4/21/2023

Signed Declaration

Bradley Cumming

S2140163 BCUMMI202

Ethical Hacking Coursework: Penetration Test of Tekkadan Inc.

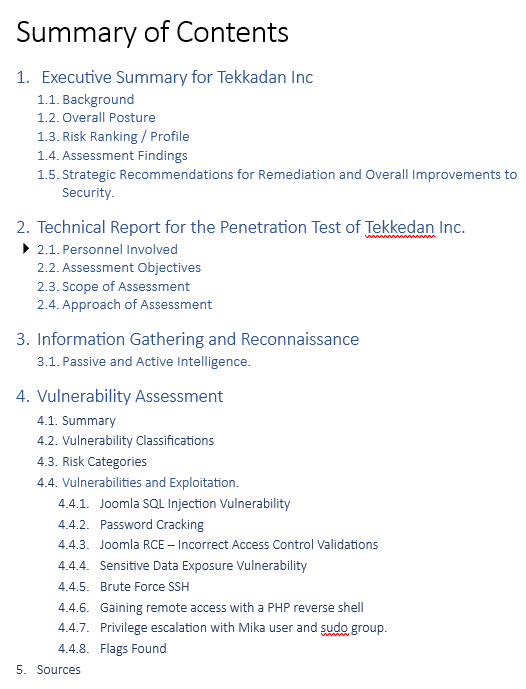
“This piece of coursework is our own original work and has not been submitted elsewhere in

fulfilment of the requirement of this or any other award.”

Signed Name: Bradley Cumming

Signature: Bradley Cumming

Date: 21/04/2023



Executive Summary for Tekkadan Inc [3]

# Background

## This penetration test aimed to assess Tekkadan Inc's security by identifying vulnerabilities and weaknesses in their network. Utilizing a systematic hacking methodology, the test involved network scans, vulnerability scanners, and an Exploit Confirmation Phase to validate vulnerabilities and evaluate their potential impact. This report summarizes Tekkadan Inc's security posture, offering an analysis of vulnerabilities, weaknesses, and attack vectors, as well as remediation recommendations. Implementing these improvements can significantly bolster Tekkadan Inc's protection and minimize potential cyber risks.

## Overall Posture

Overall, we believe that the security posture of Tekkadan Inc’s networks considerably low. The test has pointed out numerous weaknesses and vulnerabilities in the network and its security indicating it has some significant improvements to be made to reduce the risk of any future cyber-attacks on the Tekkadan Inc. Various levels of vulnerabilities were found, mostly being high-risk and could be exploited, allowing access to confidential information, servers and being able to take control of the system. We recommend that Tekkadan Inc focus on implementing the recommended remediations.

## Risk Ranking / Profile

Diagram

Description automatically generated

Figure 1 Security Risk Rating Scale [1]

We have given the Tekkadan Inc security an overall risk score of “Extreme” at level 13. Tekkadan Inc is at an extreme risk of having their security controls being compromised leading to the potential catastrophic failure and takedown of the website being hosted on the system. [1] There were various high-risk vulnerabilities found in the test including SQL Injection, RCE and critical usernames and passwords being visible on the webpages open to the public. There were also a few but not as many minor vulnerabilities, however, many of the weaknesses when exploited lead to the access to confidential information, files, and control of the system.

## Assessment Findings

Figure 2 Vulnerability Risk Factors

During the testing period, our tester identified and successfully exploited eight vulnerabilities and weaknesses within the target system. Of these, seven vulnerabilities resulted in the exposure of confidential information or allowed us to gain direct control over the system by accessing high-privilege user accounts. The following is a list of the found vulnerabilities and their risk ranking.

High Risk:

* Joomla SQL Injection
* Joomla RCE – Access Control Validations
* Sensitive Data Exposure
* Brute Force SSH
* Privilege escalation
* Remote Access with Reverse Shell

Medium Risk:

* Password Cracking

Low Risk:

* Flags Found

## Strategic Recommendations for Remediation and Overall Improvements to Security.

* It is critical that Tekkadan, update and patch their entire system in particular the Joomla server. Ensure that the operating systems and network devices are also up to date. Having up to date software and services drastically reduces the risk of vulnerabilities being exploited against the client. Patch management is a very important task which helps maintain the security integrity through regular testing and installation of up-to-date software.
* Implementing two factor authentication is another import security remediation which should be incorporated into the client’s day to day workflow. Without 2FA, any malicious attacker that gains access to a user’s account details can get into their account with no authorisation, with 2FA. Those attackers will find it very difficult to gain access even with the detail as they will need usually require access to something the accounts owner has on themselves at the time of login.
* Implementing strong passwords with password managers, means passwords are difficult to crack, if they are compromised. Passwords are also not re-used, which is a case in Tekkadan Inc’s system, that the same credentials were reused in various applications making it easy to gain unauthorised access and control.
* The implementation of firewalls is critical, as Tekkadan Inc currently lacks any in place to monitor, detect, and block malicious traffic. Firewalls filter traffic, blocking and rejecting unauthorized access attempts. They are essential tools for system security and a necessary addition to Tekkadan Inc's infrastructure. Access controls and user management should be reviewed regularly by the client to guarantee that no accounts possess excessive access to unauthorized functions. Authentication, authorization, and accounting can ensure that users only access the tools for which they have clearance. It is recommended that Tekkadan Inc employ a Discretionary Access Control Model, as the system's smaller size allows the owner to assign user privileges directly.
* Regular network and Packet monitoring will help identify any malicious traffic coming into the system and will allow the security team to mitigate and block any potential attackers from causing damage to the system*.* Intrusion detection systems can automate this process and are highly recommended.
* Data encryption is a vital method for securing data in transit, such as during user login. Attackers can view unencrypted files and data transferred across the network, potentially stealing, or exploiting this data against the client. Implementing encryption safeguards sensitive information from unauthorized access.

Technical Report for the Penetration Test of Tekkedan Inc.

## Personnel Involved

The penetration test on Tekkadan Inc (the client) was conducted by the sole tester, Bradley Cumming. Who had no access to the team at Tekkadan Inc during the period of the testing. Bradley’s contact details can be found in the contact information at the top of the report.

The penetration test took place at the residence of the tester, using a computer with virtual box hosting a network including the Attacking machine, Router and Tekkadan Target of Engagement. Topology Shown here:

Diagram

Description automatically generated

There were no integral assets provided by each party to conduct the testing. Other than the information provided in the scope specification and the download link to the Target of Engagement.

## Assessment Objectives

The assessment was performed as an internal penetration test against the Tekkadan Inc Server with the objective to validate the security measures taken by Tekkadan Inc and to point out vulnerabilities and their fixes. The assessment will simulate a real-life penetration test on a client, Tekkadan Inc, through using a selection of ethical hacking methodologies, techniques, and tools by the tester. It is required to identify, evaluate, and exploit all possible vulnerabilities on the Target of Engagement, with association to the corresponding hacking stages. The objective is to provide at least two vulnerabilities identified, with the process of each attack vector being explained and how each vulnerability can be remediated.

## Scope of Assessment

The scope of this assessment was a “grey box” hacking assessment therefor limited information was provided up front. The information on the target included: Ubuntu OS, and a link to the Target of Engagement. There was no IP information provided. The assessment took place in a virtual internal network, hosted on the tester’s machine, there were no external networks tested on this assessment.

## Approach of Assessment

A methodical approach was taken during the period of the test, making use of the key concepts behind the hacking process: Information gathering; Vulnerability Assessment; Exploitation / Vulnerability Confirmation; Post Exploitation and Maintaining Access, each will be further explained in the coming report.

The tools used during the assessment were:

* Nmap
* Ffuf
* Gobuster
* Sqlmap
* John
* Metasploit
* Searchsploit
* Cewl
* Python
* OpenVAS

Information Gathering and Reconnaissance

During the information gathering and reconnaissance stage of the penetration test, the tester gathers as much information about the target as they can which will aid them in the active phase of the assessment. This phase helps build a map of the network which the tester will use to plan the steps to their attacks.

## Passive and Active Intelligence.

As the Target of Engagement, came as a packaged VM image. There was limited passive intelligence that was able to be conducted. Once the network was setup including the ToE it necessary to find the ip address which the ubuntu target system was hosted. Using the network mapper (nmap) tool, it is possible to scan the network for live hosts and therefore find the IP address of the target system.

Using **ifconfig** to find the attacker system host ip, which can be used to find the live hosts through a subnet host ping scan of the attackers ip. In the below images we see that the attackers ip address is: 192.168.1.103. Therefore can **use nmap -sn 192.168.1.1/24** to scan the network for live hosts [14]:

Graphical user interface, text, application

Description automatically generated

Figure ifconfig

Text

Description automatically generated

Figure Subnet Scan

Using passive recon, of the host system: it was possible to find the ToE IP: **192.168.1.104**.

Active reconnaissance allows the tester to gain detail information about the target, based on live scans on the target network. This information, such as running applications, can be used to start vulnerability analysis for potential exploits on the system. In this assessment, the tester used the following tools and reconnaissance techniques to deeper research about the system (unknown at this point in the test):

* Nmap – To discover open ports, banner scans, operating system versions and live applications on the host.
* Gobuster and Dirb– To discover hidden directories within the system.
* Joomscan [15]

Text

Description automatically generated

Figure Nmap Target Scan

Running the command (**nmap -sV -sC -A -p- -O 192.168.1.104**) shown in figure 2:

* -sV: Probes open ports to determine the service / version information.
* -sC: Runs default nmap scripts.
* -A: Scans for OS, version detection, script scanning and traceroutes.
* -p-: Scans all ports.
* -O: Enables OS detection

The following information was found in the first scan of the system.

* The Operating System Version Running: Ubuntu Linux version 4.15 – 5.6.
* Port 22 open, running OpenSSH.
* Port 80 open, running a http 2.4.18 Apache service. Likely a website and will be the first place to look on the vulnerability assessment stage.
* Application is running the Joomla Content Manager.
* Discovered various directories, within the http service, such as administrator page.
* MAC Address of the system, potential spoofing attacks.

**Graphical user interface, text

Description automatically generated**

Figure Dirb scan

Dirb is used to enumerate directories within the server, potentially revealing any hidden pages that the user should not have access to. You can see the above directories which were found to exist in one the website hosted on port 80.

**Graphical user interface, text, application

Description automatically generated**

Figure GoBuster Directory Scan

GoBuster is also used to enumerate directories within the server and was run alongside dirb to enumerate all available pages within the http application.

**Text

Description automatically generated**

Figure Joomscan

Joomscan is the comprehensive OWASP Joomla scanner. The scan detected the Joomla version running being: Version 3.7.0. Among other directories which have also been covered above. The version of joomla will be a key component in finding vulnerabilities in the application which we can exploit later in the test.

From the tools and enumeration of the IP address. A few key directories were found which should not be easily accessible and expose information that an attacker can use against the target system:

/administrator /web.config.txt /robots.txt /administrator/manifests/files/joomla.xml

Vulnerability Assessment

## Summary

The vulnerability assessment’s purpose is to identify potential weaknesses which can then be used against the system as an exploit. The findings in this section highlight any discovered vulnerabilities, their risk classification and how they can be fixed. This section will provide information to Tekkadan Inc to improve and secure their system from potential attacks.

## Vulnerability Classifications

Vulnerability classifications are key when reporting vulnerabilities, as they provide you with the knowledge of which vulnerabilities are likely to cause the most harm to your organisation if they were to be exploited. This report will score the vulnerabilities found based on the Common Vulnerability Scoring System, Potential impact, and Exploitability.

The following table provides a visual representation of each score:

Table

Description automatically generated

Figure Vulnerability Levels [1]

* High: A vulnerability that if exploited will cause severe damage to Tekkadan Inc’s confidentiality, integrity, and availability of assets.
* Medium: A vulnerability that if exploited can potentially impact the Tekkadan Inc’s confidentiality, integrity, and availability of assets.
* Low: A vulnerability that if exploited will have a low level of impact the Tekkadan Inc’s confidentiality, integrity, and availability of assets. **[1]**

## Risk Categories

The following information is an extract, which briefly describes the risk categories in a pen test. Taken from: <https://www.pulsar-it.de/Pentest_Report.pdf> [1]

**Graphical user interface, text, application

Description automatically generated**

Figure Risk Categories

## Vulnerabilities and Exploitation.

Title: Joomla SQL Injection Vulnerability (CVE-2017-8917) [2]

Classification: HIGH

Vulnerability Summary:A critical SQL injection vulnerability has been identified in Joomla version 3.7, specifically within the publicly accessible com\_fields component. This security weakness stems from the improper handling of administrator-provided fields, which creates a potential attack vector for malicious actors seeking to exploit the vulnerability.

Discovery Method:Using the Searchsploit tool and specifically identifying Joomla 3.7 within the search option identified the known SQL Injection vulnerability.

A screenshot of a computer

Description automatically generated with medium confidence

Vulnerability Manipulation: Successful exploitation could result in unauthorized access to sensitive database information, potentially compromising the confidentiality and integrity of the Tekkadan Inc’s data.

Remediation Recommendations:The identified SQL injection vulnerability affects Joomla services that are running versions between 3.7.0 and 3.7.1. Therefore, it is essential that Tekkadan Inc upgrades their system software to the latest version.

### Findings and Exploitation Walkthrough:

|  |  |  |  |
| --- | --- | --- | --- |
| Username | | Email | Password Hash |
| Orga | | admin@tekkadan,com | $2y$10$oYDMd1CT.S7HBsvx8xXFi.ev3YeGMgTdgEX6tqYIlBv1joLZ4J78m |
| Biscuit | Biscuit@tekkadan.com | | $2y$10$iQ1ZrFmB11naV4dhXb08gOtPhLx6KPheEFXjYacBac7YBo6/vuG/G |
| Mika | Mikazuki@tekkadan.com | | $2y$10$iQ1ZrFmB11naV4dhXb08gOtPhLx6KPheEFXjYacBac7YBo6/vuG/G |

Graphical user interface, text

Description automatically generated

Figure SQL Injection Proof of Concept

The source code provided by Meteus Lino [2] provides the proof of concept and the url that is vulnerable. The exploitation process involves several steps that, once executed, lead to the exposure of sensitive information from the username table, including usernames, emails, password hashes, and user IDs. The entire exploitation process took approximately 10 minutes to complete, showcasing the relative ease and efficiency of this damaging exploit. [12]

**SQLMap: Finding the Table Names

**

Figure SQL Injection Databases

We can see that the SQL injection works immediately and reveals the tables: “information\_schema” and “joomla”. The command used for this is: **sqlmap -u "http://192.168.1.104/index.php?option=com\_fields&view=fields&layout=modal&list[fullordering]=updatexml" --risk=3 --level=5 --random-agent --dbs -p list[fullordering]**

Calendar

Description automatically generated with medium confidenceGraphical user interface, text

Description automatically generated

Figure 14 Joomla

Figure 13 Information\_Schema

Using the following commands, it is possible to take step further into the database this time displaying the table names within each database:

* **sqlmap -u "http://192.168.1.104/index.php?option=com\_fields&view=fields&layout=modal&list[fullordering]=updatexml" --risk=3 --level=5 --random-agent -D information\_schema --tables -p list[fullordering]**
* **sqlmap -u "http://192.168.1.104/index.php?option=com\_fields&view=fields&layout=modal&list[fullordering]=updatexml" --risk=3 --level=5 --random-agent -D joomla --tables -p list[fullordering]**

**Text

Description automatically generated**

Figure 15 Joomla Columns

Again, we take another step deeper, this time choosing to go into the “#\_\_user” table in the Joomla database displaying important column names such as name and password.

* **sqlmap -u "http://192.168.1.104/index.php?option=com\_fields&view=fields&layout=modal&list[fullordering]=updatexml" --risk=3 --level=5 --random-agent -D joomla -T '#\_\_user' --columns -p list[fullordering]**

**Text

Description automatically generated**

Figure 16 Data Dumping emails usernames and passwords]

Finally, we dump the column fields to access the confidential information stored in the database. In this case we find out some critical login information that we can use to exploit the system further. Command used:

* **sqlmap -u "http://192.168.1.104/index.php?option=com\_fields&view=fields&layout=modal&list[fullordering]=updatexml" --risk=3 --level=5 --random-agent -D joomla -T '#\_\_users' -C email id name password username -dump -p list[fullordering]**

Title: Password Cracking

### Classification: MEDIUM

Vulnerability Summary: From the previous SQL injection we uncovered 3 password hashes for each user account on the system. Using the tool: JohnTheRipper, we can try to crack the hashes to find out the passwords to access the accounts.

Discovery Method: Uncovering password hashes in the previous step means as an attacker we will try to crack them to gain access to the cracked information.

Vulnerability Manipulation: Cracking the hashes gives the attacker access to the confidential information which has been hashed for a reason so nobody can access it.

Remediation Recommendations: It is highly recommended that passwords are not common words / phrases etc. Passwords should be a variation of strings, numbers, and special characters, which would prevent password cracking through wordlists. These complex passwords can be stored in a password manager, meaning passwords can be complex and different every time, making it practically impossible for an attacker to crack the passwords.

### Findings and Exploitation Walkthrough:

|  |  |  |
| --- | --- | --- |
| **Username** | **Password** | **Hash** |
| biscuit | cookie | $2y$10$iQ1ZrFmB11naV4dhXb08gOtPhLx6KPheEFXjYacBac7YBo6/vuG/G |

Now that we have gained access to the usernames and password hashes, we use a password cracking tool known as JohnTheRipper to crack the hashes. After making a password.txt which only contains the 3 password hashes. We can run the command: **sudo john –crack-status password.txt**

Text

Description automatically generated

Figure 17 JohnTheRipper

After a long wait we crack one password. However, could not crack the other two. There will need to be another way to crack those hashed. Waiting longer may crack them but I decided to move on with the test using the newly cracked password.

We try the password on the /administrator page for each user and gain access using the biscuit:cookie credentials.

Graphical user interface, text, website

Description automatically generated

Figure 18 Access to the Joomla CMS

Title: Joomla RCE – Incorrect Access Control Validations (CVE-2020-11890)

### Classification: HIGH

Vulnerability Summary: Once we gained access to the biscuit administrator account, we found this vulnerability on Joomla versions prior to 3.9.17 which have a lack of input validation in the user’s table. This can result in a broken Access Control List (ACL) configuration, which may allow remote code execution for users with root privileges.

### Discovery Method:

**Graphical user interface, text, application, email

Description automatically generated**

Figure 18 Identifying the Vulnerability

Using the OpenVAS report we have previously discussed, we identified the CVE-2020-11890 along with a working exploit for the vulnerability at: <https://github.com/HoangKien1020/CVE-2020-11890> [3]

Vulnerability Manipulation:Through the improper admin validations, any attacker can exploit this vulnerability using the python script, which allows them to escalate to super user access if they already have access to an administrator account on the system. This would grant them full control over the website and its content, allowing them to perform various malicious activities, such as unauthorized data access, website defacement, or data manipulation.

Remediation Recommendations:Regularly update your Joomla installation to the latest version, which includes patches addressing known vulnerabilities. Employ strong, unique passwords for each user account to minimize the risk of unauthorized access. Monitor and review user access privileges, ensuring that only authorized individuals have access to the system. Implement two-factor authentication (2FA) as an additional layer of security to enhance account protection.

### Findings and Exploitation Walkthrough:

Was able to escalate privileges to gain a super-user account on the system, giving me full access to the Joomla CMS.

**Text

Description automatically generated**

Figure 19 Gaining Super User Access

Exploiting this vulnerability was simple and only one command. After downloading the exploit script from HoangKien1020 on github, we simply entered the following command: **python cve202011890.py -url** [**http://192.168.1.104:80**](http://192.168.1.104:80) **-u biscuit -p cookie**

**Graphical user interface, text, application, email

Description automatically generated**

Figure 20 Proof of SuperUser Access

The screenshot above shows the user management tab on the CMS, which now includes the newly created BCUMMI202 Super User.

Title: Sensitive Data Exposure Vulnerability

### Classification: HIGH

Vulnerability Summary: The Tekkadan Inc web pages contain an information disclosure vulnerability as they revealed sensitive information including username and passwords on the accessible webpages of the website.

Discovery Method:This vulnerability was found when enumerating the website with the Cewl keyword tool.

Vulnerability Manipulation: Attackers can use these keywords to try and login to accounts that they aren’t supposed to. In this case, the attackers can gain access to the 3 usernames and their passwords, through cracking the hashes with JohnTheRipper.

### Remediation Recommendations:

Tekkadan Inc, should remove any sensitive information that might contain usernames and passwords which can be used to exploit the system.

### Findings and Exploitation Walkthrough:

|  |  |
| --- | --- |
| **Username** | **Password** |
| Orga | isaribi |
| Biscuit | cookie |
| Mika | barbatos |

Text

Description automatically generated

Figure 21 Cewl Wordlist

Using cewl keyword tool, and the following commands we generated a list of possible keywords that could be exploited on the website: **cewl** [**http://192.168.1.104**](http://192.168.1.104) **> cewl.txt; cewl --lowercase** [**http://192.168.1.104**](http://192.168.1.104) **>> cewl.txt**

Text

Description automatically generated

Figure 22 JohntheRipper Cracking Passwords

From there we tried using JohnTheRipper to crack the remaining passwords, and we are successful. We now have access to all the users including our own bcummi202 superuser account.

Title: Brute Force SSH

### Classification: HIGH

Vulnerability Summary: OpenSSH on the server is setup so that the user can login using passwords, which we have previously found. Making it vulnerable to a brute force attack.

Discovery Method: Testing login usernames found previously meant we discovered that we could access the ToE via ssh username@address with the password. However, this also means that even without the passwords, an attacker could exploit the SSH with a tool like hydra. The below shows that you are able to login to the server with a key or password.

Text

Description automatically generated

Vulnerability Manipulation: An attacker can try brute force their way into the server, gaining access to the files, directories, and everything confidential such as users and passwords.

Remediation Recommendations: Disabling the ability to access SSH with a text password, enable only authentication with a key. Enable 2FA when logging into the server. Monitor the logging logs and traffic to identify any suspicious login attempts.

Findings and Exploitation Walkthrough:

**A picture containing text

Description automatically generated**

Using the newly found usernames and passwords for an example and proof that brute forcing the SSH is possible. We identify the usernames and passwords for the 3 accounts we know of previously. The command we use to execute this exploit is**: hydra -L username.txt -P cewl.txt 192.168.1.104 ssh -t 50**.

Where the username.txt is a file containing the usernames and the cewl.txt file includes the keywords generated in the previous step to crack the hashes.   
  
This however can be exploited without the knowledge of any usernames or passwords by changing out the files with wordlists such as rockyou.txt for the passwords and a common username wordlist to try brute force entry.

Title: Gaining remote access with a PHP reverse shell

### Classification: HIGH

Vulnerability Summary: The reverse shell vulnerability allows an attacker to execute malicious code to the server via a remote connection to the attacker machine. Using the Joomla CMS template, which was accessed through the super user we previously gained access to, we were able to add a template webpage which hosted a reverse shell also known as a backdoor to the server. From here can call the reverse shell and gain access to the system remotely.

Discovery Method: Once you gain access to a system attackers need to be able to remotely access the server without repeating the steps they initially used. Reverse shells are commonly used for this.

Vulnerability Manipulation: Where there is a reverse shell or backdoor on the system, this attacker can access the backend of the system whenever they want. This allows the attacker to explore the system, its files and configuration at their wishes. It also allows them to take steps to escalate privileges to users such as root, steal data and remotely control the system.

Remediation Recommendations: To protect against reverse shell attacks, it is essential to implement strong security measures, such as system software updates, implementing firewalls and installing intrusion detection systems to detect suspicious activity.

### Findings and Exploitation Walkthrough:

Once the attacker has access to the Joomla super user, they can exploit the Joomla templates by uploading a reverse shell to gain access to the ubuntu server from the attacker machine.

**Text

Description automatically generated**

Figure 23 Reverse Shell Editing

Editing the php-reverse-shell to include the attacker machines ip and port which will be opened, found at: /usr/share/webshells/php/php-reverse-shell.php.

**A screenshot of a computer

Description automatically generated**

Figure 24 Uploading the reverse shell.

With superuser access to the Jooma CMS we can upload the updated reverse shell to the templates, which can then be accessed to initiate the exploit.

**A screenshot of a computer

Description automatically generated**

Figure 25 Establish Netcat Listener

We then establish a Netcat listener to receive the connection from the target host server, and subsequently access the URL of the template.

**Text

Description automatically generated**

Figure 26 Gaining access with Reverse Shell

Upon successful connection, the terminal will display an update as shown in the screenshot. We then used to 'uname -a' command to determine the system version in operation. From here an attacker can identify potential exploits associated with the specific system version, leading to gaining access to the root user through known kernel vulnerabilities.

**Text

Description automatically generated**

Figure 27 Enumeration Scan

After successful exploitation of the reverse shell, I used an enumeration scan known as LinEnum.sh to gain more information about the system, with the above output.

### Classification and business impact:

Title: Privilege escalation with Mika user and sudo group.

### Classification: HIGH

### Summary: This privilege escalation makes use of basic Linux command tools to add a user with root access. As we have gained access to root this is an easy task.

**A screenshot of a computer

Description automatically generated with medium confidence**

Figure 28 Testing User Logins

It was discovered that the previously identified usernames and passwords for the Joomla system are also the same used in authentication to access the Ubuntu ToE. This presents a security risk, as the compromise of one system may inadvertently grant unauthorized access to the other.

**A screenshot of a computer

Description automatically generated with medium confidence**

Figure 29 Gain root access

We then tried to gain super user access by using the **sudo su** command on each logged-in account to assess whether they have root access. Upon examination, it was determined that the account associated with Mika had root user privileges, highlighting a potential security concern.

**Text

Description automatically generated**

Figure 30 Adding User

A new user account, bcummi202, was created, which will be used to receive the sudo privileges.

**Text

Description automatically generated**

Figure 31 Sudoers Files

**Text

Description automatically generated**

Figure 32 Sudo Privileges

Using the command **sudo visudo** the new user was added to the sudoers file, thereby granting complete access and control of the system. It is now possible to ssh into the Tekkadan ToE with the new root group user, bcummi202.

Title: Flags Found

Classification: Low

Vulnerability Summary: Throughout the Tekkadan ToE assessment, multiple flags were identified, which, although not directly representing vulnerabilities, serve as indicators of the attacker's proficiency. These flags represent ‘trophies’ within the machine, showcasing the attacker's skill and success in navigating the system.

### Findings and Exploitation Walkthrough:

A screenshot of a computer

Description automatically generated

Figure 33 Flag 1

The first flag was easy to find as it was stored on the visible webpages specifically on the about us page. Showcased above.

**Graphical user interface, text, application

Description automatically generated**

Figure 34 Flag 2

The second flag was found after gaining access to the Joomla CMS, within the content > articles section.

**Graphical user interface, text, application

Description automatically generated**

Figure 35 Flag 3

Flag 3 was found after gaining access to the ToE backend in the /home/biscuit directory. This was a hidden file which was able to be seen with the **ls -la** command.

**Graphical user interface, text

Description automatically generated**

Figure 36 Flag 4

Flag 4 was found after gaining access to the ToE backend in the /home/orga directory. This was a hidden file which was able to be seen with the **ls -la** command.

**Graphical user interface, text

Description automatically generated**

Figure 37 False Flag 5

**A screenshot of a computer

Description automatically generated with medium confidence**

Figure 38 Flag 5

Flag 5 was the most difficult to find but still relatively easy. There was a false flag5 in the /home/mika directory which contained no information. But after some searching, we discovered the final flag in the /root directory with the **ls -la** command.

Sources

[1] “Reporting - The Penetration Testing Execution Standard,” *Pentest-standard.org*, 2014. http://www.pentest-standard.org/index.php/Reporting

[2] M. Lino, “Joomla! 3.7.0 - ‘com\_fields’ SQL Injection,” *Exploit Database*, May 19, 2017. https://www.exploit-db.com/exploits/42033

[3] K. HOANG, “Made by HK,” *GitHub*, Nov. 09, 2022. https://github.com/HoangKien1020/CVE-2020-11890 (accessed Apr. 20, 2023).

[4] “5 Steps to Remediating Cyber Security Threats,” *Resolute*, May 19, 2020. https://www.resolutets.com/cyber-security-threat-remediation/

[5] “https://www.hackerone.com/sites/default/files/2021-08/hackerone-2021-pentest-report.pdf.”

[6] “How to Write an Effective Pentest Report | Cobalt,” *www.cobalt.io*. https://www.cobalt.io/blog/how-to-write-an-effective-pentest-report-vulnerability-reports

[7] “Attacking and Enumerating Joomla,” *HackerTarget.com*, Jul. 29, 2020. https://hackertarget.com/attacking-enumerating-joomla/# (accessed Apr. 20, 2023).

[8] “CVE - CVE-2020-35613,” *cve.mitre.org*. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-35613 (accessed Apr. 20, 2023).

[9] “CVE - CVE-2017-8917,” *cve.mitre.org*. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-8917 (accessed Apr. 20, 2023).

[10] “SQL Injection Vulnerability in Joomla! 3.7,” *Sucuri Blog*, May 17, 2017. https://blog.sucuri.net/2017/05/sql-injection-vulnerability-joomla-3-7.html

[11] ITasahobby, “TryHackMe: Daily Bugle,” *ITasahobby*, Aug. 03, 2020. https://itasahobby.gitlab.io/posts/dailybugle/ (accessed Apr. 20, 2023).

[12] N. Williams, “Vulnerability Adventures: DC 3,” *Medium*, Sep. 19, 2019. https://medium.com/@nickwilliams\_/vulnerability-adventures-dc-3-416afe04c531

[13] “10 STEPS TO A USEFUL PENETRATION TEST REPORT,” *CyberSecurity Services*. https://www.emagined.com/blog/10-steps-to-writing-a-useful-penetration-test-report

[14] Administrator., “Information Gathering With Nmap,” *Penetration Testing Lab*, Feb. 16, 2013. https://pentestlab.blog/2013/02/16/information-gathering-with-nmap/

[15] “OWASP/joomscan,” *GitHub*, Jun. 01, 2021. https://github.com/OWASP/joomscan