Recommended Practices for the Analysis of Web Application Vulnerabilities

Vitali V. Varenitca
Certification Department
NPO Echelon, JTC
Moscow, Russia
www@cnpo.ru

Alexey S. Markov
Information Security Department
Bauman Moscow State Technical University
Moscow, Russia
a.markov@bmstu.ru

Vladislav V. Savchenko
Certification Department
NPO Echelon, JTC
Moscow, Russia
mail@cnpo.ru

Abstract. The paper is dedicated to information security of web applications. It discusses main classes of web application vulnerabilities and topic-related regulatory documents. An original procedure for the analysis of web application vulnerabilities is suggested. Conformity of the suggested procedure with modern standards is demonstrated. The paper highlights some issues of concern associated with the analysis of vulnerabilities and identification of existing web application vulnerabilities, and suggests a few ways of how to solve them. The effectiveness and efficiency of this technique has been proved by the vulnerability statistics in the course of software certification for compliance with information security requirements.

Keywords – assessment of web application security, vulnerability assessment, vulnerability analysis technique.

I. INTRODUCTION

Timely identification of vulnerabilities is one of the most crucial tasks of web application testing [1-6]. The importance of this problem is attributed to a number of reasons, including the key ones [7-15]:

— Existing vulnerabilities imply poor security of data processed by web applications.

— It is difficult to identify various classes of web application vulnerabilities using static analyzers.

— Constantly growing complexity of modern web applications, the number of problems to be solved and the level of integration with other software and hardware make the problem of software code analysis insolvable due to limited resources allocated for testing.

— Certain classes of web application vulnerabilities cannot be identified using automation tools without a comprehensive vulnerability analysis.

— Regular vulnerability analysis helps minimize the risks associated with eventual intrusion and violation of the integrity, availability, and confidentiality of data processed by the web application.

Existing vulnerabilities mean web application vulnerabilities that are confirmed by the developer or those for which an exploitation scenario exists [16, 17].

We have analyzed information available in open information sources (OIS) in order to identify the causes of web application vulnerabilities and vulnerability exploits. To date, the open project of web application security assurance, Open Web-Application Security Project (OWASP), is one of the most comprehensive open information sources. OWASP regularly publishes information on existing web application attack techniques as well as the rating of attacks based on their implementation complexity, frequency, and criticality. Vulners (https://vulners.com/), CVE (https://cve.mitre.org/), NIST (https://www.nist.gov) databases, databank of security threats of the Federal Service for Technical and Export Control (https://bdu.fstec.ru) can also be useful.

II. METHODOLOGICAL APPROACH TO WEB APPLICATION VULNERABILITY ANALYSIS

To make the web application vulnerability analysis more effective, a vulnerability analysis technique based on web application vulnerability analysis has been developed using the information available in open sources.

At first, developer's software documents, including the source code need to be obtained. In addition to the software documents, the expert can obtain a set of tests the developer carries out during the routine analysis of the product vulnerabilities and other types of tests. The developer can also provide a test bench to enable familiarization with the product and ad hoc testing.

At this stage, the expert should study the public information sources to improve his/her awareness of the goals and tasks the tested product solves, the product purpose and its main functional features. The information should be sought for in the following publicly available sources:

- OWASP Foundation – the free and open software security community (https://www.owasp.org).

- Software developer's website.

- Other sources that contain information about the tested software and the information about similar software.

Sought for information required to expand the initial data should be based on following criteria:

- Product name and version.

- Name of similar software.

- Names of products which have the architecture similar to that of the tested product

- expert's propositions about the technologies used in the tested product based on expert's experience and qualification

The next stage includes ad hoc testing of the product.

During the exploratory testing, the expert shall perform ad hoc testing of the product.

The expert shall use a bench with the product installed and configured as required by the documents to prepare for ad hoc testing. To complete this step, the expert can:

- Use the bench prepared by the developer for ad hoc testing.

- Install and configure the product as required by the documents on his/her own.

- Use the product installed as part of the existing information system.

During this step, the expert shall:

- View the product.

- Test the product trying to disrupt the software operation or make it stop as soon as possible.

- Define the list of tools the expert is planning to use to identify defects in the code or product configuration.

A product can be tested by:

- Changing the configuration of the product and tools the product interacts with during the operation

- Using different variations of input data

- Using the product to process data known to be incorrect

- Making intentional attempts to put the product out of operation

- Studying the responses to specially formulated requests to the product

At the end of ad hoc testing, the following shall be documented:

- Potential weaknesses of the software which, in expert's opinion, may be the evidence of defects in the software code

- Name of potentially vulnerable technologies used to implement functional features of the product

- List of potentially unsafe product configurations

Then the expert shall carry out exploratory testing. At this stage, the expert shall perform the tests using the steps listed below.

The expert shall prepare for the exploratory testing. The expert shall obtain a bench with the product installed and configured as required by the documents to prepare for exploratory testing. The expert shall install and configure the product on his/her own. The bench shall allow for all types of product researches in all operation modes defined in the documents or tests required by the customer.

When preparing the test bench the expert shall perform the steps listed below.

III. Configuration of the Software Operational Environments

Software installation and configuration in compliance with the operating documentation.

Development and implementation of security measures required for software research.

Preparation of the test bench shall include the deployment and configuration of all operational environments in which the product can operate according to the operating documentation or which are specified by the customer, and identification of the tools required to perform the tests. The operational environments shall be installed, configured and adjusted in compliance with the relevant operating documentation. In case of any conflict between the requirements specified in the environment documents and the requirements for the environments in the software documents, the expert shall use the requirements defined in the environment documents and record the conflict. The expert shall analyze the conflict during the analysis of the product configuration when making further steps of this technique.

The expert shall analyze the available product documents and open information sources to obtain complete information about the product. The expert should examine the product documents and data provided in open information sources to obtain the following information:

- Identification characteristics of the product tested

- Identification characteristics of the software in which environment the test product operates

- Identification characteristics of the borrowed software

- Identification characteristics of the technologies used in the test product

After the identification characteristics are defined, the expert shall analyze the documents for the test product and perform a direct analysis of the product in order to define the set of the product input interfaces, to understand how these interfaces process the data, and to identify any additional potential vulnerabilities of the product.

During this step, the expert shall use expert analysis, documentation analysis and automated tools to identify the input interfaces of the test product, which make it possible to influence

on the product. The analysis shall result in a set of entry points the expert can use to produce a direct impact on the product.

After identification of all input interfaces of the product, the researcher shall get an idea of the structure and the type of data that can be sent to the identified interface. Then the expert shall define the input interfaces that affect the operation of the product security mechanisms.

After identification of the input interfaces of the product, the expert shall analyze the open sources for information on existing vulnerabilities of the product, its operational environment or technologies used to design the product. The expert shall document the analysis findings.

In order to complete this step, the expert shall use the unique characteristics identified previously. The expert shall search for the known (confirmed) vulnerabilities of the product using the following publicly available information sources:

- Databank of security threats of the Federal Service for Technical and Export Control of Russia (http://bdu.fstec.ru);
- Software vulnerabilities database Common Vulnerabilities and Exposures (CVE) (https://cve.mitre.org/);
- Software vulnerabilities database Vulners (https://vulners.com);
- National Vulnerability Database (NVD) (https://nvd.nist.gov/vuln/search);
- OWASP Foundation – the free and open software security community (https://www.owasp.org);
- Websites of the product developer and manufacturer and developers of borrowed components;
- Other open information sources.

Based on the analysis of the open sources, the expert shall supplement the previously identified potential weaknesses of the product.

Having identified the potential vulnerabilities described in the open information sources, the expert shall study the product documents to define the list of potentially unsafe product configurations. At this step, the expert shall read the documents for the test product to identify all possible ways of the software reconfiguration and define those configurations which can compromise the information integrity, availability, and confidentiality. At the end of this step, the expert shall correct the findings obtained earlier during the study of the product documents and in the course of ad hoc testing. The expert shall supplement the information about unsafe configurations of the test product by potentially dangerous configurations defined during this step and remove the potentially dangerous configurations for which the documents describe the techniques of how to neutralize threats caused by such configurations.

Identification of unsafe configurations shall be followed by a static analysis of the product code to identify potential vulnerabilities of the test software code.

At this step, the expert shall use static analysis tools to perform an expert assessment of the product source code. The number of false positive results can be minimized by using static analyzers which are based on symbolic execution methods and other state-of-the-art methods of false positive minimization during the static code analysis. The expert shall identify the responses which cannot be well-defined as false by the automated analysis tools as potential vulnerabilities of the product code and document them as an attachment to the vulnerability analysis certificate. The expert shall expand the information on the potential product weaknesses identified earlier with the information obtained during this step.

After the static analysis [3, 6, 18], the expert shall scan the test product using a security network scanner. The expert shall use the findings of the security scanner to identify the vulnerable components of the test product, unsafe configurations and other types of errors.

During this step, the expert shall use network scanning tools to assess the configured product in its real operating mode. If any potential configuration vulnerabilities or potentially unsafe components are identified, the expert shall correct the previous results.

On the completion of the product analysis with a network scanner [14, 18], the expert shall assess the security mechanisms of the test product for correct operation.

The expert shall examine the security mechanisms of the software under study, assess the correctness of their operation and make attempts to disrupt the claimed logic of the security mechanisms. If the expert can disrupt the normal operation of the product security mechanisms, or identify potential defects of the program using any of the methods, the expert shall add these findings to the identification results of the product potential weaknesses.

The exploratory testing shall result in a list of potential weaknesses of the product. The dynamic code analysis and fuzzing test shall be performed in relation to the potential software weaknesses identified. At the end of the completed analysis, the expert shall obtain [4, 5]:

- The list of errors in the product operation
- Fragments of data that lead to errors in the product operation.
- Sample scenarios of work with the product components whose execution causes a product behavior different from that described in the product documents.

The expert shall develop penetration tests based on the data obtained during the exploratory testing, dynamic analysis and fuzzing test, and carry out the penetration test [19, 20].

IV. Conclusions

The paper suggests a general technique and recommendations on identification of web application vulnerabilities. This technique has an applied scientific nature as it was formulated based on the findings of information security certification tests of software systems performed over many years.

Using this technique will turn the web application security assessment into a problem-oriented process, which will enable a more complete check of web resources in very a short time. This

technique complies with the state-of-the-art web application security assessment standards.

The available statistical data confirm the reliability, effectiveness and efficiency of the suggested technique [21].

REFERENCES

[1] Gaskova D., Massel A. Intelligent System for Risk Identification of Cybersecurity Violations in Energy Facility", In: Proceedings of the:2018 3rd Russian-Pacific Conference on Computer Technology and Applications (Vladivostok, Russia, August 18-25, 2018), RPC, IEEE, 2018, pp 1-5. DOI: 10.1109/RPC.2018.8482229.

[2] Kharzhevskaya A., Lomako A., Petrenko S. Representing Programs with Similarity Invariants for Monitoring Tampering with Calculations. Voprosy kiberbezopasnosti [Cybersecurity issues]. 2017. No2 (20). P. 9-20. DOI: 10.21681/2311-3456-2017-2-9-20.

[3] Markov A.S., Fadin A.A., Tsirlov V.L. Multilevel Metamodel for Heuristic Search of Vulnerabilities in the Software Source Code, International Journal of Control Theory and Applications, 2016, vol. 9, No 30, pp. 313-320.

[4] Pechenkin, A.I., Lavrova, D.S. Modeling the search for vulnerabilities via the fuzzing method using an automation representation of network protocols. (2015) Automatic Control and Computer Sciences, 49 (8), pp. 826-833. DOI: 10.3103/S0146411615080325.

[5] Reber, G., Malmquist, K., Shcherbakov, A. 2014. Mapping the Application Security Terrain. Voprosy kiberbezopasnosti [Cybersecurity issues]. 2014. N 1(2). P. 36-39. DOI: 10.21681/2311-3456-2014-2-36-39.

[6] Zegzhda, P., Zegzhda, D., Pavlenko, E., Dremov, A. Detecting Android application malicious behaviors based on the analysis of control flows and data flows (2017) ACM International Conference Proceeding Series, pp. 280-286. DOI: 10.1145/3136825.3140583.

[7] Barabanov A.V., Markov A.S., Tsirlov V.L. Information Security Controls Against Cross-Site Request Forgery Attacks on Software Application of Automated Systems. Journal of Physics: Conference Series. 2018. V. 1015. P. 042034. DOI :10.1088/1742-6596/1015/4/04203

[8] Calzavara S., Focardi R., Nemec M., Rabitti A., Squarcina M. Postcards from the Post-HTTP World: Amplification of HTTPS Vulnerabilities in the Web Ecosystem. In: 2019 IEEE Symposium on Security and Privacy (SP), IEEE, 2019, 8995551, DOI: 10.1109/SP.2019.00053

[9] Calzavara S., Focardi R., Squarcina M., Tempesta M. Surviving the Web: A Journey into Web Session Security, ACM Comput. Surv., 2017, vol. 50, no. 1, pp. 1-34, DOI: 10.1145/3038923.

[10] Nirmal K., Janet B., Kumar R. Web Application Vulnerabilities - The Hacker's Treasure. In: 2018 International Conference on Inventive Research in Computing Applications (ICIRCA), IEEE, 2018, 18358073, DOI: 10.1109/ICIRCA.2018.8597221.

[11] Petrenko, A.S., Petrenko, S.A., Makoveichuk, K.A., Chetyrbok, P.V.: Protection Model of PCS of Subway from Attacks Type «Wanna cry», «Petya» and «Bad rabbit» IoT. In: Proceedings of the 2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus 2018). IEEE, pp. 945 – 949 (2018). DOI: 10.1109/EIConRus.2018.8317245.

[12] Priya R. L., Lifna C. S., Dhanamma J., Anooja J. Rational Unified Treatment for Web application Vulnerability Assessment. In: 2014 International Conference on Circuits, Systems, Communication and Information Technology Applications (CSCITA), IEEE, 2014, 14395120. DOI: 10.1109/CSCITA.2014.6839283.

[13] Rafique S., Humayun M., Gul Z., Abbas A., Javed H. Systematic Review of Web Application Security Vulnerabilities Detection Methods, Journal of Computer and Communications, 2015. V. 3, No 9, pp. 28-40. DOI: 10.4236/jcc.2015.39004.

[14] Wang B., Liu L., Li F., Zhang J., Chen T., Zou Z. Research on Web Application Security Vulnerability Scanning Technology. In: 2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), IEEE, 2019, 19359942, DOI: 10.1109/IAEAC47372.2019.8997964.

[15] Yadav D., Gupta D., Singh D., Kumar D., Sharma U. Vulnerabilities and Security of Web Applications. In: 2018 4th International Conference on Computing Communication and Automation (ICCCA), IEEE, 2018, 18868543. DOI: 10.1109/CCAA.2018.8777558.

[16] Barabanov A.V., Markov A.S., Tsirlov V.L. Methodological Framework for Analysis and Synthesis of a Set of Secure Software Development Controls, Journal of Theoretical and Applied Information Technology, 2016, vol. 88, No 1, pp. 77-88.

[17] Howard M., Lipner S. The Security Development Lifecycle: A Process for Developing Demonstrably More Secure Software. Microsoft Press, 2006. 352 p.

[18] Dorofeev A.V., Markov A.S., Rautkin Y.V. Ethical Hacking Training. In: CEUR Workshop Proceedings, 2019, Vol-2522, pp. 47-56.

[19] Markov A., Barabanov A., Tsirlov V. Models for Testing Modifiable Systems. In Book: Probabilistic Modeling in System Engineering, by ed. A.Kostogryzov. IntechOpen, 2018, Chapter 7, pp. 147-168. DOI: 10.5772/intechopen.75126.

[20] Poltavtseva, M.A., Pechenkin, A.I. Intelligent data analysis in decision support systems for penetration tests. In: (2017) Automatic Control and Computer Sciences, 51 (8), pp. 985-991. DOI: 10.3103/S014641161708017X.

[21] Barabanov A.V., Markov A.S., Tsirlov V.L. Statistics of Software Vulnerability Detection in Certification Testing. Journal of Physics: Conference Series. 2018. V. 1015. P. 042033. DOI :10.1088/1742-6596/1015/4/042033.