UDC 004.49 **doi:** https://doi.org/10.20998/2522-9052.2021.2.20

Zhang Liqiang[1], Cao Weiling[1], Jan Rabčan[2], Viacheslav Davydov[3], Nataliia Miroshnichenko[3]

[1] Neijiang Normal University, Neijiang, China
[2] University of Žilina, Žilina, Slovakia
[3] National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine

ANALYSIS AND COMPARATIVE STUDIES OF SOFTWARE PENETRATION TESTING METHODS

Abstract. Information security is one of the most important components in any organization. The disclosure of this information can lead not only to material losses, but also to the loss of the reputation and image of the company, which ultimately, in some cases, can lead to its complete collapse. Therefore, in order to avoid these consequences, it is necessary to analyze the security and reliability of information processing systems. One of the most effective ways to do this is through the use of "penetration testing" methods. **The results obtained.** The section provides software vulnerabilities analysis. The most frequently used types of attacks and intrusions by cyber intruders are highlighted. In contrast to this, methods comparative analysis for identifying software vulnerabilities was carried out. It is concluded that it is advisable to improve the methods for identifying vulnerabilities through the recommendations complex use taking into account the existing security risks of software tools, the features of modern methodologies and software development tools, as well as the modern software penetration testing methods capabilities.

Keywords: information security; testing methods; vulnerable software; security testing.

Introduction

Currently, information security is one of the most important components in any organization, since the information processed in their information systems, to a greater or lesser extent, belongs to the categories of commercial secrets and personal data. The disclosure of this information can lead not only to material losses, but also to the loss of the reputation and image of the company, which ultimately, in some cases, can lead to its complete collapse. Therefore, in order to avoid these consequences, it is necessary to analyze the security and reliability of information processing systems. One of the most effective ways to do this is through the use of "penetration testing" methods. The term "penetration testing" means imitation of the actions of a real attacker to implement unauthorized entry into the information system [3, 7].

1. Software Vulnerabilities Analysis

Currently, modern digital computing and communication tools cannot be imagined without appropriate software. Moreover, the level of its quality is largely determined by the presence (or absence) of critical errors (bugs). This objectively existing factor cannot be ignored when assessing software vulnerabilities. Vulnerability can be the result of errors made at various stages of software development (design, coding, etc.), of outdated cryptographic systems and authentication systems use, disregard for the rules and algorithms of secure programming, etc. Analysis of different levels standards and recommendations for information security showed that the term *vulnerability* is used to denote a flaw in the system, using which you can violate its integrity or cause incorrect operation.

Studies carried out have shown that some vulnerabilities are known only theoretically, while others are actively used and have known exploits. Free software security study by Edgescan, Coverity, OWASP fund, which analyzed the security of more than 1000 projects containing more than 150 million lines of code, showed that they contain more than 60 thousand vulnerabilities [1]. And the most common types of vulnerabilities can be identified (Table 1)

As you can see in Table 1, Microsoft software products and various Web browsers have been the most vulnerable types of software for a number of years. Edgescan Vulnerability Stats Report 2020 shows vulnerabilities exist in all popular Web browsers.

The report gives a security analysis for such popular Web browsers as Mozilla Firefox, Opera and Chrome. For example, the statistics of Chrome vulnerabilities is shown in Fig. 1.

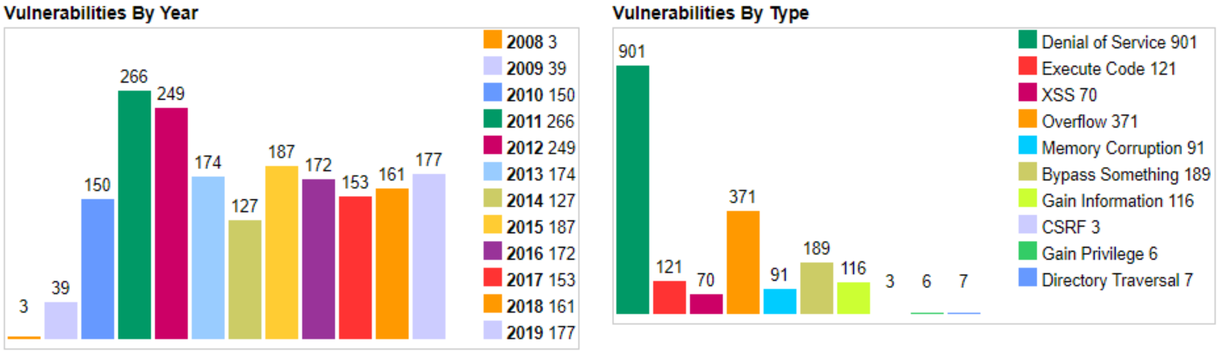

Fig. 1. Chrome Web Browser vulnerability statistics

Table 1 — **The most common types of vulnerabilities**

Vulnerability type	**The percentage of identified vulnerabilities**			**Degree of criticality**
	2018	2019	2020	
Vulnerabilities in software of private firms and non-governmental organizations				
BLUEKEEP CVE-2019-0708	15%	18%	17%	High
UNSUPPORTED SQL SERVER	14%	17%	17%	Middle
SQL INJECTION (WEB APPLICATION ATTACK)	17%	12%	11%	Middle
MS Office Memory Corruption Vulnerability CVE-2017-11882	-	8%	10%	Middle
RDP, MS12-020/, CVE-2012-0002	6%	7%	7%	High
SMB, MS17-010/, CVE-2017-0143 TO, CVE-2017-0148	6%	6%	6%	High
OTHER	42%	32%	32%	
Vulnerabilities in software of state institutions				
SQL INJECTION	41%	42%	42%	Middle
CROSS-SITE SCRIPTING (XSS)	20%	19%	20%	Middle
PHP MULTIPLE VULNERABILITIES	14%	16%	17%	High
REMOTE CODE EXECUTION	5%	7%	8%	High
SENSITIVE FILE DISCLOSURE	3%	5%	5%	High
OTHER	17%	11%	8%	

The studies carried out have shown that the most common vulnerabilities are the following:

Overriding a null pointer allows to execute a code outside the vulnerable software.

Resource leak. If after the program terminates or memory releases, the released resources are not cleared, then these memory areas may still contain the values of variables and other confidential information.

Dead code. The presence of unused sections of a code in the program allows an attacker to inject software bookmarks into these sections of the code and then use them to get unauthorized access to protected resources.

Using values before validation. If the software processes information coming from external sources without verification, then it is possible to generate such a variable value that will allow you to get full control over the vulnerable software.

Access to uninitialized variables. This vulnerability is similar in nature to the "Dead Code" vulnerability, when unused parts of software are used by an attacker to get a full control over a program.

Using an object after release allows to access the restricted information. *Buffer overflow* is a vulnerability that occurs when a computer program writes down data outside a buffer allocated in memory. Buffer overflow usually occurs due to improper handling of data received from outside and memory, without tight protection from the programming subsystem (compiler or interpreter) and the operating system. It should be noted that the mentioned above statistics and the list of vulnerabilities are not the exhaustive data. Unfortunately, the level of motivational components of cyber intruders very often exceeds the capabilities of individual IT companies which resist cyber-attacks. However, a number of methods for identifying software vulnerabilities have been developed and are currently being used. Let's conduct out their comparative analysis.

2. Comparative analysis of methods for identifying vulnerabilities

The studies carried out have shown that currently, to improve the efficiency of identifying software vulnerabilities, specific testing methods and recommendations are used. The goals, methods, algorithms and means of verification, rules for conducting and verification are indicated in them. The following ones are relevant for Ukraine: [1-8]:

– OSSTMM (Open Source Security Testing Methodology Manual);

– OWASP (Open Web Application Security Project) Testing Guide;

– PTES (Penetration Testing Execution Standard);

– NIST Special Publication 800-115: Technical Guide to Information Security Testing and Assessment (NIST SP 800-115);

– BSI – Study A Penetration Testing Model;

– ISSAF — Information System Security Assessment Framework;

– THE NATIONAL BANK OF UKRAINE BOARD RESOLUTION of 28.09.2017 № 95 «On Approval the Provision about Organization of Measures on Ensure Information Security in the Banking System of Ukraine».

Each of these documents has its own characteristics (advantages and disadvantages). The research is carried out in several stages (Fig. 2), which allow to structure knowledge and comprehensively assess the security of software. In general, the results of comparing methodologies by phases of vulnerability tests can be illustrated with Fig. 3.

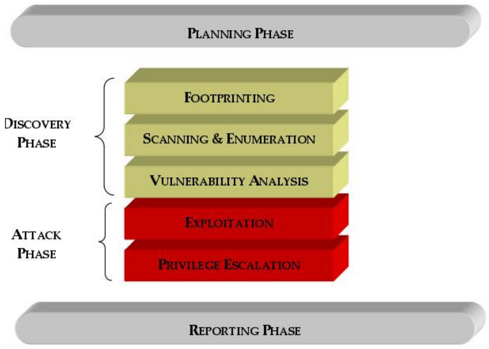

Fig. 2. Software security testing stages

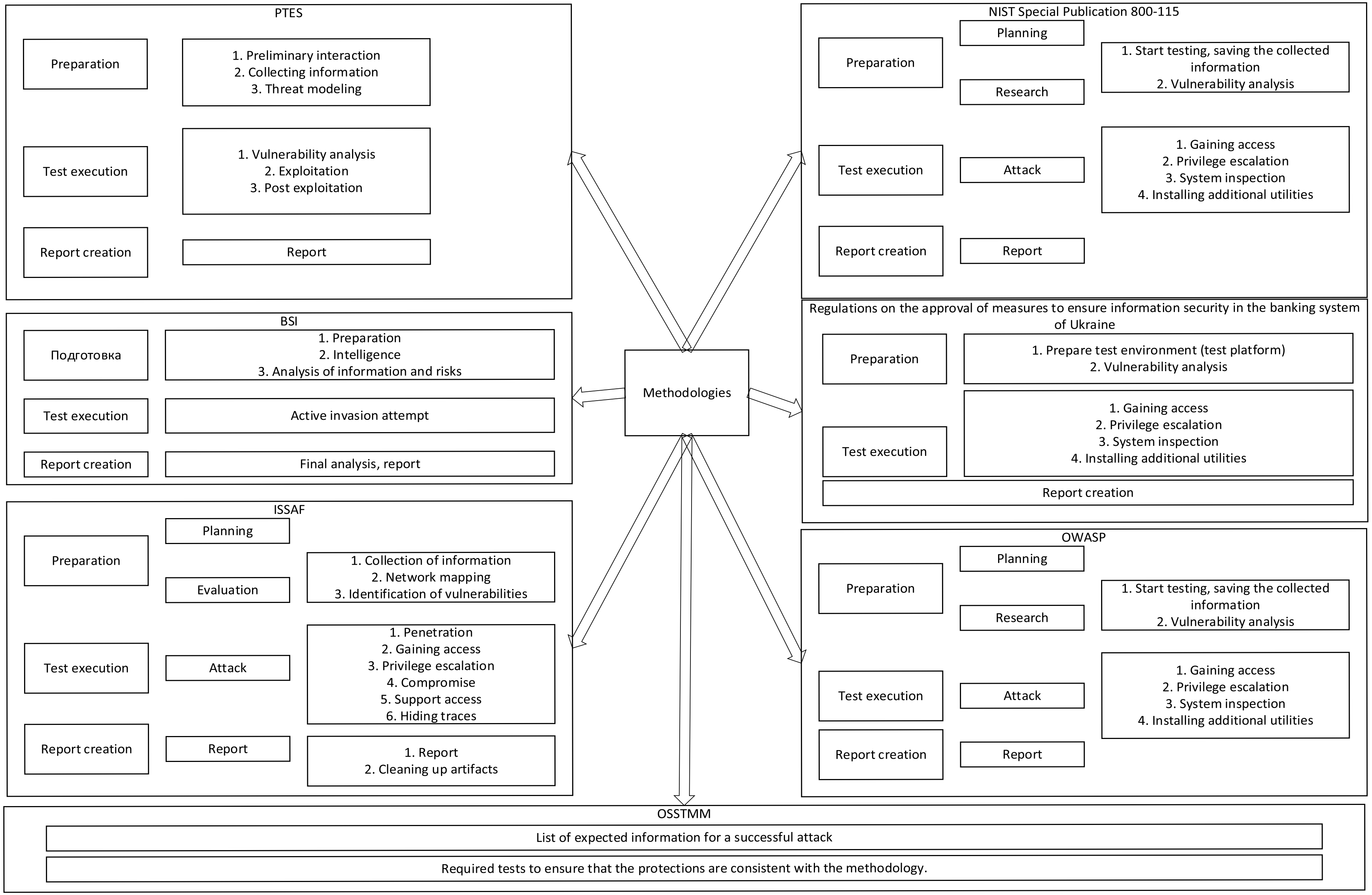

Fig. 3. Classification of methodologies by phases of vulnerability tests

Table 2 contains a comparison of the methods being studied, where the objects of research are divided into 3 standard phases:

– Preparation,

– Execution of tests

– Generation of a report.

Table 3 contains a comparison of methods according to the specified criteria using a similar system of point grades from 0 to 10.

The presented results of the methodologies assessment showed the imperfection of any of them. For example, the best methodology in the test execution and reporting phases – OWASP does not meet the requirements of the experts in the preparation phase. Vice versa, the OSSTMM methodology, which was highly appreciated by experts at the first stage of preparing the pre-test material, has a low rating at the later stages of software security research.

Table 2 – **Comparison of the investigated methodologies according to three standard phases**

Methodology, Phases	**OWASP**	**OSSTMM**	**NIST SP**	**BSI**	**ISSAF**	**PTES**	**Resolution of the NBU Board № 95**
Preparation							
Customer approval of testing modes	0	7	1	0	5	7	7
Execution and signing of the contract	0	7	1	0	5	7	8
Tests execution							
Collecting information about the object	8	1	4	8	8	7	0
Identification of vulnerabilities	8	1	3	8	8	8	1
Analysis of information and risks	8	1	2	8	8	8	2
Active invasion attempts	8	1	5	8	8	8	0
Enabling the following intrusion	8	0	0	0	8	8	0
Report creation							
Artifact cleaning	5	1	2	4	5	8	0
Report creation	5	7	4	8	7	9	2
Analysis and recommendations for found vulnerabilities elimination	10	2	4	8	4	9	1
Description of risks	10	1	3	8	4	9	2

Table 3 – **Comparison of methodologies by criteria**

Methodology, Phases	**OWASP**	**OSSTMM**	**NIST SP**	**BSI**	**ISSAF**	**PTES**	**Resolution of the NBU Board № 95**
Description of the information a cracker can obtain	5	8	1	0	2	0	2
Description of the penetration testing goals	10	4	5	10	1	5	3
Detailed description of the methodology	10	4	9	7	6	10	0

Based on this, it can be concluded that it is advisable to improve the methods for identifying vulnerabilities through the complex use of recommendations taking into account the existing security risks of software tools, the features of modern methodologies and software development tools and the capabilities of modern software penetration testing methods.

Conclusions

The section analyzes the software vulnerabilities. The priority of software security requirements and the obligation to follow these requirements at all stages of the software life cycle are shown. Research and comparative analysis of methods for identifying vulnerabilities have been carried out; lack of attention from developers to security issues has been indicated.

The expediency of improving the existing methods of software penetration testing by synthesizing a new software testing method taking into account increased security requirements is indicated.

Acknowledgment

The Slovak Research and Development Agency (Agentúra na Podporu Výskumu a Vývoja) supported this work under the contract no. APVV-18-0027 titled “New methods development for reliability analysis of complex system”.

References

1. (2020), *Edgescan’s 2020 Vulnerability Stats Report Released*, available at: https://www.edgescan.com/edgescans-2020-vulnerability-stats-report-released/
2. Kostadinov, Dimitar (2016), Introduction: Intelligence Gathering & Its Relationship to the Penetration Testing Process available at: https://resources.infosecinstitute.com/penetration-testing-intelligence-gathering/
3. Nickerson, C. (2012), *The Penetration Testing Execution Standard*, available at: http://www.pentest-standard.org/index.php/PTES_Technical_Guidelines
4. Scarfonem K., Souppayam M., Codym A. and Orebaugh, A. (2012), *NIST Special Publications 800-115 Technical Guide to Information Security Testing and Assessment*, USA, Gaithersburg, 80 p., available at: http://csrc.nist.gov/publications/nistpubs/800-41-Rev1/sp800-41-rev1.pdf
5. (2012), *Study A Penetration Tesing Model*, Germany, Bonn, 111 p., available at: https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/Studies/Penetration/penetration_pdf.pdf?__blob=publicationFile

6. (2018), *The Open Source Security Testing Methodology Manual*, available at: http://www.isecom.org/mirror/OSSTMM.3.pdf.
7. Vacca, John R. (2017), *Computer and Information Security Handbook Elsevier*, 1280 р.
8. (2018), *XPathinjection*, available at: URL:https://portswigger.net/kb/issues/00100600_xpath-injection.

Надійшла (received) 21.01.2021
Прийнята до друку (accepted for publication) 07.04.2021

ВІДОМОСТІ ПРО АВТОРІВ / ABOUT THE AUTHORS

Ліцзян Джан – викладач коледжу комп'ютерних наук, Типовий університет Нейцзяна, Нейцзян, Кітай;
Zhang Liqiang – teacher, College of Computer Science, Neijiang Normal University, Neijiang, China.
e-mail: zhangiq@njtc.edu.cn; ORCID ID: https://orcid.org/0000- 0003-1278-2209.

Цао Вейлін – викладач інформаційного центру ІТ, Типовий університет Нейцзяна, Нейцзян, Кітай;
Cao Weilin – teacher, Department of IT information Centre, Neijiang Normal University, Neijiang, China.
e-mail: caowl@njtc.edu.cn; ORCID ID: https://orcid.org/0000-0001-8230-5235.

Рабчан Ян – PhD, факультет управлінських наук та інформатики, Жилінський університет, Жиліна, Словаччина;
Jan Rabčan – PhD, Faculty of Management Science and Informatics, University of Žilina, Žilina, Slovakia;
e-mail: Jan Rabcan@fri.uniza.sk; ORCID ID: https://orcid.org/0000- 0003-2835-9114.

Давидов Вячеслав Вадимович – кандидат технічних наук, доцент, доцент кафедри "Обчислювальна техніка та програмування", Національний технічний університет "Харківський політехнічний інститут", Харків, Україна;
Viacheslav Davydov – Candidate of Technical Sciences, Associate Professor , Associate Professor of Computer Engineering and Programming Department, National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine;
e-mail: vyacheslav.v.davydov@gmail.com; ORCID ID: https://orcid.org/0000-0002-2976-8422.

Мірошніченко Наталія Миколаївна – кандидат технічних наук, доцент кафедри "Обчислювальна техніка та програмування", Національний технічний університет "Харківський політехнічний інститут", Харків, Україна;
Nataliia Miroshnichenko – Candidate of Technical Sciences, Associate Professor of Computer Engineering and Programming Department, National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine;
e-mail: natnikdr@gmail.com; ORCID ID: https://orcid.org/0000 0003-4329-7126.

Аналіз і порівняльне дослідження методів тестування програмного забезпечення на проникнення

Чжан Ліцян, Цао Вейлін, Я. Рабчан, В. В. Давидов, Н. М. Мірошніченко

Анотація. Інформаційна безпека є одним з найважливіших компонентів в будь-якій організації, оскільки інформація, що обробляється в їх інформаційних системах, в більшій чи меншій мірі відноситься до категорій комерційної таємниці і персональних даних. Розкриття цієї інформації може призвести не тільки до матеріальних втрат, а й до втрати репутації та іміджу компанії, що в кінцевому підсумку в деяких випадках може привести до її повного краху. Тому, щоб уникнути цих наслідків, необхідно проаналізувати безпеку і надійність систем обробки інформації. Один з найбільш ефективних способів зробити це - використовувати методи «тестування на проникнення». Отримані результати. У розділі проведено аналіз вразливостей програмного забезпечення. Виділено найбільш часто використовувані кіберзлоумишленнікамі види атак і вторгнень. На противагу цьому проведено порівняльний аналіз методик виявлення вразливостей ПЗ. Зроблено висновок про доцільність вдосконалення методик виявлення вразливостей шляхом комплексного використання рекомендацій з урахуванням існуючих ризиків безпеки програмних засобів, особливостей сучасних методологій і засобів розробки ПО, а також можливостей сучасних методик тестування ПО на проникнення. З метою аргументованого вибору технологій математичної формалізації процесу тестування проведені аналіз і порівняльне дослідження найбільш перспективних з них.

Ключові слова: інформаційна безпека; методи тестування; вразливе програмне забезпечення; тестування безпеки.

Анализ и сравнительное исследование методов тестирования программного обеспечения на проникновение

Чжан Лицян, Цао Вейлин, Я. Рабчан, В. В. Давыдов, Н. Н. Мирошниченко

Аннотация. Информационная безопасность является одним из важнейших компонентов в любой организации, поскольку информация, обрабатываемая в их информационных системах, в большей или меньшей степени относится к категориям коммерческой тайны и персональных данных. Раскрытие этой информации может привести не только к материальным потерям, но и к потере репутации и имиджа компании, что в конечном итоге в некоторых случаях может привести к ее полному краху. Поэтому, чтобы избежать этих последствий, необходимо проанализировать безопасность и надежность систем обработки информации. Один из наиболее эффективных способов сделать это - использовать методы «тестирования на проникновение». **Полученные результаты.** В разделе проведен анализ уязвимостей программного обеспечения. Выделены наиболее часто используемые киберзлоумышленниками виды атак и вторжений. В противовес этому проведен сравнительный анализ методик выявления уязвимостей ПО. Сделан вывод о целесообразности усовершенствования методик выявления уязвимостей путем комплексного использования рекомендаций с учетом существующих рисков безопасности программных средств, особенностей современных методологий и средств разработки ПО, а также возможностей современных методик тестирования ПО на проникновение. С целью аргументированного выбора технологий математической формализации процесса тестирования проведены анализ и сравнительное исследование наиболее перспективных из них.

Ключевые слова: информационная безопасность; методы тестирования; уязвимое программное обеспечение; тестирование безопасности.