Testing an organization's information system for unauthorized access

Ivan Tyshyk[1,†] and Hennadii Hulak[2,3,*,†]

[1] Lviv Polytechnic National University, 12 Stepana Bandery str., 79013 Lviv, Ukraine
[2] Borys Grinchenko Kyiv Metropolitan University, 18/2 Bulvarno-Kudryavska str., 04053 Kyiv, Ukraine
[3] Institute of Mathematical Machines and Systems Problems of the National Academy of Sciences of Ukraine, 42 Ac. Glushkov ave., 03680 Kyiv, Ukraine

Abstract

Security assessment of information systems is crucial for identifying protection issues in their components and determining potential attack vectors. Penetration testing is conducted by simulating what a real attacker could do against the target system and offers an effective way of obtaining such information. This approach provides an unbiased view of the actual level of protection against attacks and demonstrates the effectiveness of security solutions for the company's network infrastructure in practice. Penetration testing involves evaluating software or network infrastructure for vulnerabilities and attempting to exploit them for unauthorized access, bypassing, or damaging security components. These vulnerabilities may arise from misconfigurations of communication equipment, unsecured application code, network architecture design flaws, or the disclosure of confidential information. As a result of the testing, a comprehensive report is generated, explaining each vulnerability or chain of vulnerabilities exploited to gain unauthorized access to the target, detailing the steps taken to exploit them, and providing mitigation recommendations. Each identified vulnerability is assigned a risk rating, which is used to prioritize tasks for improving the security of the tested system. The paper examines methods for conducting penetration testing of an organization's corporate network infrastructure for unauthorized access. A simulation of information systems testing for unauthorized access was performed, and potential attacks following such access were illustrated. The most common methods of exploiting potential vulnerabilities in corporate networks are presented.

Keywords

information system, corporate network, penetration testing, virtual machine, web application, unauthorized access, network security tool, Kali Linux

1. Introduction

The rapid growth in the popularity of internet technologies is accompanied by an increase in serious threats to the disclosure of personal data, critical corporate resources, state secrets, and more. Every day, hackers and other malicious actors threaten network information resources, attempting to gain access to them through specialized attacks. These attacks are becoming increasingly sophisticated and easier to execute. Two main factors contribute to this.

Firstly, the widespread penetration of the internet. Today, billions of various devices are connected to the network, increasing the likelihood of hackers accessing these devices and their associated computer networks through their vulnerabilities. Moreover, the global spread of the internet enables hackers to exchange information on a global scale. Secondly, the widespread availability of user-friendly operating systems and development environments. This factor significantly reduces the knowledge requirements for attackers. In the past, hackers needed strong programming skills to create and distribute malicious programs. Now, to gain access to a hacking tool, one only needs to know the IP address of the desired site, and a few mouse clicks are enough to carry out an attack.

Information security breaches in corporate computer networks can be caused by human factors, vulnerabilities in the communication equipment's operating environment, server operating systems, and local workstations, as well as the possibility of executing remote attacks, especially if the corporate network is distributed and connected to public data transmission networks.

From a security perspective, distributed systems are primarily vulnerable to remote attacks, as the components of distributed systems typically use open data transmission channels. An attacker can not only perform passive eavesdropping during data transmission but also modify the traffic. While such active tampering with traffic can often be detected, passive eavesdropping is nearly impossible to identify.

Vulnerability assessment is a key task in ensuring the security of information systems, which involves regular testing. Currently, Linux distributions such as Kali Linux,

CPITS-II 2024: Workshop on Cybersecurity Providing in Information and Telecommunication Systems II, October 26, 2024, Kyiv, Ukraine
* Corresponding author.
† These authors contributed equally.
ivan.y.tyshyk@lpnu.ua (I. Tyshyk);
h.hulak@kubg.edu.ua (H. Hulak)
0000-0003-1465-5342 (I. Tyshyk);
0000-0001-9131-9233 (H. Hulak)

BackBox, Parrot Security OS, and several other tools are widely used for penetration testing of information systems [1]. However, given the diversity of unauthorized access methods and the type of object being tested, an appropriate testing methodology is required that would provide a comprehensive evaluation of the security level of the target system. Problem statement: In light of the above, the challenge arises in selecting an effective methodology for testing an organization's information system for unauthorized access to its resources. This includes conducting thorough information gathering, which will help the organization assess the current level of security of their information system, identify vulnerabilities, prioritize them based on criticality, and develop an action plan to protect the system from future cyberattacks.

2. Methods for testing unauthorized access to an organization's information system

To ensure the best testing results, regardless of the penetration tests used, the tester must follow a standardized testing methodology. The following popular testing methods will be discussed:

- Open Web Application Security Project (OWASP) [3].
- Payment Card Industry Data Security Standard (PCI DSS) [4].
- Standard Penetration Testing Execution [5].
- National Institute of Standards and Technology Special Publication (NIST SP 800-115) [6].
- Open Source Security Testing Methodology Manual (OSSTMM) [7].

2.1. Open web application security project

The Open Web Application Security Project (OWASP) is a project that brings together open-source software developers. The members of this community create programs designed to protect web applications and web services. All applications are developed based on the experience of combating malicious programs that target web services and web applications. OWASP serves as a starting point for system architects, developers, vendors, consumers, and security professionals—essentially all specialists involved in the design, development, deployment, and security testing of web services and web applications. In other words, OWASP aims to help create more secure web applications and web services.

The main advantage of the OWASP Testing Guide is that the test results provide a comprehensive description of all threats. The OWASP Testing Guide identifies all the risks that may affect the system and applications and evaluates the likelihood of their occurrence. Using the threats described in OWASP, it is possible to determine an overall risk assessment from the conducted tests and provide appropriate recommendations for mitigating these vulnerabilities [8].

The OWASP Testing Guide primarily focuses on the following areas: Methods and tools for testing web applications; Information gathering; Authentication testing; Business logic testing; Test data; Denial-of-service attack testing; Session management verification; Web services testing; AJAX testing; Risk assessment; Threat likelihood. These threats pose serious risks to cloud computing security [9]. Denial-of-service attacks can disrupt access to cloud services, misconfiguration of security can open the door to attackers, and cloud malware attacks threaten data privacy and integrity. This may allow an attacker to use the associated resources for their purposes or to steal or manipulate data stored in the cloud. All these threats require important monitoring and the provision of appropriate security measures to protect cloud services and user data [2].

2.2. Payment card data security standard

This section compiles regulations for companies that comply with PCI (Payment Card Industry) requirements. The guide contains standards not only for PCI v3.2 but was developed by the PCI Security Standards Council, outlining penetration testing methods within vulnerability management programs. The PCI Data Security Standard (PCI DSS) version 3.2 was released in April 2016 by the Payment Card Industry Security Standards Council (PCI SSC). After the update, the requirements were clarified, with additional guidelines and seven new requirements introduced.

To address issues related to breaches of cardholder personal data confidentiality and protect against existing exploits, various changes were included in PCI DSS v3.2, most of which pertain to service providers. These changes added new requirements for penetration testing, which mandate that segmentation testing for service providers be performed at least every six months or after any significant changes in segmentation controls/methods. Additionally, this standard contains several requirements obliging service providers to continuously monitor and maintain critical security controls throughout the year.

2.3. Standard penetration testing execution

The penetration testing standard consists of seven main sections. They cover all the requirements, conditions, and methods for conducting penetration tests: from reconnaissance to attempts at conducting pen tests; stages of information gathering and threat modeling, where testers work covertly to achieve the best possible testing results; stages of vulnerability assessment, exploitation, and post-exploitation, where the practical security knowledge of testers is combined with the data gathered during penetration tests; and finally, the reporting phase, where all information is presented in a format understandable to the client.

Currently, the first version is in effect, in which all standard elements have been tested under real-world conditions and approved. The second version is under development, where all requirements will be detailed, clarified, and improved.

Since the plan for each penetration test is developed individually, various tests can be applied: from web application testing to black-box testing methods. With this plan, the expected level of complexity of a particular investigation can be immediately determined and applied in the volumes and areas deemed necessary by the

organization. Preliminary research results can be seen in the section related to intelligence gathering.

Below are the main sections of the discussed standard, which form the basis for conducting penetration tests: Pre-engagement agreement; Intelligence gathering; Threat modeling; Vulnerability analysis; Exploitation; Post-exploitation; and Reporting.

2.4. National institute of standards and technology special publication

The National Institute of Standards and Technology Special Publication (NIST SP 800-115) is a technical guide for information security testing and assessment. The publication was prepared by the Information Technology Laboratory (ITL) at NIST.

The guide defines security assessment as the process of determining how effectively an organization meets specific security requirements. Upon reviewing the guide, you will find that it contains a large amount of information for conducting tests. Although the document is rarely updated, it remains relevant and can serve as a reference for building a testing methodology.

This guide provides practical recommendations for developing, implementing, and maintaining technical information security tests, as well as processes and procedures for evaluations, covering the key elements of technical security testing and evaluation. These recommendations can be used for several practical tasks, such as vulnerability assessments in a system or network and checking compliance with policies or other requirements.

NIST 800-115 provides a comprehensive framework for penetration testing, ensuring that a penetration testing program meets the necessary guidelines.

2.5. Open source security testing methodology manual

The Open Source Security Testing Methodology Manual OSSTMM is a document that is quite complex to read and understand, but it contains a vast amount of relevant and highly detailed information on security. It is also the most well-known security guide globally, with approximately half a million downloads every month. The reason for its popularity is that its guidelines are about a decade ahead of all other documents in the security industry. The purpose of the OSSTMM is to advance the standards for Internet security testing. This document is designed to provide the most detailed core framework for testing, which in turn ensures thorough penetration testing. Regardless of other organizational specifics, such as the service provider's corporate profile for penetration testing, this testing allows the client to verify the level of technical evaluation.

3. Methods for testing unauthorized access to an organization's information system

Kali Linux includes various types of programs specifically aimed at ensuring computer network security, although it is possible to install available graphical applications, text editors, image viewers, and other software. However, installing such software is not recommended due to the risk of compromising system security and anonymity during testing. This OS is in continuous development, and future updates are expected to expand its functionality and toolset, as well as improve its existing capabilities.

It can be said that Kali Linux will remain a key tool in cybersecurity going forward, as it continues to evolve, adapting to new trends while maintaining its relevance and effectiveness in the field of penetration testing.

Currently, other distributions offer similar functionality and features to Kali Linux. One of the most popular is Parrot Security OS [10]. It is also based on Debian and includes a wide range of tools for penetration testing.

Parrot Security OS includes over 700 security testing tools and utilizes more than 700 vulnerabilities. It features built-in tools for analyzing anonymous networks like Tor and I2P, tools for wireless network analysis, and intrusion detection tools, which facilitate the identification of potential attack scenarios. It offers a wide range of tools for network application security analysis and vulnerability testing. Additionally, this distribution supports the use of virtual machines, enhancing protection against network attacks and enabling the testing of various threat scenarios on information systems.

The mentioned features of Parrot Security help testers and cybersecurity professionals to assess and improve the security of information systems and data transmission networks, as well as enhance user credential protection. However, this distribution is more focused on anonymity compared to Kali Linux, which primarily emphasizes security and testing. Given this, the following sections will focus on network tools included in the Kali Linux distribution [11, 12].

One popular tool used by cybersecurity specialists, data analysts, and other professionals for gathering and analyzing open-source intelligence is Maltego [13].

The presented tool allows users to create custom entities, enabling them to represent any type of information in addition to the basic entity types that are part of the software. Its primary goal is to analyze real-world relationships (social networks, OSINT APIs, proprietary private data, and computer network nodes) between individuals, groups, web pages, domains, networks, and internet infrastructure [14]. Maltego expands its data range through integrations with various data processing partners. Data sources include DNS records, Whois, search engines, social network services, different APIs, and various metadata. Maltego can be used during the information-gathering phase for all security-related tasks.

One of the popular tools among cybersecurity professionals is Nmap, which is designed for customizable scanning of IP networks with any number of targets and determining the status of network objects (ports and corresponding services). Initially developed for UNIX systems, Nmap is now available for many operating systems. Nmap is built for quickly scanning large networks, though it also works well with single targets. It uses IP packets in unique ways to discover which hosts are available on the network, what services (application names and versions) they offer, what operating systems (and OS

versions) they are running, what types of packet filters/firewalls are in use, and many other characteristics. While Nmap is commonly used for security auditing, many network and system administrators find it useful for routine tasks such as network inventory, managing service upgrade schedules, and monitoring host or service uptime [15, 16].

According to the developer of Nmap, some of the most important features of this scanner include:

- Dynamic Timing Calculations: Some scanners require you to specify the time delay between sending packets. Nmap attempts to determine the best delay for you. It also tracks packet retransmissions to adjust the delay during the scan. For users with root access, the primary method for determining the delay is through the built-in ping function. For non-root users, it's based on the number of attempts to connect to a closed port on the target. It can also select a reasonable default value.
- Packet Retransmission: Some scanners simply send out all the query packets and collect the responses. However, this can result in false positives or negatives if packets are dropped. This is especially important for "negative" scan types like UDP and FIN, where the goal is to find ports that do **not** respond. In most cases, Nmap implements a configurable number of retransmissions for unresponsive ports.
- Parallel Port Scanning: Some scanners scan ports linearly, one at a time, until all 65,535 ports are scanned. This works for TCP on very fast local networks, but it's not acceptable on wide-area networks like the Internet. Nmap uses non-blocking I/O and parallel scanning in all TCP and UDP modes. The number of parallel scans can be adjusted with the `-M` (Max sockets) option.
- Flexible Port Specification: I don't always want to scan all 65,535 ports. Additionally, scanners that only allow scanning ports from 1 to N sometimes don't meet my needs. The `-p` option allows you to specify an arbitrary number of ports and ranges to scan. For example, `-p 21-25,80,113,60000-` does exactly what you expect (the trailing dash means up to 65,536, and the leading dash means starting from 1). You can also use the `-F` (fast) option to scan all ports registered in your `/etc/services` file (similar to **strobe**).
- Flexible Target Specification: I often want to scan more than one host, and I certainly don't want to manually list each host in a large network scan. Anything that isn't an option or option argument in Nmap is treated as a target host. As mentioned earlier, you can append a file/mask to a hostname or IP address to scan all hosts with the same initial `<mask>` bits of the 32-bit IP address.
- Unreachable Host Detection: Some scanners allow scanning large networks, but they waste a lot of time scanning all 65,535 ports on dead hosts! By default, Nmap checks each host to ensure it's alive before spending time scanning it. It also supports handling hosts that appear unreachable based on unusual scan port errors. Nmap is also tolerant of users accidentally scanning network or broadcast addresses, and similar edge cases.
- Detection of your IP address: For some reason, many scanners require you to input your IP address as one of the parameters. Nmap, however, attempts to detect your IP address during the ping phase. It uses the address that receives an echo reply, as this is typically the interface through which traffic should be routed. If it can't do this (for instance, if host pinging is disabled), Nmap tries to detect your primary interface and uses that address. You can also specify an IP address directly using the `-S` option.

Another popular penetration testing platform is Metasploit, which allows identifying, exploiting, and verifying vulnerabilities in an information system. The platform provides infrastructure, content, and tools for conducting penetration tests and offers extensive security auditing [17]. The most well-known tool within it is the open-source Metasploit Framework, an application used for developing and executing exploit code against a remote target machine [18].

This framework has become a fundamental tool for developing exploits and addressing vulnerabilities. Before Metasploit, penetration testers had to perform all checks manually, using various tools that might or might not support the testing platform, and manually write their code to deploy within networks. Remote testing was an extraordinary task, limiting security professionals to working with local companies, while organizations had to spend significant resources on in-house IT consultants or security specialists [19].

The modern version of Metasploit contains over 1,677 exploits for more than 25 platforms, including Android, PHP, Python, Java, Cisco, and others. The framework also includes around 500 payloads [20].

One of the most popular vulnerability scanners on the market is the Nessus Vulnerability Scanner, which has become a standard among vulnerability scanners. It originally started as an open-source project but was later acquired by Tenable and is now a commercial product (Professional version). Despite this, Nessus Scanner still offers a "Home" version, which is distributed for free but limited to **16 IP addresses**. This version was used for simulating penetration testing for unauthorized access within the organization [21].

Vulnerability scanners have a certain limitation: they cannot detect **0-day vulnerabilities**. Similar to antivirus software, their databases must be updated daily to ensure effective performance. Recently, even the **U.S. government** has started using it for vulnerability scanning. Almost every federal office and U.S. military base worldwide now uses Nessus [22].

Another popular platform for conducting security audits of web applications is the Burp Suite Scanner, which includes tools for mapping the web application, searching for files and directories, modifying requests, fuzzing web applications, password brute-forcing, and much more. There is also an extension store, the BApp Store, which enhances the functionality of specific applications. Notably, the latest release includes a Mobile Assistant for testing the security of iOS mobile applications [23].

Burp Suite is an integrated platform designed for auditing web applications, both manually and automatically. It features an intuitive interface with specially designed tabs to improve and speed up the attack process. The tool itself acts as a proxy mechanism that intercepts and processes all incoming browser requests. It also allows for installing a **Burp certificate** to analyze **HTTPS** connections.

Objective of the paper: To simulate testing of an organization's corporate network for unauthorized access using the tools from the **Kali Linux** distribution, to explore tools for different testing stages, and to evaluate the collected vulnerability data. The information gathered by the tester will help companies determine the current security level of their information systems, identify vulnerabilities, prioritize them by criticality, and create an action plan for responding to future cyberattacks.

The output data of Nmap is a list of scanned targets with additional information for each, depending on the specified options. The key information is the "port state table". This table includes the port number, protocol, service name, and state. The state can be open, filtered, closed, or unfiltered. An open port state means the target machine is ready to establish a connection or receive packets on that port. Filtered indicates that a firewall, network filter, or another network obstacle is blocking the port, and Nmap cannot determine if the port is open or closed. Closed ports are not associated with any application, so they may be opened at any time. Ports are considered unfiltered when they respond to Nmap requests, but Nmap cannot determine whether they are open or closed. Nmap reports "open/filtered" or "closed/filtered" combinations when it cannot determine which of the two states describes the port. This table may also provide details about software versions if requested. When performing IP protocol scanning (-sO), Nmap provides information about supported IP protocols rather than open ports. In addition to the port state table, Nmap may provide further information about targets: resolved DNS names, guesses about the operating system in use, device types, and MAC addresses [16].

Due to its wide range of applications and accessible open-source code, Metasploit is used by a diverse group of people, from cybersecurity professionals to hackers. Metasploit is valuable for anyone needing a simple-to-install, reliable tool that performs its job regardless of platform or language. This software is popular among hackers and widely available, motivating security professionals to study the Metasploit platform even if they do not use it themselves.

The modern version of Metasploit includes over 1,677 exploits for more than 25 platforms, including Android, PHP, Python, Java, Cisco, and others. The framework also contains around 500 payloads, which include the following [20]:

- Command shell payloads—allow users to execute scripts or arbitrary commands on the host.
- Dynamic payloads—enable testers to generate unique payloads to bypass antivirus software.
- Meterpreter payloads—allow users to intercept control of the device's display through the video memory controller, capture sessions, and upload or download files.
- Static payloads—facilitate port forwarding and data exchange between networks.

As a "hacker", after scanning, you receive a comprehensive list of vulnerabilities, for which you only need to find exploits. Unfortunately, vulnerability scanners are quite "noisy", and vigilant administrators can detect their activity. However, not all organizations have such administrators.

It's important to note a few key points about vulnerability scanners. They cannot detect 0-day vulnerabilities. Like antivirus products, their databases must be updated daily to remain effective.

Recently, even the U.S. government started using Nessus for vulnerability scanning. Almost every federal office and military base worldwide now employs Nessus.

The software is capable of detecting the most common types of vulnerabilities, such as (Nessus Vulnerability Scanner) [24]:

- The presence of vulnerable service or domain versions.
- Configuration errors (e.g., lack of required authentication on an SMTP server).
- Presence of default, empty, or weak passwords.

The program has a client-server architecture, which greatly expands scanning capabilities. Different editions offer varying features tailored to different clients.

3.1. Port scanning with Nmap

Having identified the target IP range with passive information gathering as well as the secmaniac.net target IP address, we can begin to scan for open ports on the target by port scanning, a process whereby we meticulously connect to ports on the remote host to identify those that are active. (Obviously, in a larger enterprise, we would have multiple IP ranges and things to attack instead of only one IP.) Nmap is, by far, the most popular port scanning tool. It integrates with Metasploit quite elegantly, storing scan output in a database backend for later use. Nmap lets you scan hosts to identify the services running on each, any of which might offer a way in. For this example, let's leave secmaniac.net behind and turn to the virtual machine described in Appendix A, with IP address 172.16.32.131. Before we get started, take a quick look at the basic nmap syntax by entering nmap from the command line on your Back|Track machine. You'll see immediately that nmap has a quite a few options, but you'll use just a few of them for the most part. One of our preferred nmap options is -sS. This runs a stealth TCP scan that determines whether a specific TCP-based port is open. Another preferred option is -Pn, which tells nmap not to use ping to determine whether a system is running; instead, it considers all hosts "alive." If you're performing Internet-based penetration tests, you should use this flag, because most networks don't allow Internet Control Message Protocol (ICMP), which is the protocol that ping uses. If you're performing this scan internally, you can probably ignore this flag. Now let's run a quick nmap scan against our machine using both the -sS and -Pn flags [25].

```
root@bt:~# nmap -sS -Pn 172.16.32.131
Nmap scan report for 172.16.32.131
Host is up (0.00057s latency).
Not shown: 990 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
25/tcp   open  smtp
80/tcp   open  http
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
443/tcp  open  https
445/tcp  open  microsoft-ds
1025/tcp open  NFS-or-IIS
1433/tcp open  ms-sql-s
3389/tcp open  ms-term-serv
Nmap done: 1 IP address scanned in 14.34 seconds
```

As you can see, nmap reports a list of open ports, along with a description of the associated service for each. For more detail, try using the -A flag. This option will attempt advanced service enumeration and banner grabbing, which may give you even more details about the target system. For example, here's what we'd see if we were to call nmap with the -sS and -A flags, using our same target system:

```
root@bt:~# nmap -Pn -sS -A 172.16.32.131
Nmap scan report for 172.16.32.131
Host is up (0.0035s latency).
Not shown: 993 closed ports
PORT     STATE SERVICE      VERSION
135/tcp  open  msrpc        Microsoft Windows RPC
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds Microsoft Windows 10 microsoft-ds
777/tcp  open  unknown
1039/tcp open  unknown
1138/tcp open  msrpc        Microsoft Windows RPC
1433/tcp open  ms-sql-s     Microsoft SQL Server
```

When you're running a complex penetration test with a lot of targets, keeping track of everything can be a challenge. Luckily, Metasploit has you covered with expansive support for multiple database systems. To ensure that database support is available for your system, you should first decide which database system you want to run. Metasploit supports MySQL and PostgreSQL; because PostgreSQL is the default, we'll stick with it in this discussion. First, we start the database subsystem using the built-in Back|Track init.d scripts.

```
root@bt~# /etc/init.d/postgresql-8.3 start
```

After PostgreSQL has started, we tell the Framework to connect to the database instance. This connection requires a username, password, the name of the host on which the database is running, and the database name we want to use. Back|Track's default PostgreSQL username is postgres with the password toor, but we'll use msfbook as the database name. Let's make the connection.

```
msf > db_connect
postgres:toor@127.0.0.1/msfbook
```

If this were the first time we connected to the database name, we would see a lot of text output as Metasploit sets up all the necessary tables. Otherwise, the command will return to the msfconsole prompt. Metasploit provides a number of commands that we can use to interact with the database, as you'll see throughout this book. (For a complete list, enter help.) For now, we'll use db_status to make sure that we're connected correctly.

```
msf > db_status
[*] postgresql connected to msfbook
```

Everything seems to be set up just fine.

When you are working with other team members, with various individuals scanning at different times and from different locations, it helps to know how to run nmap on its own and then import its results into the Framework. Next, we'll examine how to import a basic nmap-generated XML export file (generated with nmap's -oX option) into the Framework. First, we scan the Windows virtual machine using the -oX option to generate a Subnet1.xml file:

```
nmap -Pn -sS -A -oX Subnet1 192.168.1.0/24
```

After generating the XML file, we use the db_import command to import it into our database. We can then verify that the import worked by using the db_hosts command, which lists the entries of the system that have been created, as shown here:

```
msf > db_connect postgres:toor@127.0.0.1/msf3
 msf > db_import Subnet1.xml
 msf > db_hosts -c address
address       -------
192.168.1.1       192.168.1.10      192.168.1.101
192.168.1.102     192.168.1.109     192.168.1.116
192.168.1.142     192.168.1.152     192.168.1.154
192.168.1.171     192.168.1.155     192.168.1.174
192.168.1.180     192.168.1.181     192.168.1.2
```

This tells us that we've successfully imported the output of our nmap scans into Metasploit, as evidenced by the IP addresses populated when we run the db_hosts commands.

A more advanced nmap scan method, TCP idle scan, allows us to scan a target stealthily by spoofing the IP address of another host on the network. For this type of scan to work, we first need to locate an idle host on the network that uses incremental IP IDs (which are used to track packet order). When we discover an idle system that uses incremental IP IDs, the IP IDs become predictable, and we can then predict the next ID. However, when spoofing the address of an idle host while scanning a target's responses from open ports, we can see a break in the predictability of the IP ID sequence, which indicates that we have discovered an open port. To learn more about this module and IP ID sequences, visit [25] Use the Framework's scanner/ip/ipidseq module to scan for a host that fits the TCP idle scan requirements, as shown next (Fig. 1):

```
msf > use auxiliary/scanner/ip/ipidseq
msf auxiliary(ipidseq) > show options

Module options:

   Name       Current Setting  Required
   ----       ---------------  --------
   GWHOST                      no
   INTERFACE                   no
   LHOST                       no
   RHOSTS                      yes
   RPORT      80               yes
   SNAPLEN    65535            yes
   THREADS    1                yes
   TIMEOUT    500              yes
```

Figure 1: Display of required host parameters for ipidseq scanning

This figure displays the required options for the *ipidseq* scan. One notable one, RHOSTS, can take IP ranges (such as 192.168.1.20–192.168.1.30); Classless Inter-Domain Routing (CIDR) ranges (such as 192.168.1.0/24); multiple ranges separated by commas (such as 192.168.1.0/24, 192.168.3.0/24); and a text file with one host per line (such as file:/tmp/hostlist.txt). All these options give us quite a bit of flexibility in specifying our targets.

The THREADS value sets the number of concurrent threads to use while scanning. By default, all scanner modules have their THREADS value initially set to 1. We can raise this value to speed up our scans or lower it to reduce network traffic. In general, you should not set the THREADS value greater than 16 when running Metasploit on Windows, and not greater than 128 on UNIX-like operating systems.

Now let's set our values and run the module. We'll set the value for RHOSTS to 192.168.1.0/24, set THREADS to 50, and then run the scan. The result is shown in Fig. 2

```
msf auxiliary(ipidseq) > set RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
msf auxiliary(ipidseq) > set THREADS 50
THREADS => 50
msf auxiliary(ipidseq) > run

[*] 192.168.1.1's IPID sequence class: All zeros
[*] 192.168.1.10's IPID sequence class: Incremental!
[*] Scanned 030 of 256 hosts (011% complete)
[*] 192.168.1.116's IPID sequence class: All zeros
[*] 192.168.1.109's IPID sequence class: Incremental!
[*] Scanned 128 of 256 hosts (050% complete)
[*] 192.168.1.154's IPID sequence class: Incremental!
[*] 192.168.1.155's IPID sequence class: Incremental!
[*] Scanned 155 of 256 hosts (060% complete)
[*] 192.168.1.180's IPID sequence class: All zeros
[*] 192.168.1.181's IPID sequence class: Incremental!
[*] 192.168.1.185's IPID sequence class: All zeros
[*] 192.168.1.184's IPID sequence class: Randomized
[*] Scanned 232 of 256 hosts (090% complete)
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(ipidseq) >
```

Figure 2: Detection of several potential idle hosts that still perform idle scanning

Judging by the results of our scan, we see several potential idle hosts that we can use to perform idle scanning. We'll try scanning a host using the system at 192.168.1.109 shown by using the -sI command line flag to specify the idle host (Fig. 3):

msf auxiliary(ipidseq) > nmap -PN -sI 192.168.1.109 192.168.1.155
[*] exec: nmap -PN -sI 192.168.1.109 192.168.1.155

```
Idle scan using zombie 192.168.1.109 (192.168.1.109:80)
Incremental    Interesting ports on 192.168.1.155:
Not shown: 996 closed|filtered ports
PORT    STATE SERVICE
135/tcp open  msrpc
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:E4:59:7C (VMware)
Nmap done: 1 IP address (1 host up) scanned in 7.12 seconds
msf auxiliary(ipidseq) >
```

Figure 3: Detecting multiple open ports on our target system

By using the idle host, we were able to discover several open ports on our target system without sending a single packet to the system.

3.2. Port scanning with metasploit

In addition to its ability to use third-party scanners, Metasploit has several port scanners built into its auxiliary modules that directly integrate with most aspects of the Framework. In later chapters, we'll use these port scanners to leverage compromised systems to access and attack; his process, often called pivoting, allows us to use internally connected systems to route traffic to a network that would otherwise be inaccessible. For example, suppose you compromise a system behind a firewall that is using Network Address Translation (NAT). The system behind the NAT-based firewall uses private IP addresses, which you cannot contact directly from the Internet. If you use Metasploit to compromise a system behind a NAT, you might be able to use that compromised internal system to pass traffic (pivot) to internally hosted and private IP-based systems to penetrate the network farther behind the firewall. To see the list of port scanning tools that the Framework offers, enter the following [25]:

Let's conduct a simple scan of a single host using Metasploit's SYN Port Scanner. In the following Fig. 4, we start the scan by using scanner/portscan/ syn, set RHOSTS to 192.168.1.155, set THREADS to 50, and then run the scan.

From the results, you can see that ports 135, 139, and 445 are open on IP address 192.168.1.155, leveraging the portscan syn module within Metasploit.

When you are conducting a penetration test, there is no shame in looking for an easy win. A targeted scan looks for specific operating systems, services, program versions, or configurations that are known to be exploitable and that provide an easy door into a target network. For example, it is common to scan a target network for the vulnerability MS08-067, as this is (still) an extremely common hole that will give you SYSTEM access much more quickly than scanning an entire target network for vulnerabilities.

```
msf > use scanner/portscan/syn
msf auxiliary(syn) > set RHOSTS 192.168.1.155
RHOSTS => 192.168.1.155
msf auxiliary(syn) > set THREADS 50
THREADS => 50
msf auxiliary(syn) > run
[*]  TCP OPEN 192.168.1.155:135
[*]  TCP OPEN 192.168.1.155:139
[*]  TCP OPEN 192.168.1.155:445
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(syn) >
```

Figure 4: Scan of a single host using Metasploit's

Metasploit can scour a network and attempt to identify versions of Microsoft Windows using its smb_version module. I run the module, list my options, set RHOSTS, and begin scanning (Fig. 5):

```
msf > use scanner/smb/smb_version
msf auxiliary(smb_version) > show options

Module options:

    Name     Current Setting   Required
    ----     ---------------   --------
    RHOSTS                     yes
    THREADS   1                yes

msf auxiliary(smb_version) > set RHOSTS 192.168.1.155
RHOSTS => 192.168.1.155
```

Figure 5: Identification versions of Microsoft Windows using its smb version module

As you can see the smb_version scanner has pinpointed the operating system as Windows 10. Because we are scanning only one system, we leave THREADS set to 1. If we had been scanning several systems, such as a class C subnet range, we might consider upping the THREADS using the set THREADS number option. The results of this scan are stored in the Metasploit database for use at a later time and to be accessed with the db_hosts command:

```
msf auxiliary(smb_version) > db_hosts -c address,os_flavor

Hosts
=====

address         os_flavor   Svcs  Vulns  Workspace
-------         ---------   ----  -----  ---------
192.168.1.155   Windows 10  3     0      default
msf auxiliary(smb_version) >
```

We have discovered a system running Windows 10 without having to do a full scan of the network. This is a great way to target hosts quickly and quietly who are likely to be more vulnerable when our goal is to avoid being noticed.

The Simple Network Management Protocol (SNMP) is typically used in network devices to report information such as bandwidth utilization, collision rates, and other information. However, some operating systems also have SNMP servers that can provide information such as CPU utilization, free memory, and other system-specific details. Convenience for the system administrator can be a gold mine for the penetration tester, and accessible SNMP servers can offer considerable information about a specific system or even make it possible to compromise a remote device. If, for instance, you can get the read/write SNMP community string for a Cisco router, you can download the router's entire configuration, modify it, and upload it back to the router. The Metasploit Framework includes a built-in auxiliary module called scanner/snmp/snmp_enum that is designed specifically for SNMP sweeps. Before you start the scan, keep in mind that the Read-Only (RO) and Read/Write (RW) community strings will play an important role in the type of information you will be able to extract from a given device. On Windows-based devices configured with SNMP, you can often use the RO or RW community strings to extract patch levels, running services, usernames, uptime, routes, and other information that can make things much easier for you during a pen test. (Community strings are essentially passwords used to query a device for information or to write configuration information to the device.) After you guess the community strings, SNMP itself (depending on the version) can allow anything from excessive information disclosure to full system compromise. SNMPv1 and v2 are inherently flawed protocols. SNMPv3, which incorporates encryption and better check mechanisms, is significantly more secure. To gain access to a switch, you'll first need to attempt to find its community strings. The Framework's *use scanner/snmp/snmp_login* module will try a word list against one or a range of IP addresses.

```
msf > use use scanner/snmp/snmp_login
msf auxiliary(snmp_login) > set RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
msf auxiliary(snmp_login) > set THREADS 50
THREADS => 50
msf auxiliary(snmp_login) > run

[*] >> progress (192.168.1.0-192.168.1.255) 0/30208...
[*] 192.168.1.2 'public' 'GSM7224 L2 Managed Gigabit Switch'
[*] 192.168.1.2 'private' 'GSM7224 L2 Managed Gigabit Switch'
[*] Auxiliary module execution completed
msf auxiliary(snmp_login) >
```

A quick Google search for GSM7224 from the output tells us that the scanner has found both the public and private community strings for a Netgear switch. This result, believe it or not, has not been staged for this book. These are the default factory settings for this switch.

You will encounter many jaw-dropping situations like these throughout your pen testing career because many administrators simply attach devices to a network with all their defaults still in place. The situation is even scarier when you find these devices accessible from the Internet within a large corporation.

Many applications and services lack custom modules in Metasploit. Thankfully, the Framework has many features that can be useful when you're building a custom scanner, including offering access to all of its exploit classes and methods, and support for proxies, Secure Sockets Layer (SSL), reporting, and threading. It can be very useful to write your scanner during security assessments because doing so will allow you to locate every instance of a bad password or unpatched service quickly on a target system. The Metasploit Framework scanner modules include various

mixins, such as exploit mixins for TCP, SMB, and so on, and the auxiliary scanner mixin that is built into the Framework. Mixins are portions of code with predefined functions and calls that are preconfigured for you. The Auxiliary: Scanner mixin overloads the Auxiliary run method; calls the module method at runtime with run_host(ip), run_range(range), or run_batch(batch); and then processes the IP addresses. We can leverage Auxiliary: Scanner to call additional, built-in Metasploit functionality.

3.3. Vulnerability scanning

A vulnerability scanner is an automated program designed to look for weaknesses in computers, computer systems, networks, and applications. The program probes a system by sending data to it over a network and analyzing the responses received, to enumerate any vulnerabilities present on the target by using its vulnerability database as a reference. Various operating systems tend to respond differently when sent particular network probes because of the different networking implementations in use. These unique responses serve as a fingerprint that the vulnerability scanner uses to determine the operating system version and even its patch level. A vulnerability scanner can also use a given set of user credentials to log into the remote system and enumerate the software and services to determine whether they are patched. With the results it obtains, the scanner presents a report outlining any vulnerabilities detected on the system. That report can be useful for both network administrators and penetration testers [25, 26].

Vulnerability scanners generally create a lot of traffic on a network and are therefore not typically used in a penetration test when one of the objectives is to remain undetected. If, however, you are running a penetration test and stealth is not an issue, a vulnerability scanner can save you from having to probe systems manually to determine their patch levels and vulnerabilities. Whether you use an automated scanner or do it manually, scanning is one of the most important steps in the penetration testing process; if done thoroughly, it will provide the best value to your client. In this chapter, we will discuss several vulnerability scanners and how they can be integrated within Metasploit. We'll highlight some auxiliary modules in the Metasploit Framework that can locate specific vulnerabilities in remote systems [27, 28].

Let's look at how a scan works at the most basic level. In the following listing, we use netcat to grab a banner from the target 192.168.1.203. Banner grabbing is the act of connecting to a remote network service and reading the service identification (banner) that is returned. Many network services such as web, file transfer, and mail servers return their banner either immediately upon connecting to them or in response to a specific command. Here we connect to a web server on TCP port 80 and issue a GET HTTP request that allows us to look at the header information that the remote server returns in response to our request.

```
root@bt:/opt/framework3/msf3# nc 192.168.1.203 80
GET HTTP 1/1
HTTP/1.1 400 Bad Request
Server: Microsoft-IIS/5.1
```

The information returned tells us that the system running on port 80 is a Microsoft IIS 5.1-based web server. Armed with this information, we could use a vulnerability scanner, as shown in Fig. 6, to determine whether this version of IIS has any vulnerabilities associated with it and whether this particular server has been patched. Of course, in practice, it's not that simple. Vulnerability scans often contain many false positives (reported vulnerability where none exists) and false negatives (failure to log a vulnerability where one exists) due to subtle differences in system and application configurations. In addition, the creators of vulnerability scanners have an incentive to report positives: The more "hits" a vulnerability scanner finds, the better it looks to a potential buyer. Vulnerability scanners are only as good as their vulnerabilities database, and they can easily be fooled by misleading banners or inconsistent configurations. Let's take a look at some of the more useful vulnerability scanners, including NeXpose, Nessus, and some specialized scanners.

Figure 6: Vulnerability scan results against the target web server

4. Information system testing process

For conducting unauthorized access testing of the organization's information system, tools from the Kali Linux distribution were used.

During the simulation, Kali Linux will be utilized in two different modes: as a virtual machine and in Live USB mode. As a virtual machine, this operating system will be used to conduct attacks after gaining access to the organization's network. In Live USB mode, the operating system will run on a physical machine without installation, allowing the use of the hardware capabilities of the computer from which the testing is performed.

To test the information system, a virtual machine was created using Oracle VM VirtualBox. The virtual machine selected for the testing is Metasploitable 2, as this virtual machine is specifically designed to be highly vulnerable for training, exploit testing, and beginner learning. Unlike other vulnerable virtual machines, Metasploitable focuses on vulnerabilities in the Linux operating system and network services rather than individual applications [29].

The unauthorized access testing of the organization's information system began with the information-gathering process to predict possible attack vectors and methods for obtaining unauthorized access. To simulate the information-gathering process, the official website diia.gov, which belongs to the Ministry of Digital Transformation of Ukraine, was researched.

During the information gathering on public resources, data was found about individuals working in the organization, as well as the email format used within the organization (Fig. 7). This information could be useful for conducting social engineering attacks on one of the organization's employees.

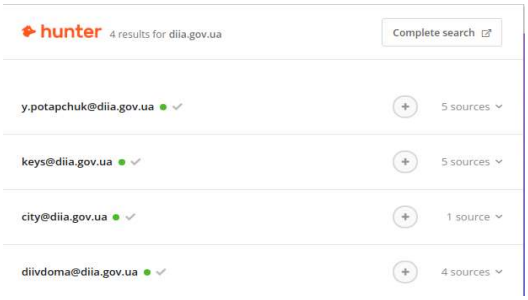

Figure 7: Information about the organization found using the EmailHunter utility

For further information gathering, Maltego was used. This is a software tool for information discovery that generates a graph based on link analysis. It is used in online investigations to automate the process and find connections between pieces of information located across different sources on the Internet. As a result of scanning the diia.gov.ua domain, the results are shown in Fig. 8. Here, we can see the physical addresses of the AWS servers hosting the Diia website, as well as servers connected to the Kyiv office of Diia.

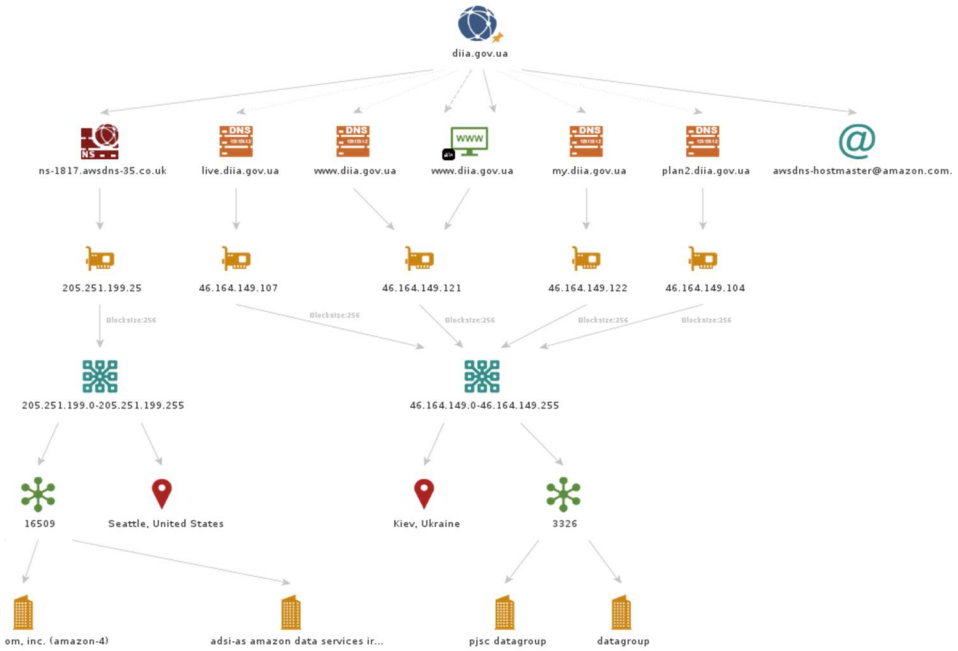

Figure 8: Graph generated as a result of the Maltego utility operation

The gathered information can be used to identify priority attack vectors, and the availability of this information enables the possibility of conducting social engineering attacks. After analyzing the website using these methods, a sufficient amount of information was collected to determine possible attack vectors. This phase is the longest, and the more information gathered, the higher the chances of finding vulnerabilities.

After gaining access to the information system, you can perform a scan of the system using the Nmap network scanner to gather information about the existing network elements and the openness of certain ports. Using the information about open ports, we can predict which attack vectors are optimal for further testing and what to focus on. The results of scanning the test network with Nmap are shown in Fig. 9. For example, from the Nmap scan, we can observe that port 80 (HTTP) is open, allowing us to continue its investigation. Since this port is HTTP and not HTTPS, meaning it is less secure, we can likely intercept and understand the information transmitted to this port.

To be able to crack the usernames and passwords of a web form, we need to identify the parameters of the login page and how the form responds to incorrect login attempts. The key parameters we must determine are:

- **IP address** of the website
- **URL**
- **Form type**
- **Username field**
- **Password field**
- **Error message**.

We can identify each of these using a proxy tool such as Tamper Data or Burp Suite.

```
ating Ping Scan at 07:34
ing 192.168.1.51 [2 ports]
eted Ping Scan at 07:34, 0.00s elapsed (1 total hosts)
ating Parallel DNS resolution of 1 host. at 07:34
eted Parallel DNS resolution of 1 host. at 07:34, 0.07s elapsed
ating Connect Scan at 07:34
ing 192.168.1.51 [1000 ports]
vered open port 445/tcp on 192.168.1.51
vered open port 3306/tcp on 192.168.1.51
vered open port 5900/tcp on 192.168.1.51
vered open port 22/tcp on 192.168.1.51
vered open port 23/tcp on 192.168.1.51
vered open port 80/tcp on 192.168.1.51
vered open port 139/tcp on 192.168.1.51
vered open port 21/tcp on 192.168.1.51
vered open port 53/tcp on 192.168.1.51
vered open port 111/tcp on 192.168.1.51
vered open port 25/tcp on 192.168.1.51
vered open port 2049/tcp on 192.168.1.51
vered open port 514/tcp on 192.168.1.51
vered open port 2121/tcp on 192.168.1.51
vered open port 1099/tcp on 192.168.1.51
  Discovered open port 5432/tcp on 192.168.1.51
  Discovered open port 6000/tcp on 192.168.1.51
  Discovered open port 8180/tcp on 192.168.1.51
  Discovered open port 1524/tcp on 192.168.1.51
  Discovered open port 512/tcp on 192.168.1.51
  Discovered open port 8009/tcp on 192.168.1.51
  Discovered open port 513/tcp on 192.168.1.51
  Discovered open port 6667/tcp on 192.168.1.51
  Completed Connect Scan at 07:34, 0.15s elapsed (1000 total ports)
```

Figure 9: Result of the Nmap utility operation

BurpSuite (Fig. 10) intercepts the request and shows us the key fields required for cracking the web form using THC-Hydra. After the login form address (/dvwa/login.php), the next field is the name of the field that accepts the username. In this case, it is “username”, but in some forms, it could be something else, such as “login”.

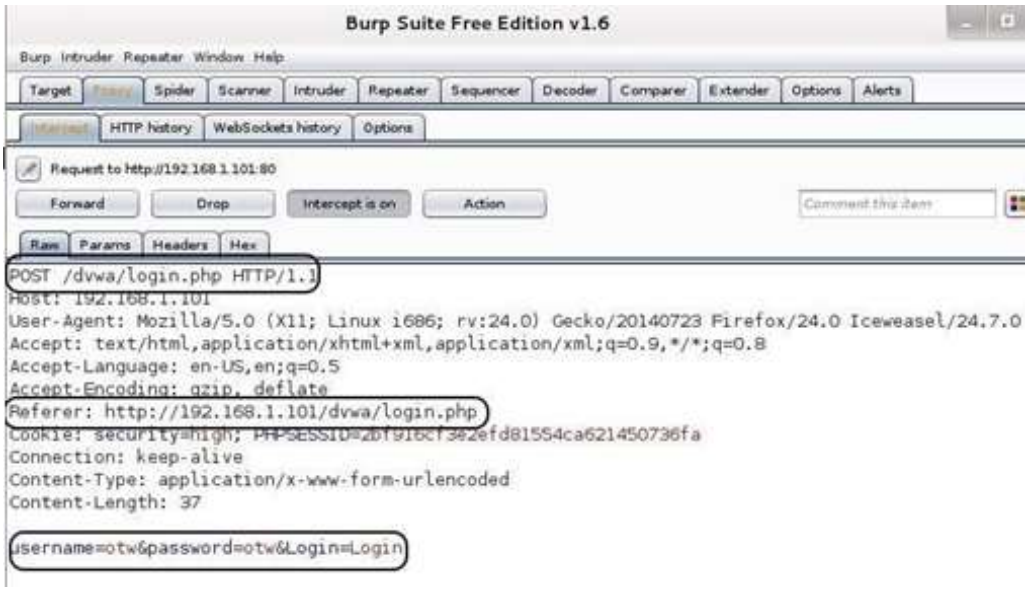

Figure 10: Result of Burp Suite operation. The image highlights the information required for further attack execution.

The command looks like this:
hydra -l admin -P /usr/share/dirb/wordlists/small.txt
192.168.1.51 http-post-form
"/dvwa/login.php:username=^USER^&password=^PASS^&
Login=Login:Login failed" –V
The result of executing the command is shown in Fig. 11.

Figure 11: Here you can see the successfully intercepted data that the user used for authentication on the web server in the information system

5. Conclusions

As a result of the conducted testing, a significant number of vulnerabilities in the information resources were identified. The information gathered by the tester about the vulnerabilities will help companies assess the current security level of their information system, identify vulnerabilities, and prioritize them based on their criticality, as well as develop a response plan for future cyberattacks. The research revealed that each system is unique in its way due to the use of different types of rules (signatures) and applications. This requires in-depth knowledge of attacks and the system documentation from the developer to configure the system for monitoring specific (non-standard) applications.

Kali Linux was chosen because it contains many tools for penetration testing, enabling periodic testing of networks and nodes, as well as security auditing of corporate networks to identify existing vulnerabilities, and misconfigurations, and mitigate them before they can be exploited by attackers.

Future research directions may focus on developing network utilities to implement protection for various types of operating environments from unauthorized interference and subsequently integrating them into a comprehensive utility system managed by the operating system. Additionally, improving the overall efficiency of monitoring the information system for identifying various types of vulnerabilities in its assets will enhance protection against many types of network attacks.

References

[1] S. V. N. Parasram, et al., Kali Linux 2018: Assuring Security by Penetration Testing Fourth Edition, Packt Publishing (2018).

[2] D. Shevchuk, et al., Designing Secured Services for Authentication, Authorization, and Accounting of Users, in: Cybersecurity Providing in Information and Telecommunication Systems II, vol. 3550 (2023) 217–225

[3] OWASP Web Security Testing Guide. URL: https://owasp.org/www-project-web-security-testing-guide/

[4] What is PCI DSS (Payment Card Industry Data Security Standard)? URL: https://www.techtarget.com/searchsecurity/definition/PCI-DSS-Payment-Card-Industry-Data-Security-Standard

[5] Penetration Testing Execution Standard (PTES). URL: https://www.geeksforgeeks.org/penetration-testing-execution-standard-ptes

[6] Security Considerations for Exchanging Files over the Internet. URL: https://csrc.nist.gov/publications/itl-bulletin

[7] Open-Source Security Testing Methodology Manual. URL: https://www.sciencedirect.com/topics/computer-science/open-source-security-testing-methodology-manual

[8] S. Shevchenko, et al., Information Security Risk Management using Cognitive Modeling, in: Workshop on Cybersecurity Providing in Information and Telecommunication Systems II, CPITS-II, vol. 3550 (2023) 297–305.

[9] D. Berestov, et al., Analysis of Features and Prospects of Application of Dynamic Iterative Assessment of Information Security Risks, in: Workshop on Cybersecurity Providing in Information and Telecommunication Systems, CPITS, vol. 2923 (2021) 329–335.

[10] Itgovernance.co.uk., Penetration Testing (2021). URL: https://www.itgovernance.co.uk/penetration-testing

[11] V. Susukailo, I. Opirskyy, S. Vasylyshyn, Analysis of the Attack Vectors used by Threat Actors during the Pandemic, IEEE 15th International Scientific and Technical Conference on Computer Sciences and Information Technologies, CSIT 2020, 2 (2020) 261–264.

[12] S. Parasram, Digital Forensics with Kali Linux - Second Edition, [S.l.]: Packt Publishing (2020).

[13] Maltego. URL: https://hackyourmom.com/kibervijna/zbir-informacziyi-pro-suprotyvnyka/osint-akademiya/4-relizy-maltego-prynczypy-roboty-ta-mozhlyvosti/

[14] R. Marusenko, V. Sokolov, P. Skladannyi, Social Engineering Penetration Testing in Higher Education Institutions, Advances in Computer Science for Engineering and Education VI, vol. 181 (2023) 1132–1147.

[15] Nmap: the Network Mapper - Free Security Scanner. URL: https://nmap.org

[16] Nmap Reference Guide | Nmap Network Scanning. Nmap.org, Chapter 15 (2021). URL: https://nmap.org/book/man.html

[17] P. Anakhov, et al., Protecting Objects of Critical Information Infrastructure from Wartime Cyber Attacks by Decentralizing the Telecommunications Network, in: Workshop on Cybersecurity Providing in Information and Telecommunication Systems, vol. 3050 (2023) 240-245.

[18] A. Singh, Metasploit Penetration Testing Cookbook, Packt Publishing (2012).
[19] A. Stoykov, Metasploitable 2 Full Walkthrough, MATRIX Labs (2021). URL: https://matrixlabsblog.wordpress.com/2019/04/02/metasploitable-2-full-walkthrough/
[20] M. Carey, et al., Nessus Network Auditing, O'reilly (2008). doi: 10.1016/B978-1-59749-208-9.X0001-9.
[21] Rapid7, Metasploit Editions: Network Pen Testing Tool (2021). URL: https://www.rapid7.com/products/metasploit/download/editions
[22] Tenable®, Nessus Product Family (2021). URL: https://www.tenable.com/products/nessus
[23] Kali.tools, Burp Suite - Kali Linux Tools (2021). URL: https://kali.tools/?p=1589
[24] Ptsecurity.com, Penetration Testing of Corporate Information Systems: Statistics and Findings, 2019 (2019). URL: https://www.ptsecurity.com/ww-en/analytics/corp-vulnerabilities-2019/
[25] D. Kennedy, et al., Metasploit: The Penetration Tester's Guide (2011).
[26] J. Hutchens, Kali Linux Network Scanning Cookbook, Packt Publishing (2014).
[27] D. R. Mathew, J. Benjamin, Penetration Testing and Vulnerability Scanning of Web Application Using Burp Suite, in: National Conference on Emerging Computer Applications (NCECA), 3(1) (2021).
[28] S. V. N. Parasram, et al., Kali Linux 2018: Assuring Security by Penetration Testing Fourth Edition, Packt Publishing (2018).
[29] Metasploitable 2. URL: https://docs.rapid7.com/metasploit/metasploitable-2