Robert Carpenter
github.com/robertmcarpenter
Sat December 14th 2024

Lab 6.5.8 Analyzing SYN Flood Attack using Wireshark

From TestOut CompTIA Security+ Course

In this lab I will be analyzing a network interface to ascertain if a SYN Flood attack is occuring on “my network” as a hypothetical corporate IT Admin. I will be using Wireshark to listen in on packets and deduce a conclusion.

The scenario for this lab is as follows:

“You are the CorpNet IT administrator. Your support team says that CorpNet's customers are unable to browse to the public-facing web server. You suspect it might be under a denial-of-service attack, possibly a TCP-SYN flood attack. Your www_stage computer is on the same network segment as your web server, so you should use this computer to investigate the problem.

In this lab, your task is to:

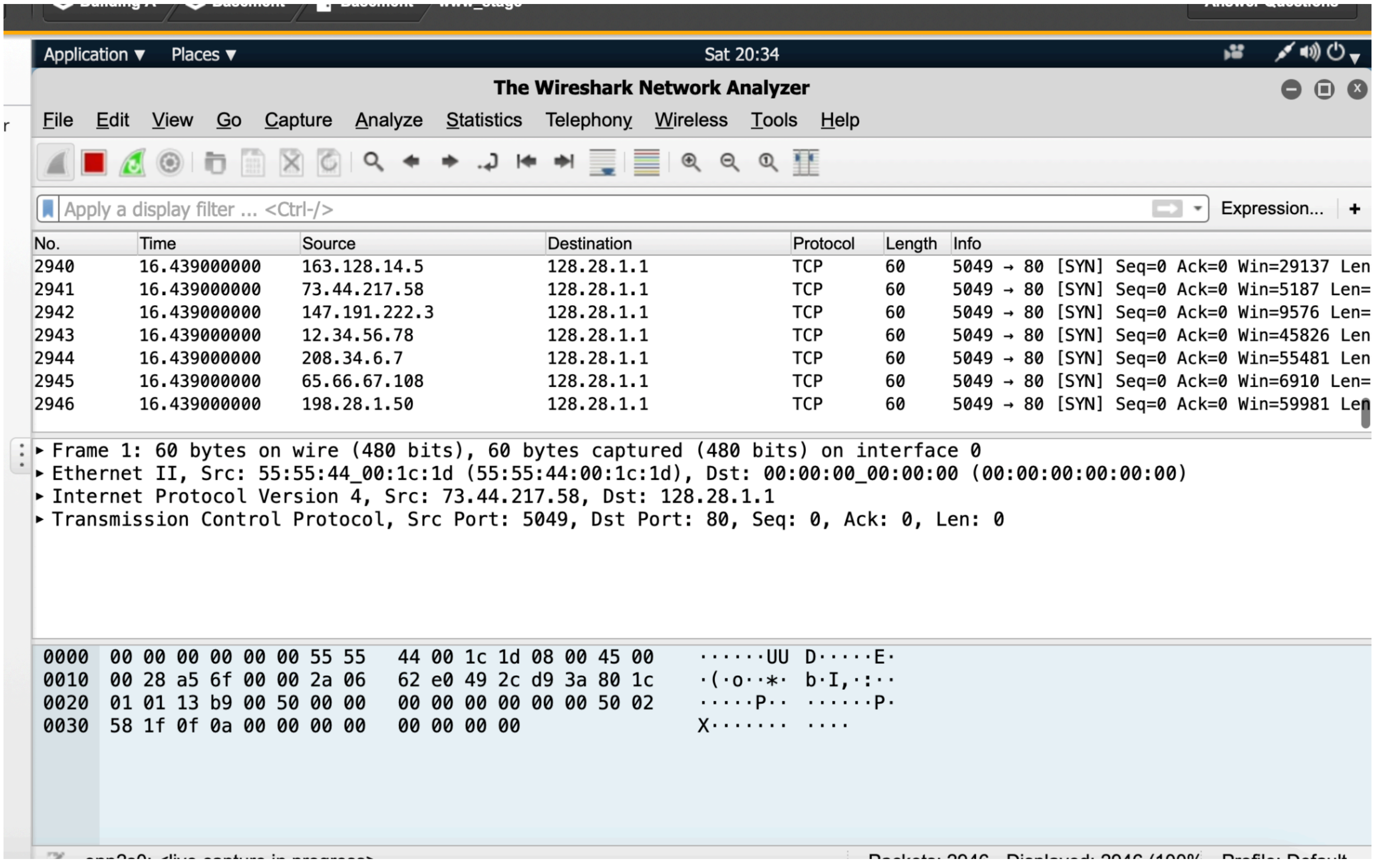
- **Capture packets from the network segment on www_stage using Wireshark.**
    - **Use the enp2s0 interface.**
- **Analyze the attack using the following filters:**
    - **tcp.flags.syn==1 and tcp.flags.ack==1**
    - **tcp.flags.syn==1 and tcp.flags.ack==0**
- **Answer the question.”**

This doesn’t sound too good! Since customers are unable to access key business resources to engage in transactions I will need to determine and troubleshoot this issue immediately!

Since my computer is on the same VLAN/Network Segment as the Webserver I can listen in on packets traveling to the web server.

In order to do that I’ll open Wireshark and select the interface given to me by the lab which is **enp2s0** and click the Blue fin in the Top left corner to start capturing packets as shown:

Robert Carpenter
github.com/robertmcarpenter
Sat December 14th 2024


Application ▾
Places ▾
Sat 20:34
The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...
+

```
No.    Time          Source           Destination   Protocol  Length  Info
2940   16.439000000  163.128.14.5     128.28.1.1    TCP       60      5049 → 80 [SYN] Seq=0 Ack=0 Win=29137 Len
2941   16.439000000  73.44.217.58     128.28.1.1    TCP       60      5049 → 80 [SYN] Seq=0 Ack=0 Win=5187 Len=
2942   16.439000000  147.191.222.3    128.28.1.1    TCP       60      5049 → 80 [SYN] Seq=0 Ack=0 Win=9576 Len=
2943   16.439000000  12.34.56.78      128.28.1.1    TCP       60      5049 → 80 [SYN] Seq=0 Ack=0 Win=45826 Len
2944   16.439000000  208.34.6.7       128.28.1.1    TCP       60      5049 → 80 [SYN] Seq=0 Ack=0 Win=55481 Len
2945   16.439000000  65.66.67.108     128.28.1.1    TCP       60      5049 → 80 [SYN] Seq=0 Ack=0 Win=6910 Len=
2946   16.439000000  198.28.1.50      128.28.1.1    TCP       60      5049 → 80 [SYN] Seq=0 Ack=0 Win=59981 Len
```

```
▸ Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
▸ Ethernet II, Src: 55:55:44_00:1c:1d (55:55:44:00:1c:1d), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
▸ Internet Protocol Version 4, Src: 73.44.217.58, Dst: 128.28.1.1
▸ Transmission Control Protocol, Src Port: 5049, Dst Port: 80, Seq: 0, Ack: 0, Len: 0
```

```
0000  00 00 00 00 00 00 55 55  44 00 1c 1d 08 00 45 00   ······UU D·····E·
0010  00 28 a5 6f 00 00 2a 06  62 e0 49 2c d9 3a 80 1c   ·(·o··*· b·I,·:··
0020  01 01 13 b9 00 50 00 00  00 00 00 00 00 00 50 02   ·····P·· ······P·
0030  58 1f 0f 0a 00 00 00 00  00 00 00 00               X······· ····
```

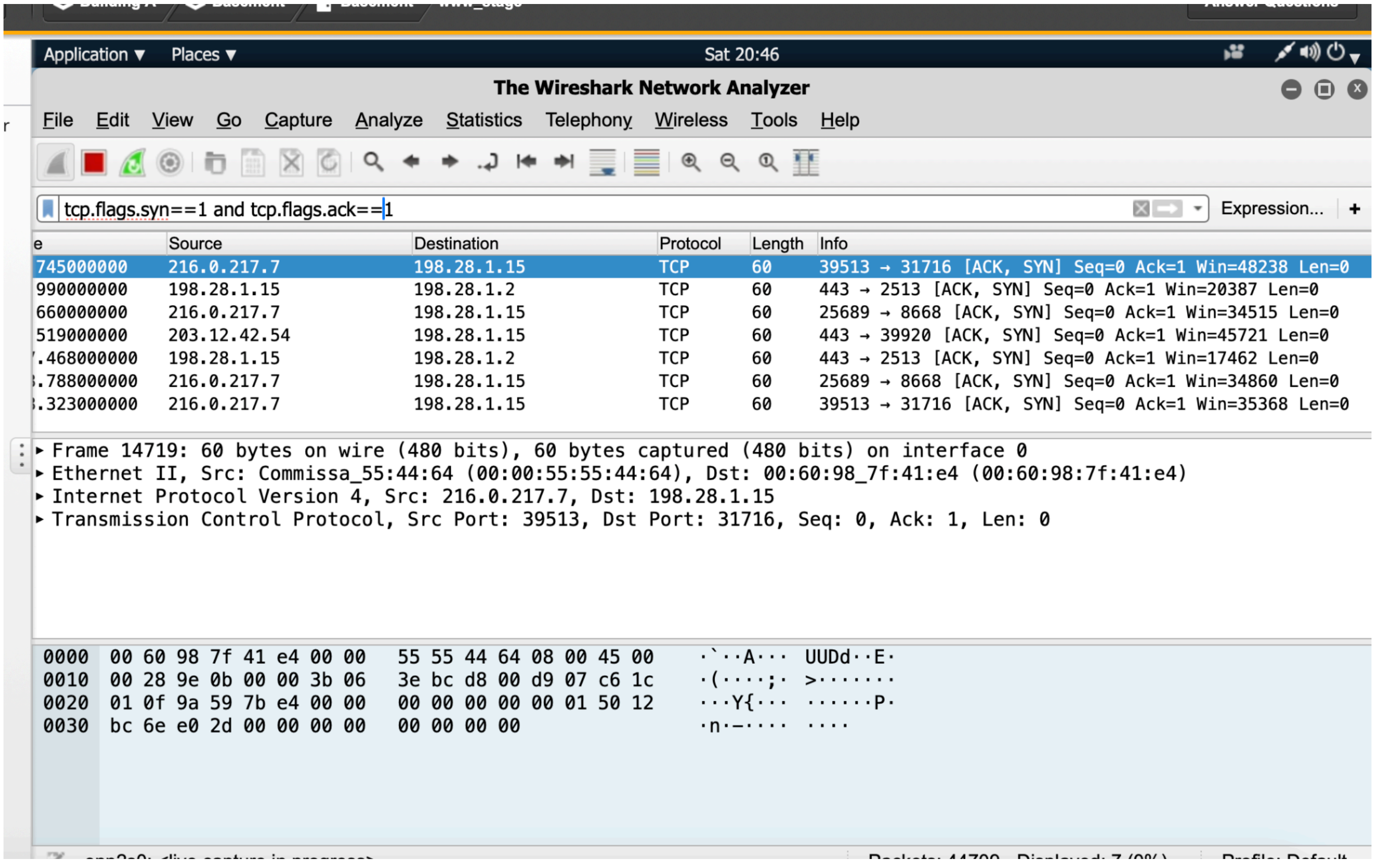
HOLY MOLY! (I actually said this out loud) In only a span of 5 seconds I can see **THOUSANDS** of SYN ACK handshake packets on their way to only 1 address which is my web server at **128.28.1.1.** I can tell this is my webserver because the IPv4 address that is listed is a public Ip address.

Now , I need to filter these packets according to the following parameters:

- **tcp.flags.syn==1 and tcp.flags.ack==1**
- **tcp.flags.syn==1 and tcp.flags.ack==0**

In the top search bar that says **Apply a Display Filter** I will add the flags to filter the packets. This filter tells us packets that have not been answered by the clients which is part of the SYN ACK handshake. These packets are considered malicious because they are the result of a SYN flood attack.

Robert Carpenter
github.com/robertmcarpenter
Sat December 14th 2024



tcp.flags.syn==1 and tcp.flags.ack==1

```
e           Source          Destination     Protocol  Length  Info
745000000   216.0.217.7     198.28.1.15     TCP       60      39513 → 31716 [ACK, SYN] Seq=0 Ack=1 Win=48238 Len=0
990000000   198.28.1.15     198.28.1.2      TCP       60      443 → 2513 [ACK, SYN] Seq=0 Ack=1 Win=20387 Len=0
660000000   216.0.217.7     198.28.1.15     TCP       60      25689 → 8668 [ACK, SYN] Seq=0 Ack=1 Win=34515 Len=0
519000000   203.12.42.54    198.28.1.15     TCP       60      443 → 39920 [ACK, SYN] Seq=0 Ack=1 Win=45721 Len=0
.468000000  198.28.1.15     198.28.1.2      TCP       60      443 → 2513 [ACK, SYN] Seq=0 Ack=1 Win=17462 Len=0
.788000000  216.0.217.7     198.28.1.15     TCP       60      25689 → 8668 [ACK, SYN] Seq=0 Ack=1 Win=34860 Len=0
.323000000  216.0.217.7     198.28.1.15     TCP       60      39513 → 31716 [ACK, SYN] Seq=0 Ack=1 Win=35368 Len=0
```

```
▸ Frame 14719: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
▸ Ethernet II, Src: Commissa_55:44:64 (00:00:55:55:44:64), Dst: 00:60:98_7f:41:e4 (00:60:98:7f:41:e4)
▸ Internet Protocol Version 4, Src: 216.0.217.7, Dst: 198.28.1.15
▸ Transmission Control Protocol, Src Port: 39513, Dst Port: 31716, Seq: 0, Ack: 1, Len: 0
```

```
0000  00 60 98 7f 41 e4 00 00   55 55 44 64 08 00 45 00    ·`··A···  UUDd··E·
0010  00 28 9e 0b 00 00 3b 06   3e bc d8 00 d9 07 c6 1c    ·(····;·  >·······
0020  01 0f 9a 59 7b e4 00 00   00 00 00 00 00 01 50 12    ···Y{···  ······P·
0030  bc 6e e0 2d 00 00 00 00   00 00 00 00                ·n·-····  ····
```

Now I will substitute the 1 in the ACK field to 0 to check responses from the Server.

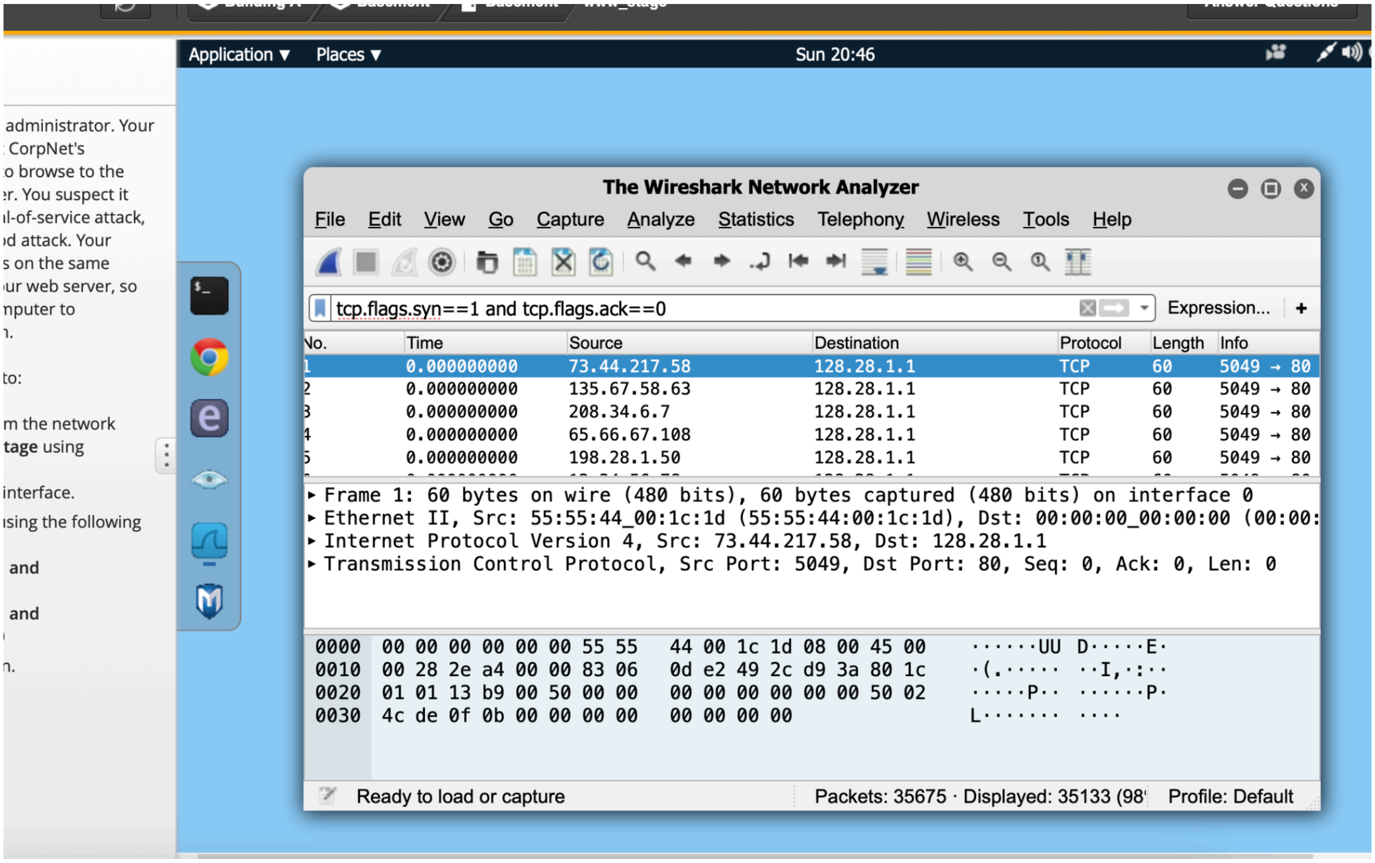
Remember that a SYN ACK handshake is 3 way and goes like this:

Step 1: Client sends **SYN** request to the server (SYN = “hey, let’s talk!)

Step 2: Server responds with **SYN ACK** to client (SYN ACK = “okay, I’ll talk to you”)

Step 3: Client responds to Server with **ACK** (ACK = “Okay cool, we are talking”)

Robert Carpenter
github.com/robertmcarpenter
Sat December 14th 2024


Application ▾
Places ▾
Sun 20:46
The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Expression...
+

administrator. Your
: CorpNet's
o browse to the
er. You suspect it
il-of-service attack,
od attack. Your
s on the same
ur web server, so
nputer to
1.

to:

m the network
tage using

interface.
ısing the following

and

and

n.

tcp.flags.syn==1 and tcp.flags.ack==0

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	73.44.217.58	128.28.1.1	TCP	60	5049 → 80
2	0.000000000	135.67.58.63	128.28.1.1	TCP	60	5049 → 80
3	0.000000000	208.34.6.7	128.28.1.1	TCP	60	5049 → 80
4	0.000000000	65.66.67.108	128.28.1.1	TCP	60	5049 → 80
5	0.000000000	198.28.1.50	128.28.1.1	TCP	60	5049 → 80

```
▸ Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
▸ Ethernet II, Src: 55:55:44_00:1c:1d (55:55:44:00:1c:1d), Dst: 00:00:00_00:00:00 (00:00:
▸ Internet Protocol Version 4, Src: 73.44.217.58, Dst: 128.28.1.1
▸ Transmission Control Protocol, Src Port: 5049, Dst Port: 80, Seq: 0, Ack: 0, Len: 0
```

```
0000  00 00 00 00 00 00 55 55  44 00 1c 1d 08 00 45 00   ······UU D·····E·
0010  00 28 2e a4 00 00 83 06  0d e2 49 2c d9 3a 80 1c   ·(.····· ··I,·:··
0020  01 01 13 b9 00 50 00 00  00 00 00 00 00 00 50 02   ·····P·· ······P·
0030  4c de 0f 0b 00 00 00 00  00 00 00 00               L······· ····
```

Ready to load or capture   Packets: 35675 · Displayed: 35133 (98'   Profile: Default

Here, we can see all of the packets with an ACK value of 0. Let's answer the questions asked of us now:

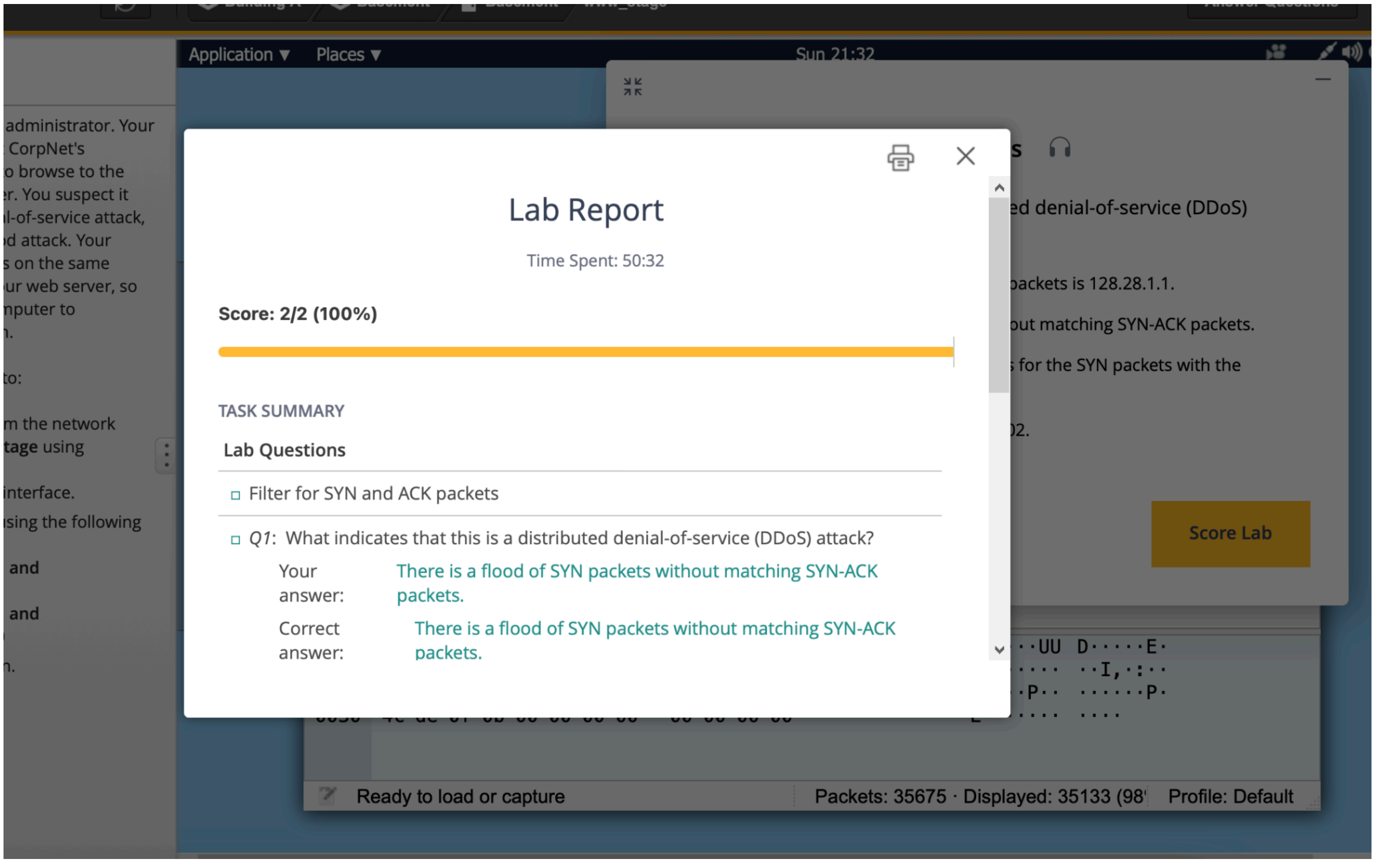
What Indicates that this is a Denial of Service Attack?

The reason I know this is a DDOS attack is because of the discrepancy between the SYN and SYN-ACK packets. If this was legitimate traffic, all SYN requests (let's say 5 SYN requests) would be answered with SYN ACK packets from the server (also 5 SYN ACK packets answering the 5 SYN packets).

Since all of these SYN requests from the clients are going unanswered the server has to wait for a ACK response from the client. This is putting the server in a "busy" state.

Going back to the 3 way handshake , it is effectively getting stuck on the 2nd step of the process.

Robert Carpenter
github.com/robertmcarpenter
Sat December 14th 2024



Lab Report

Time Spent: 50:32

Score: 2/2 (100%)

TASK SUMMARY

Lab Questions

□ Filter for SYN and ACK packets

□ *Q1:* What indicates that this is a distributed denial-of-service (DDoS) attack?

Your answer: There is a flood of SYN packets without matching SYN-ACK packets.

Correct answer: There is a flood of SYN packets without matching SYN-ACK packets.