Sam Wehunt

October 15, 2017

Demostration of TCP RST attack

1 Overview

1.1 TCP RST Attack

The focus of this report will be on the TCP RST attack. This attack is a DOS attack on the TCP protocol, and there are several real-world examples of its use to deny internet traffic and stop communication [1].
In this attack, the goal of the attacker is to inject a malicious packet into an ongoing TCP stream. This malicious packet will be just like every other packet, except the packet will have the RST flag set. In a normal TCP communication the RST flag tells a party that the connection should be closed and no further traffic should be sent. If an attacker were able to inject a packet with this bit set, then the connection would close and the attacker has caused a denial of service. This is the goal that this report will set out to accomplish.

2 Setup

If one wishes to recreate the process that went into creating this report, then this section will prove useful. Otherwise skip to the next section for an explanation of the attack and its procedure.

2.1 Network Topology

For this test, there will need to be three hosts on the same subnet. Host 1 will be the client (victim), and that machine will be at IP 192.168.13.31. Host 2 will be the server, and it will be assigned to IP 192.168.13.30. And finally host 3 will be given the IP 192.168.13.37. Internet access is not required for this attack, but it may be helpful for setting up the virtual machines. Make sure that all of the hosts can ping each other, and make sure they are all in the same subnet.

2.2 VM Setup

Each VM will require slightly different setups to account for their differing roles. It would be very helpful to install wireshark on all systems, as that will allow close monitoring of the attack. setup for each host is as follows.

2.2.1 Host 1

Host 1 is the client (victim), and so it will only require an FTP client. any TCP based communication would work here, but for this demonstration FTP will work just fine.

2.2.2 Host 2

Host 2 is the server, and naturally this means it will be running whatever server software the client will connect to. This demonstration will be using vsftpd as the server software.

2.2.3 Host 3

Host 3 is the attacker, and it will require the most setup. The scripts that will run the attack are written in python3, so make sure that is installed as well as scapy. To install all dependencies on a debian based system, use the following commands:

```
sudo apt-get install python3 python3-pip
sudo pip3 install scapy-python3
```

Now the host should have the software required to launch the attack. Next is to get the scripts that will actually run the attack. If they were not included with this PDF, then the files can be located on GitHub at "https://github.com/Samboy218/CSC6575Attack".
Either clone that repo or downloadthe archive and extract it.
Now that all of the required files have been aquired, run the following command to enable ip forwarding:

```
sudo echo -c "1 > /proc/net/ipv4/ip_forward"
```

This will make it so the the ARP poisoning does not block all traffic when we begin to intercept packets (while our goal is DOS, we don't want to stop traffic like this). After this is done, the testing environment should be all set up and ready to run the attack.

3 Attack Demonstration

Now that the testing environment is ready, the attack can be performed.

3.1 ARP Poison

The first stage of the attack involves an ARP poisoning attack. This would not normally be required, but it is here because without it we will not be able to sniff the traffic going between the victim and server. The network we are in is controlled by a switch, and this switch knows the physical addresses connected to it. so when the victim talks to the server, the packets will only be sent to the server and the attacker will never have a chance to see the communication (note: if the network was connected wirelessly then this would not be an issue and the ARP poisoning could be skipped). To bypass this issue, we will perform an ARP poison attack and become a Man-In-The-Middle. ARP is the address resolution protocol, and it is responsible for translating IP addresses into hardware (MAC) addresses. Each host on a network keeps a table of these mappings so it can quickly translate IP to MAC. To perform this attack, we will corrupt the ARP cache of both the client and the server so that when the client tries to find the

address of the server based on the IP, it will return the hardware address of the attacker, and the server will do the same when looking up the client. Corrupting this cache is easy; all we need to do is send an ARP packet to the host we want to poison, and this packet will have the 'is-at' operator, the source IP will be of the destination we want to intercept (I.E. the packet that will be sent to the client will have the source IP set to the server's IP), and the source hardware address will be the attacker's MAC. When the target receives this packet, the host will examine it and see that the location of the source IP is the source hardware address. A script has been written to perform this poisoning automatically. to run arpPoison.py, run this command at the attacker's terminal:

```
sudo python3 arpPoison.py
```

This script will craft the malicious arp packets and send them every few seconds (this is to ensure that the malicious ARP entries don't go stale). Now, if you open wireshark on the attacker, and then from the victim ping the server, the attacker should be able to see the pings go through the network. If the attacker can see the pings, then the attack worked and the next step can be done. If the attack does not work, check your configuration (ensure the values in the script for the IP addresses and the interface are correct)

3.2 TCP Sequence Prediction and RST Attack

Now that we are able to sniff the traffic going between the two hosts, we can attempt to hijack the session and inject packets into the TCP stream. One more obstacle stands in our way. In a TCP packet there is a field for a sequence number, and this number is randomly generated when a connection is made. Without knowing this sequence number, a host will be able to see that our packets are invalid. This issue is solved if we can sniff the traffic between the targets. The sequence number is generated at the beginning of the communication, but after that the sequence number for any given packet will be the ACK number of the previous packet, and the ACK number of any given packet will be the SEQ number of the previous packet plus the length of the data sent. This convention means that if we are able to capture a packet from the communication, then we should know the next valid SEQ and ACK numbers to reply with. At that point it is just a race to have our malicious packet arrive before the next real packet that is sent [2].

For this particular attack, we want to inject TCP RST packets into the communication in order to close the connection and cause a denial of service. There is a script that will perform this attack for us. rst.py will sniff all of the packets on the network until it finds a TCP packet with a destination of our victim. When a packet is found, the script will forge a packet to the sender (which would be the server) with a spoofed IP of the victim, and since we captured a packet from the server, we know what the correct SEQ and ACK numbers should be. Finally, the script will set the RST flag in the packet and send the forged packet to the server. The server should then read the packet, verify that it is legitimate, and it will close the connection with no further interaction from the client. The next time the client tries to use that connection the server will

only respond with RST packets because the connection is closed. To run rst.py use the following command:
`sudo python3 rst.py`

4 Conclusion

This attack is relatively simple once all of the required data is collected and the attacker understands how the protocol works. A basic overview of the attack is as follows: the attacker sniffs TCP traffic between two hosts, the attacker finds out the SEQ and ACK numbers of the TCP stream, and finally the attacker can use those values to inject packets into the TCP stream.

4.1 Impact

This attack has the potential to cause severe harm to a network because it only uses mechanisms built into the TCP protocol. This attack looks like normal traffic if there is no context, and sometimes context may not be enough to detect this attack. This attack is also very dangerous because the attacker can inject more than just a TCP RST if they want to. once the attacker has the ability to inject packets, They can cause denial of service to just one party and effectively hijack the session.

4.2 Mitigations

This attack is somewhat difficult to mitigate because it happens at a low level protocol, but there are ways to stop this from happening. The most effective way would be to use a VPN or another ipsec standard. This can add either encryption or authentication to the packets sent across the network, and thus when the attacker attempts to hijack the session they will not be able to properly authenticate and the hosts will not accept the malicious packets.

References

[1] Electronic Frontier Foundation, *Packet Forgery By ISPs: A Report on the Comcast Affair* November 28, 2007 https://www.eff.org/wp/packet-forgery-isps-report-comcast-affair

[2] packetlife.com *Understanding TCP Sequence and Acknowledgment Numbers* June 7, 2010 http://packetlife.net/blog/2010/jun/7/understanding-tcp-sequence-acknowledgment-numbers/