WEB APPLICATION PENETRATION TESTING COURSE OVERVIEW

The aim of the course is to prepare students to perform a full penetration test of a web application. After completing the course, participants will know the basic topics related to the entire process of a penetration test of a web application, both from the theoretical and practical side: starting from the contract and objectives of the test - Rules of Engagement, through finding and exploiting vulnerabilities, to writing a report and retesting.

A vulnerable STMAcademy Trade web application has been prepared for the course, where students will be able to test their acquired knowledge in practice by attacking a real web application in a safe, isolated environment.

1 Introduction to penetration testing

2 HTTP recap

3 Enumeration & Information Gathering

4 Client-side vulnerabilities

5 Server-side vulnerabilities

In the INTRODUCTION TO PENTESTING part, participants will be introduced to the theoretical side of penetration testing: they will learn what it is and familiarize themselves with fundamental terms and the most well-known methodologies. The course at the very beginning provides the necessary theoretical knowledge on how to conduct a penetration test and write a report.

Penetration testing fundamentals – who, why, and how pentests are performed?

Penetration test lifecycle – from the very beginning (defining the time frames, scope, and other important aspects in the Rules of Engagement) to report writing and retests.

Pentests types – basic terms related to penetration testing: vulnerability, exploitation, blackbox/graybox/whitebox, etc.

Pentesting methodologies – worldwide used, most popular methodologies in pentesting.

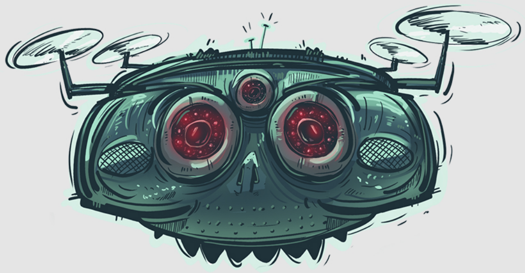
Report writing – the importance of the report, principles of writing a good report.

Before diving into security topics, HTTP RECAP will provide crucial information about the Hypertext Transfer Protocol from the inside out. Knowing the basics of how web applications work, the client-server architecture, or the meaning of security headers is a necessary foundation for gaining knowledge and skills in web application security testing.

HTTP Fundamentals – HTTP protocol basics, client-server architecture, encoding, SOP.

Modern web application architecture – from SPA (Single Page Application) to complex multi-server architectures to cloud-based web applications.

Security headers & Cookies – introduction to basic concepts of web application security.



Every penetration test begins with gathering and systematizing information. In ENUMERATION & INFORMATION GATHERING topic, students will learn techniques for gathering information, as well as tools for passive and active enumeration. The course also tries to teach the desired mindset of a pentester during the first contact with a web application, oriented towards effective enumeration. Proxy tools (Burp Suite, OWASP ZAP) will also be presented at this stage.

The first interaction with the application – the very first requests and responses can contain valuable information about the application under test.

Manual enumeration tools - many tools can be used to gather information; one of the most basic (and most useful) of these is the web browser.

Passive & Active enumeration - Differences between active and passive techniques of acquiring information; presentation and use of selected tools.

Infrastructure reconnaissance - every web application is hosted on some server that is an integral part of it; basic web enumeration techniques.

In the third section, CLIENT-SIDE VULNERABILITIES such as Cross-Site Scripting, HTML Injection, and Cross-Site Request Forgery will be presented. These vulnerabilities still found in web applications often allow attackers to interact with the application as a user victim, significantly affecting the confidentiality, integrity, and availability of both user data and the security of the application itself.

In addition, students will become familiar with the issues of some of the key aspects of today's web applications: authentication, authorization, and session management. Participants will learn about the most common access control flaws. They will also learn how web applications should operate on passwords and see how to crack acquired hashes.

HTML Injection – an attack that allows the injection of an HTML tag, which can lead to website defacement and - sometimes - also to a more serious XSS vulnerability.

Cross-Side Scripting – presentation of the cause of the vulnerability, types of XSS, and its wide impact (e.g., cookie stealing).

Cross-Site Request Forgery – the impact and CSRF scenarios; problematic nature of CSRF exploitation in today's browsers.

Authentication, Authorization, and Session management – basic concepts and potential threats related to authentication and authorization, as well as ways to bypass access control mechanisms and improper session management examples.

Password security – principles of handling passwords in web applications; hashes cracking.

In the SERVER-SIDE VULNERABILITIES topic, students will learn about vulnerabilities that target the server itself. These vulnerabilities can be extremely dangerous because they do not require user interaction. A successful attack on the server has a direct impact on the confidentiality, integrity, and availability of all processed data.

SQL Injection – the possibility of injecting into an SQL query often leads to the leakage of redundant information and, in some circumstances, even to the execution of code on the database server.

Server-Side Request Forgery – forcing the server to execute the request often enables enumeration of internal application resources and sometimes even file read.

Server-Side Template Injection – exploitation of an unsecured template engine in an application allows an attacker to gain excessive information and, in some cases, can lead to remote code execution.

Server-Side Include Injection – vulnerability often leading to the execution of commands on an improperly configured web server; rarely found in modern applications.

Path Traversal - allows an attacker to gain access to files they sho uld not be able to see, potentially leading to the leak of sensitive data.

Local & Remote File Inclusion – unsanitized input can lead to the inclusion of another file from the server or an external location, creating the risk of data leakage or even code execution.

File upload vulnerabilities – the ability to upload files to the server is an extremely sensitive point in the application, as this is where you interact directly with the file system on the webserver.