

Web Platform for Comprehensive Penetration Testing

Willi Lazarov, Zdenek Martinasek

Brno University of Technology, FEEC, Czechia

E-mail: xlazar15@vut.cz, martinasek@vut.cz

Abstract—This paper presents the design, development, and implementation of a web platform embedded in the proposed Highly Scalable Model with the main purpose of increasing the effect of penetration testing to such an extent that the time, complexity, and work required to successfully complete the entire test will be considerably lower than using currently available tools, together with greater coverage of the testing area.

Keywords—penetration testing, vulnerability assessment, web application, modularity

1. INTRODUCTION

In today's digital world, more and more organizations and users use online services. Accordingly, the number of servers hosting these services and technologies used by them is also increasing. Nowadays, servers and applications are often the targets of cyberattacks.¹ If successful, such an attack can cause damage from service disruption to theft or loss of sensitive data. In the same way, other components falling under the area of information and communication technologies are attacked, thus increasing the number of risks and the scope of the impact even further. An example of a critical impact is a cyberattack on a hospital, which can put human lives at risk [1]. The growth of threats in cyberspace has prompted the development of many penetration testing tools, which as one of the preventions against cyberattacks reveals the security risks of the tested targets. Penetration testing is a complex process and there is no complete solution on the market that would allow its full management and semi-automation through a responsive user interface together with team collaboration, individual project management, and more sophisticated visualization of the tested environment.

1.1. State of the Art

The penetration testing process may vary according to different methodologies and tested environments. For this reason, it is problematic to provide a structure for all types of tests with a single solution, because the scheme cannot be simply generalized [3]. An example is the comparison of penetration testing between network infrastructure and a web application. In the case of a computer network, the tester will work mainly with the type, subnet mask, and default gateway of the tested network. During penetration testing, the tester can discover network or endpoint devices (e.g., router, server, etc.) and different running services. On the other hand, testing a web application can expose vulnerabilities related to programming languages, frameworks, and other websites with various user inputs. Considering each possible test target, more than thousands of different test sections would need to be developed to cover the entire area. Development on this scale would likely require a large amount of time, money, and human resources. In addition, future modifications of such a solution are very problematic due to the large-scale environment. Another disadvantage is the tendency to produce a duplicate and less readable source code. Also, the process of testing and optimizing the entire solution is more demanding. All these disadvantages and shortcomings make the gradual development of such a complex tool very difficult, and therefore the problem has to be solved in a completely different way.

1.2. Contribution

The paper discusses a web-based platform as a solution to the mentioned problem with a new approach compared to available tools that are commonly used in penetration testing. The main goal was to design, develop and deploy a comprehensive environment for effective and complete penetration testing that would allow testers to collaborate with each other as well as with the developers, system administrators, managers, and other relevant persons of the tested subject. During the design and development of the platform, emphasis was placed on high modularity, scalability, and optimization of all parts of the solution. The focus was also on user experience and usability of the client part.

¹In fact, an estimated 17,6 million websites could be infected with malicious code until 2019, based on SiteLock's analysis. Compared to the previous year, this is a 59% increase with an average of 62 attacks per day [2].

2. PROPOSAL OF A HIGHLY SCALABLE MODEL

The solution to the problem mentioned in the previous chapter is the proposal of a Highly Scalable Model, which is intended for the development and subsequent efficient expansion of the platform. The actual design of the model is shown in Figure 1 and the following text explains its main parts in a general way. The model consists of three main sides – data, application, and client. The schema for using the platform is formed in the data side, where the input data is defined and the relationships between them are created. The output of this process is structured data that represents the schema of the entire platform. The structured data then determines how the dynamic data will be represented, which will enter the database only when the platform is used in the production environment and their flow is therefore completely separated from the initial phase of creating the main structure. Both of these parts are located in the database with which the application side of the proposed model communicates.

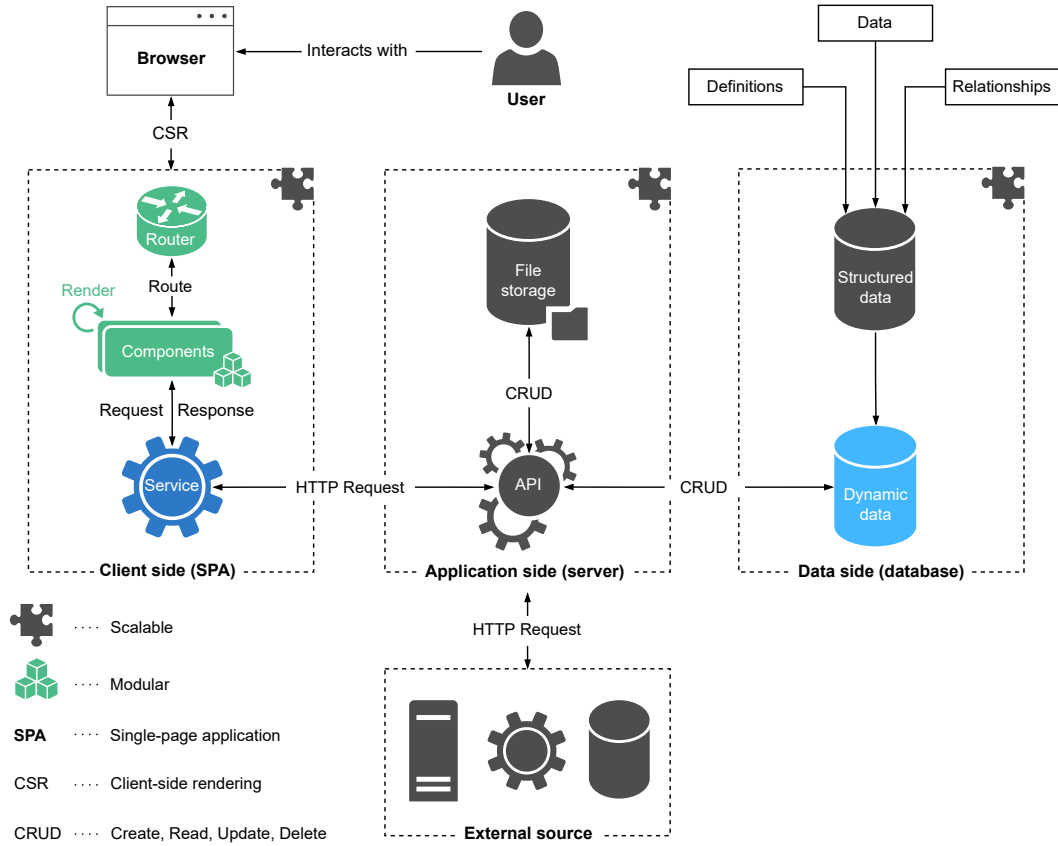


Figure 1: Highly Scalable Model

The second part of the proposed model is the application side, which consists of one or more servers that represent the model logic and communicate with the client and data side through the API (Application Programming Interface). The application server handles user authentication, authorization of all requests, and can also contain file storage. The main part of the application side is an API that returns structured data from the database for the client part based on an HTTP request or processes the corresponding dynamic data. The application server can be additionally extended with another API for communication with an external source, which can be a database, cloud storage, or another remote service. The logical part of the model can be further subdivided, following the microservices architecture, into multiple independent services with their interfaces based on different technologies [4].

The third part of the model is the client-side containing the SPA (Single-page application), with which the user interacts and sends requests to the application server. Once the user is authenticated, the application server sends the SPA, which is then loaded into the web browser. The SPA works on the principle of CSR (Client-side rendering) and therefore makes all changes on the client-side and requests only new data from the server. In the proposed model the SPA is highly modular. Based on client requests or structured data, it builds itself from its components, which then request the necessary dynamic data.

3. IMPLEMENTATION OF A WEB PLATFORM

According to the proposed model, the main goal was to implement a web platform to support penetration testing. For this purpose, a scheme for different possible test areas and targets is defined in the initial phase of the development. This process is very fast and it is possible to create a structure for different types of tests, environments, and other related parts of penetration testing in a short time. Dynamic data created or obtained during testing are then represented by nodes related to the structured data. The entire data side is managed by the relational database system MySQL. The application part is deployed on the Apache HTTP Server, which runs on Ubuntu operating system.² The main function of this part is handled by an API that communicates with SPA, server for automated penetration testing, data side, and authentication server. The entire application interface was developed using the Laravel PHP framework. On the client-side, the SPA is implemented as a modular Vue.js application programmed in TypeScript, which also ensures strict type checking of the source code. SPA handles client requests and forwards them to the application server for processing through its services.

The implemented platform communicates with other parts of the production environment. Users interact with the web platform through the web browser, into which the SPA is automatically loaded. Before the AS (Application Server) provides the SPA to the client-side, the user must authenticate himself against the UMS (User Management Server). Once the user is successfully authenticated, he is returned to the AS with the authentication token. The AS verifies the token against the UMS, and if the token is valid, the SPA is provided to the user, which is then loaded into his web browser. From now on, all further requests are sent by the SPA based on user interaction and other application events. There is only one UMS to authenticate all deployed application servers, which can be hosted on the cloud, custom servers, or client machines. The scheme of the production environment is shown in Figure 2.

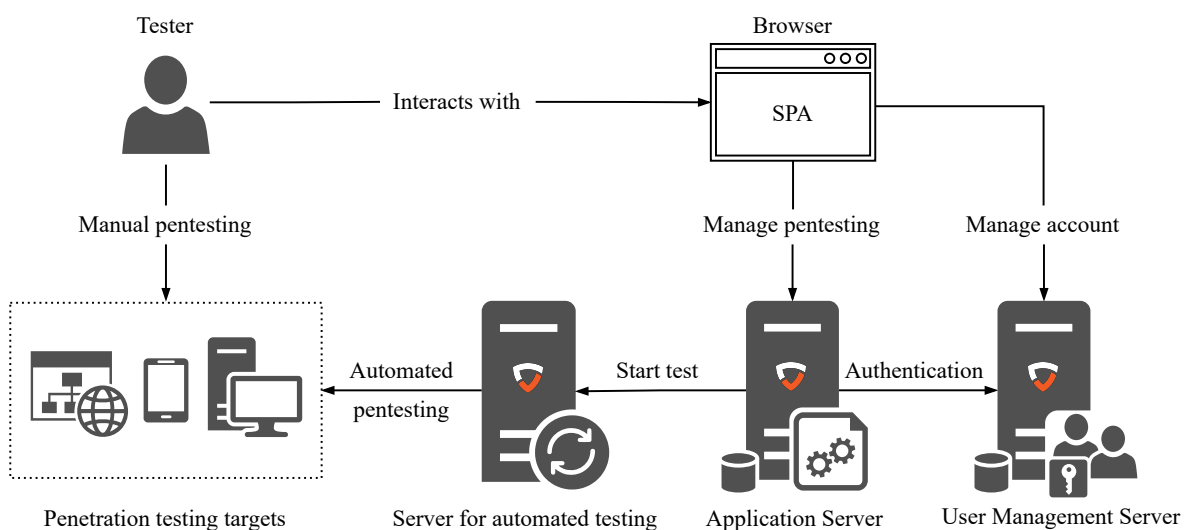


Figure 2: Web platform in production environment

An important part of the production environment is the SAT (Server for automated testing). Users either perform the penetration testing manually and upload the results to the web platform, or run automated tests against the specified target, which are executed by the SAT according to their parameters and the results are then returned to the application server for further processing. The target of a manual or automated test can be almost anything that is accessible from the Internet, and the SAT has a suitable tool for testing the specified target. The web platform, in order to support penetration testing to the maximum extent, allows testing to be managed not only by testers but by other users, such as developers, who can, for example, fix the found vulnerabilities and testers then re-verify their presence. The key final part is generating the report of the entire testing, which is very time-consuming when using a text editor. All this can be done within the web platform without the need to use additional tools.

² It is also possible to use another web server, such as Nginx running on a Linux distribution or Microsoft IIS (Internet Information Services). The only condition for full functionality is the presence of an interpreter and a database system.

Figure 3 shows the user interface that the SPA has built for the client based on the received data. On the left side of the picture is a tree of all tested nodes and their descendants. For each node, a dashboard with relevant sections (e.g. tables, tests, vulnerabilities, attachments, etc.) is built at the moment of selection. In the case of this example, SPA built a web page node with a table of tests.

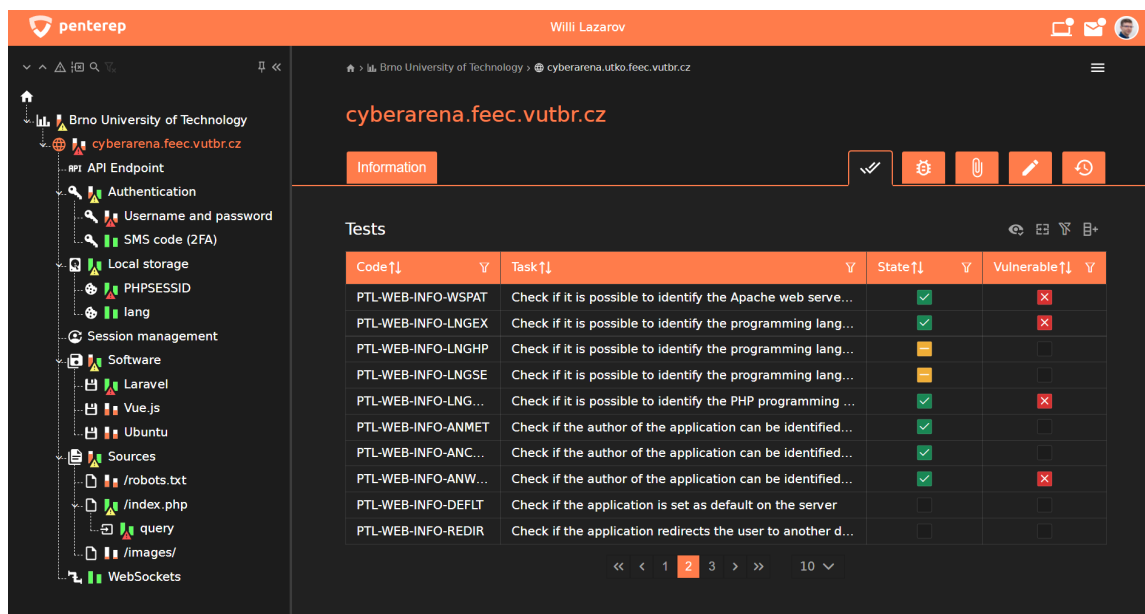


Figure 3: Single-page application user interface

4. CONCLUSION

Based on the demanding requirements on the required solution, a Highly Scalable Model was proposed, which proved to be fully suitable for the development of a web platform whose purpose is comprehensive penetration testing. From the beginning, the platform was developed with a primary focus on modularity, scalability, and optimization. Thanks to its embedding in the proposed model, it was possible to create a considerably large environment in the limited time of one year, which is gradually being extended by other parts and its possibilities of use are thus increasing over time. The main contribution of the platform is the unification of all penetration tester tasks into one place, automation of testing processes, and team collaboration between testers and other users. The result is a solution that improves the effect of penetration testing to such an extent that the time, complexity, and work required to successfully complete the entire test will be significantly lower than using the currently available tools.

ACKNOWLEDGMENT

The research described in this paper was financially supported by the Technology Agency of the Czech Republic, project No. TJ04000456.

REFERENCES

- [1] S. J. Choi, M. E. Johnson and C. U. Lehmann, "Data breach remediation efforts and their implications for hospital quality," *Health Services Research*, 54(5), 971-980, 2019, doi: [10.1111/1475-6773.13203](https://doi.org/10.1111/1475-6773.13203).
- [2] "Cybersecurity And Website Report," *SiteLock*. [Online]. Available: <https://www.sitelock.com/resources/security-report/>. [Accessed Feb. 10, 2022].
- [3] M. Denis, C. Zena and T. Hayajneh, "Penetration testing: Concepts, attack methods, and defense strategies," 2016 IEEE Long Island Systems, Applications and Technology Conference (LISAT), 2016, pp. 1–6, doi: [10.1109/LISAT.2016.7494156](https://doi.org/10.1109/LISAT.2016.7494156).
- [4] A. Sill, "The Design and Architecture of Microservices," *IEEE Cloud Computing*, 3(5), 76-80, 2016, doi: [10.1109/MCC.2016.111](https://doi.org/10.1109/MCC.2016.111).