide an overview on Pentest, showing its application scenarios, models, methodologies, and tools from published papers. Thereby, this work may help researchers and people that work with security to understand the aspects and existing solutions related to Pentest. A systematic mapping study was conducted, with an initial gathering of 1145 papers, represented by 1090 distinct papers that have been evaluated. At the end, 54 primary studies were selected to be analyzed in a quantitative and qualitative way. As a result, we classified the tools and models that are used on Pentest. We also show the main scenarios in which these tools and methodologies are applied to. Finally, we present some open issues and research opportunities on Pentest.

Keywords: Security testing, Penetration test, Systematic mapping study

Background

Introduction

The security risks for companies, organizations, and entities that work with sensitive data, from the public sector or not, are more than evident. In many situations, these companies are not able to understand the extension of the actual complex communication structures and have just a little or no control of them [1]. Furthermore, these risks are even bigger when applications that run on their computing infra-structures are taken into consideration. The risks that are not controlled may increase the number of security attacks that can become big financial losses.

Usually, security can be guaranteed by some protection mechanisms: prevention, detection, and response. Prevention is the process of trying to stop intruders from gaining access to the resources of the system. The detection occurs when the intruder has succeeded or is in the process of gaining access to system. Finally, response is an aftereffect mechanism that tries to respond to the failure of the first two mechanisms. It works by trying to stop and/or prevent future damage or access to a facility [2].

*Correspondence: dalalana@gmail.com
Pontifical Catholic University of RS (PUCRS), Porto Alegre, Brazil

However, assessing the security state is a continuous and necessary task to understand the risks there exist. This assessing is usually performed through security tests. So, the use of the right techniques for security testing is an important task to minimize the existing security risks in any corporation [3].

One of the known forms to assess the state of security and reduce security risks is called penetration test (Pentest). Pentest is a controlled tentative to penetrate into a system or network in order to identify vulnerabilities. Pentest applies the same techniques that are used in a regular attack by a hacker. This alternative allows that appropriate measures are taken in order to eliminate the vulnerabilities before they can be explored by unauthorized people [4].

These regular attacks are made with the aim to read, damage, or steal data. The attacks can be classified as follows [5]:

- Denial of service (DoS): an attacker makes some computing resources too busy to handle legitimate requests.
- Remote to user (R2L): an attacker who does not have an account on a remote machine sends packets to that machine over a network and exploits some

vulnerability to gain local access as a user of that machine.
- User to root (U2R): an attacker starts out with access to a normal user account on the system and is able to exploit system vulnerabilities to gain root access to the system.
- Probing: an attacker scans a network of computers to gather information or find known vulnerabilities. An attacker with a map of machines and services that are available on a network can use this information to look for exploits.

Based on this classification, some of the activities are related to the Pentest process. Usually, the Pentest process may be divided into the following activities: data gathering of the target system; scaning the target system to identify the available services/protocols; identifying existing systems and applications that are running on the target system; and identifying and exploit the known vulnerabilities on the systems and applications [6]. Further to the objective mentioned in the previous paragraph, Pentest can be applied also to understand whether the security team is performing their task appropriately or whether the companies security process is comprehensive.

The process to apply Pentest can be a way to evaluate the security level of a system. The stronger the Pentest is, the more complete is the evaluation of the weakness/strength of a system. Regarding the activities and criteria of Pentest, there are several issues that have to be taken into consideration, for example, legal implications and type of information that is being accessed. As such, the application of Pentest can be classified as follows [4]:

- Information base: level of knowledge about the company before the execution of Pentest.
- Aggressiveness: depth level of the test, i.e., determine whether it is trying to identify the main vulnerabilities or whether it should exploit all possible attacks.
- Scope: set for a specific environment or to a general environment.
- Technique: what are the techniques and methodologies used on Pentest.

In order to understand how Pentest is being investigated or how Pentest has evolved in the past years, this work presents a systematic mapping study (SMS) [7] that was conducted to map out the Pentest field. Moreover, this paper aims also to identify research trends, methodologies, scenarios, and tools in Pentest. An SMS is considered a secondary study to find and aggregate evidences available about a specific subject. Therefore, it provides an overview of a research area, identifies the quantity, quality, kind of research, and the available results. Hence, this study will be able to serve as base for primary studies, once the results may identify the answers related to available models, scenarios, and tools. Also, it provides a discussion about the existing open issues in the area. The main contribution of this paper is to provide an overview about the studies on penetration test.

This paper is organized as follows. The "Related work" section describes some related studies considering the mapping of the concepts in a Pentest context. "Systematic mapping study" section describes the *SMS planning* that presents the systematic mapping planning; *conduction* that presents the activities related to the SMS; *results* that describes all obtained results; and *threats to validity* that lists the possible threats to the validity of this study. "Discussion" section presents the discussions based on the defined research questions. The "Lessons learned and future directions" section discusses the main contributions and lessons learned in this study and point out some open issues. The "Removing vulnerabilities: before deployment" section presents some discussion on other ways that might be used for removing vulnerabilities before the system is deployed. Finally, the "Conclusion" section presents the final considerations about this research.

Related work

In the last years, Pentest became an important area and several studies have been developed and applied to improve security in data, systems, and networks. However, there are just a few mapping studies, surveys, or overviews that gather this information in order to show researchers what has been done and what directions they should follow.

Mirjalili and Alidoosti [8] present a survey about Web Pentest, discuss models, and compare vulnerability scanning tools. Besides, they gather works that have new proposals of methods or tools for Web Pentest. Their work shows a selection of primary studies identified in three different ways: studies comparing methods and tools that already exist, studies suggesting a new method or tool, and studies that suggest test environments for Web Pentest. Firstly, the research shows a comparison between 13 different open-source scanning tools, evaluating different criteria regarding their structure (interface, settings, usability, stability, and performance) and their features (spider, manual crawl, file analysis, logging, and reports). A comparison regarding the same criteria is also performed among seven commercial scanning tools, evaluating only their features. In general, the authors insert their main contributions around the relationship between the operation of the vulnerability scanning tools and its application scenarios, target environments, and limitations.

Al-Ghamdi [9] discusses the existing security testing techniques. The study focuses on Pentest considering other test techniques, such as, fuzz testing, binary code analyses, and vulnerability scanning. Conceptually, the

author treats Pentest as ethical hacking and highlights the division of Pentest in black box, white box, and gray box.

Bishop [10] treats the details in Pentest, discussing the correct interpretation of the Pentest, and reiterates the need of a detailed analysis about the activities that are part of a Pentest. In the same way, Geer and Harthorne [11] show the main approaches and opinions about Pentest in a study that is used by several different studies as a conceptual base.

Systematic mapping study

Planning

In particular, the idea behind a SMS is to provide a process to identify and to investigate a specific research area. A systematic literature review, on the other hand [12], aims to analyze, evaluate, and interpret all the available research papers for a determined research question. The SMS provides a broader approach in relation to the existing primary studies for a research theme.

For this objective, this SMS follows the process proposed by Petersen et al. [7], as shown in Fig. 1.

The process shown in Fig. 1 is divided in three phases: planning phase, conduction phase, and reporting phase. Each phase is composed of activities. The SMS planning is described in this section, while the study conduction is presented in the "Conduction" section and the SMS report phase is discussed in "Result analysis" section. The activities inside each phase of the process result in artifacts.

Scope and objective

In this SMS, we focus on identifying the main contributions regarding penetration tests and to provide an overview about models, methodologies, and tools used in this research area. Therefore, the aim of this research is to provide foundation about the Pentest process and its general structure. The results can allow a comprehensive analysis for researchers, security analysts, and other correlated professionals through the discussion about such models, methodologies, and tools.

Question structure

The structure of this SMS is based on the PICO (Population, Intervention, Comparison, Outcome) criteria [12].

- Population: establishes the target population of the research method execution. In this paper, the published research papers are on information security.
- Intervention: represents the specific issue related to the research objective. Here, the intervention is penetration test.
- Comparison: defines what will be compared with the intervention. In this systematic mapping, the comparison is not applied.
- Outcome: the obtained results, like type and quantity of the evidences regarding penetration tests, in order to identify the tools, models, methodologies, scenarios, and main challenges in this area.

Research questions

We defined the following research questions (RQ):

RQ1. What are the main tools used in Pentest? Question defined to identify the tools that are used for Pentest, since in Security Testint, the tool set is very broad.

RQ2. What are the target scenarios in Pentest? Question to identify the environments, contexts, and applications that normally represent the Pentest target.

RQ3. What are the models used in Pentest? Question to determine if the standardization in Pentest area is a consolidated alternative and what are the related methodologies and standards.

RQ4. What are the main challenges in Pentest? Question to map the main open problems, challenges, and possibilities to new studies in Pentest.

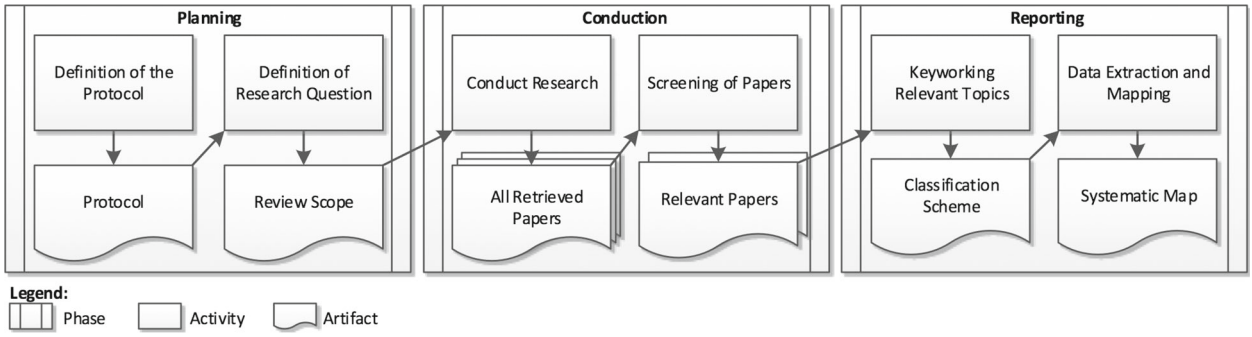

Fig. 1 Systematic mapping study process

Research process

Databases. In order to perform our research, we selected databases that (1) have a web-based search engine; (2) have a search engine able to use keywords; and (3) contain computer science papers. Our selection includes ACM Digital Library, IEEE Xplore, Scopus, and Springer Link.

Terms and synonyms. Based on the RQ, we have used structured terms (shown in Table 1) to construct the search string. The adopted terms are suggested considering an evaluation to identify and map the whole context of Pentest. Although terms related to attack methods or specific tools are not considered, generic terms are specified to find the largest number possible of related studies in Pentest.

String. We have used the logical operator "OR" to select alternate words and synonyms, and the logical operator "AND" to select terms for population, intervention, and outcome (see Fig. 2).

Inclusion and exclusion criteria

One of the essential activities during the SMS planning is the definition of the inclusion criteria (IC) and the exclusion criteria (EC). Such criteria are responsible for supporting the selection of the appropriate papers and are employed to reduce the number of papers that will be analyzed. For example, if a paper is classified in at least one IC, it will be included as a primary study; on the opposite, if a paper is related at least one EC, it will be excluded. Whenever there was a conflict between IC and EC criteria, the researchers involved in this SMS would have a discussion to resolve the conflict. In our SMS, we defined the following IC and EC:

Table 1 Terms used to construct the search string

Structure	Terms	Synonyms
Population	Security information	
Intervention	Penetration test	Pentest
		Penetration testing
		Pentesting
Outcome	Tool	Tools
		Software
		Suite
	Model	Process
		Methodology
		Standard
		Framework
	Environment	Context
	Challenges	Open research topics
		Open problems

```
(("penetration test" OR "penetration testing" OR
"pentest") AND (tool OR tools OR software OR suite)
AND (model OR process OR framework OR methodology OR
standard) AND (environment OR context) AND ("open
research topics" OR challenges OR "open problems"))
```

Fig. 2 Search string

- IC1. The primary study discusses one or more tools for Pentest
- IC2. The primary study suggests a model, process, framework, or methodology for Pentest
- EC1. The primary study is not direct related to Pentest
- EC2. The study shows a Pentest methodology but does not provide enough information about its use and application
- EC3. The study does not have any kind of evaluation to demonstrate outcomes, e.g., case study, experiment, or proof of correctness

The whole SMS was conducted by two researchers, in which, papers would be included or excluded only after a discussion between them achieved an agreement. Basically, one researcher would list all the papers and apply the inclusion and exclusion criteria, and the second researcher would also check whether the papers should be included or excluded. After a meeting to check for discrepancies, the final list of papers that should be analyzed was produced.

Quality assessment criteria

The purpose of the quality assessment criteria (QA) is to ensure the appropriated evaluation of the studies, as a way to measure the relevance of each of them. The quality assessment criteria are:

- QA1. Does the study present a contribution to Pentest?
- QA2. Is there any kind of evaluation based on analysis or discussion about the use of the models or tools for Pentest?
- QA3. Does the study describe the used tools or models?

For each one of the quality assessment criteria questions, we applied the following score: Y (yes) = 1; P (partly) = 0.5; N (no) = 0. Thereby, the total score (result of QA1 + QA2 + QA3) can result in as follows: 0 or 0.5 (limited), 1 (regular), 1.5 (good), 2 (very good), and 2.5 or 3 (excellent).

In order to grade each paper, the reader has to respect the following criteria:

- QA1. Y, the contribution is explicitly defined in the study; P, the contribution is implicit; and N, the

contribution cannot be identified and/or it is not established;
- QA2. Y, the study has explicitly applied an evaluation (for example, a case study, an experiment, or another); P, the evaluation is a short example; and N, no evaluation has been presented;
- QA3. Y, the tools or models are clearly specified; P, the tools or models are barely specified; N, the tools or models were not specified

The quality criteria are applied on each evaluated paper. Besides, these criteria do not consider the details of tools or models described in the selected research papers.

Selection process

Our selection process is divided into six steps, as shown in Fig. 3:

Step 1. To search databases. Initially, the search strings are generated based on keywords and their synonyms. After that, an initial selection occurs based on the inclusion and exclusion criteria mentioned in the "Inclusion and exclusion criteria" section.
Step 2. To eliminate redundancies. As the results come from different search engines, the redundant studies are eliminated and stored.
Step 3. Intermediate selection. The title and the abstract of each selected study are read (introduction and conclusion are also read when it is necessary).
Step 4. Final selection. In this step, all studies are completely read.
Step 5. To eliminate divergences. If there are any divergences or doubts about the studies, a second Pentest specialist reads the studies and discusses its inclusion or not in the final selection;
Step 6. Quality assessment. Based on the quality criteria previously mentioned, the quality of the studies in the final selection are evaluated.

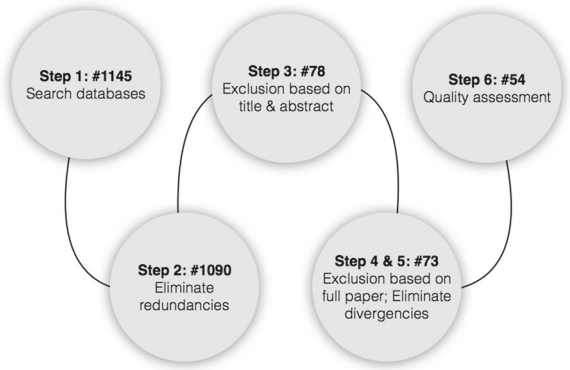

Fig. 3 Selection studies process

Data analysis

The collected data was tabulated to:

- Identify the tools used on Pentest and their characteristics (RQ1);
- Map the main Pentest application domains (RQ2);
- Enumerate the studies that have been selected by models or specifications (RQ3);
- Gather the studies selected by research type and contribution (RQ4).

Conduction

The SMS conduction describes the execution of the search process based on the previously defined string. We conducted the SMS in two periods. The former ended in June 2015, and 1019 papers, published between 2005 and 2015, were retrieved. The latter started in August 2016 and finished in October 2016 when 126 papers, published between 2015 and 2016, were retrieved. In total, 1145 papers were retrieved. In this section, we present in details the steps "Search databases" and "Quality assessment".

Search databases

The 1145 returned papers were retrieved through the submission of the search string to the four databases mentioned in the "Research process" section. With the exclusion of 55 duplicated studies, the title and the abstract of 1090 were read and 78 were selected, which fulfilled one of the inclusion criteria. During the fourth step, all 78 papers were read and 5 of them were removed because they had no straight relation to the expected contributions. From the 73 remaining papers, 19 were excluded after the quality assessment step (step 6). Table 2 shows the total of the remaining primary studies from each database.

Study quality assessment

The inclusion/exclusion criteria, previously mentioned, provide the basis to discuss the applied quality assessment criteria. These criteria help, as the main objective, to evaluate the reliability of the primary studies. Table 3 shows the quality score of the studies. Each study is identified by the column ID, its reference is shown on the column *Reference* and the year of publication in the column Year. Columns 1, 2, and 3 show the scores from the

Table 2 Search engine and retrieved, not duplicated and selected primary studies

Database	Retrieved	Not duplic.	Selected	Prec. rate	Rate index
ACM DL	144	141	8	0.0555	0.1481
IEEE Xplore	531	523	32	0.0602	0.5925
SCOPUS	128	90	3	0.0234	0.0555
Springer Link	342	336	11	0.0321	0.2037
Total	1145	1090	54		

Table 3 Quality studies scores

	Studies		QA			Quality			Studies		QA			Quality	
ID	Reference	Year	1	2	3	Sc	Des	ID	Reference	Year	1	2	3	Sc	Des
01	[13] Austin	2013	Y	Y	Y	3.0	E	28	[62] Line	2008	Y	P	P	2.0	V
02	[40] Hsu	2008	P	P	P	1.5	G	29	[37] Mainka	2012	Y	P	Y	2.5	E
03	[53] Holm	2011	P	Y	N	1.5	G	30	[11] Geer	2002	Y	Y	Y	3.0	E
04	[41] Bechtsoudis	2012	Y	P	Y	2.5	E	31	[48] Traore	2011	Y	P	N	1.5	G
05	[42] Sarraute	2011	Y	Y	Y	3.0	E	32	[39] Benkhelifa	2013	P	P	P	1.5	G
06	[14] Khoury	2011	P	Y	N	1.5	G	33	[22] Salas	2014	Y	Y	Y	3.0	E
07	[34] Antunes	2015	Y	Y	N	2.0	V	34	[23] Büchler	2012	Y	Y	Y	3.0	E
08	[15] Xu	2012	P	P	Y	2.0	V	35	[56] Sandouka	2009	Y	P	P	2.0	V
09	[43] Shen	2011	Y	P	Y	2.5	E	36	[24] Liu	2012	Y	Y	Y	3.0	E
10	[35] Mendes	2011	P	Y	P	2.0	V	37	[52] Masood	2011	Y	P	Y	2.5	E
11	[16] Fong	2008	Y	Y	P	2.5	E	38	[25] Igure	2008	Y	Y	P	2.5	E
12	[65] Williams	2012	Y	Y	P	2.5	E	39	[64] Khoury	2011	Y	P	N	1.5	G
13	[44] Bou-harb	2014	P	Y	Y	2.5	E	40	[26] Leibolt	2010	P	P	P	1.5	G
14	[45] Kasinathan	2013	P	P	P	1.5	G	41	[27] Fonseca	2010	Y	P	P	2.0	V
15	[46] Xing	2010	Y	P	Y	2.5	E	42	[49] Jajodia	2005	P	P	Y	3.0	E
16	[36] Antunes	2009	Y	Y	P	2.5	E	43	[50] Blackwell	2014	Y	Y	Y	3.0	E
17	[54] Holik	2014	Y	Y	Y	3.0	E	44	[28] Prandini	2010	Y	Y	Y	3.0	E
18	[17] Avramescu	2013	Y	Y	Y	3.0	E	45	[59] Dimkov	2010	Y	Y	Y	2.0	V
19	[57] Ridgewell	2013	P	P	P	1.5	G	46	[60] Stepien	2012	Y	P	Y	2.5	E
20	[18] Walden	2008	P	Y	P	2.0	V	47	[29] Badawy	2013	P	P	P	1.5	G
21	[19] Mink	2006	P	P	P	1.5	G	48	[30] Curphey	2006	P	P	P	1.5	G
22	[55] Tondel	2008	P	Y	P	2.0	V	49	[31] Huang	2005	P	P	P	1.5	G
23	[20] Armando	2010	Y	P	P	2.0	V	50	[32] Doupé	2010	P	Y	P	2.0	V
24	[63] Dahl	2006	Y	Y	Y	3.0	E	51	[51] Vegendla	2016	Y	Y	Y	3.0	E
25	[47] Mclaughlin	2010	Y	Y	Y	3.0	E	52	[61] Casseli	2016	Y	Y	Y	3.0	E
26	[58] Somorovsky	2012	Y	Y	Y	3.0	E	53	[38] Antunes	2016	Y	Y	Y	3.0	E
27	[21] Garn	2014	Y	Y	Y	1.5	G	54	[33] Awang	2015	Y	Y	P	2.5	E

Legend - Y: Yes, N: No, P: Partly, Sc: Score, Des: Description, G: Good, V: Very Good, E: Excellent

QA. Column Sc shows the final score for each study, while column Des describes the classification of the study based on the score. As a final result, it is possible to identify that the selected studies are the studies that got at least a score of 1.5.

Result analysis

Classification schemes

One of the activities of the SMS, i.e., "keywording relevant topics" (see Fig. 1), is to decide how the studies will be classified. This activity was executed in two steps: firstly, we read the abstracts of the papers (introduction and conclusion, when necessary) and identified keywords, concepts, and the research context. In the second step, keywords are merged and combined for a more detailed understanding of each selected study. This second step helps the definition of some aspects in the mapping process, where each activity provides the identification of the following aspects: (i) target-scenarios, for example, web applications, web services, network and communication protocols, software and applications, and others; (ii) research type, for example, empirical study, experimental study, industrial experience, opinion papers, proof of concept, and theoretical; (iii) contribution type, for example, tools, frameworks, models, methodologies, strategies, techniques, or approaches; (iv) Pentest methodologies, for example, OSSTMM (Open Source Security Testing Methodology Manual), OWASP Testing Guide, ISSAF (Information Systems Security Assessment Framework), among others. This classification can be seen in Figs. 4 and 5 and the "Discussion" section.

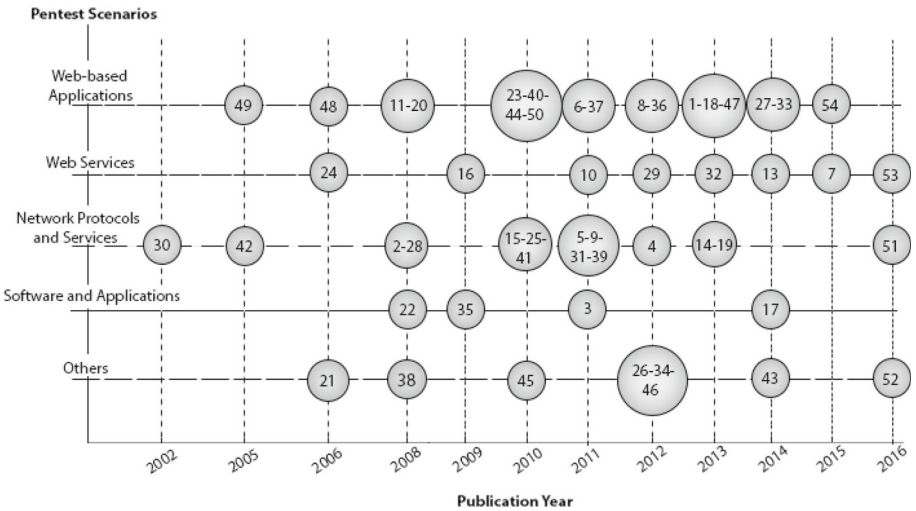

Fig. 4 Bubble plot of the distribution of the studies by scenario and by year

Mapping

This section presents a qualitative assessment of the literature regarding the research questions. Firstly, Fig. 4 shows a bubble graph with the domain distribution of target scenarios in relation to the publication year, where the bubble size indicates the number of related studies at each intersection of the axes.

Studies are grouped by year, so it is possible to visualize how the Pentest has been developed in the last years. As can be seen in Fig. 4, only one study is older that 10 years and it was not ruled out once it is characterized as a primary reference for this SMS. Another characteristic that can be noticed in the figure is that Pentest applied to the web application context is the theme that has showed the biggest frequency in the last years. Therefore, it shows that web application scenarios is one of the main research topics and that it is a live one. Nevertheless, we can see that other Pentest scenarios are still relevant, e.g., network protocols and services.

The analysis of Fig. 4 is related to research question RQ2. Naturally, these scenarios have a strong influence on the tools that were developed or applied to each context, as mentioned in the selected studies.

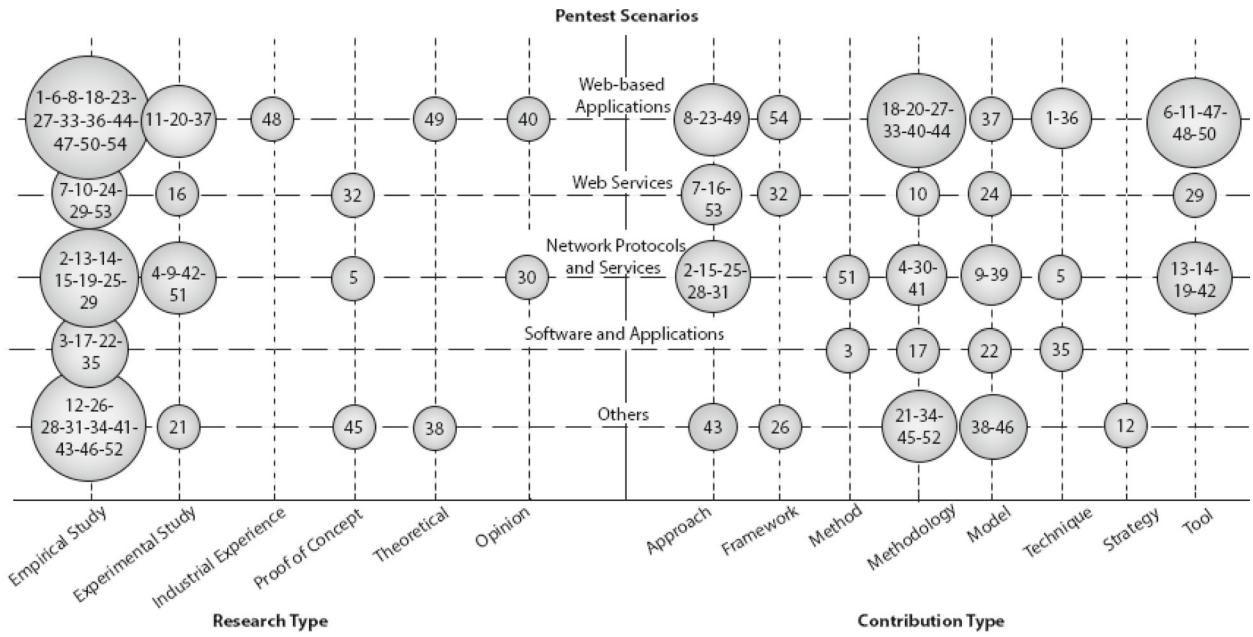

Fig. 5 Bubble Plot of the Scenarios distribution by search type and contribution

Meanwhile, Fig. 5 shows the relationship of the target scenario with the contribution and research type. Hence:

- Discussion on methodologies for Pentest: 15 of the selected and analyzed primary studies present as their main contribution discussion on methodologies. This point encourages discussions about the existing methodologies for the application of Pentest, dealing mainly the deep level of the testing knowledge, since for certain scenarios it becomes interesting to use more models for each security testing process.
- Distribution of the types of research: 37 studies, representing 68,5% overall, were analyzed and characterized as empirical studies. This results seems coherent with the way normally studies on the Pentest area are performed, i.e., research papers are applied to a specific area and therefore are not general strategies that can be applied to any context.

Threats to validity

The main identified threats that can compromise the validity of our SMS in Pentest are:

1. Publication bias: refers to the possibility of some papers that are not selected or published because the research results did not yield the desired outcome or because the research was conducted on topics that do not fit into the common computing conferences and journals. As we analyzed 1019 papers on Pentest, our SMS was not restricted to a small sample of the available papers; thus, it minimizes the risk that some unpublished papers during the searching process impact the SMS results.
2. Primary studies selection bias: usually, the SMS authors cannot guarantee that all relevant primary studies were returned during the search process and during the evaluation. In this sense, the established quality criteria as well as the allocation of scores aim to mitigate that threat.
3. Unfamiliarity with other fields: a search string was defined based on experience and authors' knowledge, but we cannot completely avoid the possibility that some terms defined in the search string have synonyms that we have not identified.

Discussion

In this section, we present and discuss the answers of our research questions.

RQ1—what are the main tools used in Pentest?

After the analysis of the selected studies, we identified 72 tools that are used in Pentest. Among the 72, twelve (12) are categorized as tools for static analysis, which is a technique for security analysis. These tools are relevant because of their usefulness in the analysis and identification of code vulnerabilities, an important task in the Pentest process.

The Pentest process is divided, basically, into three phases: pre-attack, attack, and post-attack. The three stages are formed by five phases, according to the hacking process: reconnaissance (pre-attack), scanning (pre-attack), gaining access (attack), maintaining access (attack), and covering tracks (post-attack). Each phase can be briefly described as follows [1]:

- Reconnaissance: Reconnaissance is the process of obtaining essential information about a target organization. In most cases, attackers will find out as much as they can usually by obtaining public information or masquerading as a normal user.
- Scanning: In this phase, remotely accessible hosts are mapped. Network scanning can also sometimes reveal the vendor brands of systems being used, as well as identify operating system types and versions. Network scanning helps to determine firewall location, routers in use, and the network's general structure.
- Gaining access: Vulnerabilities exposed during the reconnaissance and scanning phases are exploited to get access to the target system.
- Maintaining access: Once the access to a target system was achieved, it is necessary to keep this access for a future exploitation and attack.
- Covering tracks: The last phase covers tracks to avoid detection after the hacker has achieved the access.

The other 60 tools are mainly used for vulnerability scanning. Usually, they are tools used in the early stages of Pentest. We can also mention that tools for traffic monitoring, or intrusion phase, are also part of a significant portion of the analyzed studies.

Based on the analyzed papers, we identified 13 tools as the most cited ones. Hence, Table 4 presents the main tools used in Pentest, showing for each tool: its manufacturer, type of license, category, and phase in which it is applied during Pentest.

It is important to mention that detailed information about the tools are not present in most studies. The studies show the relevant contribution of the each tool within specific contexts along with the Pentest process. Thus, we had to look for their features and documentation directly in their websites or repositories.

RQ2—what are the target-scenarios in Pentest?

The SMS results show that the Pentest process is applied to several specific target scenarios. These scenarios can be divided in web-based applications and systems [13–33], web services [34–39] network protocols and devices [11, 14, 40–52], software and desktop applications

Table 4 Main tools used on Pentest

Tool name	Manufacturer	License	Category	Phase on Pentest
Acunetix WS	Acunetix	Commercial	Web vulnerability scanner	Pre-attack and attack
WebInspect	HP	Commercial	Web vulnerability scanner	Pre-attack and attack
AppScan	IBM	Commercial	Web vulnerability scanner	Pre-attack and attack
Metasploit	Rapid7	Open Source	Vulnerability exploitation tool	Attack
Nessus	Tenable	Commercial	Vulnerability scanner	Pre-attack
NeXpose	Rapid7	Commercial	Vulnerability scanner	Pre-attack
Nikto	CIRT	Open Source	Web vulnerability scanner	Pre-attack
Nmap	–	Open Source	Port scanner	Pre-attack
Paros	–	Open Source	Web vulnerability scanner; Web proxy	Pre-attack
QualysGuard	Qualys	Commercial	Vulnerability scanner	Pre-attack
WebScarab	OWASP	Open Source	Web vulnerability scanner	Pre-attack
Wireshark	Wireshark	Open Source	Packet crafting tool	Pre-attack

[53–56], network game devices [57], SAML frameworks [58], physical penetration [59], operating system [60], critical infrastructure [61], and process control system [62]. Figure 4 shows the different target scenarios that have a diversity in relation to the number of studies, and as mentioned before, most of the studies are related to web-based applications, network devices, and protocols contexts.

RQ3—what are the models of Pentest?

Regarding security testing models, the results obtained in this SMS are classified on methodologies and categories.

The categories describe, based on the taxonomy of the security testing process (see the "Background" section), how/what is the knowledge about the target information for the test execution. The security test models are categorized into white box, gray box, and black box. White box describes the test in which the tester has the complete knowledge about the infrastructure to be tested [24, 63]. Black-box, in contrast, assume that there is no prior knowledge about the environment. Most of the studies and research papers, mainly around vulnerability discovery tools, perform black box tests [32, 34, 64]. Gray box test represents the middle ground between black box and white box, in which the amount of information about the target is not complete but it is also not non-existent. Among the analyzed papers, Avramescu et al. [17] give an example of gray-box test application.

Analyzing the returned studies, it was possible to identify the following methodologies, frameworks, and security testing models: OSSTMM [24, 28, 54, 59, 61, 65], ISSAF [28, 61], PTES (Penetration Testing Execution Standard) [24], NIST (National Institute of Standards and Technology) Guidelines [28, 61], and OWASP Testing Guide [24, 54, 65]. Concerning the classification of the models, there exist three approaches to Pentest [1]: Exploratory Manual Pentest, Automated Pentest, and Systematic Manual Pentest.

OSSTMM is an international standard methodology for security testing, maintained by ISECOM (Institute for Security and Open Methodologies). The test begins by settings that are established from the scope, representing all possible operational security environment for interaction with any asset. The scope consists of three classes: *COMSEC* (communications security channel), *PHYSSEC* (physical security channel), and *SPECSEC* (spectrum security channel). These classes are divided into five channels before being used by the tester: human, physical, wireless, telecommunications, and data networks. Those channels are used to conduct the test and contain specifications for the security assessment according to the test scenario. There are no indicated tools for the testing process, only information about the tasks to be executed for each channel. Finally, the test ends with the Security Test Audit Report (STAR), which contains data obtained during the activities.

The ISSAF methodology provides a framework able to model the internal control requirements for information security and aims to assess the security of networks, systems, and applications. Its design is structured in three main areas: planning and preparation, evaluation and report, and cleaning and destruction of artifacts. The first area covers the steps required to set the test environment, test tools, contracts and legal aspects, definition of engagement team, deadlines, requirements, and structure of the final reports. The evaluation area is the core of the methodology, where security tests are executed. This phase has other nine main activities, which follow the basic flow of an attack (recognition, invasion, and post-invasion), previously mentioned.

The PTES methodology describe the steps to perform the activities that are required to accurately test the security state in an environment. The purpose of the methodology is not to establish rigid patterns for a penetration test. The community of analysts and security professionals responsible for creating the methodology suggest that the guidelines for the security evaluation process of an environment should be comprehensible for organizations. Therefore, the technical guidelines help to define procedures to follow throughout a Pentest, enabling the methodology to provide a basic structure to initiate and conduct a security test. The methodology consists of seven phases: pre-engagement interactions, which defines the testing scope (goal, target, test type, date, and time); intelligence gathering, which deals with the enumeration and scanning information of the target system; threat modeling, where the attack vectors are analyzed from the information obtained in the previous phases; vulnerability analysis, which deals with the detection of vulnerabilities of the target system; exploitation, used to exploit found vulnerabilities; post-exploitation, which covers the tracks and also performs additional exploitations; and reporting, which is to write the final report to be sent to the customer.

The methodology proposed by NIST (National Institute of Standards and Technology) was initially introduced as a GNST (Guideline on Network Security Testing), reproduced in the Special Publication 800-42, and its continued version is presented in Special Publication 800-115 as "Technical Guide to Information Security Testing and Assessment". Basically, the structure follows four stages: planning, where the system is analyzed to find the most interesting test targets; discovery, where the tester looks for vulnerabilities in the system; attack, where the tester verifies that the found vulnerabilities can be exploited; and report, where each result from the actions taken in the previous step is reported. In the attack stage, the following activities are also present: gaining access, escalating privileges, system browsing, and install additional tools.

OWASP has a methodology driven by the idea of making secure software a reality, and therefore, the guidelines are directed towards testing security for web applications. In most software development organizations, security concerns are not present in the development process. Then, the methodology idealizes the use of security testing as a means of awareness and is based on other projects provided by the OWASP as the Code Review Guide and Development Guide. The methodology is divided into three main stages: the introductory stage, which deals with the preconditions for testing web applications and also the testing scope; the intermediate stage, which presents the OWASP Testing Framework with its techniques and tasks that are related to the different phases of the Software Development Life Cycle; and the conclusive stage that describes how vulnerabilities are tested by Code Review and Penetration Testing.

Based on that, the evaluation of methodologies is performed using some of the features described next. Figure 6 shows a comparison on the methodologies discussed in this SMS.

The classification of the features is categorized as follows:

- Meet (M): Provides detailed definitions and concepts to deal with that feature in an appropriate manner.
- Partly meet (PM): Issues about the feature are mentioned, but without the necessary robustness.
- Not meet (NM): The methodology does not mention anything related to the feature.

Coverage. Initially, one of the important criteria for a security test is the scope. Scope refers to the concerns of the test range over possible scenarios. The OSSTMM [66], ISSAF [67], PTES [68], and NIST [69] methodologies are easily integrated and can be tailored to applications and operating systems, databases, physical security assessments, and web applications. However, the OWASP Testing Guide [70] model has a precisely defined focus: web applications and services. In this sense, the coverage of this methodology can represent a limitation.

Flexibility. The possibility of integrating new items and additional directions at security testing from the results obtained in each step or phase of the methodology is an important feature in the current context of security checks. In this sense, even if a static definition of plans and steps to be followed is a prime requirement, the flexibility to include new items makes a methodology more interesting. For this feature, the model provided by NIST allows the testers to have greater dynamism throughout the test, since they can consider and reevaluate their artifacts in each activity. In contrast, some methodologies, such as, ISSAF, OSSTMM, and OWASP Testing Guide, while consolidated and extremely robust, limit such flexibility by treating the execution scenarios.

Modeling. By defining the detailed aspects and concepts for guiding the testing process, the model may even limit the flexibility but increments the quality of modeling. These key concepts facilitate the tester in their activity to model the entire flow of test actions, in addition to modeling the system and target environment. This confirms a crucial point for security testing, which is the elimination of possible ambiguity in respect of each subsequent step that will be performed. For this characteristic, OSSTMM, OWASP Testing Guide, and PTES models meet this, especially in the way they approach the planning stage of its testing process.

		OSSTMM	ISSAF	PTES	NIST Guidelines	OWASP Testing Guide
Entity		ISECOM	OISSG	-	NIST	OWASP
Last update		2010	2006	2012	2008	2014
Features	Intended to Pentest	No	No	Yes	No	No
	Coverage	M	PM	PM	PM	NM
	Flexibility	NM	NM	PM	M	NM
	Modeling	PM	M	PM	NM	PM
	Adaptation	NM	PM	NM	PM	PM
	Planning	NM	M	M	NM	M
	Documentation	M	M	PM	M	M

Legend - **M**: Meet, **PM**: Partly Meet, **NM**: Not Meet.

Fig. 6 Comparison between the models for Pentest

Adaptation. It is important to have well-defined concepts in order to avoid possible ambiguities and, therefore, impact the adaptation factor. Moreover, the possibility to adapt models and actions for different environments produces a more complete security test flow. Among the possible adaptations are, for example, the choice of test type, test plan, or test scope. From the studied methodologies, on one hand, OSSTMM is the one that can better fulfill this feature, since it has a process with well-defined activities. On the other hand, the PTES methodology presents some limitations for not detailing how adaptations could be performed.

Planning. The whole set of requirements defined in a security test must be properly planned prior to the start of the test execution. Thus, planning is a feature that is the support provided to the tester for the definition phase, implementation of activities, prerequisites for continuation, and progress of the test, choice of tools to be used and also the expected return for each activity within the test. PTES is a methodology that provides this type of feature. It describes, carefully, all the planning that must be defined, in addition to establishing the set of tools, and how to operate them, that will be used in each activity from the Pentest. OSSTMM and NIST, since they try to provide great flexibility, do not focus on providing a very detailed planning.

Documentation. Finally, we can also consider the documentation as part of the key features of setting up a Pentest. All studied methodologies provide how the documentation has to be produced. Only PTES does not provide a complete description of how to produce a documentation that contains detailed explanations of each process and activity. For this reason, it is the only one of the models that does not fully meet this feature.

RQ4—what are the main challenges on Pentest?

In the previous sections, we analyzed the relationship among the target scenarios, tools, and models. Based on that we can draw some initial research challenges on Pentest. One of the main problems discussed in some of the analyzed studies is regarding *the efficacy in the process of vulnerability assessment*. Another challenging research area is *how to provide models and tools to ensure high security levels to some specific target scenarios*. These challenges are related to different types of problems, for example, the complexity of some attacks, discovery of new vulnerabilities, and changes in the environments can change the applicability of Pentest.

Furthermore, *the automation of activities execution for Pentest* can also be considered a challenge. Several of the studies, presented in the previous sections, discuss or present ways to increase the efficiency and efficacy of Pentest throughout automation, for example, for the activity

of vulnerabilities discovery. This will help to avoid bias by the testers when they are executing such activity.

The formalization of methodologies/models disseminated in the security community provides the robustness required for best practices in Pentest. Still, another challenge is precisely related to this: *the specific lack of models that address the Pentest process*.

More on the challenges for Pentest will be discussed in the "Lessons learned and future directions section."

Lessons learned and future directions

Through the research questions answered in the previous sections, we present in this section some future directions and some discussions on tools, target scenarios, models, and main challenges in Pentest:

- Target scenarios: One of the main goals of a security test is to assess the security of the resources, devices, controls, or systems, considering a great diversity of target-scenarios. The majority of the studies that we found considers the web context as top priority when testing security; to a minor extent, network environment and its protocols are also considered important. However, there is almost none discussion on security testing in scenarios such as cloud computing, mobile devices, or solutions related to IoT (Internet of Things). Therefore, studies about security testing applications—especially Pentest—in those scenarios, for example, present the possibility of groundbreaking discoveries and improvements through new studies.
- Models and methodologies: As presented previously, the existing methodologies for security testing contain several variations in their characteristics, objectives, and procedures; however, those methodologies also have limitations regarding target scenarios since they are tailored to serve distinct purposes. Therefore, we believe that none of the so-called "standard" methodologies could be used to execute Pentest considering the variety of target-scenarios. This could be considered one of the core lessons of this systematic mapping since it presents an open challenge in the security testing area. Creating a new methodology or strategy that could manage the diversity of target scenarios and the aspects—advantages and disadvantages—of any existing methodology could potentially point towards a new and interesting path for future studies in security.
- Tools and task automation: During the security testing process, several tools are used for each activity, and tools listed when answering research question 1 (RQ1) are some of the most consolidated in the current research context. Those tools have specific purposes in each testing phase, and the testers can determine when and how those tools will be utilized according to their preferences. Among the tools, it was possible to notice that applications that scan and identify vulnerabilities are the ones that are most cited/mentioned in the research papers. Sometimes those tools are not as adequate for some of the strategies testers use; hence, it is necessary to have some study to verify to what extend those automated tools solve the testers goals. In this sense, the idea of attack graphs is considered a topic related to automation in Pentest. Sarraute et al. [42] discuss that attack graphs have been proposed as a tool to help testers understand the potential weaknesses in the target network, once that assessing network security is a complex and difficult activity. A better explanation about attack graphs is described in [71]. According to their review, attack graphs are used to determine if designated goal states can be reached by attackers attempting to penetrate computer networks from initial starting states. The graphs are made by nodes and arcs, representing the attacker actions (normally involve exploits or exploit steps that take advantage of vulnerabilities) and the changes in the network state caused by these actions. The goal of these actions is for the attacker to obtain typically restricted privileges on one or more target hosts. An attack graph must show all possible sequences of attacker actions that lead to the desired level of privilege on the target. It is possible to use nodes to represent network states and arcs to represent attack actions, while some use other representations like nodes for both actions and networks states and also with actions that are nodes and network states that are arcs [71]. The idea of tools or frameworks that help the tester in the most insightful way during the entire process is an interesting possibility; future studies could study how to bring a better balance to the complexity of testing and the comprehension of the results.
- Dynamics and test reprocessing: Since a Pentest requires the identified vulnerabilities to be exploited, the test activities can be modified according to the consequences of this exploitation. This change affects directly the test dynamics and flow, and some decisions during the activities execution depend on the tester discernment. Nevertheless, a point that is not considered in the related studies, mentioned in this systematic mapping, refers to the flexibility of security testing applications allied to the concerns of reprocessing the stages during the test. In this sense, a continuous evaluation of the executed tasks with the intention of installing verification cycles could result in an increased test efficacy or efficiency, which

could potentially facilitate the enumeration of new attack vectors.

Removing vulnerabilities: before deployment

Despite the main objective of this paper, i.e., to find tools and strategies to test vulnerabilities when the application has already been deployed, there has been a lot of work also on removing vulnerabilities from an application before it is deployed. This is performed during design, development, and testing phases of the software development life cycle. Therefore, this section discusses some of the works that seek to remove vulnerabilities using, basically, testing strategies. It is important to mention that the papers mentioned in this section were not found through an SMS, since this could be subject for a completely new paper.

Avgerinos et al. [72] present a system for automatic vulnerability scanning, called *AEG*. The study describes some important contributions and shows how to generate exploits for hijacking attacks that can be formally modeled. AEG tool was implemented because of insufficient and inadequate source code analysis, a type of evaluation that does not fall into the category of security testing. AEG is designed to work in the process of bug-finding and to generate exploits.

Hossen et al. [73] propose an approach for generating test driver using a crawler that identifies the needed information. The article is a study directed to the context of model-based testing and model inference and not for penetration testing. Other similar publications have also been excluded from the first phase of our mapping, through the inclusion and exclusion criteria.

Felderer and Schieferdecker [74] present a taxonomy of risk-based testing providing a framework to understand, categorize, assess, and compare risk-based testing approaches to support their selection and tailoring for specific purposes. The discussion on this study is based on the fact that software testing has often to be performed under severe pressure due to limited resources and a challenging time schedule. The taxonomy presented is aligned with the consideration of risks in all phases of the test process and consists of the top-level classes risk drivers, risk assessment, and risk-based test process. In general, the authors mention that risk-based testing uses risk re-assessments to steer all phases of the test process to optimize testing efforts and limit risks of the software-based system.

Botella et al. [75] introduce an approach guided by risk assessment to perform and automate vulnerability testing for web applications, called risk-based vulnerability testing. This approach is intended to security testing and adapts model-based testing techniques, which are mostly used currently to address functional features. The paper also mentions that the proposed approach extends model-based vulnerability testing techniques by driving the testing process using security test patterns selected from risk assessment results. In general, the study describes a model used for automated test generation that captures some behavioral aspects of the web applications and includes vulnerability test purposes to drive the test generation process.

Doupé et al. [76] present discussions on a category of tools called "web vulnerability scanner", responsible for finding security vulnerabilities in web applications. The purpose of this study is to detect vulnerabilities that other scanners do not detect, by inference a state machine that controls the changes of the web application. Actually, the use of these tools is usual in the security testing context, mainly in the pre-attack phase. Several other studies are similar to this work [32], because the automation to find vulnerabilities (whether known or not) is a complex and constantly evolving subject. Nonetheless, the authors do not mention any security testing, since the article is focused on the operation of the tools. At the search for vulnerabilities in web applications, the authors only mention the static code analysis, which as mentioned above, does not fit in the discussions around our systematic mapping.

Bouquet et al. [77] discuss the behavior of systems that are tested and executed through a set of selected stimuli, observing if the behavior conforms to the specification. The paper also defines that testing is a strategic activity at the heart of software quality assurance, highlighting that it is today the principal validation activity in industrial context to increase the confidence in the quality of systems. Nevertheless, they give an overview of the test data selection techniques and provide a state-of-the-art about model-based approaches for security testing.

Duchene et al. [78] present the KamaleonFuzz, a fuzzer for web applications designed to XSS detection (cross site scripting). The main idea is based on the concept of "fuzz testing," which consists in the generation and automatic sending of malicious entries to achieve a vulnerability. A fuzz test can be categorized as a security testing, but with a different purpose from a penetration test, i.e., audit or vulnerability analysis. There is an approximation of the issues when the topic addresses to identify vulnerabilities, usually the main objective of security testing.

Godefroid et al. [79] present a solution developed as an alternative to black box fuzzing idea through the definition of white box fuzzing, called SAGE (Scalable Automated Guided Execution). White box fuzzing consists of a symbolic execution of a program, collecting restrictions on inputs found during execution. The proposal is directly related to security testing (in software), but does not point to Pentest. That solution type is characterized as a tool that seeks to discuss security in application development,

rather than related to methods for assessing security states in companies and organizations.

McAllister et al. [80] present an automated testing tool to find XSS vulnerabilities in web applications. In this case, the study treats the bug detection before deployment, like other works previously discussed. This work discusses XSS attacks and presents a comparative of the proposed tool against other tools, e.g., Acunetix tool, that perform vulnerability scanning in web applications.

Kals et al. [81] present SecuBat, a generic web vulnerability scanner that automatically analyzes web sites with the aim of finding exploitable SQL injections and XSS vulnerabilities. The authors also discuss the types of security testing, black box and white box, relating to the tool operation. In addition, they analyze the differences between XSS attack types and conduct a case study to validate the study. As presented in other works discussed in our study, the contributions of this article are related to security testing performed in web applications, although not specifically in Pentest.

Huang et al. [82] describe some software testing techniques and suggest mechanisms to apply these techniques in web applications. The authors also discuss the evaluation of inputs that allow fault injection, and they propose algorithms to perform that. In this case, the security tests are similar to vulnerability assessments that consist of using exploits on application breaches.

Although there are several other studies related to intrusion detection system or security testing, e.g. [83, 84], they were not included in our SMS because they were out of the scope of this paper.

Conclusion

The relevance of the penetration testing (Pentest) is clear from the research point of view. This subject has been widely targeted by researchers of testing and safety, mainly because the number of flaws and vulnerabilities has increased in the last years. This paper focused on mapping the Pentest field, identifying the application scenarios, usual tools and methodologies in different contexts, the main contributions and related challenges.

It was possible to draw some conclusions on how tools or methodologies are used to vulnerability assessment, network scanning, pre-invasion, post-invasion, and web analytics. From that, the results can help testers to define, within their testing scope, which tools or methodologies are indicated depending on the context or scope they will be applied to (see the "Lessons learned and future directions" section).

Based on the lessons learned, it was possible to notice that it would be important to have a set of recommendations aimed to improve and/or complement Pentest. This set of recommendations can be based on the existing methodologies. Thus, a proposed set of recommendations would address the strengths and limitations of the models and also would provide a flexible, dynamic, and many activities choices, steps, and other aspects inherent to a Pentest. Some preliminary results on a new methodology for Pentest that can be applied in different target scenarios is Tramonto [85].

Abbreviations

EC: Exclusion criteria; IC: Inclusion criteria; ISSAF: Information Systems Security Assessment Framework; NIST: National Institute of Standards and Technology; OSSTMM: Open Source Security Testing Methodology Manual; Pentest: Penetration test; PICO: Population, Intervention, Comparison, Outcome; PTES: Penetration testing execution standard; QA: Quality assessment criteria; SMS: Systematic mapping study

Acknowledgements

The authors thank the support from Hewlett Packard Enterprise. We also thank the anonymous reviewers for their comments and suggestions.

Authors' contributions

DDB performed all activities of the proposed SMS and the study analysis. AFZ executed the planning, selection, and revision activities of the SMS. Both authors drafted, read, and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Received: 20 June 2016 Accepted: 25 January 2017
Published online: 06 February 2017

References

1. Lam K, LeBlanc D, Smith BI (2004) Assessing network security. Redmond, Wash. Microsoft Press, Washington
2. Kizza JM (2010) Guide to computer network security. Springer, London
3. Zhao JJ, Zhao SY, Zhao SY (2010) Opportunities and threats: a security assessment of state e-government websites. Gov Inf Q 27(1):49–56
4. Whitaker A, Newman D (2005) Penetration testing and Cisco network defense. Cisco Press, Indianapolis
5. Peddabachigari S, Abraham A, Grosan C, Thomas J (2007) Modeling intrusion detection system using hybrid intelligent systems. J Netw Comput Appl 30(1):114–132
6. Henry KM (2012) Penetration testing: protecting networks and systems. IT Governance Publishing, UK
7. Petersen K, Feldt R, Mujtaba S, Mattsson M (2008) Systematic mapping studies in software engineering. In: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering. EASE'08. British Computer Society, Swinton. pp 68–77
8. Mirjalili M, Nowroozi A, Alidoosti M (2014) A survey on web penetration test. Adv Comput Sci Int J 3(6):107–121
9. Al-Ghamdi ASA-M (2013) A survey on software security testing techniques. Int J Comput Sci Telecommun 4:14–18
10. Bishop M (2007) About penetration testing. IEEE Secur Priv 5(6):84–87
11. Geer D, Harthorne J (2002) Penetration testing: a duet. In: Proceedings of the 18th Annual Computer Security Applications Conference. IEEE. pp 185–195
12. Kitchenham B, Charters S (2007) Technical report title: Guidelines for performing Systematic Literature Reviews in Software Engineering, EBSE 2007-001. Keele University and Durham University Joint Report
13. Austin A, Holmgreen C, Williams L (2013) A comparison of the efficiency and effectiveness of vulnerability discovery techniques. Inf Softw Technol 55(7):1279–1288
14. Khoury N, Zavarsky P, Lindskog D, Ruhl R (2011) An analysis of black-box web application security scanners against stored sql injection. In: IEEE Third International Conference on Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third International Conference on Social Computing (SocialCom). IEEE. pp 1095–1101
15. Xu D, Tu M, Sanford M, Thomas L, Woodraska D, Xu W (2012) Automated security test generation with formal threat models. IEEE Trans Dependable Secure Comput 9(4):526–540

16. Fong E, Gaucher R, Okun V, Black PE, Dalci E (2008) Building a test suite for web application scanners. In: Proceedings of the 41st Annual Hawaii International Conference on System Sciences. IEEE. pp 478–478
17. Avramescu G, Bucicoiu M, Rosner D, Tapus N (2013) Guidelines for discovering and improving application security. In: Proceedings of the 2013 19th International Conference on Control Systems and Computer Science. CSCS '13. IEEE Computer Society, Washington. pp 560–565
18. Walden J (2008) Integrating web application security into the it curriculum. In: Proceedings of the 9th ACM SIGITE Conference on Information Technology Education SIGITE '08. ACM, New York. pp 187–192
19. Mink M, Freiling FC (2006) Is attack better than defense? teaching information security the right way. In: Proceedings of the 3rd Annual Conference on Information Security Curriculum Development. InfoSecCD '06. ACM, New York,. pp 44–48
20. Armando A, Carbone R, Compagna L, Li K, Pellegrino G (2010) Model-checking driven security testing of web-based applications. In: Third International Conference on Software Testing, Verification, and Validation Workshops (ICSTW). IEEE. pp 361–370
21. Garn B, Kapsalis I, Simos DE, Winkler S (2014) On the applicability of combinatorial testing to web application security testing: a case study. In: Proceedings of the 2014 Workshop on Joining AcadeMiA and Industry Contributions to Test Automation and Model-Based Testing. JAMAICA 2014. ACM, New York. pp 16–21
22. Salas MIP, Martins E (2014) Security testing methodology for vulnerabilities detection of XSS in web services and WS-security. Electron Notes Theor Comput Sci 302:133–154
23. Büchler M, Oudinet J, Pretschner A (2012) Semi-automatic security testing of web applications from a secure model. In: IEEE Sixth International Conference on Software Security and Reliability (SERE). IEEE. pp 253–262
24. Liu B, Shi L, Cai Z, Li M (2012) Software vulnerability discovery techniques: a survey. In: Proceedings of the 2012 Fourth International Conference on Multimedia Information Networking and Security. MINES '12. IEEE Computer Society, Washington. pp 152–156
25. Igure VM, Williams RD (2008) Taxonomies of attacks and vulnerabilities in computer systems. IEEE Commun Surv Tutorials 10(1):6–19
26. Leibolt G (2010) The complex world of corporate CyberForensics investigations. Humana Press, New York
27. Fonseca J, Vieira M, Madeira H (2010) The web attacker perspective—a field study. In: ISSRE '10 Proceedings of the 2010 IEEE 21st International Symposium on Software Reliability Engineering. IEEE. pp 299–308
28. Prandini M, Ramilli M (2010) Towards a practical and effective security testing methodology. In: ISCC '10 Proceedings of the The IEEE Symposium on Computers and Communications. IEEE. pp 320–325. doi:10.1109/ISCC.2010.5546813
29. Badawy MA, El-Fishawy N, Elshakankiry O (2013) Vulnerability scanners capabilities for detecting windows missed patches: comparative study. In: Advances in Security of Information and Communication Networks: First International Conference, SecNet 2013, Cairo, Egypt, September 3-5, 2013. Proceedings. Springer, Berlin. pp 185–195. doi:10.1007/978-3-642-40597-6_16
30. Curphey M, Arawo R (2006) Web application security assessment tools. IEEE Secur Priv 4(4):32–41. doi:10.1109/MSP.2006.108
31. Huang YW, Lee DT (2005) Web application security—past, present, and future. In: Computer Security in the 21st Century. Springer, Boston. pp 183–227. doi:10.1007/0-387-24006-3_12
32. Doupé A, Cova M, Vigna G (2010) Why Johnny can't pentest: an analysis of black-box web vulnerability scanners. In: Proceedings of the 7th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment. DIMVA'10. Springer, Berlin. pp 111–131
33. Awang NF, Manaf AA (2015) Automated security testing framework for detecting SQL injection vulnerability in web application(Jahankhani H, Carlile A, Akhgar B, Taal A, Hessami AG, Hosseinian-Far A, eds.). Springer, Cham
34. Antunes N, Vieira M (2015) Assessing and comparing vulnerability detection tools for web services: benchmarking approach and examples. IEEE Trans Serv Comput 8(2):269–283
35. Mendes N, Durães J, Madeira H (2011) Benchmarking the security of web serving systems based on known vulnerabilities. In: 5th Latin-American Symposium on Dependable Computing, LADC 2011, 25-29 April 2011. IEEE, São José Dos Campos. pp 55–64
36. Antunes N, Laranjeiro N, Vieira M, Madeira H (2009) Effective detection of SQL/XPath injection vulnerabilities in web services. In: IEEE International Conference on Services Computing, 2009. SCC '09. IEEE. pp 260–267
37. Mainka C, Somorovsky J, Schwenk J (2012) Penetration testing tool for web services security. In: SERVICES '12 Proceedings of the 2012 IEEE Eighth World Congress on Services. IEEE. pp 163–170
38. Antunes N, Vieira M (2016) Designing vulnerability testing tools for web services: approach, components, and tools. Int J Inf Secur:1–23. http://link.springer.com/article/10.1007/s10207-016-0334-0
39. Benkhelifa E, Welsh T (2013) Security testing in the cloud by means of ethical worm. In: 2013 IEEE Globecom Workshops (GC Wkshps). IEEE. pp 500–505
40. Hsu Y, Shu G, Lee D (2008) A model-based approach to security flaw detection of network protocol implementations. In: IEEE International Conference on Network Protocols, 2008. ICNP 2008. IEEE. pp 114–123
41. Bechtsoudis A, Sklavos N (2012) Aiming at higher network security through extensive penetration tests. IEEE Lat Am Trans 10(3):1752–1756
42. Sarraute C, Richarte G, Lucángeli Obes J (2011) An algorithm to find optimal attack paths in nondeterministic scenarios. In: Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence. AISec '11. ACM, New York. pp 71–80
43. Shen L, Liang X, Bo Y, Xia C (2011) Automatic generation for penetration testing scheme analysis model for network. In: ICCIS '11 Proceedings of the 2011 International Conference on Computational and Information Sciences. IEEE. pp 821–826
44. Bou-Harb E, Debbabi M, Assi C (2014) Cyber scanning: a comprehensive survey. IEEE Commun Surv Tutorials 16(3):1496–1519
45. Kasinathan P, Pastrone C, Spirito MA, Vinkovits M (2013) Denial-of-service detection in 6LoWPAN based Internet of Things. In: IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob). IEEE. pp 600–607
46. Xing B, Gao L, Zhang J, Sun D (2010) Design and implementation of an XML-based penetration testing system. In: International Symposium on Intelligence Information Processing and Trusted Computing (IPTC). IEEE. pp 224–229
47. McLaughlin S, Podkuiko D, Miadzvezhanka S, Delozier A, McDaniel P (2010) Multi-vendor penetration testing in the advanced metering infrastructure. In: Proceedings of the 26th Annual Computer Security Applications Conference. ACSAC '10. ACM, New York,. pp 107–116
48. Traore MD, Jin H, Zou D, Qiang W, Xiang G (2011) Rapn: Network attack prediction using ranking access petri net. In: Sixth Annual Chinagrid Conference (ChinaGrid). IEEE. pp 108–115
49. Jajodia S, Noel S, O'Berry B (2005) Topological analysis of network attack vulnerability. In: Managing cyber threats: issues, approaches, and challenges. Springer, Boston. pp 247–266
50. Blackwell C (2014) Towards a penetration testing framework using attack patterns. In: Cyberpatterns: unifying design patterns with security and attack patterns. Springer, Cham. pp 135–148. doi:10.1007/978-3-319-04447-7_11
51. Vegendla A, Søgaard TM, Sindre G (2016) Extending HARM to make test cases for penetration testing(Krogstie J, Mouratidis H, Su J, eds.). Springer, Cham
52. Masood R, Um-e-Ghazia, Anwar Z (2011) SWAM: Stuxnet worm analysis in metasploit. In: 2011 Frontiers of Information Technology, FIT 2011, Pakistan, December 19-21, 2011. IEEE, Islamabad. pp 142–147
53. Holm H, Sommestad T, Almroth J, Persson M (2011) A quantitative evaluation of vulnerability scanning. Inf Manag Compute Secur 19(4):231–247
54. Holik F, Horalek J, Marik O, Neradova S, Zitta S (2014) Effective penetration testing with metasploit framework and methodologies. In: IEEE 15th International Symposium on Computational Intelligence and Informatics (CINTI). IEEE. pp 237–242
55. Tondel IA, Jaatun MG, Jensen J (2008) Learning from software security testing. In: IEEE International Conference on Software Testing Verification and Validation Workshop, 2008. ICSTW '08. IEEE. pp 286–294
56. Sandouka H, Cullen AJ, Mann I (2009) Social engineering detection using neural networks. In: Proceedings of the 2009 International Conference on CyberWorlds. CW '09. IEEE Computer Society, Washington. pp 273–278
57. Ridgewell WW, Kumar V, Kinshuk (2013) Immersive and authentic learning environments to mitigate security vulnerabilities in networked game devices. In: Proceedings of the 2013 International Conference on

Signal-Image Technology & Internet-Based Systems. SITIS '13. IEEE Computer Society, Washington. pp 1042–1048
58. Somorovsky J, Mayer A, Schwenk J, Kampmann M, Jensen M (2012) On breaking SAML: be whoever you want to be. In: Proceedings of the 21st USENIX Conference on Security Symposium. Security'12. USENIX Association, Berkeley. pp 21–21
59. Dimkov T, van Cleeff A, Pieters W, Hartel P (2010) Two methodologies for physical penetration testing using social engineering. In: Proceedings of the 26th Annual Computer Security Applications Conference. ACSAC '10. ACM, New York. pp 399–408. doi:10.1145/1920261.1920319
60. Stepien B, Peyton L, Xiong P (2012) Using TTCN-3 as a modeling language for web penetration testing. In: IEEE International Conference on Industrial Technology (ICIT). IEEE. pp 674–681. doi:10.1109/ICIT.2012.6210016
61. Caselli M, Kargl F (2016) A security assessment methodology for critical infrastructures(Panayiotou CG, Ellinas G, Kyriakides E, Polycarpou MM, eds.). Springer, Cham
62. Line MB, Jaatun MG, Cheah ZB, Faruk ABMO, Garnes HH, Wedum P (2008) Penetration testing of OPC as part of process control systems. In: Ubiquitous Intelligence and Computing: 5th International Conference, UIC 2008, Oslo, Norway, June 23-25, 2008 Proceedings. Springer, Berlin. pp 271–283
63. Dahl OM, Wolthusen SD (2006) Modeling and execution of complex attack scenarios using interval timed colored petri nets. In: Proceedings of the Fourth IEEE International Workshop on Information Assurance. IWIA '06. IEEE Computer Society, Washington. pp 157–168
64. Khoury N, Zavarsky P, Lindskog D, Ruhl R (2011) Testing and assessing web vulnerability scanners for persistent SQL injection attacks. In: Proceedings of the First International Workshop on Security and Privacy Preserving in e-Societies. SeceS '11. ACM, New York. pp 12–18
65. Williams GP (2012) Cost effective assessment of the infrastructure security posture. In: 7th IET International Conference on System Safety, incorporating the Cyber Security Conference. IET. pp 1–6
66. Hertzog P (2010) OSSTMM—Open Source Security Testing Methodology Manual. Institute for Security and Open Methodologies (ISECOM), Barcelona. http://www.isecom.org/osstmm
67. ISSAF (2006) Information Systems Security Assessment Framework Open Information Systems Security Group. OISSG
68. PTES (2012) Penetration testing execution standard. http://www.pentest-standard.org
69. Stouffer K, Falco J, Scarfone K (2008) NIST SP 800-115: technical guide to information security testing and assessment. National Institute of Standards and Technology, Maryland
70. Meucci M, Muller A (2014) OWASP testing guide V.4. 4th edn. OWASP Foundation, USA
71. (2005) An annotated review of past papers on attack graphs, ESC-TR-2005-054. Massachusetts Institute of Technology - Lincoln Laboratory
72. Avgerinos T, Cha SK, Rebert A, Schwartz EJ, Woo M, Brumley D (2014) Automatic exploit generation. Commun ACM 57(2):74–84
73. Hossen K, Groz R, Oriat C, Richier JL (2013) Automatic generation of test drivers for model inference of web applications. In: Softw Testing Verification Validation Workshop IEEE Int Conf. pp 441–444. doi:10.1109/ICSTW.2013.57
74. Felderer M, Schieferdecker I (2014) A taxonomy of risk-based testing. Int J Softw Tools Technol Transfer 16(5):559–568
75. Botella J, Legeard B, Peureux F, Vernotte A (2014) Risk-based vulnerability testing using security test patterns(Margaria T, Steffen B, eds.). Springer, Berlin
76. Doupé A, Cavedon L, Kruegel C, Vigna G (2012) Enemy of the state: a state-aware black-box web vulnerability scanner. In: Proceedings of the 21st USENIX Conference on Security Symposium. Security'12. USENIX Association, Berkeley. pp 26–26
77. Bouquet F, Peureux F, Ambert F (2014) Model-based testing for functional and security test generation(Aldini A, Lopez J, Martinelli F, eds.). Springer, Cham
78. Duchene F, Rawat S, Richier JL, Groz R (2014) Kameleonfuzz: evolutionary fuzzing for black-box XSS detection. In: Proceedings of the 4th ACM Conference on Data and Application Security and Privacy. CODASPY '14. ACM, New York,. pp 37–48
79. Godefroid P, Levin MY, Molnar D (2012) Sage: whitebox fuzzing for security testing. Queue 10(1):20–202027
80. McAllister S, Kirda E, Kruegel C (2008) Leveraging user interactions for in-depth testing of web applications(Lippmann R, Kirda E, Trachtenberg A, eds.). Springer, Berlin
81. Kals S, Kirda E, Kruegel C, Jovanovic N (2006) Secubat: a web vulnerability scanner. In: Proceedings of the 15th International Conference on World Wide Web. WWW '06. ACM, New York,. pp 247–256
82. Huang YW, Huang SK, Lin TP, Tsai CH (2003) Web application security assessment by fault injection and behavior monitoring. In: Proceedings of the 12th International Conference on World Wide Web. WWW '03. ACM, New York. pp 148–159
83. Sekar R (2009) An efficient black-box technique for defeating web application attacks. In: Network and Distributed System Security Symposium (NDSS). The Internet Society, Geneva
84. Milenkoski A, Payne BD, Antunes N, Vieira M, Kounev S, Avritzer A, Luft M (2015) Evaluation of intrusion detection systems in virtualized environments using attack injection(Bos H, Monrose F, Blanc G, eds.). Springer, Cham
85. Bertoglio DD, Zorzo AF (2016) Tramonto: Uma estratégia de recomendações para testes de penetração. In: XVI Simpósio Brasileiro de 1315 Segurançã da Informação e Sistemas Computacionais (SBSeg 2016). SBC, Porto Alegre