UDC 004. 8 **doi: 10.20998/2522-9052.2020.3.18**

Mykhailo Mozhaiev[1], Viacheslav Davydov[2], Zhang Liqiang[3]

[1] Prof. M.S. Bokarius Kharkiv Research Institute of Forensic Examinations, Kharkiv, Ukraine
[2] National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine
[3] Neijiang Normal University, Neijiang, China

ANALYSIS AND COMPARATIVE RESEARCHES OF METHODS FOR IMPROVING THE SOFTWARE

Abstract. The results analysis of main methods for identifying software vulnerabilities presents in the article. The results of authors' research, synthesizing and regulating knowledge about systems for detecting software vulnerabilities, are presented. The software analysis methods used during certification tests are considered. It is shown that the methods and techniques existing for software security analysis use do not ensure the result accuracy under fuzzy input data conditions. This drawback is aggravated by strict requirements for the test scenarios implementation speed. This is largely due to the fact that experts, in order to a decision make, have to conflicting information large amounts analyzed. Consequently, it is necessary to develop a system for identifying vulnerabilities, the main task of which will be to the conflicting information amount minimize used by an expert when making a decision. The most promising direction the existing identifying vulnerabilities systems efficiency increasing is seen in reducing the burden on an expert by methods for identifying vulnerabilities and implementing a decision support system improving. This will significantly reduce the time spent on a decision making on software security, and, as a result, will the software security testing procedure accessible to a developer's wide range make more.

Keywords: computer systems; Software; security risks; security threats.

The requirements and security risks analysis of computer systems software

The modern information space is a complex, heterogeneous structure that performs society various functions and needs. At the same time, a significant part of the automation and intellectualization functions is assumed by computer systems (CS). Malicious impacts on the CS during their operation are carried out with purposes various malicious of security services violation (deterioration). The tasks solution related to the prevention of unauthorized influences on the CS and the information that is processed and stored in them is carried out as part of comprehensive programs to improve security. At the same time, the safety CS improving problem is high relevance.

A generalized CS security model can be represented as Fig. 1, which is based on the following objects: security improving methods and means – security software increasing models, methods and means; user's interaction computerized system – hardware and software, computing, information, linguistic, communication and other resources for interacting with users, as well as the users themselves; software – is one of the most important and vulnerable computer systems components; security increasing mechanisms and means – information protecting mechanisms and means; security threats – a potential event, action, process or phenomenon that could damage the user's interaction computerized system; security risks – the potential possibility exploiting vulnerabilities CS of a specific threat to cause damage.

An existing computer systems integral component is software. And, in many respects, the operation CS quality depends on the operating software quality. As noted in the explanations, one of the most vulnerable, from the view of security point, components the CS is software. Moreover, the task of increasing (ensuring) software security the complexity is compounded by the need to take into account security risk factors throughout the entire software development life cycle.

An international standards number of analysis and regulatory documents [10, 11, 12, 14] has shown a significant increase in software quality requirements recently. This only confirms the fact the expert community increase attention to the software operation issues and tasks. The conducted studies allowed classifying software requirements and presenting them in the diagram form Fig. 2. This scheme clearly illustrates the fact that now, in addition to the ensuring tasks completeness and use ease, accessibility, reliability and other characteristics, the requirements for software security are becoming increasingly important among operators. As the developing and operating secure software importance evidence, we can

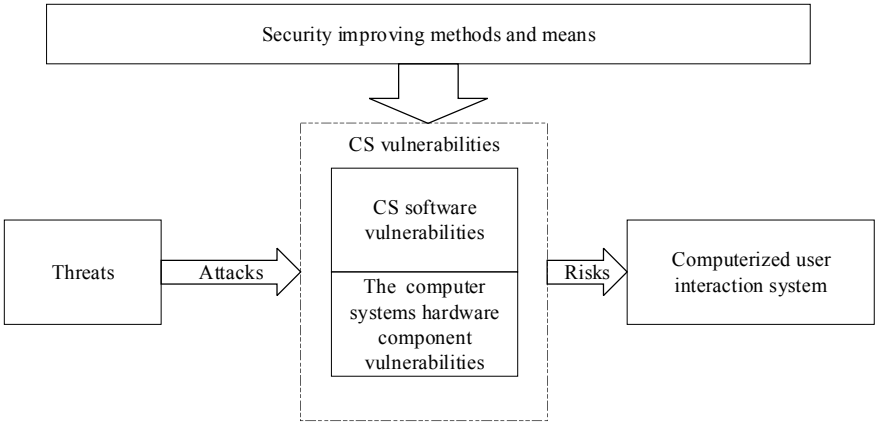

Fig. 1. Generalized computer system security model

© Mozhaiev M., Davydov V., Liqiang Zhang, 2020

note the annual growth its vulnerabilities identified. So, in Fig. 3, statistics are presented for only one software type – Web applications with risk varying degrees vulnerabilities, noted by IBM over the past two years.

Open sources information research [1, 8, 16] made it possible to present the most common vulnerabilities types. The software security threats classification is presented in Fig. 4.

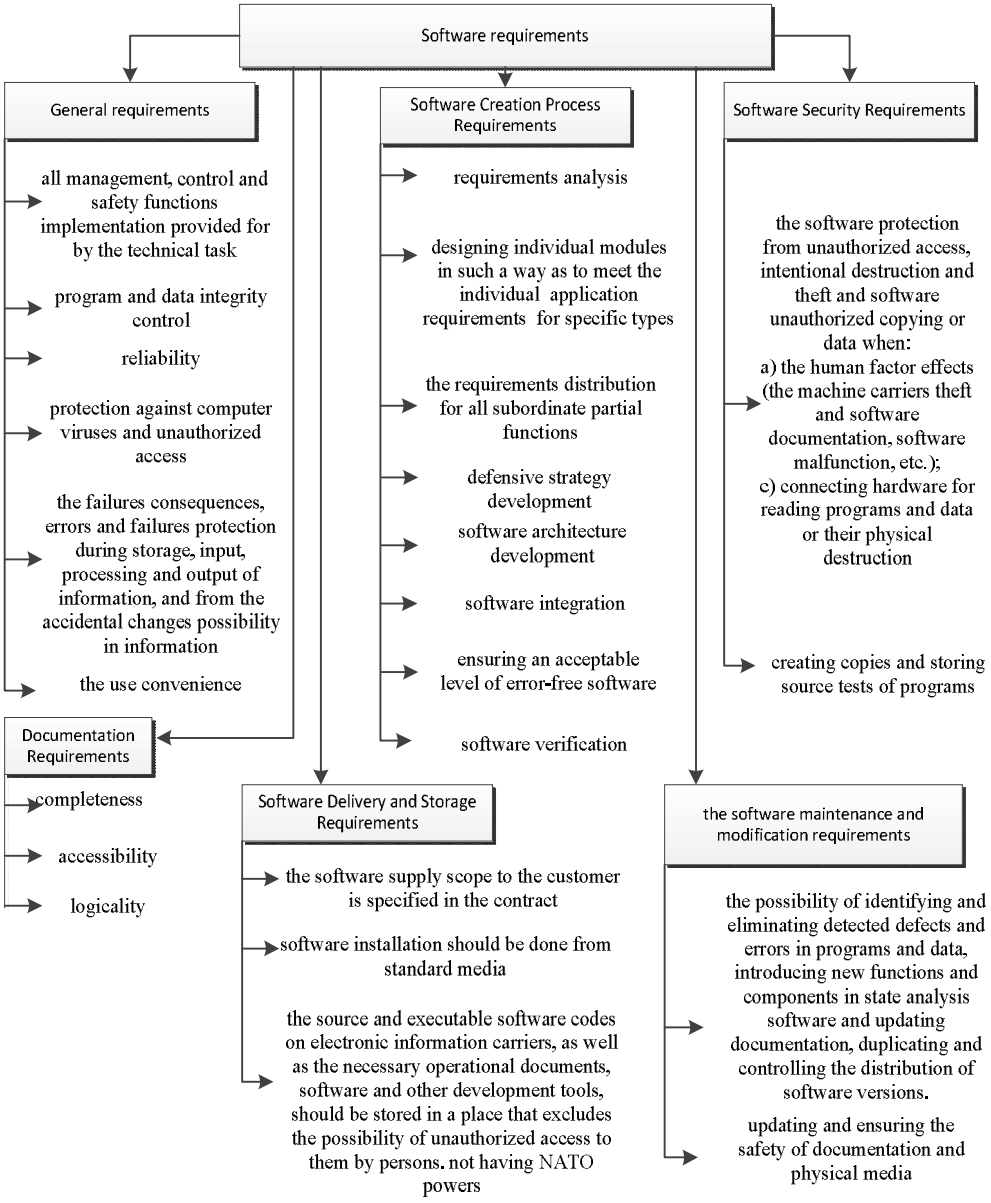

Fig. 2. Software requirements classification

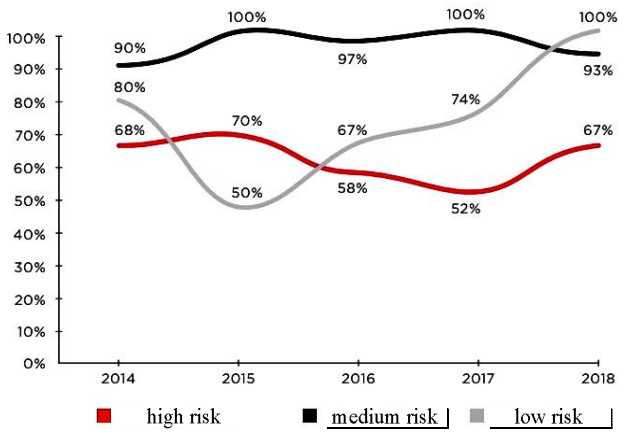

Fig. 3. The Web applications percentage with vulnerability varying degrees

The presented classification is based on an international standards number of software development issues fixing. At first, these are the standards: ISO 10007-2007 “Organization management. Guidelines for configuration management”, ISO / IEC 12207-2010 “Information technology. System and software engineering. Software life cycle processes”, German Information Security Agency. IT Baseline Protection Manual – Standard security safeguards (Guide to the information technology basic protection level), GOST 58412-2019 “Secure software development”, etc.

As can be seen from the presented classification, at present there is a wide information range security threats during software development. In addition, in recent years there has been a significant increase in the cyberattacks intensity, and hence an even greater these

threats spectrum expansion. Such a variety very often leads to ambiguity in assessing the software security level among experts and those responsible for information security issues. Taking into account an increasing number of factors and threats leads to the uncertainty factors introduction, complicating decision-making processes and reducing the results accuracy.

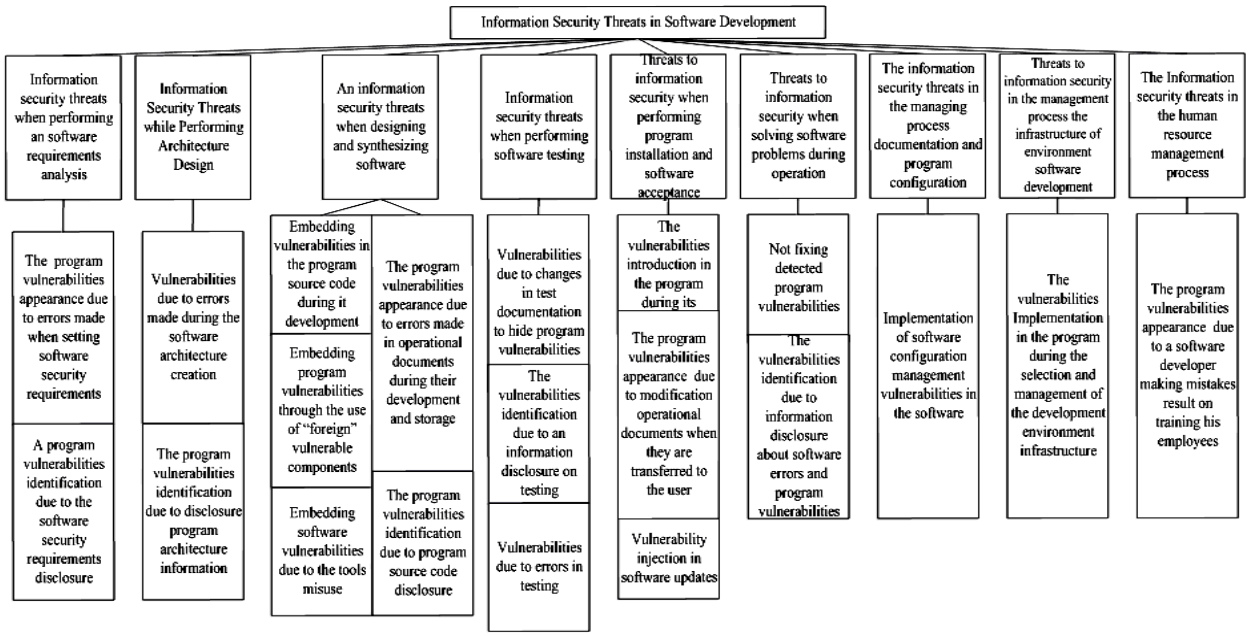

Fig. 4. The software security threats classification

Therefore, it becomes important to develop and implement models and methods for improving software security, taking into account uncertainties.

The software vulnerability detection methods analysis

The literature [1, 3] comparative review showed that software traditional methods analysis are somehow connected with the defects absence proof. Moreover, these methods can be divided into two main categories: inspection-testing and logical-linguistic.

It is known from [1] that such a classification is based on their focus on the action object. So inspection-testing methods are aimed at fixing the software security violation fact, and logical-linguistic methods are aimed at the software under study identifying deviations from indicators declared in the technical documentation.

Research has shown that there are currently many different options for classifying methods for identifying software vulnerabilities. At the same time, noting their wide range and variety, one can also indicate their action direction and identify target and auxiliary methods.

Let us present the methods classification for identifying software vulnerabilities in the form of Fig. 5.

As can be seen from this figure, most of the methods can be synthesized according to the expert; dynamic, static and combined analyzes principles. At the same time, about 45% of the methods are targeted. Let us present the comparative characteristics of the proposed methods.

Since the greatest effect in the software security testing process is provided by target methods, we will focus on these methods, evaluate them and highlight their use limitations. To highlight these methods quality characteristics using, we use the CWE (Common Weakness Enumeration) database developed and recommended by MITRE (mitre.org) Corporation and the US Department of Homeland Security (DHS) Cyber-security and Infrastructure Security Agency (CISA) [6].

We will select the characteristics recommended by MITRE from the specified base: the input / output user data processing errors (CWE - 78, 79, 89, 119, 134, 189, 352, 434); security functions errors (CWE - 21, 200, 255, 264, 287, 310); synchronization errors (CWE - 162, 399, 829, 834); the use software interfaces errors (CWE - 583, 684); environment errors (CWE - 16, 733); the error handling disadvantages (CWE - 703); encapsulation errors (CWE - 653); low code quality (CWE - 477).

The targeted methods applicability pie diagrams for identifying software vulnerabilities, as applied to the most common security threats analysis, are shown in Fig. 6.

The critical software vulnerabilities statistics analysis, in accordance with MITRE 2019 CWE Top, OWASP Top 10−2019, WASC (Web Application Security Statistics Project), made it possible to present the most dangerous software errors brief description (vulnerabilities) in Table 1.

As can be seen from the diagrams Fig. 6, the listed and researched targeted methods for identifying vulnerabilities cover most of the error spectrum. However, the problem of identifying the vulnerabilities most, at this stage, using existing methods has not been completely resolved. In addition, it should be borne in mind that most, even targeted, methods for identifying vulnerabilities do not fully solve the assigned tasks. However, some of them are applicable for detecting a limited errors number.

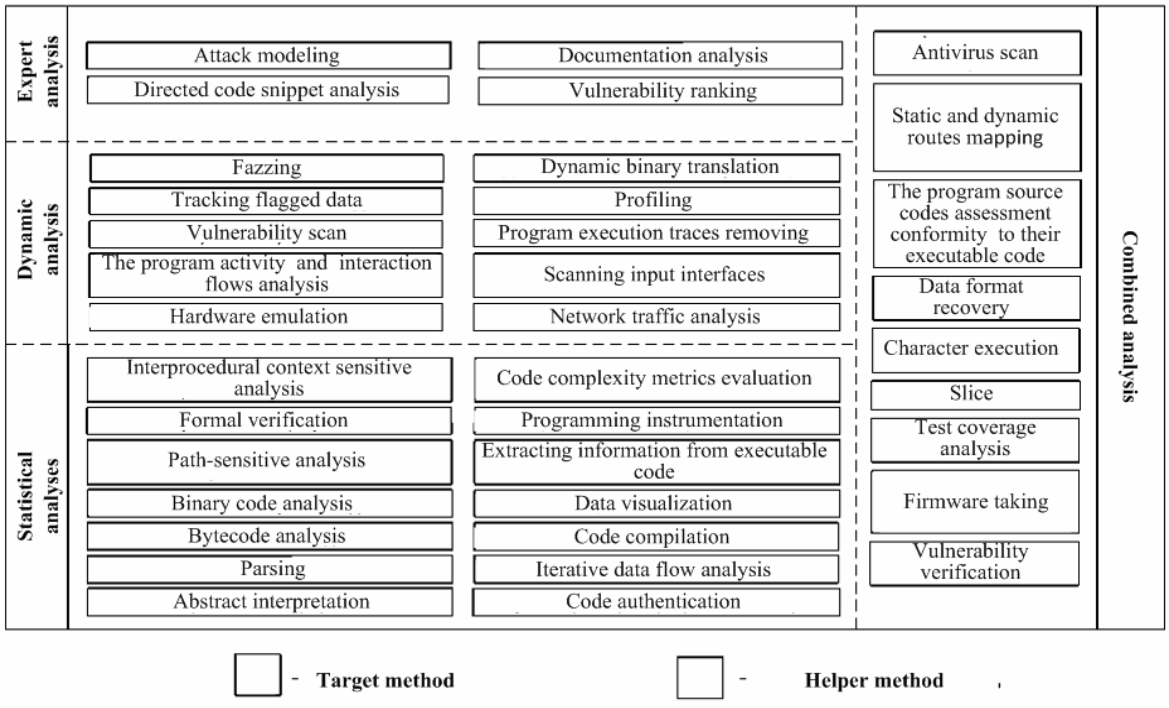

Fig. 5. Methods classification identifying software vulnerabilities

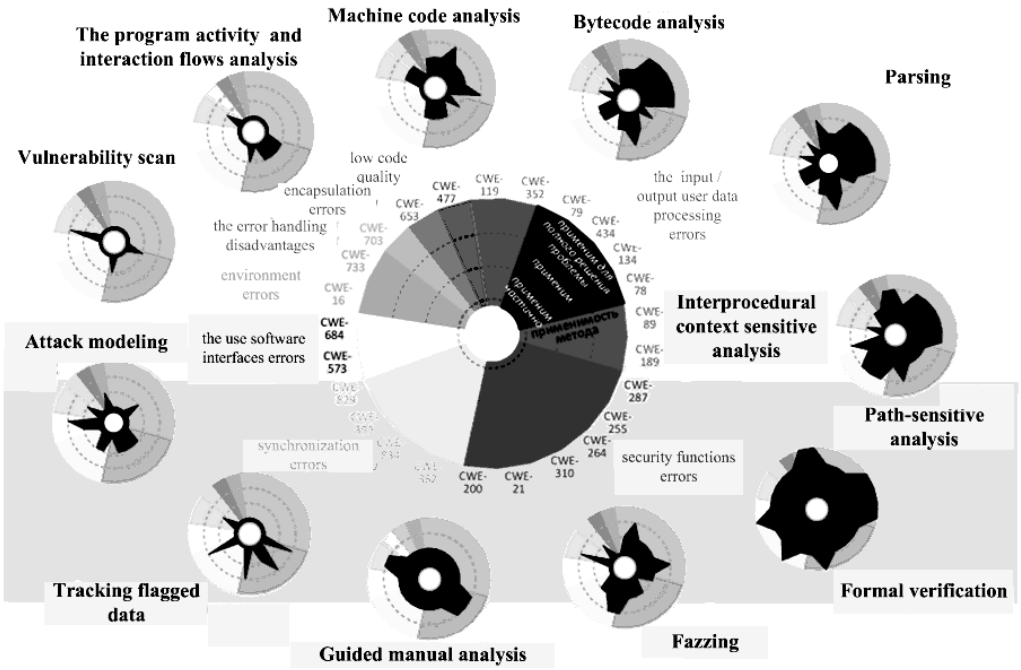

Fig. 6. The targeted methods applicability pie charts for identifying software vulnerabilities

Consequently, experts (developers) need to take into account the fact that it is advisable to use the entire possible the proposed methods spectrum. It is only necessary to determine the methodology for their most effective use in the security testing process.

This hypothesis is confirmed by the availability methods with a sufficiently high applicability degree. For example, formal verification is applicable to identify more than 70% of errors, and interprocedural context-sensitive analysis covers about 40% of software vulnerabilities.

It should be noted that very the three-dimensional scale introduction for assessing the presented methods applicability introduces a fuzziness element in the decision-making process about possible software vulnerability. This is especially true when different methods of identifying vulnerabilities show opposite, conflicting results. The advertised software errors variety, their occurring at all software development stages possibility, the ambiguity of understanding the rules and specifications by software developers (for example, CWE-684), as well as other factors, reduce the error detection accuracy, introduce ambiguity in the conditions for identifying software vulnerabilities, complicate by experts the adoption process software security solutions.

Table 1 – **The most dangerous software errors (vulnerabilities) brief description**

№	Index	Characteristics description	Software vulnerability brief description
1	CWE-119	Improper Restriction of Operations within the Bounds of a Memory Buffer	The software performs operations on a memory buffer, but it can read from or write to a memory location that is outside of the intended boundary of the buffer.
2	CWE-79	Improper Neutralization of Input During Web Page Generation	The software does not neutralize or incorrectly neutralizes user-controllable input before it is placed in output that is used as a web page that is served to other users.
3	CWE-78	Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')	The software constructs all or part of an OS command using externally-influenced input from an upstream component, but it does not neutralize or incorrectly neutralizes special elements that could modify the intended OS command when it is sent to a downstream component.
4	CWE-200	Exposure of Sensitive Information to an Unauthorized Actor	The product exposes sensitive information to an actor that is not explicitly authorized to have access to that information.
5	CWE-162	Improper Neutralization of Trailing Special Elements	The software receives input from an upstream component, but it does not neutralize or incorrectly neutralizes trailing special elements that could be interpreted in unexpected ways when they are sent to a downstream component.
6	CWE-684	Incorrect Provision of Specified Functionality	The code does not function according to its published specifications, potentially leading to incorrect usage.
7	CWE-733	Compiler Optimization Removal or Modification of Security-critical Code	The developer builds a security-critical protection mechanism into the software, but the compiler optimizes the program such that the mechanism is removed or modified.
8	CWE-703	Improper Check or Handling of Exceptional Conditions	The software does not properly anticipate or handle exceptional conditions that rarely occur during normal operation of the software.
9	CWE-653	Insufficient Compartmentalization	The product does not sufficiently compartmentalize functionality or processes that require different privilege levels, rights, or permissions.
10	CWE - 477	Use of Obsolete Function	The code uses deprecated or obsolete functions, which suggests that the code has not been actively reviewed or maintained.

There is a need to synthesize knowledge about software vulnerabilities and develop appropriate models and methods to support decision-making about software security, taking into account a fuzzy input data factors.

The models and methods analysis for software security decision support

The literature analysis [4, 5, 7] and research conducted have shown the approaches, models, methods and decision support systems (DSS) wide range.

Based on the generally accepted classification systems, three main the decision making theory approaches can be distinguished, divided according to the conditions for making decisions:

- in certainty conditions taken;
- in risk conditions taken;
- in uncertainty conditions taken.

At the same time, the conducted research has shown the alternatives priority to the DSS models and methods occurring under a risk and uncertainty conditions. It should be noted that most often, in practice, in order to improve the DSS systems quality and accuracy results, it becomes necessary to synthesize knowledge in a risks and uncertainties reducing for solving problems.

Research have shown that in the DSS systems development, taking into account the reducing risks tasks is reduced to minimizing the first and second kind errors probability that arise when making decisions.

It is known from [4] that it is customary to refer the first kind mistakes as a wrong solutions making in the current conditions, to the second kind errors - not to accept the right one. Quantitatively, these errors ratio consequences are estimated by the risk formula:

$$R = C_{12}\left(1 - P\left(H_{12}\right)\right) + C_{21}P\left(H_{12}\right) \to \min, \quad (1)$$

where C_{12} – the expected "useful value" in the actions taken result; $P(H_{12})$ – the probability that the actions strategy is chosen correctly and the event will be completed with a positive result; C_{21} – losses arising as a breakdown result or unreasonable decision to act; $P(H_{21})$ – the probability of wrong choice strategy, leading to a negative result.

In the works [7, 13] the risk reduction approach when testing the computer systems security was considered. The ROC-analysis apparatus was taken as the basis for solving the problem. Using the data and research results, the authors carried out a comparative analysis of the developed methods for identifying anomalies and abuses of critical computer systems. The hypothesis was confirmed that the methods for identifying anomalies, which were based on the fuzzy mathematics theory rules (fuzzy discriminant and cluster analysis and a fuzzy expert system based on the Bayesian classifier), satisfy the established quality criteria. At the same time, the computer systems software component architectures and semantics development rapid pace requires an increase in the using efficiency this well-known mathematical apparatus.

It should be noted that the certain methods possibility risk reduction is determined by the fact that their occurrence nature is in the same area in which the information uncertainty arises. Risks are determined both by making decisions based on incomplete initial data and by the confrontational systems actions algorithms ignorance. Accordingly, the risk reduction calculation methods can be considered from a reducing uncertainty methods set. The DSS systems synthesizing

task in this situation is reduced to the specific methods choice for reducing uncertainty, their combination or integrated use. The reducing uncertainties methods classification is shown in Fig. 7. As you can see from the figure, the uncertainties reducing methods can be conditionally divided into two large groups: based on the starter data modifying methods and algorithmic. The conducted research has shown that the initial data modifying methods choice is implemented taking into account the requirements for the data processing quality (for example, taking into account the requirements for the problem solving promptness).

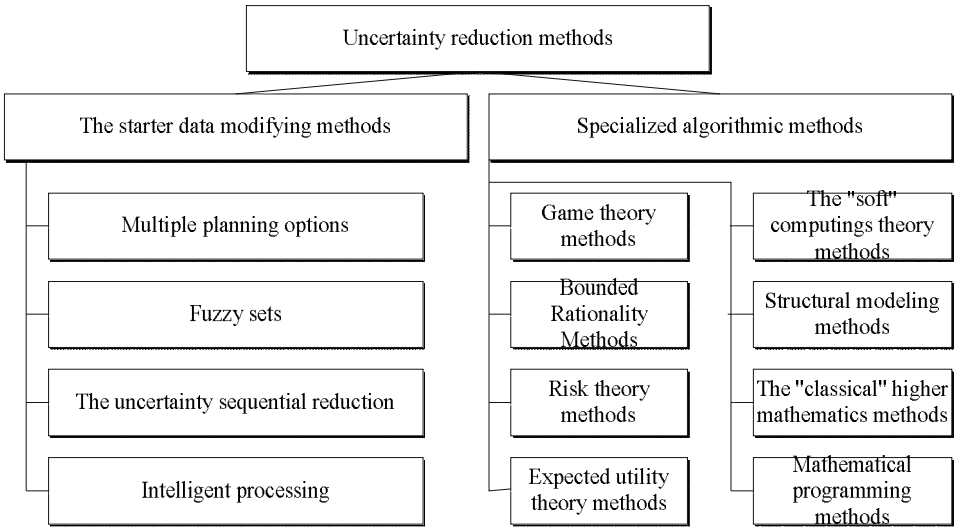

Fig. 7. The reducing uncertainties methods classification

For example, multivariate planning methods are very popular in the financial and economic sector. And some works in this industry [5] emphasize their use appropriateness in solving optimization problems of allocating resources and forces. However, the significant time spent on their implementation, as well as the taking impossibility into account the possible alternatives entire range, reduce their capabilities when solving problems in increasing software security. However, the significant time spent on their implementation, as well as the taking impossibility into account the possible alternatives entire range, reduce their capabilities when solving problems in increasing software security.

The uncertainty successive reduction methods have a similar disadvantage, although such the methods accuracy, as seen from the sources [18], is quite high.

The intellectual processing methods, despite their diversity and positive reviews [15, 18], ultimately most often come down to separate structural subtasks of the overall task consistently reducing uncertainty. Based on this, we can conclude about the relatively these methods low efficiency.

At the same time, the fuzzy sets theory at this stage has significant prospects for its development. So in the works [4, 7] high obtaining accuracy and efficiency the resulting data processed using the fuzzy sets theory the methods is noted.

With an algorithmic approach to reducing uncertainty, the methods choice is most often limited to the decision theory functionality. Each of these methods has a advantages and disadvantages set.

So in the works [9, 15], it is noted on the one hand the solving multi-alternative problems possibility of the game theory mathematical apparatus (namely, in this problem formulation we most often encounter when modeling the software security testing process), and on the other hand, the uncertainty in the indicators and optimization criteria. Also, high computational complexity significantly limits the implementing practical this modeling mechanism possibilities.

And, for example, in the works [2] devoted to the expected utility theory, it is noted that this theory main provisions application to reduce the uncertainty, the source of which is the information asymmetry, contributed to the agency relations normative theory formation. But it is this, according to the authors that often become a factor in reducing the resulting data accuracy. A shift in the research vector only towards empirical knowledge reduces other theoretical approaches value, which ultimately leads to a decrease in the modeling results quality.

Largely, the bounded rationality methods [4] have similar disadvantages, which recommend minimizing the expected result, based on the person's (expert's) socio-psychological characteristics making the decision.

The literature review conducted [4, 15, 17] led to the works conclusion about a large number devoted to the mathematical programming methods. So, to the mathematical programming problems solve, more than two dozen mathematical methods are used: from the simplest "northwest corner" to computationally complex gradient methods for finding an extremum. Taking this into account, the DSS systems efficiency improving problem using this mathematical apparatus is not in the new mathematical programming methods development, but rather in the complexity of formalizing the solution conditions and choosing methods for them that most adequately describe the controlled system functioning.

Research have shown that the structural modeling methodology, "classical" higher mathematics, the "soft" computing's theory for a classes of problems and

models wide range solving are described in sufficient detail in the scientists works. So, for example, it is described in the works devoted to the DSS systems software creation [7, 8], the calculations general mathematical methods development in the decision support interests [4], and mathematical modeling interests [17, 18]. Despite the rather long history and these methods variety, their improvement and use will allow one to fulfill a particular scientific tasks number of increasing software security. So, for example, structural modeling methods will allow formalizing the DSS process, and the methods of the "soft" computing theory and "classical" higher mathematics will the basis of the developed method become for software security increasing.

Thus, the forming alternatives problem in DSS systems about software safety in a priori uncertainty conditions and the resulting risks can be solved by improving mathematical methods [7, 15, 17, 18].

Research problem statement

The research conducted on existing systems for identifying software vulnerabilities have shown that they have a disadvantages number associated with the complexity of processing a input data large amount with a high contradiction degree. In addition, the lack of improved decision support mechanisms reduces the expert group effectiveness. Confirming this hypothesis, we present a generalized model for identifying vulnerabilities in Fig. 8.

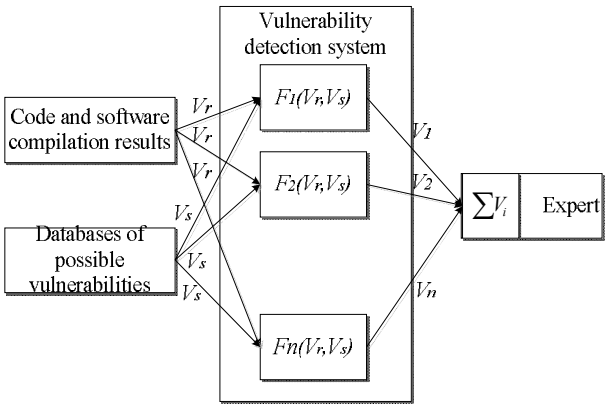

Fig. 8. Generalized vulnerability identification model

As can be seen from the figure, the information amount obtained during software testing V_Σ the expert's decision-making process significantly complicated makes, the time for implementing test scenarios increases, and the software security procedures effectiveness reduces. V_Σ is defined as the software security tests results sum carried out by various methods.

$$V_\Sigma = \sum_{i=1}^{n} V_i , \qquad (2)$$

where i – the method number; n – the test cases number.

The value V_i depends on the source code volume and software compilation results V_r, the possible vulnerabilities database volume V_s, as well as the algorithm that implements the check $V_i = F_i(V_r, V_s)$.

The time t_{test} required to implement test scenarios for analyzing software for vulnerabilities is determined similarly to the value V_i. The value t_{test} depends on the vulnerability detection system operating time $t_{сбу}$, as well as the time $t_э$ required for an expert to make a decision.

The time $t_{сбу}$ is determined by the maximum time values spent on searching for vulnerabilities, by each of the methods implemented in the system for identifying vulnerabilities.

Assuming that software security tests start running at the same time $t_{без}$, the total time to determine software security is:

$$t_{без} = \max\left(f_1(V_r, V_s) \ldots f_n(V_r, V_s)\right) + t_э . \qquad (3)$$

The values V_i, based on the problem formulation, should be free from uncertainty signs. However, the resulting data that determine software security decisions due to the data possible polarization may again become fuzzy.

Proceeding from this, increasing the software security testing procedures efficiency, as well as the making decisions efficiency about security, lies in the optimization problem plane solving of minimizing the fuzzy sets fuzziness index.

The fuzzy sets fuzziness index, on the one hand, directly depends on the fuzzy set power, that is, the analyzed data volume V_Σ, and on the other hand, it depends on the conflicting data set power $V_\Sigma^{(\text{пр})}$. Accordingly, a particular optimization problem can be transformed into the result:

$$V_\Sigma^{(\text{пр})} \to \min , \qquad (4)$$

when limited $R \le R_{доп}$.

To reduce the importance there is proposed to improve the method for identifying vulnerabilities in the work, as well as DSS methods as the system integral part for identifying vulnerabilities use. These methods implementation will allow you to achieve the required result by removing from V_Σ the data that do not affect decision making, automate the data processing process and focus on important information for decision making. It is proposed to use these methods as the vulnerability detection system modules components. In this case, the generalized model for identifying vulnerabilities is transformed into Fig. 9.

In the presented figure, it is not possible to display the detecting software security promptness factors account. However, the efficiency characteristic significantly affects the software quality and its security. For the optimization problem representation completeness, we introduce the software safety indicator, and denote it as the software secure factor $K_{без}^{(\text{ПО})}$.

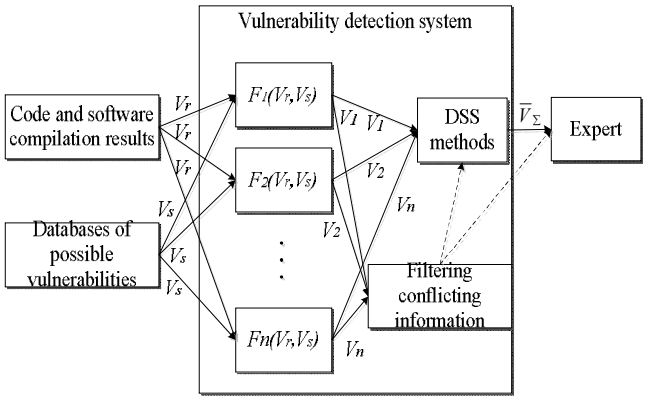

Fig. 9. Improved generalized vulnerability identification model

Based on this, we present the software security increasing task in the mathematical expression form:

$$K_{без}^{(ПО)} = \arg\left\{t_{без}\left(f\left(V_{\Sigma}^{(пр)}\right)\right) \to \min \,/\, R \le R_{доп}\right\}.$$

Thus, in order to a scientific problem solve and this goal achieve, it is necessary to the following research tasks solve:

– to conduct an existing methods analysis comparative and research for identifying vulnerabilities and making decisions on software compliance with security requirements;

– to improve the model and method for identifying software vulnerability;

– no develop a method for making a decision on software security;

– to evaluate the developed methods effectiveness and the research results obtained reliability;

– to develop practical recommendations for the software security increasing developed methods using.

Conclusions

The results analysis of main methods for identifying software vulnerabilities presents in the article. The results of author's research, synthesizing and regulating knowledge about systems for detecting software vulnerabilities, are presented. The software analysis methods used during certification tests are considered.

It is shown that the methods and techniques existing for software security analysis use do not ensure the result accuracy under fuzzy input data conditions. This drawback is aggravated by strict requirements for the test scenarios implementation speed. This is largely due to the fact that experts, in order to make a decision, have to large conflicting information amounts analyze. Consequently, it is necessary to develop a system for identifying vulnerabilities, the main task of which will be to the conflicting information amount minimize used by an expert when making a decision.

The most promising direction the existing identifying vulnerabilities systems efficiency increasing is seen in reducing the burden on an expert by methods for identifying vulnerabilities and implementing a decision support system improving. This will significantly reduce the time spent on a decision making on software security, and, as a result, will the software security testing procedure accessible to a developer's wide range make more.

Also, the software security testing procedures availability increasing have to the both developers and information systems users interest increase to the security problem, which, in turn, will give a new impetus to the methods and systems for identifying vulnerabilities development and verifying software compliance with security requirements.

REFERENCES

1. Kazarin, O. V. (2003), *Security of computer systems software*, MGUL, Moscow, 212 p.
2. Podshivalov, G.K., Ternovskov, V.B., Demidov, L.N. and Tarasov B.A. (2016), "Economic security in the face of uncertainty", *Economics: yesterday, today, tomorrow*, No. 2, pp. 242-257.
3. Savin, R. (2007), *Testing Dot Com or A Manual on Bug Abuse in Internet Startups*, Delo, Moscow, 312 p.
4. Tikhanychev, O. V. (2018), *Theory and practice of decision support automation*, Editus, Moscow, 76 p.
5. Yuzvovich, L.I. and Yudina, E.A. (2014), "An integrated approach to the study of the essence, principles and methods of financial planning at enterprises in the economic system", *Fundamental research*, No. 9, pp. 1596-1601.
6. (2020), *CWE List Version 4.1*, available at: https://cwe.mitre.org/data/.
7. Gavrylenko, S., Chelak, V., Hornostal, O. and Vassilev, V. (2020), "Development of a method for identifying the state of a computer system using fuzzy cluster analysis", *Advanced Information Systems*, Vol. 4, No. 2, pp. 8-11, DOI: https://doi.org/10.20998/2522-9052.2020.2.02.
8. Imtiaz, N., Murphy, B. and Williams L. (2019), "How Do Developers Act on Static Analysis Alerts? An Empirical Study of Coverity Usage", *2019 IEEE 30th International Symposium on Software Reliability Engineering (ISSRE)*, Berlin, Germany, pp. 323-333, DOI: https://doi.org/10.1109/ISSRE.2019.00040.
9. Ishizaka, Alessio and Philippe, Nemery (2013), *Multi-criteria Decision Analysis: Methods and Software*, SAP Labs – China, Shanghai, PRC 2013, 310 p.
10. (2020), *ISO/IEC 27034-1:2011 Information technology – Security techniques – Application security*, available at: https://www.iso.org/standard/44378.html.
11. (2020), *ISO/IEC 15408-1:2009 Information technology — Security techniques — Evaluation criteria for IT security*, available at: https://www.iso.org/standard/50341.html.
12. (2020), *New ISA/IEC 62443 standard specifies security capabilities for control system components*, available at: https://www.isa.org/intech/201810standards/.
13. O'Connell, and James, L.M. (2013), "SzalmaRoc-Estimator Software and Roc Analysis", *Proceedings the Human Factors and Ergonomics Society Annual Meeting*, Vol. 57 is. 1, pp. 1432-1434.
14. Sanjab, Anibal and Walid, Saad (2016), "On bounded rationality in cyber-physical systems security: Game-theoretic analysis with application to smart grid protection", *Computer Science, Mathematics, 2016 Joint Workshop on Cyber-Physical Security and Resilience in Smart Grids (CPSR-SG)*, Vienna, 2016, pp. 1-6, DOI: https://doi.org/10.1109/CPSRSG.2016.7684101.

15. (2020), *OWASP Proactive Controls*, available at: https://owasp.org/www-project-proactive-controls/.
16. Semenova, Z.V., Danilova, O.T. and Kovshar, I.R. (2019), "The analysis of security of a stack of technologies for development of web-resources", *Dynamics of systems, mechanisms and machines*, Vol. 7, No. 4, pp. 98-105.
17. Sinha, S.M. (2006), *Mathematical Programming. Theory and Methods*. Elsevier Science, 572 p.
18. Zhang, Yuchen and Liu, Jing (2019), "Optimal Decision-Making Approach for Cyber Security Defense Using Game Theory and Intelligent Learning", *Security and Communication Networks Volume*, Article ID 3038586, 16 p., DOI: https://doi.org/10.1155/2019/3038586.

Надійшла (received) 25.06.2020
Прийнята до друку (accepted for publication) 02.09.2020

ВІДОМОСТІ ПРО АВТОРІВ / ABOUT THE AUTHORS

Можаєв Михайло Олександрович – кандидат технічних наук, завідувач сектором комп'ютерно-технічних, телекомунікаційних досліджень та досліджень відео-, звукозапису Харківського науково-дослідного інституту судових експертиз ім. засл. проф. М. С. Бокаріуса, Харків, Україна;
Mykhailo Mozhaiev – Candidate of Technical Sciences, Head of Department of Computer Engineering, Telecommunications, Video- and Audio-recording Research, Hon. Prof. M.S. Bokarius Kharkiv Research Institute of Forensic Examinations, Kharkiv, Ukraine;
e-mail: mikhail.mozhayev@hniise.gov.ua; ORCID ID: http://orcid.org/0000-0003-1566-9260.

Давидов Вячеслав Вадимович – кандидат технічних наук, доцент кафедри "Обчислювальна техніка та програмування", Національний технічний університет "Харківський політехнічний інститут", Харків, Україна;
Viacheslav Davydov – Candidate of Technical Sciences, 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.

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.

Аналіз та порівняльні дослідження методів підвищення рівня безпеки програмного забезпечення

М. О. Можаев, В. В. Давидов, Джан Ліцзян

Анотація. У статті представлені результати аналізу основних методів виявлення вразливостей програмного забезпечення. Представлені результати досліджень ряду авторів, синтезуючих та регламентуючих знань про системи виявлення вразливостей програмного забезпечення. Розглянуті методи аналізу програмного забезпечення, що використовуються при проведенні сертифікаційних випробувань. Показано, що використання існуючих методів та методик аналізу безпеки програмного забезпечення не забезпечує точність результатів на умовах нечисельних вхідних даних. Цей недолік ускладнює жорсткі вимоги до оперативності реалізації тестових сценаріїв. Отже, необхідне розроблення системи виявлення вразливих користувачів, основною задачею якої буде мінімізація кількості протирічної інформації, що використовується експертом при прийнятті рішень. Найкраще перспективне створення підвищення ефективності існуючих систем виявлення вразливостей полягає у зменшенні навантажень експертів за рахунок випробувань методів виявлення вразливостей, що впроваджені до систем підтримки прийняття рішень. Це дозволить істотно знизити затримки часу при прийнятті рішень щодо безпеки програмного забезпечення, а також, як наслідок, зробить процедуру тестування безпеки програмного забезпечення більш доступною широкому колу розробників. Підвищення доступності процедур тестування безпеки програмного забезпечення повинно підвищувати інтерес як розробників, так і користувачів інформаційних систем до проблем безпеки, що, в свою чергу, дасть новий імпульс розвитку методів та систем виявлення вразливостей і перевірки відповідності програмним вимогам безпеки.

Ключові слова: комп'ютерна система; програмне забезпечення; ризики безпеки; загрози безпеки.

Анализ и сравнительные исследования методов повышения безопасности программного обеспечения

М. А. Можаев, В. В. Давыдов, Джан Лицзян

Аннотация. В статье представлены результаты анализа основных методов выявления уязвимостей Software. Приведены результаты исследований ряда авторов, синтезирующие и регламентирующие знания о системах выявления уязвимостей Software. Рассмотрены методы анализа Software, используемые при проведении сертификационных испытаний. Показано, что использование существующих методов и методик анализа безопасности Software не обеспечивает точности результата в условиях нечетких входных данных. Этот недостаток усугубляется жесткими требованиями к оперативности реализации тестовых сценариев. Следовательно, необходимо разработать систему выявления уязвимостей, основной задачей которой будет минимизация количества противоречивой информации, используемой экспертом при принятии решения. Наиболее перспективным направлением повышения эффективности существующих систем выявления уязвимостей видится снижение нагрузки на эксперта за счет усовершенствования методов выявления уязвимостей и внедрения системы поддержки принятия решения. Это позволит существенно снизить затраты времени на принятие решения о безопасности Software, и, как следствие, сделает процедуру тестирования безопасности Software более доступной широкому кругу разработчиков. Повышение доступности процедур тестирования безопасности Software должно повысить интерес как разработчиков, так и пользователей информационных систем к проблеме безопасности, что, в свою очередь, даст новый импульс развитию методов и систем выявления уязвимостей и верификации соответствия Software требованиям безопасности.

Ключевые слова: компьютерная система; программное обеспечение; риски безопасности; угрозы безопасности.