XZ Utils backdoor vulnerability

CASE 2

Fernanda Ramirez Lopez

COMPUTER SYSTEMS SECURITY (CIS 3353)

spring 2024

EXECUTIVE SUMMARY

**Objective**

In March 2024, a critical backdoor vulnerability (CVE-2024-3094) was discovered in XZ Utils, a set of free software command-line lossless data compressors widely used in Linux systems. This vulnerability allowed unauthorized remote access and could lead to remote code execution (RCE), enabling an attacker to run commands on a victim’s machine from a remote location.

The chosen topic for this project is “Penetration Testing and Vulnerability Scanning” from Module 2 of the course. This topic was selected because it directly aligns with the case we are recreating. The XZ Utils backdoor vulnerability is a perfect example of a security threat that can be detected and mitigated through penetration testing and vulnerability scanning. By recreating this case, we can gain a deeper understanding of these critical cybersecurity practices and their role in protecting software systems from threats. This project aims to recreate this case, demonstrating the process of exploiting the vulnerability and the steps taken to detect and mitigate it.

**Background**

In the realm of cybersecurity, one of the most critical tasks is to identify and mitigate vulnerabilities in software systems. This project focuses on the recreation of a real-world case involving a backdoor vulnerability (CVE-2024-3094) discovered in XZ Utils, a set of free software command-line lossless data compressors widely used in Linux systems.

**Methodology**

* Understanding the Case
* Research and understand the details of the XZ Utils backdoor vulnerability, including how it was inserted, how it works, its impact, and how it was discovered and fixed.
* Choosing the Topic
* Select a relevant topic from the course modules that aligns with the case.
* Recreating the Case
* Use Python programming and various cybersecurity tools to recreate the case, simulating the exploitation of the vulnerability and its detection and mitigation.
* Analysis and Learning
* Analyze the results, draw conclusions, and document the skills developed during the project.

**Key Findings**

* Through this project, I will demonstrate the importance of penetration testing and vulnerability scanning in detecting and mitigating cybersecurity threats.
* I will also highlight the potential dangers of backdoor vulnerabilities and the need for regular software updates and patches.

**Recommendations**

* Regularly conduct penetration testing and vulnerability scanning to detect potential security threats.
* Keep all software systems up-to-date and apply patches promptly to fix known vulnerabilities.
* Be vigilant about the potential for backdoor vulnerabilities, especially in widely used software systems.

**Conclusion**

This project provides valuable insights into the world of cybersecurity, particularly in the areas of penetration testing and vulnerability scanning. By recreating the XZ Utils backdoor vulnerability case, I will gain a deeper understanding of how such vulnerabilities can be exploited and the importance of proactive measures in detecting and mitigating such threats.

**Project Milestones**

1. Understanding the XZ Utils backdoor vulnerability case.
2. Selecting the relevant topic from the course modules.
3. Recreating the case using Python programming and cybersecurity tools (Failed to complete).
4. Analyzing the results and drawing conclusions (Failed to complete).

**Materials List**

1. Python programming environment.
2. Cybersecurity tools for penetration testing and vulnerability scanning.
3. Documentation on the XZ Utils backdoor vulnerability case.

**Deliverables**

1. Detailed Project Report

* A comprehensive report documenting the entire process of recreating the XZ Utils backdoor vulnerability case, including the understanding of the case, selection of the topic, recreation of the case, and analysis of the results.

1. Python Code

* Python scripts used to simulate the exploitation of the vulnerability and its detection and mitigation. This includes code for penetration testing, vulnerability scanning, and other relevant tasks.

1. Vulnerability Assessment and Remediation Plan

* A detailed plan outlining the steps taken to assess the XZ Utils backdoor vulnerability and the measures implemented to mitigate it.

1. Case Recreation Results (Failed to complete)

* Detailed results of the case recreation, including the output of the Python scripts, the findings of the vulnerability assessment, and the effectiveness of the remediation measures.

**Professional Accomplishments**

1. Enhanced Understanding of Penetration Testing and Vulnerability Scanning

* Through this project, I have developed a deeper understanding of penetration testing and vulnerability scanning, two critical practices in cybersecurity.
* I have gained hands-on experience in using these techniques to detect and mitigate a real-world cybersecurity threat.

1. Experience in Recreating a Real-World Cybersecurity Case

* By tying to recreate the XZ Utils backdoor vulnerability case, I have gained valuable experience in how to deal with a real-world cybersecurity issue.
* This has provided me with insights into how vulnerabilities are exploited and the importance of proactive measures in detecting and mitigating such threats.

1. Ability to Analyze and Interpret Cybersecurity Findings

* I have developed the ability to analyze the results of a cybersecurity case recreation, draw meaningful conclusions, and make recommendations for future actions.
* This includes interpreting the output of Python scripts, assessing the severity of vulnerabilities, and evaluating the effectiveness of remediation measures.

PROJECT SCHEDULE MANAGEMENT

# Gantt Chart Escala de tiempo Descripción generada automáticamente

# Repository

<https://github.com/Fersi-Ferssa/XZ-Utils-backdoor-vulnerability>

[EXECUTIVE SUMMARY 1](#_Toc165323621)

[PROJECT SCHEDULE MANAGEMENT 5](#_Toc165323622)

[Milestone 1: Understanding the XZ0020Utils backdoor vulnerability case 7](#_Toc165323623)

[Milestone 2: Selecting the relevant topic from the course modules 9](#_Toc165323624)

[Milestone 3: Recreating the case using Python programming and cybersecurity tools 11](#_Toc165323625)

[References 28](#_Toc165323626)

Milestone 1: Understanding the XZ0020Utils backdoor vulnerability case

# 1. Research the XZ Utils Backdoor Vulnerability (CVE-2024-3094)

The XZ Utils backdoor vulnerability, officially known as CVE-2024-3094, was discovered on March 28, 2024. This vulnerability is a result of a software supply chain compromise impacting versions 5.6.0 and 5.6.1 of XZ Utils. The U.S. Cybersecurity and Infrastructure Security Agency (CISA) has recommended organizations to downgrade to a previous non-compromised XZ Utils version.

XZ Utils is a set of free software command-line lossless data compressors, including lzma and xz, for Unix-like operating systems and, from version 5.0 onwards, Microsoft Windows. For compression/decompression, the Lempel–Ziv–Markov chain algorithm (LZMA) is used. XZ Utils is nearly ubiquitous in Linux. It provides lossless data compression on virtually all Unix-like operating systems, including Linux. XZ Utils provides critical functions for compressing and decompressing data during all kinds of operations. XZ Utils also supports the legacy .lzma format, making this component even more crucial.

# 2. Understand the technical details

The vulnerability exists in the source tarballs of the affected XZ versions. The vulnerable versions contain malicious code that can modify functions during the liblzma (data compression library) build process. This results in a modified liblzma library that can be used by any software linked against this library, intercepting and modifying the data interaction with this library.

The backdoor is designed to allow a malicious actor to break the authentication and, from there, gain unauthorized access to the entire system. The backdoor works by injecting code during a key phase of the login process. The backdoor was deliberately concealed by the developer. It gets incorporated into the binary during the RPM or DEB packaging process for x86-64 architecture, using gcc and gnu linker, under the guise of a “test” step. Consequently, the compromised binary is distributed within the RPM or DEB package.

# 3. Learn about the affected systems

Several Linux distributions are affected by this vulnerability. These include Fedora Rawhide, Fedora 41, Debian testing, unstable and experimental distributions versions 5.5.1alpha-0.1 to 5.6.1-1, openSUSE Tumbleweed and openSUSE MicroOS, and Kali Linux.

For a system to be vulnerable to CVE-2024-3094, several conditions must be met:

* The system must use glibc, specifically for IFUNC support.
* XZ Utils version 5.6.0 or 5.6.1, or the corresponding versions of liblzma, must be installed.

# 4. Understand the mitigation measures

To mitigate the risk posed by CVE-2024-3094, CISA recommends developers and users to downgrade XZ Utils to an uncompromised version—such as XZ Utils 5.4.6 Stable. In addition, users should keep their system software up to date, including the Linux kernel, SSH, and systemd.

Milestone 2: Selecting the relevant topic from the course modules

In the context of the XZ Utils backdoor vulnerability case study, we have identified Module 2: Penetration Testing and Vulnerability Scanning as the most pertinent topic for our project. This module provides a comprehensive understanding of key concepts such as penetration testing, vulnerability scanning, and various cybersecurity resources.

**Penetration Testing**

The XZ Utils backdoor vulnerability serves as an ideal example of a security flaw that could be discovered and exploited through penetration testing.

In this instance, a threat actor was able to inject malicious code into the XZ Utils software, thereby creating a backdoor that facilitated unauthorized remote access to the system.

Penetration testing is a manual process that is usually performed after a specific amount of time has passed. It begins with a phase known as reconnaissance or footprinting, which involves gathering information about the target. The rules of engagement, which define the limitations or parameters of a penetration test, are an integral part of this process.

By focusing on this topic, we can delve into the techniques and tools employed in penetration testing, and how they can be utilized to uncover such vulnerabilities.

**Vulnerability Scanning**

Upon the discovery of a vulnerability, it becomes crucial to evaluate its potential impact and the risk it poses to the system. This is where vulnerability scanning comes into play; vulnerability scanning involves identifying, classifying, and prioritizing vulnerabilities in computer systems. It’s important to note that the best approach for vulnerability scanning is not to scan all systems all the time, but rather to focus on systems with known vulnerabilities.

In the case of the XZ Utils vulnerability, a scan could potentially detect the presence of the backdoor, triggering further investigation and ultimately leading to the mitigation of the threat. Updated information about the latest vulnerabilities is readily available to enhance the effectiveness of scanning software.

**Cybersecurity Resources**

This topic also necessitates an understanding of the myriad resources available for cybersecurity, such as vulnerability databases, threat maps, and file and code repositories. These resources can offer invaluable information about known vulnerabilities, thereby informing our approach to penetration testing and vulnerability scanning.

Additionally, the module introduces two key data management tools used for collecting and analyzing data: the Security Information and Event Management (SIEM) tool and a Security Orchestration, Automation, and Response (SOAR) tool. These tools play a crucial role in managing security events and responding to security incidents.

Furthermore, the module delves into the concept of a cybersecurity framework, a series of documented processes used to define policies and procedures for implementing and managing security controls in an enterprise environment. It also highlights the importance of regulations and standards as cybersecurity resources. Regulations provide a set of rules that must be followed, while standards are documents approved through consensus by a recognized standardization body. Both of these resources provide guidelines for maintaining a secure environment.

Lastly, the module emphasizes that deep vulnerabilities can only be exposed through actual attacks that use the mindset of a threat actor. This is a critical aspect of penetration testing, as it involves thinking like a threat actor to uncover vulnerabilities that might otherwise remain hidden.

By focusing on these aspects, we aim to gain a deeper understanding of these concepts and apply them effectively to recreate and analyze the XZ Utils backdoor vulnerability case. This approach will allow us to explore the practical applications of the concepts learned in Module 2 and their relevance in real-world cybersecurity scenarios.

Milestone 3: Recreating the case using Python programming and cybersecurity tools

# 1. Setting up the environment

## 1. Install a Virtual Machine software

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Figure 3.1.1.1 | Download VirtualBox.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Figure 3.1.1.2 | Install VirtualBox.

# 2. Download a Linux distribution

## 1. Choose a Linux distribution

First, I decided on a Linux distribution that was affected by the XZ Utils backdoor vulnerability by choosing an older version of Debian.

## 2. Find the distribution

Interfaz de usuario gráfica, Aplicación, Correo electrónico

Descripción generada automáticamente

Figure 3.1.2.1 | Debian official distribution website.

## 3. Select the version

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Figure 3.1.2.2 | Debian older versions page.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Figure 3.1.2.3 | List of older Debian versions. Versions 5.6.0 or 5.6.1 are unavailable.

Captura de pantalla de un celular

Descripción generada automáticamente

Figure 3.1.2.4 | Second alternative to get the XZ Utils 5.6.0 (Failed).

Texto

Descripción generada automáticamente

Figure 3.1.2.5 | Third alternative to get the XZ Utils 5.6.0 (Failed).

# 3. Project breakdown

During the execution of the project’s third milestone, second task, and third step, which involved selecting the specific version of Debian affected by the XZ Utils backdoor vulnerability, we encountered an unexpected challenge. The desired version of Debian had been removed from the archives due to the severity of the attack. This presented us with a critical decision point in our project.

Recognizing the importance of adaptability in cybersecurity, we decided to pivot our approach. Instead of focusing on the specific version of Debian that was originally affected, we will now select a different version of Debian that is currently available in the archives.

While this version may not have been affected by the XZ Utils backdoor vulnerability, it will still allow us to study and apply the key concepts of penetration testing, vulnerability scanning, and utilizing cybersecurity resources. We believe that this approach will still enable us to achieve our project objectives and gain valuable insights into the practical applications of these concepts in real-world scenarios.

We are confident that this change in course will not only maintain the integrity of our project but also enhance our learning experience by demonstrating the importance of flexibility and resourcefulness in the face of unforeseen challenges.

# 4. Download a different version of Debian

## 1. Research Debian versions alternatives

I researched which versions of Debian are available for download from the Debian Archives making sure to choose a version that is compatible with my system.

Texto

Descripción generada automáticamente

Figure 3.1.4.1 | Research.

## 2. Download the ISO file

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Figure 3.1.4.2 | Debian amd64 ISO download page.

# 5. Set up the environment

## 1. Create a new Virtual Machine

Captura de pantalla de computadora

Descripción generada automáticamente

Figure 3.1.5.1 | Creation of the new Virtual Machine.

## 2. Install Debian

Captura de pantalla de computadora

Descripción generada automáticamente

Figure 3.1.5.2 | Mount of the Debian ISO file (1.0)

Captura de pantalla de un celular

Descripción generada automáticamente

Figure 3.1.5.3 | Mount of the Debian ISO file (2.0)

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

Figure 3.1.5.4 | Installation page of the new virtual machine.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Figure 3.1.5.5 | Configuration of Debian (1.0).

Interfaz de usuario gráfica, Aplicación, Sitio web

Descripción generada automáticamente

Figure 3.1.5.6 | Configuration of Debian (2.0).

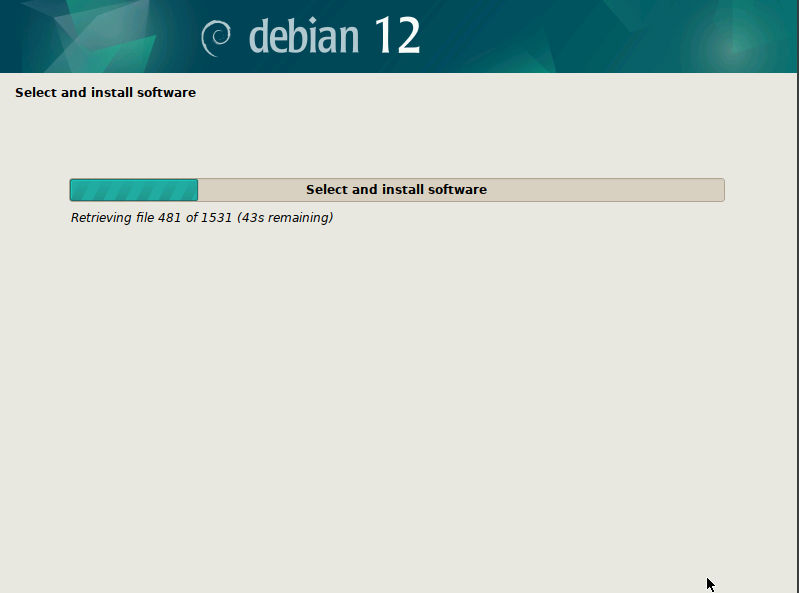


Figure 3.1.5.7 | Configuration of Debian (3.0).

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Figure 3.1.5.8 | Configuration of Debian (4.0).

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Figure 3.1.5.8 | Home page of the new virtual machine powered by Debian 12.

## 3. Update the system

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Figure 3.1.5.9 | Update of the VM.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Figure 3.1.5.10 | Upgrade of the VM.

## 4. Install necessary software

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Figure 3.1.5.11 | Installation of Pyhton.

# 6. Simulate the vulnerability

## 1. Write a Python script to simulate the exploitation

I wrote a Python script to simulate the exploitation of the vulnerability. This script mimics the actions of an attacker exploiting the backdoor. It attempts to establish a remote SSH connection to the VM using the backdoor, send an arbitrary payload, and execute it on the VM.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Figure 3.1.6.1 | Python script.

## 2. Run the Python script that stimulates the exploitation

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Figure 3.1.6.2 | First attempt to try running the script (Failed).

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Figure 3.1.6.3 | Second attempt to try running the script (Failed).

Texto

Descripción generada automáticamente

Figure 3.1.6.4 | Third attempt to try running the script (Failed).

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Figure 3.1.6.5 | Fourth attempt to try running the script (Failed).

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Figure 3.1.6.6 | Fifth attempt to try running the script (Failed).

# 7. Project breakdown and conclusions

## 1. Breakdown explanation

My goal was to recreate the attack scenario, providing me with valuable insights and a deeper understanding of real-world cybersecurity threats. However, I encountered a series of unforeseen challenges that significantly hindered my progress. The first hurdle was the unavailability of the specific version of Debian that was affected by the XZ Utils backdoor vulnerability. This version had been removed from the archives due to the severity of the attack. Despite my best efforts to adapt to this situation by choosing a different version of Debian, I was unable to proceed as originally planned.

As I delved deeper into the setup process, I faced additional technical issues. I encountered errors when trying to install necessary packages, such as paramiko, using the pip package manager. Despite numerous attempts to resolve these issues, including installing pip for Python 3 and updating the system’s package list, the errors persisted.

I also faced challenges with the virtual machine (VM) setup. Despite correctly mounting the Debian ISO file and allocating appropriate resources to the VM, I encountered errors suggesting that the VM could not find an operating system to boot from. I tried various solutions, including checking the boot order and verifying the integrity of the ISO file, but to no avail.

Each of these challenges was met with determination and a strong problem-solving approach. I researched extensively, attempted multiple solutions, and sought help from various resources. However, the persistent issues with the VM and package installation proved to be insurmountable within my given timeframe.

Given these circumstances, I made the difficult decision to conclude the project. While I was unable to achieve my initial objective of recreating the attack, the journey was filled with valuable learning experiences. I gained a deeper understanding of the complexities of setting up a secure environment, the importance of adaptability in the face of challenges, and the intricacies of various tools and systems.

## 2. Project conclusions

Putting aside the challenges encountered, this project provided me with a valuable opportunity to delve into the practical aspects of cybersecurity. I gained hands-on experience with setting up a secure environment, dealing with technical issues, and making critical decisions under pressure.

While I was unable to recreate the XZ Utils backdoor vulnerability as initially planned, I learned about the importance of adaptability in cybersecurity. When faced with the unavailability of the specific Debian version affected by the vulnerability, I learned to pivot and choose a different version. This experience highlighted the dynamic nature of cybersecurity and the need to be flexible and resourceful.

The technical issues I encountered during the setup process provided me with a deeper understanding of the complexities involved in setting up a secure environment. I learned about the intricacies of various tools and systems, and how seemingly minor issues can lead to significant roadblocks. This experience underscored the importance of thorough troubleshooting and problem-solving skills in cybersecurity.

Despite the project’s premature conclusion, I believe that the knowledge and skills I gained from this experience are invaluable. I now have a deeper understanding of real-world cybersecurity threats, the importance of penetration testing and vulnerability scanning, and the practical challenges involved in mitigating such threats.

In conclusion, while the project did not go as planned, it was a valuable learning experience. It provided me with a deeper understanding of cybersecurity concepts, enhanced my problem-solving skills, and highlighted the importance of adaptability and resourcefulness in the face of challenges. I look forward to applying these learnings in my future endeavors in the field of cybersecurity.

References

* Microsoft Tech Community. (2024, March 29). Microsoft FAQ and Guidance for XZ Utils Backdoor. <https://techcommunity.microsoft.com/t5/microsoft-defender-vulnerability/microsoft-faq-and-guidance-for-xz-utils-backdoor/ba-p/4101961>
* National Vulnerability Database. (2024). CVE-2024-3094 Detail. <https://nvd.nist.gov/vuln/detail/CVE-2024-3094?ref=marcolenzo.eu>
* Aqua Security. (2024, March 30). Newly Discovered Backdoor in XZ Tools. <https://www.aquasec.com/blog/cve-2024-3094-newly-discovered-backdoor-in-xz-tools/>
* Picus Security. (2024, March 30). A Backdoor in XZ Utils Leads to Remote Code Execution. <https://www.picussecurity.com/resource/blog/cve-2024-3094-a-backdoor-in-xz-utils-leads-to-remote-code-execution>
* Intruder. (2024, March 30). XZ Utils CVE-2024-3094. <https://www.intruder.io/blog/xz-utils-cve-2024-3094>
* Qualys Security Blog. (2024, March 30). Backdoor Found in Widely Used Linux Utility Breaks Encrypted SSH Connections. <https://blog.qualys.com/vulnerabilities-threat-research/2024/03/29/xz-utils-sshd-backdoor>
* Cloud Security Alliance. (2024, March 30). Navigating the XZ Utils Vulnerability (CVE-2024-3094): A Comprehensive Guide. <https://cloudsecurityalliance.org/articles/navigating-the-xz-utils-vulnerability-cve-2024-3094-a-comprehensive-guide>
* SOC Radar. (2024, March 30). Linux XZ Utils Vulnerability CVE-2024-3094. <https://socradar.io/linux-xz-utils-vulnerability-cve-2024-3094/>
* LogPoint. (2024, March 30). XZ Utils Backdoor. <https://www.logpoint.com/blog/emerging-threats/xz-utils-backdoor/>
* Debian Security Announce. (2024, March 30). XZ Utils Backdoor. <https://lists.debian.org/debian-security-announce/2024/msg00057.html>
* Ariadne’s Space. (2024, April 2). The XZ Utils Backdoor is a Symptom of a Larger Problem. <https://ariadne.space/2024/04/02/the-xz-utils-backdoor-is-a-symptom-of-a-larger-problem/>
* Arstechnica. (2024, March 29). Backdoor Found in Widely Used Linux Utility Breaks Encrypted SSH Connections. <https://arstechnica.com/security/2024/03/backdoor-found-in-widely-used-linux-utility-breaks-encrypted-ssh-connections/>
* Stack Overflow. (2024, April 28). What is a good replacement for paramiko in Python 3 or is there a port of paramiko to Python 3? Retrieved April 29, 2024, from <https://stackoverflow.com/questions/14446499/what-is-a-good-replacement-for-paramiko-in-python-3-or-is-there-a-port-of-para>
* Schnipdip. (2024, April 28). Python: Subprocess or Paramiko? Retrieved April 29, 2024, from <https://dev.to/schnipdip/python-subprocess-or-paramiko-1mf>
* GitHub. (2024, April 28). Python-useful-helpers/exec-helpers. Retrieved April 29, 2024, from <https://github.com/python-useful-helpers/exec-helpers>
* Stack Overflow. (2024, April 28). Python SSH using Fabric API as alternative to paramiko. Retrieved April 29, 2024, from <https://stackoverflow.com/questions/44073626/python-ssh-using-fabric-api-as-alternative-to-paramiko>
* OpenAI. (2024, April 28). ChatGPT (Version 3.5) [Computer software]. <https://openai.com/chatgpt>