https://doi.org/10.31891/csit-2024-3-5
UDC 519.2

Oleksandr REVNIUK, Andrii POSTOLIUK
Ternopil Ivan Puluj National Technical University

# RESEARCH ON THE APPLICATION OF ADAPTIVE RISK ASSESSMENT METHODS FOR WEB APPLICATIONS

*The paper analyzes the significance of security issues in modern web applications and emphasizes that the major threats in this area include low awareness among employees regarding information security, weak password policies or widespread non-compliance, deficiencies in software update management processes, use of unsafe configurations, and paradoxically, ineffective inter-network access segmentation.*

*The testing methods of «white-box,» «gray-box,» and «black-box» are described. It is argued that gray-box testing combines techniques used in black-box testing along with reverse engineering technologies and methods. The value of source code in vulnerability discovery lies in its representation of the program's logic in a comprehensible form for researchers. Analyzing source codes, in addition to black-box and gray-box methods, allows for the identification of more vulnerabilities for each application. Specifically, white-box testing on average identifies 3.5 times more medium-risk vulnerabilities compared to black-box and gray-box methods.*

*Based on the identified list of most common threats to web applications and the application of an enhanced cumulative risk methodology, a detailed analysis of threat data was conducted, and risk factors specific to each threat were identified. These factors were determined based on available statistics. A comparison of security risk assessment methods for web applications was conducted using an example from the banking sector. Criteria for translating indicators from quantitative to qualitative values for the researched enterprise are provided. Recommendations are made to reduce threat levels regarding reported vulnerabilities: reducing the automatic system logout time during inactivity; implementing multi-factor authentication on the web application, such as password and card, or password and fingerprint; installing additional protective software (e. g., vipnet); enabling quick revocation of privileges, minimizing damage by swiftly identifying and stopping unauthorized actions; any changes in an employee's position that affect their rights should promptly reflect in their actual rights in the computer system.*

*Keywords: information security, web attack, SQL injections, OWASP, databases, information security management system, information protection, risk, risk management.*

Олександр РЕВНЮК, Андрій ПОСТОЛЮК
Тернопільський національний технічний університет імені Івана Пулюя

# ДОСЛІДЖЕННЯ ЗАСТОСУВАННЯ АДАПТИВНИХ МЕТОДИК ОЦІНЮВАННЯ РИЗИКІВ БЕЗПЕКИ ДЛЯ ВЕБ-ДОДАТКІВ

*Проаналізовано вагомість проблеми забезпечення безпеки веб-додатків в сучасних умовах та наголошено, що найбільшими загрозами в цій області є низька обізнаність співробітників у питаннях інформаційної безпеки, слабка парольна політика або повсюдне її невиконання, недоліки в процесах управління оновленням програмного забезпечення, використання небезпечних конфігурацій, і, як це може здатися парадоксальним, неефективним міжмережевим розмежуванням доступу.*

*Дано опис тестування методами «білого», «сірого» та «чорного ящиків». Висловлена думка, що тестування за принципом «сірого ящика» являє собою комбінацію методів, що використовуються при тестуванні за принципом «чорного ящика», а також технологій і прийомів реверс розробки. Цінність вихідного коду в процесі пошуку вразливостей полягає в тому, що він представляє логіку роботи програми в зрозумілому для дослідника поданні. Аналіз вихідних кодів, на додаток до аналізу методами чорного і сірого ящика дозволяє виявити більше вразливостей для кожного додатка. Зокрема, тестування методом білого ящика в середньому знаходить в 3,5 рази більше вразливостей середнього ступеня ризику в порівнянні з методами чорного і сірого ящика.*

*На основі виявленого переліку найпопулярніших загроз для веб-додатків і застосування удосконаленої методології сукупного ризику був проведений детальний аналіз даних загроз, а також були виявлені фактори ризику, характерні для кожної із загроз. Ці фактори визначалися на основі доступної статистики.*

*На прикладі підприємства банківської сфери проведено порівняння методик з оцінки ризиків безпеки для веб-додатків. Наведено критерії переводу показників з кількісних в якісні величини для досліджуваного підприємства. Зроблено такі рекомендації для зниження рівня загроз, щодо заявлених вразливостей: зменшення часу автоматичного виходу з системи при бездіяльності; багатофакторна аутентифікація на веб-додатку. Наприклад, пароль і карта, або пароль і відбиток пальця; установка додатково захисного програмного забезпечення (vipnet та ін.); можливість швидкого відкликання прав, тобто мінімізація збитку за рахунок швидкого з'ясування та припинення несанкціонованих дій; будь-які зміни в позиції співробітника, що тягнуть зміни в його правах, повинні якомога швидше відбиватися на його реальних правах в комп'ютерній системі.*

*Ключові слова: інформаційна безпека, Web-атака, SQL-ін'єкції, OWASP, бази даних, система управління інформаційною безпекою, захист інформації, ризик, управління ризиками.*

## Introduction

The vulnerability of web applications remains one of the most common shortcomings in information security. Among other frequently encountered issues are low employee awareness of information security, weak password policies or their widespread non-compliance, deficiencies in software update management processes, the use of insecure configurations, and, paradoxically, ineffective network access segmentation. Despite the fact that web application vulnerabilities have been repeatedly described in contemporary scientific and specialized literature,

preventive protective mechanisms that reduce the risks of exploitation are rarely encountered [2]. Malicious actors can harm a business or organization by exploiting a web application. Such methods of exploiting a web application pose a serious threat to the user of the application, potentially leading to negative consequences. In some cases, web application security vulnerabilities can be easily identified and remedied, while in others, even detecting the existence of a security vulnerability can be challenging. The potential damage from exploiting security vulnerabilities in a web application can vary from minor financial losses to the complete bankruptcy of a company. To assess the security vulnerability risks of a web application for a specific organization, it is necessary to evaluate the probabilities associated with threat sources, attack vectors, and security vulnerabilities, and then combine these with an assessment of the technical and reputational damage to the organization [3–9].

**Research Objective**

The objective of this study is to explore the application of adaptive risk assessment methodologies for web application security.

**Related works**

The issues of data preservation and protection in information systems are currently being addressed by a significant number of both Ukrainian and foreign researchers, including I. R. Maltseva [3], C. Grigoriadis [13], S. Rafique [24], V. Lakhno [19], D. Yadav [31], and S. Kumar [17].

**Research Methods**

The methodological foundation of this study consists of a set of methods, approaches, and techniques for scientific analysis, including methods of induction and deduction, analysis and synthesis, the unity of historical and logical approaches, abstraction, generalization, a systematic approach to the phenomena under study, as well as comparative and statistical methods.

**Problem Statement**

Risk assessment involves determining qualitative indicators, calculating quantitative indicators, creating a risk registry, and classifying risks according to their impact on information security. The risk assessment process begins with the inventory of visual information and the determination of its value. Next, based on the current threat model and existing prerequisites, the organization must identify a list of potential risks. This list is ranked by determining the level of risk—a value traditionally derived from the comparison of potential damage and the likelihood of its occurrence [10–14].

Indeed, a risk that is highly likely to cause 1 million UAH in damage will be given higher priority than a risk that could potentially cause 2 million UAH in damage but with a very low probability. Following this, options for mitigating the assessed risks are selected—for example, reducing their level by implementing appropriate protective measures—and the budget is calculated. The core of data leakage lies in the uncontrolled transfer of confidential information through acoustic, light, electromagnetic, radiation, and other fields, as well as material objects.

Globally, the development of standards, technical reports, guidelines, and recommendations in the field of information security (IS) is a continuous process; projects and versions of standards addressing various aspects of IS are consistently published at different stages of agreement and approval. The development of regulatory documents on IS, fully or partially dedicated to incident management, is carried out by a number of specialized international organizations and consortia, such as the Computer Emergency Response Team (CERT), ISO, IEC, Internet Engineering Task Force (IETF), International Telecommunication Union Telecommunication (ITU-T), Institute of Electrical and Electronics Engineers (IEEE), OMG, SANS Institute, X/Open, among others. In Ukraine, recommendations for web application security are described in the standard ND TZI 2.5–010–03 "Requirements for Protecting Information on a WEB Page from Unauthorized Access" [1].

**Presentation of the main material.**

Currently, testing is being conducted using the white-box method (which utilizes internal system data, including source code analysis) and is being compared with the results of black-box and gray-box testing methods (where the analysis is conducted with privileges similar to those of a potential attacker).

White-box testing has a significant advantage—code coverage. Since the source codes are available, they can be analyzed for potential vulnerabilities. However, one of the drawbacks of this method is its complexity, as existing tools are not perfect and generate a large number of false positives. Therefore, the report generated by the tool must be carefully reviewed by a competent specialist. Given the volume of code that modern programs contain, such reports can be extremely lengthy [15–21].

During manual testing using the "black-box" approach, the researcher navigates through application pages using a standard internet browser, inserting special characters into input fields and request parameters, such as a single quote, to identify scripts potentially vulnerable to SQL injection. Automated testing (fuzzing) is based on the brute-force method, and the main drawback of this approach is compensated by its simplicity and effectiveness.

Essentially, fuzzing involves sending a large number of random input data to the application under test and analyzing the results of its operation.

*Advantages of black-box testing* include:

1) Accessibility. This method can be applied in any situation and can be useful even when the source code of the application is available.

2) Versatility. Since the approach does not rely on specific information about a particular software product, a tool designed, for example, to assess the security of one web server can be used for any other.

3) Simplicity. At its most basic level, fuzzing does not require any knowledge of the internal structure of the application. However, it is clear that the most complex errors are practically impossible to detect using this method.

*This testing method also has several disadvantages*:

1) Coverage. One of the most difficult questions to resolve during fuzzing is when to stop testing and how effective it is.

2) Primitiveness. Fuzzing is not effective at detecting complex vulnerabilities, such as those that require multiple stages to place the program in a certain state and then trigger an error. Such vulnerabilities are usually identified through source code analysis.

Currently, there are fuzzers that generate input data that is not entirely random but is based on the specifications of the protocols and file formats being tested. These tools can also be classified under the "gray-box" testing methods.

Gray-box testing represents a combination of methods used in black-box testing, along with techniques and approaches from reverse engineering. The value of the source code in the vulnerability detection process lies in its ability to present the program's logic in a format understandable to the researcher [22–28]. The main goal of the analysis is to determine the internal logic of the secure application. There is no tool that can retrieve the original source code from a protected file (due to obfuscation).

However, using reverse engineering techniques, it is possible to present the program in a form that is comprehensible, even though it is not the complete source code. This method inherits one of the advantages of black-box testing—accessibility. Another significant advantage is code coverage. Information obtained through reverse analysis can significantly improve the quality of the input data generated by the fuzzer. A major drawback of this method is its complexity. Among the vulnerability detection technologies considered, this one imposes the highest requirements on the analyst's qualifications.

Source code analysis, in addition to black-box and gray-box methods, allows for the identification of more vulnerabilities in each application. Specifically, white-box testing, on average, detects 3.5 times more medium-risk vulnerabilities compared to black-box and gray-box methods.

In the field of information risk assessment and management, expert methods of evaluation currently prevail. Expert assessments usually involve estimating the probability of events occurring, as well as approximate values of damage corresponding to these events. Based on this data, system risk is calculated.

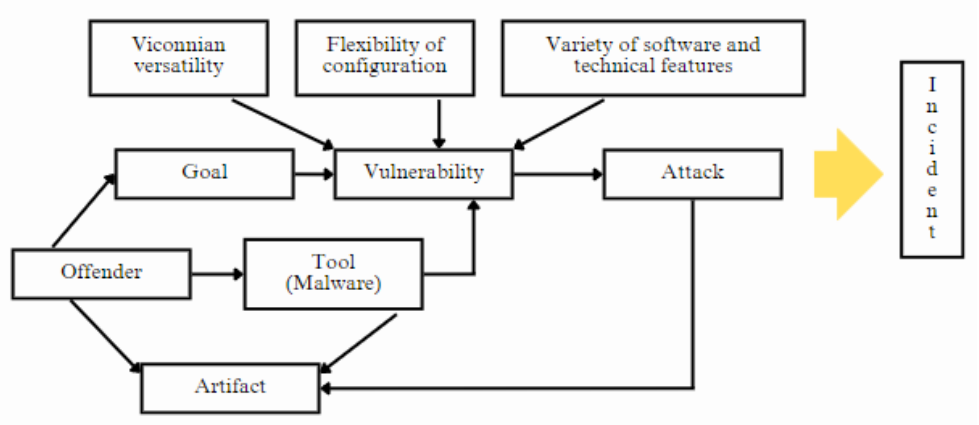


**Fig. 1. Diagram of Incident Occurrence for a Web Application**

The application of expert evaluation methods has obvious drawbacks, such as their subjectivity and significant errors when used in analytical calculations. It is also important to note the existing quantitative methods designed for risk assessments. These methods typically use accumulated statistics and operate with probabilities derived from statistical calculations. A disadvantage of these methods is the need to accumulate sufficiently large volumes of statistical data to obtain accurate risk level forecasts.

Penetration testing (pentesting) or ethical hacking is often employed, which involves identifying vulnerabilities in information activity objects (IAOs) and conducting controlled attacks. These attacks can be directed at individual information systems, such as a content management system (CMS), customer relationship management (CRM) system, enterprise resource planning (ERP) system, and internet client-bank, or at the entire

IAO infrastructure as a whole—such as the external network perimeter (IP address and website perimeter), wireless networks, internal or corporate networks, etc. [32].

Pentesting is essentially the simulation of a penetration process into an information environment within controlled boundaries, or in other words, the modeling of a basic hacking process with clear results.

*Conducting pentesting allows for the following*:

1) Understanding the feasibility of information security threats.
2) Assessing the consequences of a targeted hacker attack.
3) Identifying vulnerabilities in the information system's protection.
4) Evaluating the effectiveness of information protection measures.
5) Assessing the effectiveness of information security management.
6) Estimating the likely skill level of an attacker needed to successfully execute an attack.
7) Providing justification for further investment in information security.
8) Developing a list of countermeasures to reduce the likelihood of successful attacks.

As a result, this approach helps obtain objective information about the level of protection of a company's resources and provides a realistic basis for implementing a multi-layered defense system, considering the motivations of third parties—whether financial, political, or moral.

The risk assessment methodology from the Open Web Application Security Project (OWASP) can be presented as the following algorithm:

1) For each threat, the prevalence rate (based on statistics) and its danger factor (technical consequences) are determined.
2) Based on the description of threats in the Common Vulnerabilities and Exposures (CVE) database, the complexity of detection and exploitation is assessed for each threat.
3) The criticality level of each factor is then classified on a scale from 1 (low) to 3 (high).
4) The analyzed threats are compiled into a list with numerical values for the prevalence factor [30].

The advantages of this security risk assessment methodology for web applications include the fact that it provides generalized data on popular threats but not on specific vulnerabilities in real applications and Application Programming Interfaces (APIs). Therefore, only the owner or manager of the application can accurately assess the risks to a particular application. The owner alone possesses the most comprehensive knowledge necessary to evaluate the criticality of their data, the presence of potential threats, and the system's operational and usage principles.

The primary drawback of this security risk assessment methodology for web applications is its strong focus on the prevalence of threats. Some common threats may have weak technical consequences and should therefore be given lower priority during the development of security measures.

Based on the identified list of the most common threats to web applications and the application of an enhanced methodology for cumulative risk, a detailed analysis of threat data was conducted, and risk factors specific to each threat were identified. These factors were determined based on available statistics.

Additionally, combined methods of protecting web resources are being developed, which are based on a heuristic approach that highlights abnormal behavior, thereby increasing the likelihood of protection compared to signature analysis. The use of threat agent models for protecting web resources from attacks is also considered promising, as it allows for the formalization of vulnerability detection in information systems at all stages of the threat agent's interaction with the web resource [31].

A Web Application Firewall (WAF) applies a set of security rules to higher-level (application-level) protocols such as HTTP/HTTPS and FTP/FTPS. The traditional placement of a WAF in a network is in reverse proxy mode, positioned in front of the protected web servers. The typical functions of a WAF include the following security mechanisms: protocol validation, signature analysis, session and cookie protection, data leakage prevention, attack detection (based on a negative model, including attacks on the application, network, web server, and operating system), the ability to create custom security rules, and machine learning.

A firewall generally establishes a barrier between a protected internal network and an unprotected external network. Its primary purpose is to protect the internal network or specific nodes within it from unauthorized access. The firewall controls access to network resources using a positive control model (only traffic permitted by the rules is allowed into the internal network, and all other traffic is blocked).

To determine which security risk analysis and assessment methodology for web applications is the best for evaluating security risks, it is necessary to conduct a comparative analysis of the available options using various criteria. When selecting the indicators to focus on during the comparison, all possible specifics, such as those of credit organizations, should be considered to ensure that the developed methodology meets the information security needs of such companies against external interference.

*Let's outline the main stages of risk assessment*.

1) Identification of At-Risk Assets: In the first stage, based on survey data, technical documentation, and automated network analysis, a list of assets located within the risk zone is compiled.
2) Threat Identification: Different approaches can be used to compile a list of threats:

2.1) Conventional Method: In this case, experts create checklists of potential threats, from which the most relevant to the specific system are selected.

2.2) Statistical Method: This involves analyzing the statistics of events related to the information security of the given information system (IS) and similar systems, assessing their average frequency, and then evaluating the risk points.

2.3) Brainstorming: Conducted by the company's employees, this method differs from the first in that it is done without the involvement of external experts.

3) Collection of Risk Statistics: After compiling a list of potential threats, statistics are gathered on the occurrence of each risk: the frequency of certain situations and the level of damage caused. Based on this data, experts assess the impact degree using the following parameters: probability of threat occurrence (High Probability, Medium Probability, and Low Probability) and the severity of the damage (High Impact, Medium Impact, and Low Impact).

The first step in applying the proposed risk assessment methodology is calculating interaction coefficients for each web application development scenario. Figure 2 shows the process flow diagram for handling customer orders for products.

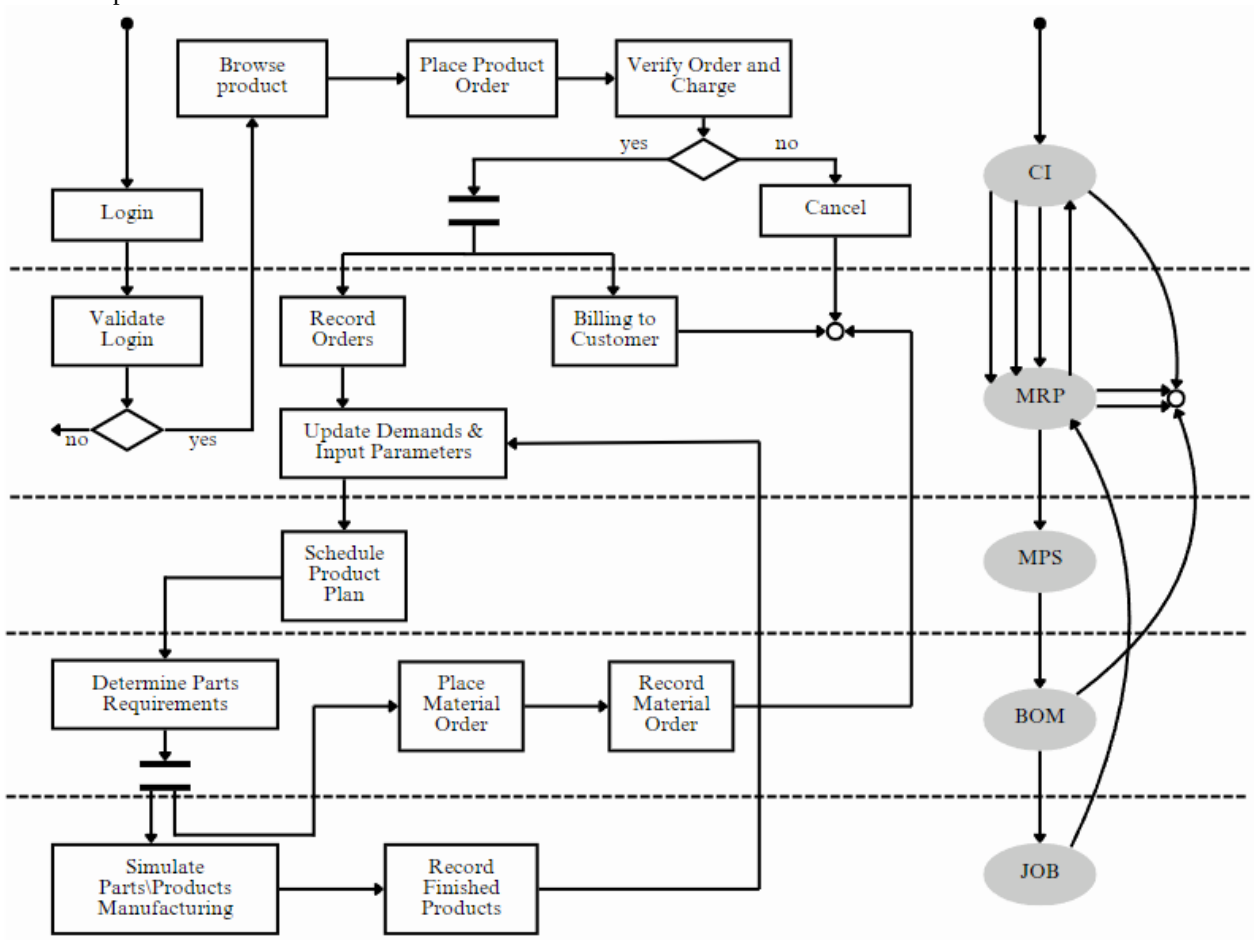


**Fig. 2. Workflow diagram of the client use case**

We assume that clients have established business relationships with the manufacturing enterprise, and therefore Material Requirements Planning (MRP) is available only to these clients. Consequently, clients and suppliers must access the system through a web interface. The workflow starts with a dark circle and ends with a white circle. It represents both sequential actions (indicated by directed links) and parallel actions (marked by double lines) between different components. For example, after determining the details needed for product manufacturing, the specification sends the results to the workshop for simulating the production orders and simultaneously places raw material orders with the respective suppliers. The workflow diagram is extended and includes software components responsible for the functions/execution of these actions. The components are depicted as ovals on the right side of the diagram. Arrows between the components indicate the interactions between them. MRP has one interaction with the Master Production Schedule (MPS) but has four overall interactions (two from MRP to the end state, one to the Customer Interface (CI), and one to MPS), along with interactions with the Bill of Material (BOM), Job Shop Simulator (JOB), and Supplier Interface (SI).

By transforming the relationships in Figure 2 according to the risk assessment methodology, we simplify the B2B process by assuming that all payments between the customer and the manufacturer or between the

manufacturer and the supplier are offline transactions. To facilitate understanding of our approach to analysis, we present the workflow in three use cases, each of which has one scenario. The three use cases are: the client orders a product, the manager adjusts the requirements, and the supplier confirms the material order. We assume that the probability of each scenario occurring is 0.4, 0.2, and 0.4, respectively.

As illustrated in Figure 2, the workflow diagram for processing customer orders for products can be characterized as follows: We assume that clients have established business relationships with the manufacturing enterprise, and therefore MRP is available only to these clients. Consequently, clients and suppliers must access the system through a web interface. The workflow starts with a dark circle and ends with a white circle. It represents both sequential actions (indicated by directed links) and parallel actions (marked by double lines) between different components. For example, after determining the details needed for product manufacturing, the specification sends the results to the workshop for simulating the production orders and simultaneously places raw material orders with the respective suppliers. The workflow diagram is extended and includes software components responsible for the functions/execution of these actions. The components are depicted as ovals on the right side of the diagram. Arrows between the components indicate the interactions between them.

The first step in applying the proposed risk assessment methodology is to calculate interaction coefficients for each web application development scenario. As shown in Figure 2, MRP has one interaction with MPS, but overall it has four interactions (two from MRP to the end state, one to CI, and one to MPS).

| $M_1$ | CI | MRP | MPS | BOM | JOB | T |
|---|---|---|---|---|---|---|
| CI | 0 | 3/4 | 0 | 0 | 0 | 1/4 |
| MRP | 1/4 | 0 | 1/4 | 0 | 0 | 1/2 |
| MPS | 0 | 0 | 0 | 1 | 0 | 0 |
| BOM | 0 | 0 | 0 | 0 | 1/2 | 1/2 |
| JOB | 0 | 1 | 0 | 0 | 0 | 0 |

| $M_2$ | CI | MRP | MPS | BOM | JOB | T |
|---|---|---|---|---|---|---|
| $M_1$ | 0 | 2/3 | 0 | 0 | 0 | 1/3 |
| MRP | 1/3 | 0 | 1/3 | 0 | 0 | 1/3 |
| MPS | 0 | 0 | 0 | 1 | 0 | 0 |
| BOM | 0 | 0 | 0 | 0 | 1/2 | 1/2 |
| JOB | 0 | 1 | 0 | 0 | 0 | 0 |

| $M_1$ | SI | MRP | BOM | T |
|---|---|---|---|---|
| SI | 0 | 1/2 | 1/2 | 0 |
| MRP | 1/2 | 0 | 0 | 1/2 |
| BOM | 0 | 0 | 0 | 1 |

**Fig. 3. Interaction metric tables for each scenario**

Thus, the interaction frequency between MRP and MPS is 1/4, as shown in the MI table in Figure 3. MI represents the interaction frequency between components derived from the "client orders product" scenario, where T denotes the end state of the workflow. Similarly, we can derive the M2 and M3 tables, representing interaction levels in the manager and supplier scenarios, respectively.

Next, we construct a dependency graph of the software components. For each pair of components, we calculate the transition rate using the corresponding interaction frequencies obtained from each scenario (MI, M2, and M3) and the given probability of each scenario, as described in Figure 3.

In the subsequent step, we identify vulnerabilities of the hardware platforms from the specified deployment plan by applying a scanning tool (if the necessary system host and network configurations are already operational) or, alternatively, by performing a lookup in public vulnerability databases. Table 1 presents the sources of vulnerabilities and their quantities in our application deployment plan. As previously described, the search for vulnerability sources in the table is based on the software and environments required by each host to provide its services.

Table 1

**Vulnerabilities in the equipment deployment plan**

| Host | Vulnerability Sources | Count |
|---|---|---|
| *W* | Apache-Chunk, PHP 4.2 | 2 |
| *A1* | Jboss, JRE 1.4.2, Windows, Tomcat-3.2.1 | 4 |
| *A2* | Jboss, JRE 1.4.2, Windows | 3 |
| *A3* | Telnetd | 1 |
| *D*1 | Oracle, TNS Listener | 2 |
| *D*2 | Oracle, TNS Listener | 2 |
| *D*3 | Oracle, TNS Listener | 2 |
| *D*4 | Oracle, TNS Listener | 2 |

Next, to determine the vulnerability coefficient ViV_iVi of component iii, we identify the hosting services required by the component. For example, the Customer Interface (CI) requires a web server WWW for logging in and ordering products, as well as accessing product information from the database D2D2D2 for viewing. Similar results can be derived for other components, as shown in Table 2

Table 2

**Component Requirements and Vulnerability Coefficients**

| | W | Al | A2 | A3 | D1 | D2 | D3 | D4 | #Vuls | Vi |
|---|---|---|---|---|---|---|---|---|---|---|
| Cl | 1 | | | | | 1 | | | 4 | 0.09 |
| SI | 1 | | | | | | 1 | 1 | 6 | 0.14 |
| Ml | | 1 | | | | 1 | | 1 | 8 | 0.2 |
| MRP | | 1 | | | 1 | 1 | 1 | 1 | 10 | 0.23 |
| MPS | | | 1 | | | 1 | 1 | | 7 | 0.16 |
| BOM | | | 1 | | | | 1 | 1 | 7 | 0.16 |
| JOB | | | | 1 | | | | | 1 | 0.02 |

In the second-to-last column, the number of vulnerabilities associated with each component is indicated. This value is calculated by summing the total number of vulnerabilities on each of the hosts required for the component. The last column shows the vulnerability coefficients with a total coefficient equal to one, representing the overall system vulnerability.

Figure 4a displays the final graph obtained in step 2, annotated with transition coefficients and vulnerability coefficients (highlighted in bold), along with heuristic estimates of the project components' vulnerabilities. Here, we assume that at the initial point S (solid circle), the system has no vulnerabilities, while at the endpoint T (white circle), it has the overall system vulnerability coefficient (i.e., one).

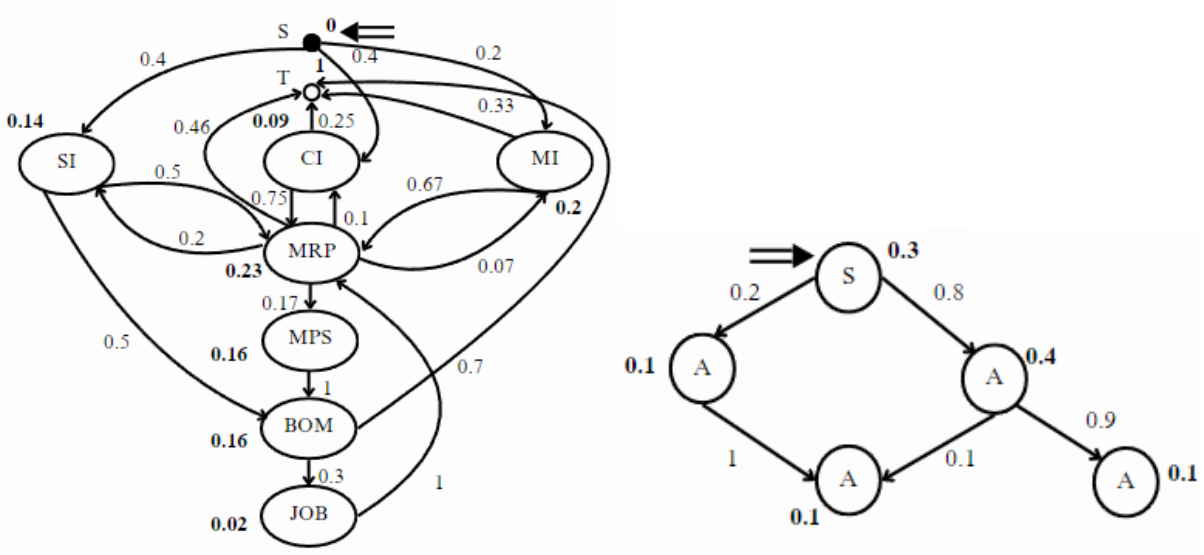


**Fig. 4. a) Dependency graph of software components; b) Probability of security breach in components**

Next, we determine the probability of attacks on each component (or security breaches within it) using the approach described in the example in Figure 3b. For instance, the probability of an attack on CI (via path 5 — CI) is calculated as 1 x 0.4 x 0.09 = 0.036. Note that the path to CI from MRP (via 5 — CI — MRP — CI) is ignored because it results in a lower probability and is, therefore, not the worst-case scenario as desired. The probability of an attack on MRP is calculated as the sum of the probabilities of reaching MRP from CI, MI, SI, and JOB, i.e., 0.06 + 0.02 + 0.04 + 0.01 = 0.14. The summary information about the probability of security breaches in the components is provided in the first row of Table 3.

Table 3

**Risk Analysis Results**

| | CI | SI | MI | MRP | MPS | BOM | JOB | T |
|---|---|---|---|---|---|---|---|---|
| Likelihoods | 0.04 | 0.04 | 0.04 | 0.14 | 0.07 | 0.08 | 0.13 | 0.665 |
| Severity | 0.75 | 0.5 | 0.25 | 0.95 | 0.75 | 0.75 | 0.5 | 0.75 |
| Risk | 0.03 | 0.03 | 0.01 | 0.13 | 0.05 | 0.06 | 0.07 | 0.499 |
| %Risk | 7.13 | 7.39 | 2.64 | 34.31 | 14.51 | 16.77 | 17.25 | |

In the next stage, we assess the impact of adverse consequences in each component, categorizing the consequences into five categories: CT (catastrophic: 0.95), CR (critical: 0.75), MG (marginal: 0.5), and MN (minor: 0.25) (ISO, 2002).

By selecting the maximum severity level obtained for each component as the severity of the component and applying the standard quantitative assessment, we can determine the risk for each component, as shown in Table 3. Here, MRP is identified as the component with the highest risk (34.31%), while MI has the lowest risk (2.64%). The overall system risk is calculated by multiplying the probability of reaching the endpoint, T, by its vulnerability and

the overall system severity. In this context, we use CR, which is the severity category for the majority of all components.

Table 4

| Component | Malicious Actions | Consequences |
|---|---|---|
| CI | Change product order | Unverified order |
| | Escalate access level | Increases vulnerabilities |
| MRP | Change product order | Incorrect billing, planning |
| | Modify requirements | Custom order implementation |
| | Adjust pricing | Loss of profit |
| | Access client information | Loss of financial information |
| | Access product information | Loss of product information |
| MPS | Access schedule changes | Incorrect plans |
| | Access product information | Loss of product information |
| | Access supplier information | Loss of supplier information |
| BOM | Change components | Incorrect product plan |
| | Escalate access to product information | Increases vulnerabilities |
| | Access supplier information | Loss of product and supplier information |
| JOB | Change product production completion time | Delays in delivery leading to loss of customers |
| | Access product information | Loss of product information |
| MI | Change product production stages | Incorrect update of product information |
| | Escalate access level | Increases vulnerabilities |
| SI | Escalate access level | Increases vulnerabilities |

**Vulnerability and Impact Analysis of Web Applications**

The results presented in Tables 3 and 4 show that the overall security risk for the system is approximately 50%, which can be used as a baseline for comparison with alternative options. These risks provide relative indicators that assist in making decisions about security options in different projects.

Thus, the proposed adaptive security risk assessment methodology for web applications can be used to evaluate security risks systematically and at an early stage. The risks associated with components connected to attacked components are determined in the same way as those that are not under attack.

*Key recommendations for reducing threat levels related to identified vulnerabilities* include:

1) Reducing the time for automatic system logout during inactivity;

2) Implementing multi-factor authentication for the web application, such as a combination of a password and a card, or a password and a fingerprint;

3) Installing additional protective software (e.g., ViPNet and others);

4) Enabling quick revocation of access rights, minimizing damage by quickly identifying and stopping unauthorized actions;

5) Ensuring that any changes in an employee's position, which entail changes in their access rights, are reflected in their actual system rights as quickly as possible.

**Conclusions**

The importance of ensuring web application security in modern conditions has been analyzed, and a description of testing methods using "white-box," "gray-box," and "black-box" techniques has been provided. It is suggested that gray-box testing combines the methods used in black-box testing with reverse engineering techniques.

Based on the identified list of the most common threats to web applications and the application of an improved cumulative risk methodology, a detailed analysis of threat data was conducted, and risk factors specific to each threat were identified. A comparison of security risk assessment methodologies for web applications was conducted using an example from the banking sector. Criteria for translating quantitative indicators into qualitative values for the studied enterprise were also provided.

**References**

1. Kapiton A. (2021). Perspektyvy rozvytku kiberprostoru ta jogho socialjno-psykhologhichni naslidky. Systemy upravlinnja, navighaciji ta zv'jazku. *Zbirnyk naukovykh pracj. Poltava*: PNTU. T. 3 (65). S. 89–91. doi: https: // doi. org / 10.26906 / SUNZ. 2021.3.089.

2. Leghominova S., Ghajdur Gh. (2023). Analiz suchasnykh zaghroz informacijnij bezpeci orghanizacij ta formuvannja informacijnoji platformy protydiji jim. Elektronne fakhove naukove vydannja «Kiberbezpeka: osvita, nauka, tekhnika». # 2 (22), S. 54–67. https: // doi. org / 10.28925 / 2663–4023.2023.22.5467.

3. Maljceva I. R. (2022). Analiz dejakykh kiberzaghroz v umovakh vijny. *Kiberbezpeka: osvita, nauka, tekhnika*. # 4 (16). S. 37–43. doi: 10.28925 / 2663–4023.2022.16.3744.

4. Onishhenko S., Ghlushko A. (2022). Analitychnyj vymir kiberbezpeky Ukrajiny v umovakh zrostannja vyklykiv ta zaghroz. *Naukovyj zhurnal «Ekonomika i reghion»,* # (1 (84), S. 13–20. https: // doi. org / https: // doi. org / 10.26906 / EiR. 2022.1 (84). 2540.

5. Tkach Ju. (2021). Konceptualjna modelj bezpeky kiberprostoru. *Tekhnichni nauky ta* tekhnologhiji. # 4 (22). S. 96–108. https: // doi. org / 10.25140 / 2411–5363–2020–4 (22) – 96–108.

6. Khlaponin I., Kozubcova L. (2022). Funkciji systemy zakhystu informaciji i kiberbezpeky krytychnoji informacijnoji infrastruktury. *Elektronne fakhove naukove vydannja «Kiberbezpeka: osvita, nauka, tekhnika»*. # 3 (15), S. 124–134. https: // doi. org / 10.28925 / 2663–4023.2022.15.1241341.

7. Abozeid, A., AlHabshy, A. A., ElDahshan, K. (2021). A Software Security Optimization Architecture (SoSOA) and Its Adaptation for Mobile Applications. *International Journal of Interactive Mobile Technologies*, 15 (11). https: // doi. org / 10.3991 / ijim. v15i11.20133.

8. Agrawal, A., Alenezi, M., Kumar, R., and Khan, R. A. (2019). Measuring the sustainable- security of Web applications through a fuzzy-based integrated approach of AHP and TOP-SIS. *IEEE Access*, 7: 153936–153951. https: // doi. org / 10.1109 / ACCESS. 2019.2946776.

9. Alali, M., Almogren, A., Hassan, M. M., Rassan, I. A. L., Bhuiyan, M. Z. A. (2018). Improving risk assessment model of cyber security using fuzzy logic inference system. *Computers & Security*, 74, 323–339. https: // doi. org / 10.1016 / j. cose. 2017.09.011.

10. Al zahrani A., Alqazzaz A. Web Application Security Tools Analysis, 2017 ieee 3rd international conference on big data security on cloud (big data security), *ieee international conference on high performance and smart computing (hpsc), and ieee international conference on intelligent data and security* (ids), Beijing, China, 2017, pp. 237–242, doi: 10.1109 / BigDataSecurity. 2017.47.

11. Cagliano, A. C., Grimaldi, S., Rafele, C. (2015). Choosing project risk management techniques. A theoretical framework. *Journal of Risk Research*, 18 (2), 232–248. doi: 10.1080 / 13669877.2014.896398.

12. Goutam, A., and Tiwari, V. (2019). Vulnerability Assessment and Penetration Testing to Enhance the Security of Web Application. *4th International Conference on Information Systems and Computer Networks (ISCON)*, November, IEEE, pp. 601–605. https: // doi. org / 10.1109 / ISCON47742.2019.9036175.

13. Grigoriadis C, Laborde R, Verdier A, Kotzanikolaou P. (2021). An Adaptive, Situation-Based Risk Assessment and Security Enforcement Framework for the Maritime Sector. Sensors (Basel). Dec 29;22 (1): 238. doi: 10.3390 / s22010238. PMID: 35009781; PMCID: PMC8749908.

14. Hammoudeh, M. A. A., Alobaid, A., Alwabli, A., Alabdulmunim, F. (2021). The Study on Assessment of Security Web Applications. *International Journal of Interactive Mobile Technologies (iJiM)*, 15 (23), pp. 120–135. https: // doi. org / 10.3991 / ijim. v15i23.27357.

15. Holik F., Neradova S. (2017). Vulnerabilities of modern web applications, *40th International Convention on Information and Communication Technology, Electronics and Microelectronics* (MIPRO), Opatija, Croatia, pp. 1256–1261, doi: 10.23919 / MIPRO. 2017.7973616.

16. Korobiichuk, I., Hryshchuk, R., Mamarev, V., Okhrimchuk, V., Kachniarz, M. (2017). Cyberattack Classificator Verification. *International Conference on Diagnostics of Processes and Systems DPS 2017: Advanced Solutions in Diagnostics and Fault Tolerant Control*, pp. 402–41 doi: 10.1007 / 978–3–319–64474–5_34.

17. Kumar S., Mahajan R., Kumar N. (2017). A study on web application security and detecting security vulnerabilities, 6th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, pp. 451–455, doi: 10.1109 / ICRA TO. 2017.8342469.

18. Kuzminykh, I.; Ghita, B.; Sokolov, V.; Bakhshi, T. (2021). Information Security Risk Assessment. Encyclopedia 1, 602–617. https: // doi. org / 10.3390 / encyclopedia1030050.

19. Lakhno V., Kozlovskyi V., Klobukov V., Kryvoruchko O., Chubaievskyi V., Tyshchenko D. (2022). Software Package for Information Leakage Threats Relevance Assessment. Cybernetics Perspectives in Systems, vol. 503, pp. 290. doi: 10.1109 / USBEREIT51232.2021.9454994.

20. Muzaki, R. A., Briliyant, O. C., Hasditama, M. A., and Ritchi, H. (2020). improving security of web-based application using modsecurity and reverse proxy in web application rewall. International Workshop on Big Data and Information Security (IWBIS) IEEE, pp. 85–90. https: // doi. org / 10.1109 / IWBIS50925.2020.9255601.

21. Nagpure, S., and Kurkure, S. (2017). Vulnerability assessment and penetration testing of Web application. International Conference on Computing, Communication, Control and Automation (ICCUBEA), August 2017, IEEE, pp. 1–6. https: // doi. org / 10.1109 / ICCUBEA. 2017.8463920.

22. Pradeep Kumar J., Ravi T. Nagendra K. V. Analysis of security vulnerabilities for web based application, Fourth International Conference on Advances in Recent Technologies in Communication and Computing (ARTCom2012), Bangalore, India, 2012, pp. 233–236, doi: 10.1049 / cp. 2012.2535.

23. Radu, R., Săndescu, C., Grigorescu, O., and Rughiniș, R. (2020). Analyzing Risk Evaluation Frameworks and Risk Assessment Methods. 19th RoEduNet Conference: Networking in Education and Research (RoEduNet), December, IEEE, pp. 1–6. https: // doi. org / 10.1109 / RoEduNet51892. 2020.9324879.

24. Rafique S., Humayun M., Hamid B. (2015). Web application security vulnerabilities detection approaches: A systematic mapping study, IEEE / ACIS 16th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel / Distributed Computing (SNPD), Takamatsu, Japan, pp. 1–6, doi: 10.1109 / SNPD. 2015.7176244.

25. Riaz, A. R.; Gilani, S. M. M.; Naseer, S.; Alshmrany, S.; Shafiq, M.; Choi, J. – G. (2022). Applying Adaptive Security Techniques for Risk Analysis of Internet of Things (IoT) – Based Smart Agriculture. Sustainability, 14, 10964. https: // doi. org / 10.3390 / su141710964.

26. Sánchez-García, I. D.; Mejía, J.; San Feliu Gilabert, T. (2023). Cybersecurity Risk Assessment: A Systematic Mapping Review, Proposal, and Validation. Appl. Sci. 13, 395. https: // doi. org / 10.3390 / app13010395.

27. Shrivastava, A., Choudhary, S., and Kumar, A. (2016). XSS vulnerability assessment and prevention in web application. 2nd International Conference on Next Generation Comput-ing Technologies (NGCT), October, IEEE, pp. 850–853. https: // doi. org / 10.1109 / NGCT. 2016.7877529.

28. Svitlana Shevchenko, Yuliia Zhdanova (2022). Information protection model based on information security risk assessment for small and medium-sized business. Cybersecurity Education Science Technique. January 158–175. doi: 10.28925 / 2663–4023.2021.13.158157.

29. Tsilyna M. (2021). Modern technologies of protection and processing of confidential documentary information in organizations and institutions of different forms of ownership. Library science. Record Studies. Informology. 4, 15–23 [in Ukrainian].

30. Vandana D., Himanshu Y. (2014). Web Application Vulnerabilities: A Survey, International Journal of Computer Applications, vol. 108, no. 1, pp. 25–31, https: // doi. org / 10.5120 / 18877–0144.

31. Yadav D., Gupta D., Singh D. (2018). Vulnerabilities and Security of Web Applications, 4th International Conference on Computing Communication and Automation (ICCCA), Greater Noida, India, pp. 1–5, doi: 10.1109 / CCAA. 2018.8777558.

32. Wang B., Liu L., Li F. (2019). Research on Web Application Security Vulnerability Scanning Technology, IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chengdu, China, pp. 1524–1528, doi: 10.1109 / IAEAC47372. 2019.8997964

| | | |
|---|---|---|
| **Oleksandr Revniuk**<br>**Олександр Ревнюк** | graduate student of Cybersecurity Department. Ternopil Ivan Puluj National Technical University Ternopil, Ukraine<br>e-mail: revo0708@gmail.com<br>https://orcid.org/0009-0005-0511-5354 | аспірант кафедри кібербезпеки. Тернопільський національний технічний університет імені Івана Пулюя, Тернопіль, Україна |
| **Andrii Postoliuk**<br>**Андрій Постолюк** | graduate student of Cybersecurity Department. Ternopil Ivan Puluj National Technical University Ternopil, Ukraine<br>e-mail: mrapostoliuk@gmail.com<br>https://orcid.org/0009-0004-0169-3379 | аспірант кафедри кібербезпеки. Тернопільський національний технічний університет імені Івана Пулюя, Тернопіль, Україна |