Penetration Testing: Exploiting software to save millions of dollars

Balthazar West, bwest@kth.se
Christer Winge, cwing@kth.se

April 26, 2020

1 Introduction

In today's society information technology is all around us. Using a plethora of online services each and every day is part of the norm for a lot of people around the world. With the continued increase in internet connectivity, there is no surprise that the amount of people affected by computer crimes has kept up with the progress [1]. Unlike getting scammed via social engineering or phishing, which in most instances can be faulted at the individual victim, data breaches in services can have severe consequences for individuals whose data is included in the breach. Apart from not using any sensitive information online, which is nearly impossible, the burden is entirely on the company who provides the service. Making sure that a service is fully secure against all types of threats is certainly an impossible task, but good tools and practices can lessen the chance and the impact of a potential breach. Penetration testing tries to solve this problem, aiding in the discovery of potential vulnerabilities before changes are exposed to its users, enabling the vulnerabilities to be patched beforehand.

2 Penetration testing

2.1 What it is

Penetration testing is a method used to assess the security of a computer system. Typically, it does this by trying to find vulnerabilities in a target while trying to reach a particular goal. The outcome of this process may be used to measure the difficulty for users to penetrate the target and gain unauthorized access. It can be automated or manual. There are various kinds of penetration testing depending on the information and context given to the tester [2].

Target knowledge

There are terms describing testing based on the amount of knowledge the tester has about the target. They are similar to those used in the topic of fuzzing.

Black box testing is done when there is no knowledge of the target. In other words, the tester must gather information by itself, simulating the behaviour of a real attacker with no information. White box testing is done when there is complete knowledge of the target. This will include anything the tester desires, e.g. source code. Grey box testing is done when there is partial knowledge of the target [3].

Internal or external

Internal or external refers to the position of the user relative to the target.

Internal testing is a way to estimate the damage an employee on the inside could cause. The tester will typically have standard access privileges. External testing is performed outside the owner of the target system, simulating what an attacker could access and what damage that might cause [4].

Area

The area typically concerns the part or workflow of the target that is being tested. It can be split into three types which are network, application and social engineering.

Network penetration identifies weaknesses in the organization's network. The tester attempts to exploit device connections and assesses how access to the target can be obtained. Application penetration tests expose the faults the target may have in its security checks. Social engineering involves exploiting humans to gain information or access to systems related to the target. This tests the effectiveness of organizations security protocols [2].

2.2 How it is done

Penetration testing can be divided into multiple phases. However, there is no established way of distinguishing between phases. This section presents one possible structure with elements common across many of the references. Figure 1 shows possible phases and their relation to each other.

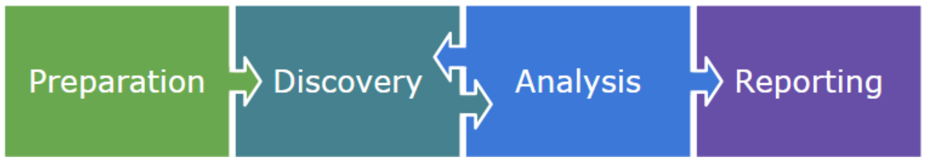


Figure 1: A common phase structure used in penetration testing

Preparation

Before beginning the penetration testing, one must establish the target computer system and goal, i.e. the objective that is to be achieved during testing. If agreeing on more details is necessary, such as test duration or signing legal documents, it should be done here. The tester will be handed some amount of information depending on the choice of penetration test. As an example, in an internal white box test the tester may be given the target's source code and login information to the organization [4].

Discovery

The actual testing begins in this phase. Testing can be manual and or automated using tools like *fuzzers*. During this phase the objective for the tester is to discover vulnerabilities that potentially could be exploited. This could be a wide variety of things like memory leaks, buffer overflows or errors caused by concurrency. Anything that produces an error or unexpected behaviour may be worth looking at in the next phase [2].

Analysis

At this point the tester will have gathered a set of discoveries that are to be analyzed. The implications of each discovery should be considered, i.e. whether a fault would lead to an exploit. If a discovery leads to an exploit, then the tester returns to the discovery phase to see if the exploit leads to additional discoveries. For instance, this could happen if the analysis of a discovered vulnerability allows the attacker to gain escalated system privileges, letting the attacker access more of the target system than intended [3].

Reporting

Of course, the tester must share the results of the testing process. The discoveries, exploits and implications should be presented in a document that is systematic, complete and easy to digest. They could be organized depending on type, severity or likelihood of being exploited. A review of these factors would have to be done and presented in a readable manner, for instance as a matrix. Avoiding simply listing the vulnerabilities in bullet points, without any clear weight attached to them [3].

2.3 Why it is used

There are many reasons as to why companies might consider the use of penetration testing practices on their systems and applications. The exploitation of a service can have a severe impact depending on what could be leveraged out of the weakness. Leaking personal information about its users and customers can easily lead to huge monetary losses and damage its reputation, which can be detrimental and hard to recover from in a competitive market [5][6]. Good practice of penetration testing before the release of a service can help ensure that the likelihood of such problems arising down the line is minimized.

Penetration testing should be seen as an ongoing process that is to be repeated at certain key points during release, as well as performed at certain intervals. If the service is continuously developed and new releases or updates keep being pushed, then there is always the possibility that a new bug crops up, which might be uncovered by the penetration testing and prevent the release of a potentially severe vulnerability. Even if the developed service does not contain any such vulnerability, there is always the chance that some part of the infrastructure used to deploy and run the service could be leveraged by an attacker. Figure 2 shows just how many commonly used and established applications have a large amount of vulnerabilities reported to Common Vulnerabilities and Exposures (CVE) each year [7].

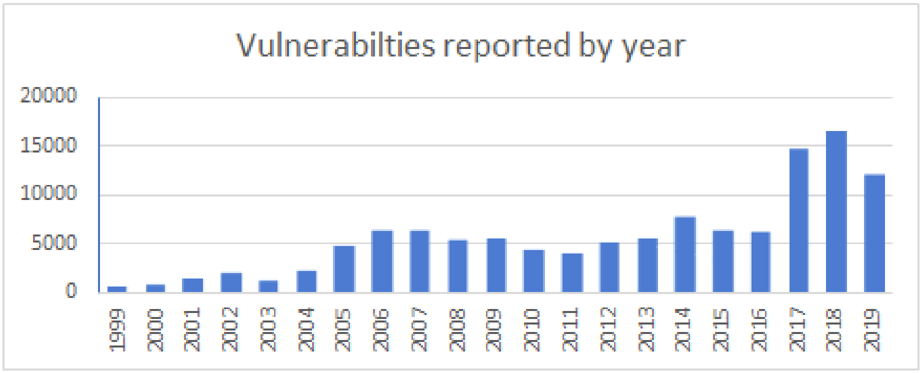


Figure 2: Vulnerabilities reported and registered by CVE by year

To lessen this entry vector, it is therefore crucial to keep all used infrastructure software up-to-date, otherwise you stay susceptible to these now widely known vulnerabilities and become a much easier target than previously. Some penetration tools have these vulnerabilities built in, which would identify some of these issues, but that depends highly on what has been implemented by the tool. Depending on what type of service is provided, penetration testing might even be mandatory to fulfil certain industry standards and regulations, where requirements and testing frequency can vary greatly [8][9][10][11][12].

It is not always a cut-and-dried process when deciding where the security concern should be focused. Penetration testing can assist with the decision making by employing a risk-prioritization approach. By employing a risk-prioritization approach the results could be presented as a list of vulnerabilities ordered by assessed severity, making them easier to communicate. This builds a strong case that the focus should be directed by the prioritization, leading to that the vulnerabilities are dealt with in correlation to their severity.

2.4 Tools

As mentioned earlier, there is a wide selection of tools to perform penetration testing. There are even several specialized Operating System distributions, that come preinstalled with many of the most popular penetration testing tools, that ease the burden of having to set up a test environment [13]. As with most tools there are a few that do something similar. Even when accounting for this, there is still quite a substantial number of categories of penetration testing, that each cover one or more different use cases. These use cases range from simple *port scanning* and injecting commonly harmful *SQL statements* into forms, to executing complex scripts that targets known vulnerabilities. This section will cover a few different tools and how they might be used.

Finding vulnerable hosts using Nmap

In internet's infancy, knowledge and tools to help with setting up a secure firewall was not as widely available as it is today. Having open ports exposed to anyone on the internet can lead to less than ideal situations. A common prank was to scan for exposed printers, connect to them and then print something hilarious. These days the default firewalls and settings that come with most systems are often relatively secure unless the users specifies otherwise.

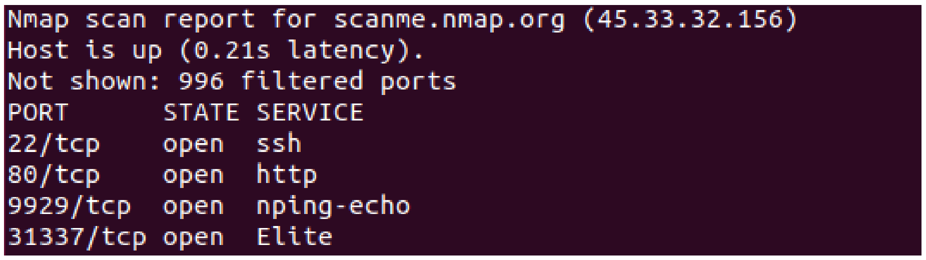
Nmap is a tool that can be used to help with securing networks by running its different features to discover potential weaknesses. It discovers open hosts and services by sending specially crafted packets and is able to determine a wide range of things depending on the responses. Nmap was first released in 1997 and has had many updates ever since, gaining massive popularity and is one of the most known tools of its kind.

Nmap includes a sizeable amount of commands and options. Most of the features mentioned below can add additional options and flags to get the most out of the tool. Correct usage of certain options can significantly reduce the time required to perform certain actions. Despite these options it is not overwhelming to new users as most features can easily be combined or performed separately with their default options, giving valuable results without any prior knowledge.

The *host discovery* feature is used to discover hosts, it does so by pinging and sending different type of packets to see if a response can be triggered and received. If a response is received, it is now known that on a specific *IP-address* there is a potential target that is online which can be further probed.

Another feature is its *port scan*, which goes through ports on selected hosts, to see which ports are active and open. This would be a suitable second step, used on the resulting list of hosts gathered from the use of the *host discovery* feature. The result would be a list of ports where some kind of information was learnt about the services the host employs on certain ports. Figure 3 shows an example of what the output could look like.

Through Nmap's *service/version detection* feature, it is possible to detect what service and what version a host machine is running. For a service and version to accurately be detected depends on if it is present in the *nmap-services* database. The database keeps expanding, covering more and more services and versions. There are about 2200 entries according to the *reference guide*, but at the time of writing the current database contains over 12000 known entries and over 15000 unidentified entries [14]. The ports discovered in a previous phase are probed using the *nmap-serivce-probes* database, which contains the



```
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.21s latency).
Not shown: 996 filtered ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
9929/tcp  open  nping-echo
31337/tcp open  Elite
```

Figure 3: Result of default nmap scan targeted at scanme.nmap.org

used queries and expected responses that are matched to identify services. The resulting output could include information such as the service's service protocol, version number, hostname, device type and OS family [15].

Nmap is well-known for its OS detection feature. By sending a series of probes and exhaustively examining the responses, it is able to detect what OS and version a host is running by employing *TCP/IP stack fingerprinting* using several techniques. Some of these techniques are *TCP ISN sampling*, *TCP options support and ordering*, *IP ID sampling* and an initial *windows size check* [16]. At the time of writing the *nmap-os-db* contains 6180 different fingerprints, including OS's used by a large variety of devices, such as regular computer systems, routers, switches and the ever more popular *smart* doorbells [17].

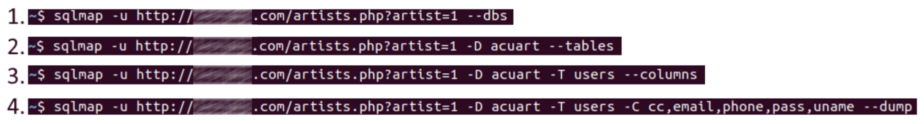
Lastly, the *Nmap Scripting Engine* (NSE) provides a feature that allows users to write and share their own Lua scripts that allows for automation of different networking tasks. These scripts extended what features *Nmap* can perform quite substantially. Some examples of scripts are vulnerability scanning (more specific vulnerabilities than described above), fuzzing, exploiting, intrusion detection and many more. At the time of writing, there are 601 included NSE scripts in *Nmap* [18].

Database dumping using sqlmap

Any website accessible on the internet must defend itself against a wide variety of attack vectors. If a website uses a database and allows users to input some type of data through a GET or POST request, special care must be taken to ensure it is not susceptible to *SQL injections.* For instance, if the input data is not correctly sanitized, it can allow an attacker to simply probe the database using regular SQL syntax through the point of entry. This can often allow an attacker to completely or partially dump any information that the database contains, which is a huge security breach that could lead to enormous information leaks, all due to the trivial mistake of not sanitizing a single input field.

The quite popular open source tool *sqlmap* can be used to discover these vulnerabilities, as well as exploit them to show what type of damage an attacker could cause. It has full support for many different databases, such as *MySQL*, *PostgreSQL* and *Apache Ignite* [19]. It provides a long list of features, with one of the main ones obviously being its implementation of six different SQL injection techniques [19].

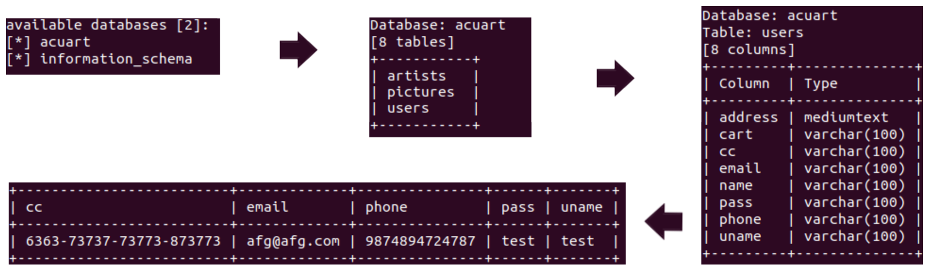
To perform a quite extensive SQL injection probe with *sqlmap*, not much effort is required. To get started, an URL to probe is required, with for instance a *GET* field that can be used as target. From there on it is a step by step process with using the results found in previous steps, to gain access to more information required for the next step. Figure 4 and 5 shows a very simple example of how a website can be searched for vulnerabilities and then systematically continue with the results from each step, to finally retrieve sensitive information from the database.



```
1. ~$ sqlmap -u http://[redacted].com/artists.php?artist=1 --dbs
2. ~$ sqlmap -u http://[redacted].com/artists.php?artist=1 -D acuart --tables
3. ~$ sqlmap -u http://[redacted].com/artists.php?artist=1 -D acuart -T users --columns
4. ~$ sqlmap -u http://[redacted].com/artists.php?artist=1 -D acuart -T users -C cc,email,phone,pass,uname --dump
```

Figure 4: The used commands run one after another to get the results shown in figure 5 from a targeted vulnerable website

As shown, the process was able to first find the databases used by the website, carry on to find what tables it contains, the columns existing within a table and then finally able to extract the selected data.



```
available databases [2]:
[*] acuart
[*] information_schema
```

```
Database: acuart
[8 tables]
+-----------+
| artists   |
| pictures  |
| users     |
+-----------+
```

```
Database: acuart
Table: users
[8 columns]
+---------+--------------+
| Column  | Type         |
+---------+--------------+
| address | mediumtext   |
| cart    | varchar(100) |
| cc      | varchar(100) |
| email   | varchar(100) |
| name    | varchar(100) |
| pass    | varchar(100) |
| phone   | varchar(100) |
| uname   | varchar(100) |
+---------+--------------+
```

```
+-----------------------+-------------+---------------+------+-------+
| cc                    | email       | phone         | pass | uname |
+-----------------------+-------------+---------------+------+-------+
| 6363-73737-73773-873773 | afg@afg.com | 9874894724787 | test | test  |
+-----------------------+-------------+---------------+------+-------+
```

Figure 5: The results gained from each step shown in figure 4, to end up extracting potentially valuable information on its users from the database

If the website is secure enough to not yield any valuable results, there are plenty of more advanced features available at its disposal, using different techniques. With tools like *sqlmap* and other similar widely available, there really is no excuse for a website to leak data through this type of attack. It is quite an ancient and well-known problem and many development tools and frameworks now assist with ensuring better practices around this area.

The Metasploit Project

The Metasploit project is the most used penetration testing framework in the world [20]. It is a collection of sub-projects, all owned by the security company Rapid7. There are multiple editions, free and commercial, all with the common goal to provide an exploit development environment. Due to its completeness, their most premium edition (Metasploit Pro) could be considered a general-purpose penetration testing tool. Because the project is so large and encompassing only an overview and comparison of four major editions will be given. To learn more there are plenty of resources freely available [21][22].

Metasploit Framework Edition is the most basic version. It is open-source and free and includes a command-line interface that allows for network scanning and access to over 1500 exploits. However, testing must be done manually because it lacks any automation [23].

Metasploit Community Edition is also free and open source. It is essentially the framework edition with a web-based interface, resembling the paid versions but with less features. There is some additional functionality not available in the framework edition like module browsing [24].

Metasploit Express is a commercial and open-core version. It is aimed towards security teams to verify vulnerabilities, offering a graphical user interface, smart brute-forcing and automated evidence collection and more. Express was dropped to focus efforts on the pro version [25].

Metasploit Pro is also commercial and open-core and is the most complete version of Metasploit. It adds to the features in express, like more automation, social engineering, web application testing, the pro console and more [23].

3 Conclusion

In the connected world that is today, penetration testing is not optional, it is a must. As shown, security flaws costs hundreds of millions of dollars each year, they breach privacy and can devastate an organization's reputation. There is a multitude of tools to aid in the testing process that vary in price and purpose. Due to the many free tools it only makes sense to always include this kind of testing in the development process. However, great expertise is required to minimize risks as there is a great amount of points of entry and many angles to consider.

References

[1] Godwin Udo, Kallol Bagchi, and Peeter Kirs. Analysis of the growth of security breaches: A multi-growth model approach. *Issues in Information Systems*, 19(4), 2018.

[2] Rafay Baloch. *Ethical Hacking and Penetration Testing Guide.* Auerbach Publications, 2014.

[3] Daniel Geer and John Harthorne. Penetration testing: A duet. pages 185–195, 2002. doi: 10.1109/CSAC.2002.1176290.

[4] Bei Chu Monique Jones Aileen Bacudion, Xiaohong Yuan. An overview of penetration testing. *International Journal of Network Security & Its Applications*, pages 19–38, 2011. doi: 10.5121/ijnsa.2011.3602.

[5] Ponemon Insitute. Cost of a data breach study, 2020. URL `https://www.ibm.com/security/data-breach`. accessed 2020-04-24.

[6] Christina Y. Jeong, Sang-Yong Tom Lee, and Jee-Hae Lim. Information security breaches and it security investments: Impacts on competitors. *Information & Management*, 56(5):681 – 695, 2019. ISSN 0378-7206. doi: https://doi.org/10.1016/j.im.2018.11.003.

[7] MITRE Corporation. Current CVSS score distribution for all vulnerabilities, 2020. URL `https://www.cvedetails.com/`. accessed 2020-04-24.

[8] ISO/IEC 27001 INFORMATION SECURITY MANAGEMENT, 2020. URL `https://www.iso.org/isoiec-27001-information-security.html`. accessed 2020-04-24.

[9] CYBERSECURITY — NIST, 2020. URL `https://www.nist.gov/topics/cybersecurity`. accessed 2020-04-24.

[10] FEDERAL INFORMATION SECURITY MODERNIZATION ACT, 2020. URL `https://www.cisa.gov/federal-information-security-modernization-act`. accessed 2020-04-24.

[11] Summary of the HIPAA Security Rule, 2020. URL `https://www.hhs.gov/hipaa/for-professionals/security/laws-regulations/index.html`. accessed 2020-04-24.

[12] Official PCI Security Standards Council Site - Verify PCI Compliance, Download Data Security and Credit Card Security Standards, 2020. URL `https://www.pcisecuritystandards.org/`. accessed 2020-04-24.

[13] Backbox - a free open source community project, 2020. URL `https://www.backbox.org/`. accessed 2020-04-25.

[14] nmap-services database, 2020. URL `https://svn.nmap.org/nmap/nmap-services`. accessed 2020-04-26.

[15] Service and Version Detection — Nmap Network Scanning, 2020. URL `https://nmap.org/book/man-version-detection.html`. accessed 2020-04-26.

[16] OS Detection — Nmap Network Scanning, 2020. URL `https://nmap.org/book/man-os-detection.html`. accessed 2020-04-26.

[17] nmap-os-db database, 2020. URL `https://svn.nmap.org/nmap/nmap-os-db`. accessed 2020-04-26.

[18] Public NSE scripts, 2020. URL `https://nmap.org/nsedoc/scripts/`. accessed 2020-04-26.

[19] sqlmap - Automatic SQL injection and database takeover tool, 2020. URL `http://sqlmap.org/`. accessed 2020-04-25.

[20] Metasploit — penetration testing software, pen testing security — metasploit, 2020. URL `https://www.metasploit.com/`. accessed 2020-04-26.

[21] Metasploit unleashed - free online ethical hacking course, 2020. URL `https://www.offensive-security.com/metasploit-unleashed/`. accessed 2020-04-26.

[22] Quick start guide, 2020. URL `https://metasploit.help.rapid7.com/docs`. accessed 2020-04-26.

[23] Metasploit editions: Network pen testing tool, 2020. URL `https://www.rapid7.com/products/metasploit/download/editions/`. accessed 2020-04-26.

[24] Metasploit community edition - metasploit unleashed, 2020. URL `https://www.offensive-security.com/metasploit-unleashed/msf-community-edition/`. accessed 2020-04-26.

[25] Announcement: End of life for metasploit express edition, 2020. URL `https://blog.rapid7.com/2018/06/04/announcement-end-of-life-for-metasploit-express-edition/`. accessed 2020-04-26.