

PFSENSE SETUP AND DOS DEFENSE DEMO

WRITEUP BY : ALI MALIK

This document outlines the configuration of a pfSense firewall to protect a target system from a TCP SYN flood attack, demonstrating its effectiveness through pre and post-attack analysis.

1. Network Configuration

The network setup involves three virtual machines: Ubuntu, Kali Linux, and pfSense, configured as follows:

Machine	IP Address	Network	Role	Adapter Type
Ubuntu	192.168.18.85	WAN (External)	Attacker	Bridged via VMnet0
Kali	192.168.1.128	LAN (Internal)	Victim	Host-only (VMnet1)
pfSense WAN	192.168.18.87	WAN (bridged)	Firewall	Bridged (VMnet0)
pfSense LAN	192.168.1.1	LAN	Gateway	Host-only (VMnet1)

- Network Segmentation:** Ubuntu and pfSense WAN are on the same physical network (192.168.18.0/24). Kali and pfSense LAN are on an isolated host-only network (192.168.1.0/24).
- pfSense Role:** pfSense acts as a bridge and firewall between these two networks , with its WAN interface connected to VMnet0 (Ubuntu and Wi-Fi router) and its LAN interface connected to VMnet1 (Kali).

pfSense Interface Configuration:

Interface	Device	IP Address	Network	Purpose
WAN	em0	192.168.100.101	External (VMnet0)	Ubuntu / Internet side
LAN	em1	192.168.1.1	Internal (VMnet1)	Kali side

2. Baseline Connectivity (Without Block Rules)

Before implementing any firewall rules, a ping test was conducted from the Ubuntu machine (attacker) to the Kali machine (victim) to confirm network connectivity.

- **Ping Command (from Ubuntu):** ping 192.168.1.128

```
ali@ali-VMware-Virtual-Platform: ~ ping 192.168.1.128
PING 192.168.1.128 (192.168.1.128) 56(84) bytes of data.
64 bytes from 192.168.1.128: icmp_seq=1 ttl=63 time=1.56 ms
64 bytes from 192.168.1.128: icmp_seq=2 ttl=63 time=2.70 ms
64 bytes from 192.168.1.128: icmp_seq=3 ttl=63 time=1.46 ms
64 bytes from 192.168.1.128: icmp_seq=4 ttl=63 time=8.23 ms
64 bytes from 192.168.1.128: icmp_seq=5 ttl=63 time=1.27 ms
64 bytes from 192.168.1.128: icmp_seq=6 ttl=63 time=1.53 ms
64 bytes from 192.168.1.128: icmp_seq=7 ttl=63 time=2.42 ms
64 bytes from 192.168.1.128: icmp_seq=8 ttl=63 time=2.68 ms
64 bytes from 192.168.1.128: icmp_seq=9 ttl=63 time=2.94 ms
64 bytes from 192.168.1.128: icmp_seq=10 ttl=63 time=1.72 ms
64 bytes from 192.168.1.128: icmp_seq=11 ttl=63 time=5.46 ms
64 bytes from 192.168.1.128: icmp_seq=12 ttl=63 time=3.07 ms
64 bytes from 192.168.1.128: icmp_seq=13 ttl=63 time=1.66 ms
64 bytes from 192.168.1.128: icmp_seq=14 ttl=63 time=1.08 ms
64 bytes from 192.168.1.128: icmp_seq=15 ttl=63 time=3.11 ms
64 bytes from 192.168.1.128: icmp_seq=16 ttl=63 time=2.70 ms
64 bytes from 192.168.1.128: icmp_seq=17 ttl=63 time=2.97 ms
64 bytes from 192.168.1.128: icmp_seq=18 ttl=63 time=4.07 ms
64 bytes from 192.168.1.128: icmp_seq=19 ttl=63 time=2.56 ms
64 bytes from 192.168.1.128: icmp_seq=20 ttl=63 time=4.08 ms
64 bytes from 192.168.1.128: icmp_seq=21 ttl=63 time=5.85 ms
64 bytes from 192.168.1.128: icmp_seq=22 ttl=63 time=2.83 ms
64 bytes from 192.168.1.128: icmp_seq=23 ttl=63 time=2.58 ms
64 bytes from 192.168.1.128: icmp_seq=24 ttl=63 time=2.92 ms
64 bytes from 192.168.1.128: icmp_seq=25 ttl=63 time=6.98 ms
64 bytes from 192.168.1.128: icmp_seq=26 ttl=63 time=1.59 ms
64 bytes from 192.168.1.128: icmp_seq=27 ttl=63 time=1.96 ms
64 bytes from 192.168.1.128: icmp_seq=28 ttl=63 time=2.62 ms
64 bytes from 192.168.1.128: icmp_seq=29 ttl=63 time=2.24 ms
64 bytes from 192.168.1.128: icmp_seq=30 ttl=63 time=4.39 ms
64 bytes from 192.168.1.128: icmp_seq=31 ttl=63 time=3.79 ms
64 bytes from 192.168.1.128: icmp_seq=32 ttl=63 time=2.74 ms
64 bytes from 192.168.1.128: icmp_seq=33 ttl=63 time=2.00 ms
```

No.	Time	Source	Destination	Protocol	Length	Info
7	2.810384426	192.168.1.85	192.168.1.128	ICMP	98	Echo (ping) request id=0x001b, seq=1/256, ttl=63 (reply in 8)
8	2.810431609	192.168.1.128	192.168.1.85	ICMP	98	Echo (ping) reply id=0x001b, seq=1/256, ttl=64 (request in 7)
10	3.808290446	192.168.1.85	192.168.1.128	ICMP	98	Echo (ping) request id=0x001b, seq=2/256, ttl=63 (request in 9)
11	3.812049417	192.168.1.128	192.168.1.85	ICMP	98	Echo (ping) reply id=0x001b, seq=2/256, ttl=64 (request in 10)
17	4.815177224	192.168.1.85	192.168.1.128	ICMP	98	Echo (ping) request id=0x001b, seq=3/256, ttl=63 (request in 11)
18	4.815216407	192.168.1.128	192.168.1.85	ICMP	98	Echo (ping) reply id=0x001b, seq=3/256, ttl=64 (request in 12)
19	5.827855960	192.168.1.85	192.168.1.128	ICMP	98	Echo (ping) request id=0x001b, seq=4/256, ttl=64 (request in 13)
20	5.827903376	192.168.1.128	192.168.1.85	ICMP	98	Echo (ping) reply id=0x001b, seq=4/256, ttl=63 (request in 14)
22	6.825023442	192.168.1.85	192.168.1.128	ICMP	98	Echo (ping) request id=0x001b, seq=5/256, ttl=63 (request in 15)
23	6.825073379	192.168.1.128	192.168.1.85	ICMP	98	Echo (ping) reply id=0x001b, seq=5/256, ttl=64 (request in 16)
24	7.826996385	192.168.1.85	192.168.1.128	ICMP	98	Echo (ping) request id=0x001b, seq=6/256, ttl=63 (request in 17)
25	7.827046446	192.168.1.128	192.168.1.85	ICMP	98	Echo (ping) reply id=0x001b, seq=6/256, ttl=64 (request in 18)
27	8.829637938	192.168.1.85	192.168.1.128	ICMP	98	Echo (ping) request id=0x001b, seq=7/256, ttl=63 (request in 19)
28	8.829791296	192.168.1.128	192.168.1.85	ICMP	98	Echo (ping) reply id=0x001b, seq=7/256, ttl=64 (request in 20)
31	8.830723052	192.168.1.85	192.168.1.128	ICMP	98	Echo (ping) request id=0x001b, seq=8/256, ttl=63 (request in 21)
32	9.831787364	192.168.1.128	192.168.1.85	ICMP	98	Echo (ping) reply id=0x001b, seq=8/256, ttl=64 (request in 22)
34	10.833486741	192.168.1.85	192.168.1.128	ICMP	98	Echo (ping) request id=0x001b, seq=9/256, ttl=63 (request in 23)
41	10.833638955	192.168.1.128	192.168.1.85	ICMP	98	Echo (ping) reply id=0x001b, seq=9/256, ttl=64 (request in 24)
43	11.835329741	192.168.1.85	192.168.1.128	ICMP	98	Echo (ping) request id=0x001b, seq=10/256, ttl=63 (request in 25)
44	11.835393582	192.168.1.128	192.168.1.85	ICMP	98	Echo (ping) reply id=0x001b, seq=10/256, ttl=64 (request in 26)
46	12.841098446	192.168.1.85	192.168.1.128	ICMP	98	Echo (ping) request id=0x001b, seq=11/256, ttl=63 (request in 27)
47	12.841143284	192.168.1.128	192.168.1.85	ICMP	98	Echo (ping) reply id=0x001b, seq=11/256, ttl=64 (request in 28)
49	13.841092440	192.168.1.85	192.168.1.128	ICMP	98	Echo (ping) request id=0x001b, seq=12/256, ttl=63 (request in 29)
50	13.841133800	192.168.1.128	192.168.1.85	ICMP	98	Echo (ping) reply id=0x001b, seq=12/256, ttl=64 (request in 30)

- **Result:** The ping was successful, demonstrating communication between the Ubuntu and Kali machines through the pfSense firewall. This was further verified by observing ICMP echo requests and replies in Wireshark on the Kali Linux machine.

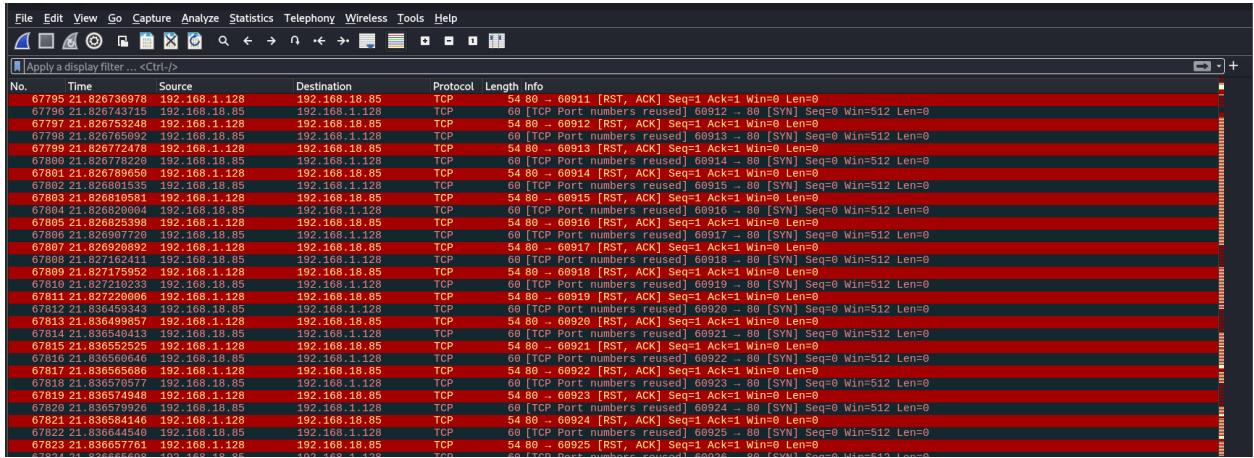
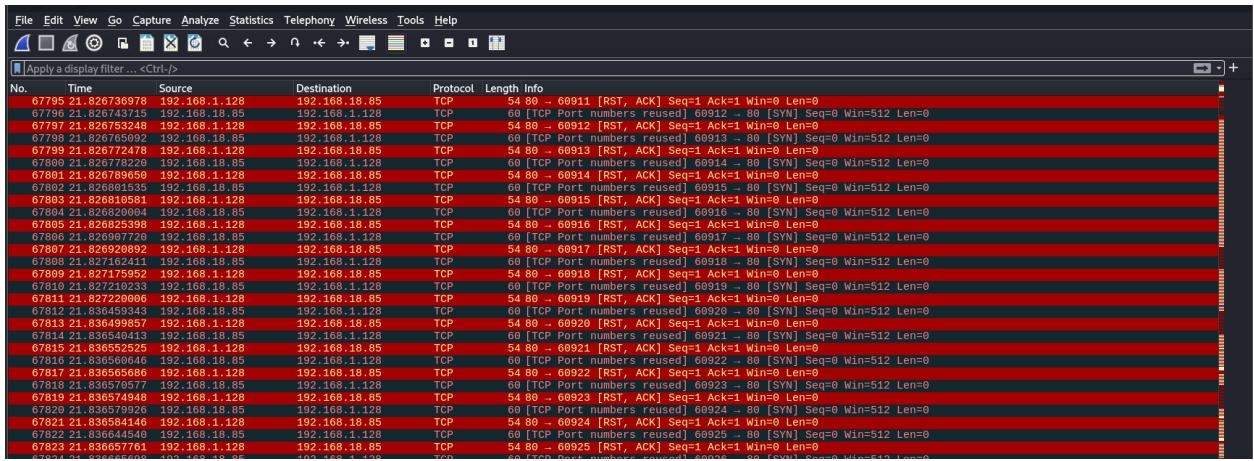
3. TCP SYN Flood Attack (Without Block Rules)

A TCP SYN flood attack was initiated from the Ubuntu machine targeting the Kali machine.

- **Attack Command (from Ubuntu):** sudo hping3 -S -p 80 --flood 192.168.1.128

```
ali@ali-VMware-Virtual-Platform:~$ sudo hping3 -S -p 80 --flood 192.168.1.128
[sudo] password for ali:
HPING 192.168.1.128 (ens33 192.168.1.128): S set, 40 headers + 0 data bytes
hp ping in flood mode, no replies will be shown
^C
--- 192.168.1.128 hp ping statistic ---
211973 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

- **hping3 Statistics:** The hping3 tool reported a 100% packet loss for transmitted packets, as it was operating in flood mode and not expecting replies.
- **Wireshark Analysis (on Kali):** Wireshark on the Kali machine showed a high volume of TCP packets, indicating the SYN flood attack. These packets were predominantly SYN requests, as expected during a SYN flood.



4. Implementing a Block Rule on pfSense

To mitigate the SYN flood attack, a block rule was created in pfSense.

- **Action:** Block (This drops packets silently, unlike "reject" which sends a response back to the sender).
- **Interface:** WAN (Packets matching this rule must originate from the WAN interface).
- **Address Family:** IPv4
- **Protocol:** TCP
- **Source:** 192.168.18.85 (The IP address of the Ubuntu attacker machine).
- **Destination:** 192.168.1.128 (The IP address of the Kali victim machine).
- **Description:** block syn flood
- **Logging:** Enabled (To log packets handled by this rule for verification).

Firewall / Rules / Edit

Edit Firewall Rule

Action	Block
Choose what to do with packets that match the criteria specified below. Hint: the difference between block and reject is that with reject, a packet (TCP RST or ICMP port unreachable for UDP) is returned to the sender, whereas with block the packet is dropped silently. In either case, the original packet is discarded.	
Disabled	<input type="checkbox"/> Disable this rule Set this option to disable this rule without removing it from the list.
Interface	WAN
Choose the interface from which packets must come to match this rule.	
Address Family	IPv4
Select the Internet Protocol version this rule applies to.	
Protocol	TCP
Choose which IP protocol this rule should match.	

Source

Source	<input type="checkbox"/> Invert match	Address or Alias	192.168.18.85
Display Advanced The Source Port Range for a connection is typically random and almost never equal to the destination port. In most cases this setting must remain at its default value, any.			

Destination

Destination	<input type="checkbox"/> Invert match	Address or Alias	192.168.1.128
Destination Port Range	any	From	Custom
	any	To	Custom
Specify the destination port or port range for this rule. The "To" field may be left empty if only filtering a single port.			

5. Post-Block Rule TCP SYN Flood Attack

After saving the new block rule, the TCP SYN flood attack was re-initiated from the Ubuntu machine.

- **Attack Command (from Ubuntu):** sudo hping3 -S -p 80 --flood 192.168.1.128

```
ali@ali-VMware-Virtual-Platform:~$ sudo hping3 -S -p 80 --flood 192.168.1.128
HPING 192.168.1.128 (ens33 192.168.1.128): S set, 40 headers + 0 data bytes
hp ping in flood mode, no replies will be shown
^C
--- 192.168.1.128 hp ping statistic ---
7633068 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

- **hping3 Statistics:** Similar to the previous attack, hping3 reported 100% packet loss.
- **Wireshark Analysis (on Kali):** Wireshark on the Kali machine displayed no incoming packets from the attacker, indicating that the SYN flood was successfully blocked by pfSense.

No.	Time	Source	Destination	Protocol	Length	Info
1250 161.779434469	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=470456 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1251 161.779434477	192.168.1.1	192.168.1.1	TCP	1514	17286 → 443	[ACK] Seq=3916 Ack=470456 Win=514 Len=0
1252 161.780133863	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=444118 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1253 161.780378289	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=445578 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1254 161.780545913	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=447038 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1255 161.780549858	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=448498 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1256 161.780749278	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=449958 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1257 161.780749692	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=451118 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1258 161.781045406	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=452224 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1259 161.781045889	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=453224 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1260 161.783377749	192.168.1.1	192.168.1.10	TLSv1.2	499	Application Data	
1261 161.788210712	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=453461 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1262 161.788737873	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=454921 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1263 161.789419391	192.168.1.10	192.168.1.1	TCP	1514	443 → 17286	[ACK] Seq=456381 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1264 161.790516158	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=456381 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1265 161.790636221	192.168.1.10	192.168.1.1	TCP	1514	443 → 17286	[ACK] Seq=457841 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1266 161.790644994	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=457841 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1267 161.790647003	192.168.1.10	192.168.1.1	TCP	1514	443 → 17286	[ACK] Seq=457841 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1268 161.790647003	192.168.1.10	192.168.1.1	TCP	1514	443 → 17286	[ACK] Seq=457841 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1269 161.80178416	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=460404 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1270 161.801380052	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=461864 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1271 161.801393059	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=463324 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1272 161.801896853	192.168.1.10	192.168.1.1	TCP	1514	443 → 17286	[ACK] Seq=464784 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1273 161.803450436	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=464784 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1274 161.803450436	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=464784 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1275 161.803469979	192.168.1.1	192.168.1.10	TCP	1514	443 → 17286	[ACK] Seq=467713 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1276 161.803473627	192.168.1.1	192.168.1.1	TCP	1514	443 → 17286	[ACK] Seq=469164 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1277 161.803396933	192.168.1.1	192.168.1.1	TCP	1514	443 → 17286	[ACK] Seq=469164 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]
1278 161.804332994	192.168.1.1	192.168.1.1	TCP	1514	443 → 17286	[ACK] Seq=470624 Ack=3916 Win=514 Len=1460 [TCP segment of a reassembled PDU]

- **pfSense Logs:** The pfSense firewall logs confirmed that packets from the Ubuntu attacker (192.168.18.85) destined for the Kali machine (192.168.1.128) on TCP port 80 were actively being blocked by the "block syn flood" rule.

Action	Time	Interface	Rule	Source	Destination	Protocol
✗	Jul 29 21:23:50	WAN	👤 block syn flood (1753805059)	i [192.168.18.85:61094]	i [192.168.1.128:80]	TCP:S
✗	Jul 29 21:23:50	WAN	👤 block syn flood (1753805059)	i [192.168.18.85:61095]	i [192.168.1.128:80]	TCP:S
✗	Jul 29 21:23:50	WAN	👤 block syn flood (1753805059)	i [192.168.18.85:61096]	i [192.168.1.128:80]	TCP:S
✗	Jul 29 21:23:50	WAN	👤 block syn flood (1753805059)	i [192.168.18.85:61097]	i [192.168.1.128:80]	TCP:S
✗	Jul 29 21:23:50	WAN	👤 block syn flood (1753805059)	i [192.168.18.85:61098]	i [192.168.1.128:80]	TCP:S
✗	Jul 29 21:23:50	WAN	👤 block syn flood (1753805059)	i [192.168.18.85:61099]	i [192.168.1.128:80]	TCP:S
✗	Jul 29 21:23:50	WAN	👤 block syn flood (1753805059)	i [192.168.18.85:61100]	i [192.168.1.128:80]	TCP:S
✗	Jul 29 21:23:50	WAN	👤 block syn flood (1753805059)	i [192.168.18.85:61101]	i [192.168.1.128:80]	TCP:S
✗	Jul 29 21:23:50	WAN	👤 block syn flood (1753805059)	i [192.168.18.85:61102]	i [192.168.1.128:80]	TCP:S
✗	Jul 29 21:23:50	WAN	👤 block syn flood (1753805059)	i [192.168.18.85:61103]	i [192.168.1.128:80]	TCP:S
✗	Jul 29 21:23:50	WAN	👤 block syn flood (1753805059)	i [192.168.18.85:61104]	i [192.168.1.128:80]	TCP:S
✗	Jul 29 21:23:50	WAN	👤 block syn flood (1753805059)	i [192.168.18.85:61105]	i [192.168.1.128:80]	TCP:S
✗	Jul 29 21:23:50	WAN	👤 block syn flood (1753805059)	i [192.168.18.85:61106]	i [192.168.1.128:80]	TCP:S
✗	Jul 29 21:23:50	WAN	👤 block syn flood (1753805059)	i [192.168.18.85:61107]	i [192.168.1.128:80]	TCP:S
✗	Jul 29 21:23:50	WAN	👤 block syn flood (1753805059)	i [192.168.18.85:61108]	i [192.168.1.128:80]	TCP:S

Conclusion:

The successful blocking of the TCP SYN flood attack, as verified by both Kali's Wireshark capture and pfSense's firewall logs, demonstrates the effectiveness of the configured pfSense rule in mitigating Denial of Service (DoS) attacks.