CO2515 Coding for Penetration Testing Worksheet 1

Aim – To introduce students to some basic PT coding using Python3 and Scapy with IPv6. Only basic programming skills are assumed. All code is written to be as simple as possible.

NOTE – The techniques discussed in this worksheet should only be used on networks with the full written permission of the network owner otherwise you may be in breach of the GDPR.

|  |  |
| --- | --- |
| 1. | In this lab you will be introduced to some packet manipulation techniques using Python3.  Python is just one of several ‘scripting’ languages.  One of Python’s strengths is the large number of ‘modules’ available that can extend the basic Python tool set.  Module is the term used for Python program scripts.  One such tool is ‘Scapy’ which is used for packet crafting and sniffing. |
| 2. | Access the university server and extract the ‘Linux light’ and ‘Gserver3’ virtual machines. |
| 3. | Run VMware as administrator and open the virtual machines. The network adapters should be set to ‘VMnet4’.  Go to ‘Edit’ and ‘Virtual network editor’ in VMware.  Create a ‘Vmnet4’ and set it to ‘Host-only’. Make sure the following two statements are **unchecked**:  ‘Connect a host virtual network adapter to this network’  ‘Use local DHCP service to distribute IP address to VMs.’  Click ‘Apply’. |
|  | |
| 4. | Power on Linux Light and log on using the password ‘scapy’. |
| 5. | Start the interactive environment for Scapy by typing **sudo scapy** at a command prompt.  Enter the following code below. |
| 6. | **>>> pkt = sniff(iface="ens33", filter="icmp6", count=10)** |
| 7. | The code is written in Python3 and uses Scapy to **sniff** packets on a network constrained by 3 defined fields:  **iface =** (the network interface which will be used.)  **filter =** (The protocol which will be filtered.)  **count =** (The maximum number of sniffed packets required.)  The content received by the sniffer is stored in **pkt**.  Other fields are available and can be added to the code if required. |
| 8. | Power on Gserver and wait for it to boot up.  Now from the attacker, stop the sniffing process with **ctrl+c**.  Enter the following code and examine the output. |
| 9. | **>>> pkt.summary( )** |
| 10. | The summary should present you with a series of icmpv6 neighbour solicitations, this is the IPv6 replacement for ARP. Take notice of the Link-local and unique-local addresses being sent out from Gserver.  Use **ctrl+d** to exit Scapy. |
| 11. | Scapy also has the capability to be run through python scripts. Create a new file and call it **‘Sniff6.py’**.  Copy the following code and read the comments the output of the file is slightly different to what was seen before. |
| 12. | **#!/usr/bin/python3** #This tells the machine to run Python script  **from scapy.all import \*** #Import modules from the python library which may be needed  **def print\_pkt (pkt):** #Define a function which stores pkt and shows it  **pkt.show()**  **pkt=sniff(iface="ens33", filter="icmp6", count=10, prn=print\_pkt)** #Run the packet sniffer and store it in function |
| 13. | Exit the file and run the script using:  **sudo python3 ./sniff6.py**  Examine the output of the script. What do you notice? |
| 14. | The output of the file is used to display raw packet data from the ones captured with scapy.  With the output of the script find the unique-local address of Gserver target machine. |
| 15. | Assign the machine with a temporary IPv6 address using: **ip -6 addr add <ipv6address>/<prefixlength> dev <interface>** |
| 16. | Ping the target using: **ping6 <target IP>** |
| 17. | Open Wireshark on GServer and filter for ICMP packets. Open it up full screen so you can see all the information in the 3 sections. |
| 19. | Run Scapy. |
| 20. | Send a ping packet to GServer by typing the following command.  **>>> send(IPv6(dst=”FD80:1234:5678:2::254”)/ICMPv6EchoRequest())**  NOTE – include the quotes round your message.  NOTE – single and double quotes are both accepted by Python.  What do you see in Wireshark? |
| 21. | The **send** command is only able to send packets one way, in order to receive packets use **sr.**  Now try:  **>>> sr(IPv6(dst=”fd80:1234:5678:2::254”)/ICMPv6EchoRequest())** |
| 22. | Examine Wireshark’s capture, can you see the message? Check in the hex dump section. |
| 23. | Change the ICMP field to the following and add your own text in the message field:  **>>> send(IPv6(dst=”FD80:1234:5678:2::254”)/ICMPv6EchoRequest()/”Write a message”)** |
| 24. | What changes when the packet is sent?  Check the packet in Wireshark. |
| 25. | Start a new packet capture in Wireshark.  From scapy change the source address of the packet to an address which does not match the firewall rule. This can be achieved by adding **src** information. This must be in the form of hex digits (0-9, A-F).  **>>> send(IPv6(src=”<Enter a source address here>”, dst=”FD80:1234:5678:1::254”)/ICMPv6EchoRequest()/”this is a message”)** |
| 26. | Check the Packet IP information in Wireshark. What does this say about the reliability of packet data?  Exit Scapy. |
| 27. | Create a file called **myping.py** in the home directory.  Enter the following code below into the python file. |
| 28. | **#!/usr/bin/python3**  **"""**  **This module sends an echo reply with a user defined message.**  **"""**  **from scapy.all import \***  #This is a ping packet we created with the imported scapy modules.  **send(IPv6(dst="FD80:1234:5678:2::254")/ICMPv6EchoRequest()/"This is a message")** |
| 29. | NOTE – (“””) below is Python’s way of including a comment spread over several lines.  Breakdown of module structure:  NOTE - Python like a lot of languages uses # to include comments. Hint from Philip Whiteside – when starting a script write the comments first and then the code!  NOTE – If the script will be used more than once include comments for yourself and anyone else who may use it.  NOTE – Python doesn’t use ‘;’ in the way many other languages do, but it is very sensitive to indenting and spacing.  Close and save the file. |
| 30. | There are different ways of running the module  Quickest – **python3 ./myping.py**  If you want to keep and run the module multiple times, make it executable: chmod u+x myping.py  then type: **./myping.py**  If you don’t want to bother with the **./** put myping.py into **/usr/local/bin** **directory then it will run from anywhere.**  NOTE – Python is an interpreted language so there is no compile process. |
| 31. | Now at the moment, the tool created is not very practical to use as it requires you to manually alter the source file in order to change the source and destination address. This can however be solved with the use of system arguments in the code.  Examine and copy the following code and add it to the **myping.py** file previously created. |
| 32. | **#!/usr/bin/python3**  **"""**  **This script sends an echo reply**  **With a user defined message.**  **"""**  **from scapy.all import \***  **import sys**  **if len(sys.argv)!=4:**  **print ('\n Usage: myping2.py <Source IP> <Destination IP> <"Message">\n')**  **sys.exit(1)**  **send(IPv6(src=sys.argv[1],dst=sys.argv[2])/ICMPv6EchoRequest()/sys.argv[3])** |
| 33. | Run the script: **sudo ./myping.py**  What does it say? |
| 34. | NOTE: The system argument function allows command line arguments to be passed into certain field in the script. In this case the code requests 4 arguments. Remember that programming languages start from 0 and **sys.argv[0]** is the name of the script itself.  To execute the code, look at the example below: |
| 35. | **sudo ./myping.py ABCD:1234:5678:2::100 fd80:1234:5678:2::254 "Hello"** |
| 36. | **./myping.py = sys.argv[0]**  **ABCD:1234:5678:2::100 = sys.argv[1]**  **fd80:1234:5678:2::254 = sys.argv[2]**  **“Hello” = sys.argv[3]** |
| 37. | For short and simple module, a text editor is fine, and many programmers use nothing more but an IDE (Integrated Development Environment) can be helpful.  There are several IDEs for Python. A popular program is Spyder, which is designed for scientists. You can load it from the menu, or by typing spyder in a terminal. If you don’t like Spyder you could try IDLE or PyCharm both of which are free.  You may need to install these if they are not available on your machine. |
| 38. | For short and simple module, a text editor is fine, and many programmers use nothing more but an IDE (Integrated Development Environment) can be helpful.  There are several IDEs for Python. A popular program is Spyder, which is designed for scientists. You can load it from the menu, or by typing spyder in a terminal. If you don’t like Spyder you could try IDLE or PyCharm both of which are free.  You may need to install these if they are not available on your machine. |
| 39. | Run your code by using Run on the menu or F5.  NOTE - If you are not comfortable using IDEs than simply using the command terminal is viable for the rest of the worksheets. |