Systems, Software, and Applications Updating for avoiding Cyber Attacks: A Pentest Demonstration

Luiz V. I. C. Casagrande, Evandro C. Vilas Boas, Guilherme P. Aquino

***Abstract*— This work exploits vulnerabilities in an outdated version of Pandora FMS software through penetration testing (pentest) to demonstrate the relevance of updating systems, software, and applications to avoid cyber attacks. The practical approach is based on a pentest black box in an environment with an outdated version of the Pandora FMS. The SQL injection and the remote file inclusion are exploited, allowing administrative access to the software by inserting a session cookie on the server. Therefore, malware is introduced into the network to control the server.**

***Keywords*— Cyber security, Pandora FMS, Pentest, SQL injection, vulnerabilities.**

I. Introduction

Updating operating systems, programs, and applications is a standard and inherent process to their useful life, allowing improving the user experience. Furthermore, this practice also includes the correction of security aspects, which is essential within the scope of cybersecurity to prevent attacks by correcting vulnerabilities. Therefore, it is crucial to update operating systems, programs, and applications immediately as recommended by the OWASP (Open Web Application Security Project) *frameworks* [1], [2].

In the development phase of any software, the program developers can make mistakes compromising its security or the system using itself. Furthermore, cybercriminals can discover and exploit these vulnerabilities to yield economic and social impairment to system users. For instance, a study by CHAOS briefed that 66% of programs have some vulnerabilities [3]. In the context of cybersecurity, those newly discovered are called zero-day vulnerabilities and are unknown to program developers, without immediate corrections releases [4]. As a result, cybercriminals exploit these vulnerabilities to attack operating systems, programs, and applications, such as the incident involving the malware known as Wannacry [5]. However, update packages are developed to fix the software vulnerability as soon as it is analyzed.

Despite the updating packages, companies have suffered attacks due to outdated or misconfiguration devices and systems. For example, Equifax exposed approximately 143 million customer data through an outdated consumer complaints portal in 2017. San Francisco State University notified a cybersecurity incident due to outdated systems in 2015 [6]. Recently, the verified vulnerability in the open-source library Log4j from the Apache Logging Services Project has served as a gateway to malware on many networks [7]. These examples highlight the importance of recurrently updating operating systems, programs, and applications at a personal, corporate or industrial level.

Luiz V. I. C. Casagrande, Evandro C. Vilas Boas, Guilherme P. Aquino, Inatel Cyber Security Center (Centro de Segurança Cibernética do Inatel - CxSC Telecom), National Institute of Telecommunication (Instituto Nacional de Telecomunicações - Inatel), Santa Rita do Sapucaí - MG, e-mail: luiz.casagrande@get.inatel.br, evandro.cesar@inatel.br, guilhermeaquino@inatel.br.

On the other hand, updating routines are not a conventional practice. It concerns several aspects, such as the systems application and incompatibility issues with versions of other dependent programs. For instance, industries that use automation systems can be exposed to cyberattacks since updates are scheduled instead of executing them immediately, aiming to reduce losses by avoiding interrupting the activities [5]. Updating programs can also cause performance and functionality issues in applications with mutual dependence. For example, the update of PHP or Apache is mentioned that can cause incompatibility with each other, which may demand future updates. This situation helps the end-user avoid or delay the update task, exposing their device or machine to cyber attacks.

This work exploits vulnerabilities in an outdated version of the Pandora FMS software to highlight the significance of keeping updated systems, programs, and applications to prevent cyber attacks. The Pandora FMS vulnerabilities are exploited to accomplish complete control over the application and, consequently, over the server. This software is often used for network monitoring by large companies from different segments, such as Toshiba, Rakuten, Allianz, and Logicalis. The practical approach is based on a penetration test (pentest black box) in an environment with an outdated version of the Pandora FMS. The process is divided into three phases: scanning and enumeration, exploration, and post-exploitation. First, the SQL injection and the remote file inclusion are exploited, allowing administrative access to the software by inserting a session cookie on the server [8], [9]. Therefore, malware is introduced into the network to control the server.

This work is structured in four sections as follows. Section II discusses concepts related to pentest and the vulnerabilities found in version 742 of the Pandora FMS software, highlighting the SQL injection. Section III presents a pentest practical demonstration performance over the program's outdated version and discusses the results. Finally, comments and conclusions are presented in Section IV.

II. Penetration Test and Pandora FMS Vulnerabilities

This section introduces important concepts about pentest and discusses the vulnerabilities found in the Pandora FMS software and patch-released update.

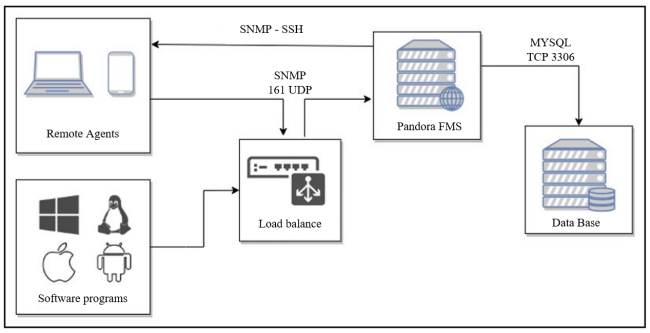

Fig. 1. Test environment scenario.

A. Penetration Test

Companies that develop technology, manipulate or store sensitive data on a local or virtual network must undergo periodic cybersecurity verifications or consultations. This practice allows adapting the information cycle processes to comply with current local legislation on data protection and avoid sanctions in case of incidents with data necessary for its business development.

Penetration testing (pentest) represents a possible approach tool during consulting, allowing exploiting vulnerabilities in a company's systems and software similar to a cyber attack. There are three possible approaches to executing penetration testing. First, the pentest white box is the most straightforward procedure and demands prior knowledge of the technologies involved in the systems architecture and access to the application's source code. Next, realistic simulations of an attack scenario are used without this information, defining the pentest black box. Finally, the intrusion test known as the grey box includes two possibilities: unavailability of the source code and retention of information about the system or login to validate the application.

A pentest consists of six phases: i) pre-engagement interaction, ii) threat modeling, iii) vulnerability identification, iV) exploration, v) post-exploitation, and vi) reporting. The pre-engagement interaction phase obtains essential information about the target available on the Internet. It uses scanning tools to identify and enumerate open ports and available active services open to the Internet and/or internal network. Based on the information gathered, the threat modeling stage is responsible for developing strategies similar to a legitimate attacker. Therefore, tests are carried out to prove the effectiveness of attack plans and asses successful strategies, comprising the vulnerability analysis stage. Tools are used to automate this process with the critical analysis of the person responsible for testing to obtain the best results.

The exploration phase uses programs available on the Internet or previously developed to exploit vulnerabilities in the client's system. In the post-exploitation step, the criticality level of each exploited vulnerability is identified. According to the scope, horizontal displacement in the network or escalation of privileges may occur to obtain complete control over the systems. The final step comprises reporting the results by addressing the vulnerabilities, critical level, and the corrections.

B. Pandora FMS Vulnerabilities

The Pandora FMS tool is an open-source solution used by several companies for network monitoring, event analysis, and management of applications and devices on Unix and Windows systems. This tool was scanned by SonarSource, who reported numerous instances exposed on the Internet running the software version 742. Additionally, critical vulnerabilities were identified in this version that allows complete control over the application, highlighting the SQL injection vulnerability.

The SQL injection issue, cataloged as CVE-2021-32099 and classified as a critical-level vulnerability, is exploited without the need for system privileges [8]–[10]. In other words, anyone could use this vulnerability for systems authentication and access with administrative privileges. Consequently, one can manipulate files, identify network devices, exploit other vulnerabilities such as executing remote commands, create a *backdoor* and *upload* malicious files to contaminate the entire network. This failure is characterized by poor code sanitation when handling the user session, causing the system to misinterpret the user as having an administrative level in its database. In [8], there is complete detail on the sanitation failures of the Pandora FMS program, including excerpts that allowed the vulnerability of the system, as well as its solution [11].

Version 743 introduces the Pandora FMS update that fixes the SQL injection flaw vulnerability, allowing authentication as an administrator without application credentials. However, this version has other vulnerabilities classified as XSS (*Cross-site Scripting*) and remote file injection flaws, which exploration is out of the scope of this work.

III. Pentest Demonstration

This section presents the practical execution of a pentest using the black box approach on a server running version 742 of the Pandora FMS software, which retains the vulnerabilities discussed in Section II.

A. Test environment

Figure 1 shows the test environment scenario. There is a server running the Pandora FMS tool, whose communication to the external environment uses the SSH (*Secure Shell*) and SNMP (*Simple Network Management Protocol*) protocols. There is also direct communication with the database to store information about Pandora itself and the managed devices and software. A load balance is used to maintain the servers stability when the traffic is substantial, including the configuration of ports that can communicate with the server. These ports are open to the Internet, such as ports 22 (SSH) and 161 (SNMP), or only on the server's local network, such as port 80 (HTTP).

B. Scanning and Enumeration

The Nmap software analyzed and listed the possible active services and their respective ports for the Pandora FMS application. The scan included the possible 65535 ports open on the `panda.htb` server, whose IP is 10.10.11.136 [12].

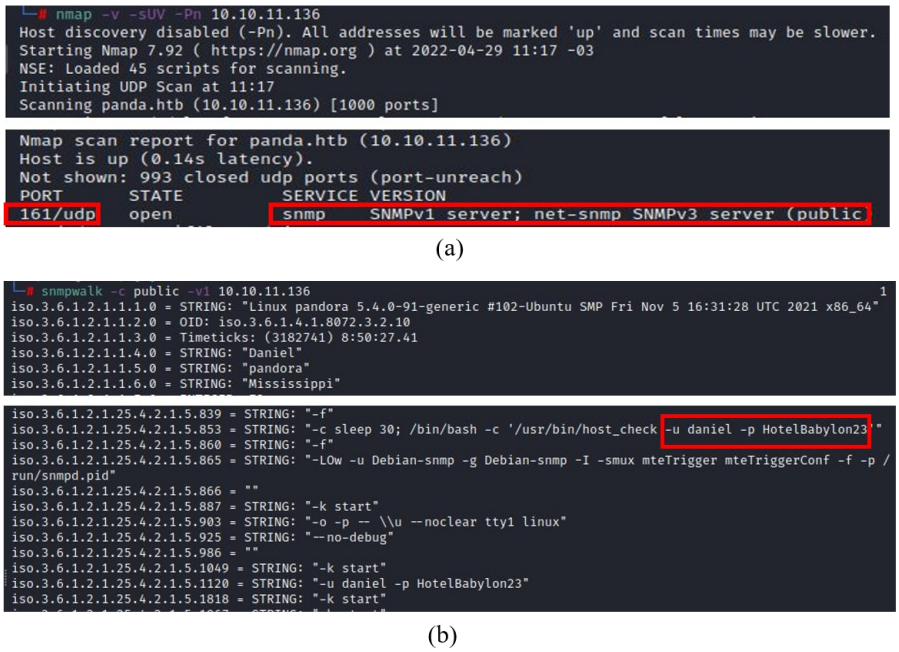

Fig. 2. (a) UDP scan results usign Nmap software, (b) snmpwalk enumeration with Daniel user credentials as results.

Transmission Control Protocol (TCP) scanning was initially considered without returning any vulnerability in services for easy exploitation. Afterward, the scan was performed with the User Datagram Protocol (UDP), identifying the SNMP active on UDP port 161, as shown in Figure 2(a). In addition, the SSH was identified on the TCP port 22 and the HTTP on port 80. For the analysis, the parameters `-v` were used to activate the verbose mode and return the information; `-sUV` to show the version of each service/protocol found, the `U` stands for UDP; `-sC` to test standard scripts and try to identify some simple vulnerability; and `-Pn` to avoid using the Internet Control Message Protocol.

The SNMP is used for managing and monitoring network devices, representing a risk to application security since exposed. Therefore, this work exploited this vulnerability to obtain credentials and sensitive information about the network using a simple protocol enumeration. The snmpwalk tool was used for the complete enumeration process, comprising the Community (public, in this case) and the protocol version (identified as version 1 during the scanning) as parameters. Then, the command `snmpwalk -c public -v1 10.10.11.136` was executed, which returned the credential of a user named `daniel:HotelBabylon23`, as shown in Figure 2(b).

This credential enabled authentication to the server via the SSH service, which allowed us to access the codes referring to the web application and identify possible ways to exploit privileges. As a result, we found the pandora directory referring to the Pandora FMS program, which is located at `/etc/apache2/sites-enabled/pandora.conf` directory. The file reading returned the local execution of the application and user texttttmatt with administrative privileges, as depicted in Figure 3. Based on port forwarding, we mirrored the traffic from the local application to the attacking machine executing the command `ssh -L 8080:127.0.0.1:80 daniel@10.10.11.136` to provide access to the application for exploration. The `-L` parameter is used to mirror the traffic from the servers port 80 to local port 8080.

C. Exploration

The port forwarding allowed accessing the applications login page, whose version is 7.0NG.742, classified as insecure configuration by the OWASP framework and included in category `A05:2021-Configuration insecure`. This version is outdated and included in the class `A06:2021-Outdated and Vulnerable Component`. It comprises the following vulnerabilities: SQL Injection (CVE-2021-32099), Phar deserialization (CVE-2021-32098), Remote File Inclusion (CVE-2021- 32100), and Cross-Site Request Forgery [8]. This work exploits SQL injection and remote file inclusion.

The SQL injection allows inserting a session cookie on the server to acquire admin privileges. An injection code provided in [13] was used, and changes were inserted through the icyberchef website using the URL decode filter. Figure 4(a) shows the change in the payload. A blank page is displayed without any error after inserting this code in a URL session. Afterward, the remaining code after `/pandora_console/` was deleted, and the page reloaded to access as an admin user, as seen in Figure 4(b).

Furthermore, platform access enabled remote file inclusion by inserting a malicious code (exploit) in PHP containing the code to obtain a reverse shell from the server. The malicious code file is available at [14], requiring only the IP address and port change. The PHP file containing this exploit must be compressed in .zip format. The file was

```
daniel@pandora:/var/www/pandora/pandora_console$ cat /etc/apache2/sites-enabled/pandora.conf
<VirtualHost localhost:80>
  ServerAdmin admin@panda.htb
  ServerName pandora.panda.htb
  DocumentRoot /var/www/pandora
  AssignUserID matt matt
  <Directory /var/www/pandora>
    AllowOverride All
  </Directory>
  ErrorLog /var/log/apache2/error.log
  CustomLog /var/log/apache2/access.log combined
</VirtualHost>
daniel@pandora:/var/www/pandora/pandora_console$
```

Fig. 3. Pandora software configuration file.

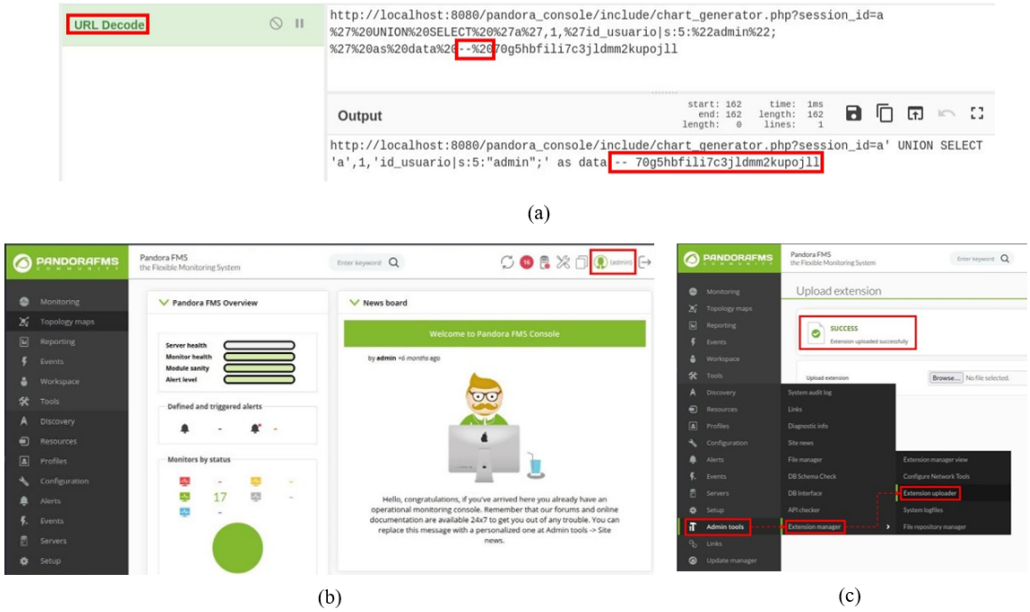

Fig. 4. (a) Code changing for SQL injection using icyberchef, (b) Administrator access to Pandora FMS, (c) Malicious file upload containing reverse shell.

inserted in `Admin tools -> Extension manager -> Extension uploader`," as indicated in Figure 4(c). Searched for the files storage location through the Daniel user access, discovered during the penetration test. In this case, we ran the command `find / | php-reverse-shell.php` to search the system root (/) filtering by the name of the file (grep) inserted. The file was identified in the `/var/www/pandora/pandora_console/extensions/` directory, executing it by searching the URL. An active connection was maintained on the chosen port to receive the shell from the server during the malicious file execution.

The public SSH key was saved in the authorized keys to provide easy authentication on the server through an SSH connection after receiving the server shell. The command `ssh-keygen` was used to generate the SSH public and private keys. For example, if there is no .ssh directory for the user Daniel, the directory can be created and the permissions set to 700; the authorized_keys file must have permission 600 and contain a copy of the public key. Finally, we authenticated the server via SSH connection, ending the exploration step.

D. Post-Exploration

The post-exploitation step seeks to escalate privileges to the highest degree, i.e., becoming a root user. A simple way to start post-exploration is to search for binaries that have Set User ID (SUID) permission to execute files with the permissions of other users. In this case, we searched for the execution of files that have root permissions running the command `find / -perm -u=s -type f 2>/dev/null`, which allows searching in the system root (/) for files with SUID permission (parameter `-perm -u=s`) and defined the file type (parameter `-type f`). As a result, it was identified that the file `pandora_backup` fits these conditions, as shown in Figure 5(a). Therefore, the file was used to escalate root privileges when executed, identifying an error in the response because `tar` does not fill files with a slash / by default (Figure 5(b)); it is necessary to rewrite the `tar` file to escalate privileges by `$PATH`.

Running a program from the command line on Linux operating systems results in a search for the programs binary in the system root with the help of the `$PATH` variable. By default, this variable fetches the binary from left to right. If a directory contains a malicious file with the same name as the search file, the operating system executes it instead of the original file. This approach was carried out to replace the `tar` file, identifying a directory with writable permission using the command `find / -writable -type d 2>/dev/null`. This command searches the operating system

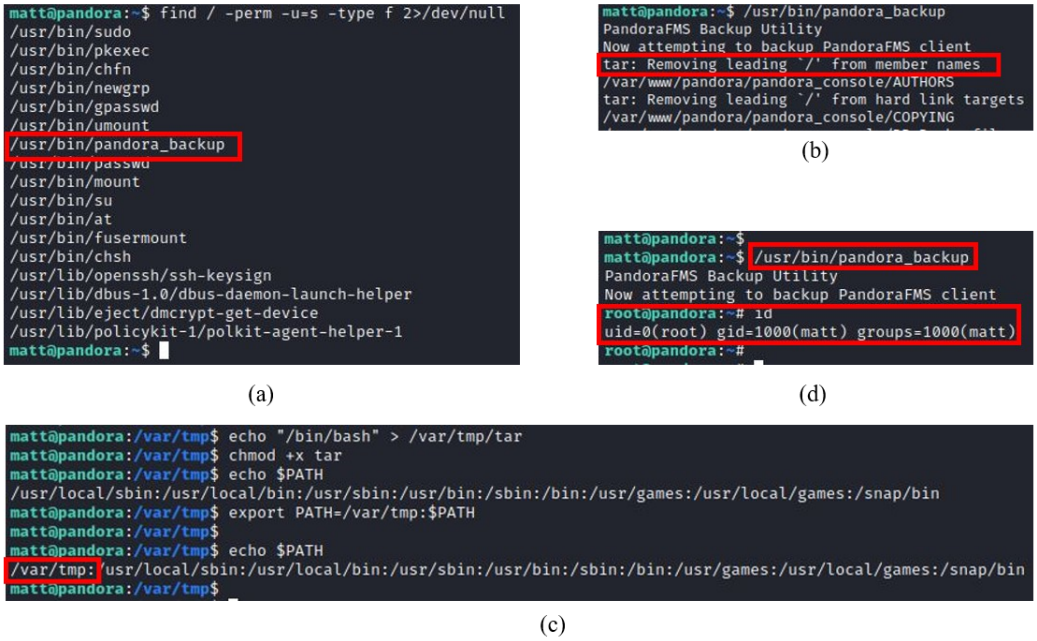

Fig. 5. (a) Binary searching with SUID permission, (b) $PATH privileges escalation identification, (c) $PATH variable changing with a malicious file creation for privileges escalation, (d) User privileges escalation.

root (/) for directory (parameter `-type d`) with writable permission (parameter `-writable`).

The commands `echo $PATH` were executed to check the directories defined for the variable `$PATH` and `export PATH=/tmp:$PATH` to concatenate the variable with the directory containing the malicious file. Therefore, the malicious `tar` file was created: `echo "/bin/bash" > /tmp/tar` to run a `shell` as `root` and obtain maximum user privileges operating system, giving full permission to the `tar` file created through the command `chmod +x /tmp/tar`, as shown in Figure 5(c). Execution of the `pandora_backup` file resulted in the reading of the malicious `tar` file, which promoted root privilege escalation to user. In Figure 5(d), the `id` command was used to verify the root privileges.

IV. Conclusions

This work aimed to demonstrate the importance of preserving operating systems, software, and application updating to avoid cyber attacks. Therefore, a pentest black box was performed on a local server containing an outdated version of the Pandora FMS software with SQL injection and remote file insertion vulnerabilities. As a result, it was possible to escalate root user privileges in the operating system. By considering the Pandora FMS version 743, the pentest conducted in this work would be able to achieve similar results since this release includes the vulnerabilities corrections.

Acknowledgment

The authors thanks the financial support of the Inatel Cyber Security Center (Centro de Segurança Cibernética do Inatel - CxSC Telecom) of the National Institute of Telecommunication (Instituto Nacional de Telecomunicações - Inatel) and Huawei for providing the means to support this work.

References

[1] C. G. da Internet no Brasil. Cartilha de Segurança para Internet, versão 4.0 / CERT.br. [Online]. Available: https://cartilha.cert.br/livro/cartilha-seguranca-internet.pdf
[2] OWASP. OWASP TOP10 – 2021. [Online]. Available: https://owasp.org/Top10/
[3] P. Global. Why Software Development Projects Fail. [Online]. Available: https://www.3pillarglobal.com/insights/why-software-development-projects-fail
[4] Kaspersky. O que é um ataque de dia zero? – Definição e explicação. [Online]. Available: https://www.kaspersky.com.br/resource-center/definitions/zero-day-exploit
[5] T. Branquinho and M. Branquinho, *Segurança Cibernética Industrial*, 1st ed. Alta Books, 2021.
[6] D. Swinhoe. 7 falhas de cibersegurança que custaram os empregos dos CISOs. [Online]. Available: https://itforum.com.br/noticias/7-falhas-de-ciberseguranca-que-custaram-os-empregos-dos-cisos/
[7] J. Korn. Falha de segurança do Log4j pode afetar toda a Internet. [Online]. Available: https://www.cnnbrasil.com.br/tecnologia/falha-de-seguranca-do-log4j-pode-afetar-toda-a-internet-o-que-voce-precisa-saber/
[8] D. Brinkrolf. Pandora FMS 742: Critical Code Vulnerabilities Explained. [Online]. Available: https://blog.sonarsource.com/pandora-fms-742-critical-code-vulnerabilities-explained
[9] C. Osborne. Multiple vulnerabilities in Pandora FMS could trigger remote execution attack. [Online]. Available: https://portswigger.net/daily-swig/multiple-vulnerabilities-in-pandora-fms-could-trigger-remote-execution-attack
[10] CVE. CVE-2021-32099. [Online]. Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-32099
[11] G. Weidman, *Penetration Testing: A Hands-On Introduction to Hacking*, 7th ed. Novatec, 2019.
[12] TheCyberGeek and dmw0ng. Pandora. [Online]. Available: https://www.hackthebox.com/
[13] ibnuuby. CVE-2021-32099. [Online]. Available: https://github.com/ibnuuby/CVE-2021-32099
[14] pentestmonkey. Php-Reverse-Shell. [Online]. Available: https://github.com/pentestmonkey/php-reverse-shell/blob/master/php-reverse-shell.php