Cryptography And Cyber security risk analysis
(CY702)

Dr. Mohamed Saif

Project Part 2 (Defense Stratgies)

-Information Gathering

-Exploitation

Made by:

Ahmed Elsayed

Kareem Yasser

Part 2: Defense Strategies

Phase 1: Intrusion Detection (Snort)

Due to the limited availability of exploits within the Metasploit framework, certain vulnerabilities were not available in the database. This limitation restricted our ability to exploit a wide range of vulnerabilities. On the other hand, we conducted a test of potential vulnerabilities, relying on our readings and understanding of the vulnerabilities' nature. At the end, we chosen two vulnerabilities that can easily be replicated using Metasploit.

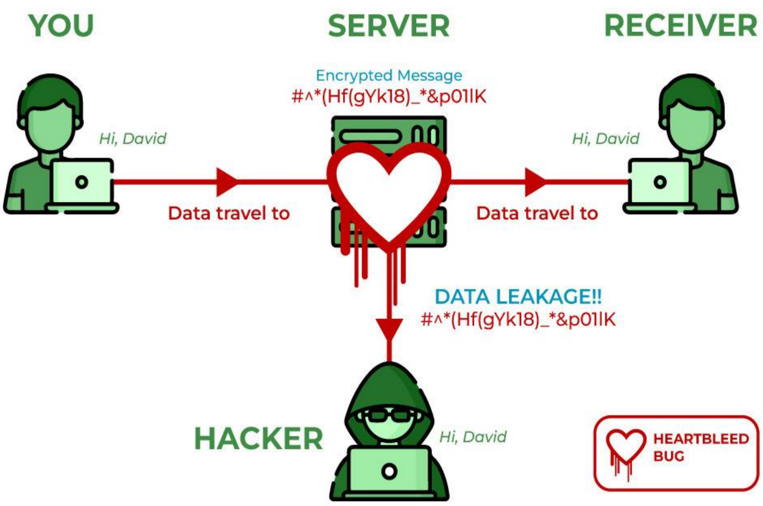
We conducted tests to validate their exploitability, and the results confirmed their potential for a Metasploit-based attacks. This decision was made after considering the available options, taking into account the ease of exploitation.

We have successfully used Metasploit to exploit the chosen vulnerabilities. The successful results confirm the vulnerability of the targeted systems; we have succeeded in our main goal of creating a strong defense system against these weaknesses. The results are shown below

1) Vulnerability Selection

a. 1st Vulnerability: OpenSSL Heartbeat Information Disclosure (Heartbleed)

In 2014, the encryption software, OpenSSL was found to have a serious security vulnerability known as the Heartbleed issue. According to security experts, this vulnerability is among the most serious ones in the past ten years for cybersecurity. Significant businesses and services were impacted, including Google, Yahoo, Dropbox, to name a few.



The Heartbleed vulnerability was assigned the CVE-2014-0160 classification under the Common Vulnerabilities and Exposures. A buffer overread occurs when a system permits access to data that ought to be controlled.

A computer can use the "heartbeat" feature implemented in the SSL standard to send a message to the other end of the connection and get a "beat" back, which indicates that the other computer is online. Attackers have the ability to send malicious messages in place of legitimate ones by using the heartbeat request capabilities. By doing this, the receiving computer may be forced to accept and send secret data, even the memory contents.

Risk	Probability	Impact
Medium (5.0)	Low (2.9)	Critical (10.0)

b. ***2nd Vulnerability: OpenSSL 'ChangeCipherSpec' MiTM Vulnerability***

The discovery of the CCS injection vulnerability (CVE-2014-0224), an OpenSSL flaw. OpenSSL versions 0.9.8, 1.0.0, and 1.0.1 are impacted.

A flaw in the way the 'ChangeCipherSpec' message handling during the SSL/TLS handshake occurs is known as the OpenSSL 'ChangeCipherSpec' MiTM (Man-in-the-Middle) vulnerability. It is possible that this vulnerability will result in a successful MiTM attack. By pretending to be one of the parties, an attacker can intercept and alter communications between two parties in this kind of assault. An essential part of the SSL/TLS handshake procedure is the 'ChangeCipherSpec' message.

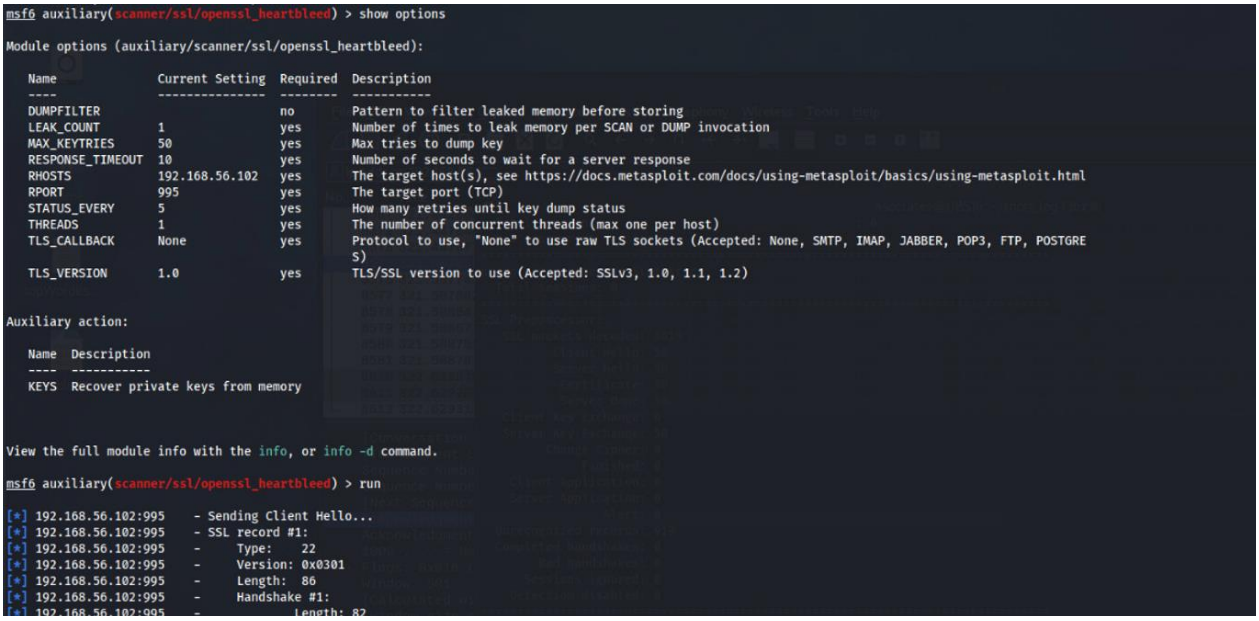
It initiates the change of communications from unencrypted to encrypted. However, an attacker can intercept and change network data can take advantage of this vulnerability because of a fault in the OpenSSL library's handling of this message. The attacker will then be able to decode the SSH tunnel, crack the encryption technique, and obtain sensitive data and passwords.

Risk	Probability	Impact
Medium (6.8)	Medium (6.4)	High (8.6)

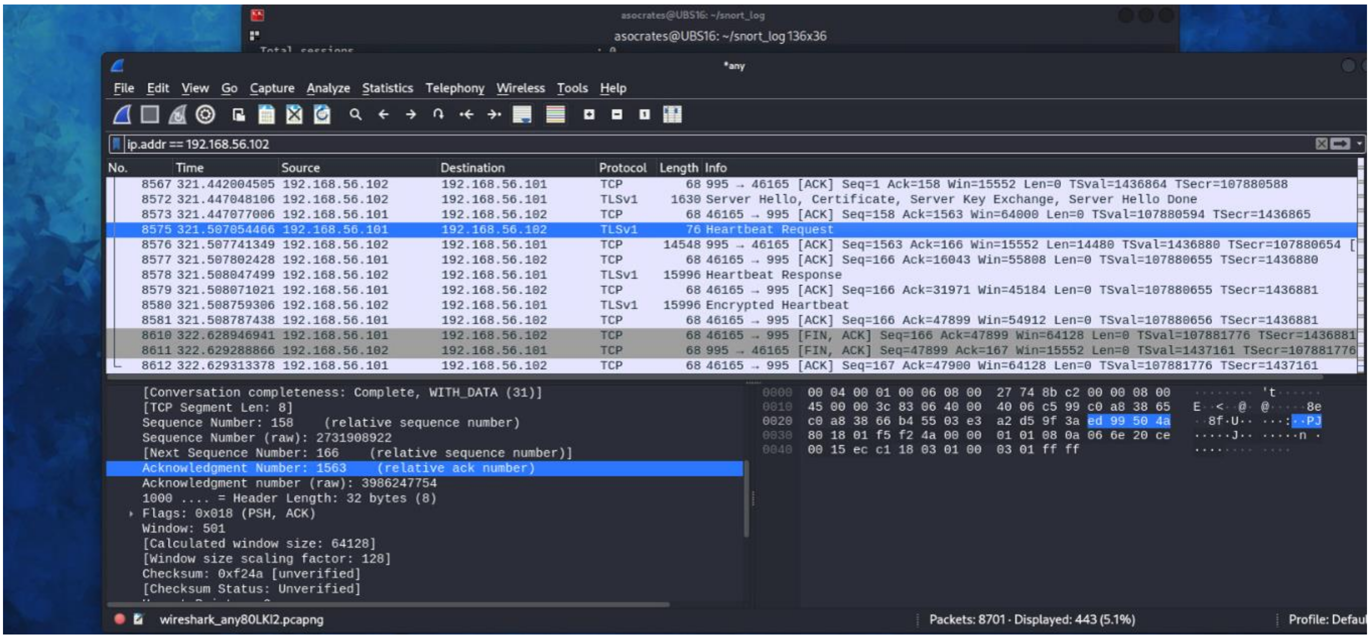
2) Snort Rules and 3) Exploit Execution:

We firstly, Configured Metasploit to run the attack, then had WireShark running in the background to capture the traffic, and then we examined the packets and created rules based on the packet content, screenshots will be provided below:

A. OpenSSL Heartbeat Information Disclosure (Heartbleed)



As shown above we configured the options to run on the vulnerable machine, then WireShark began capturing the traffic



Finally based on this info we created a new snort rule:

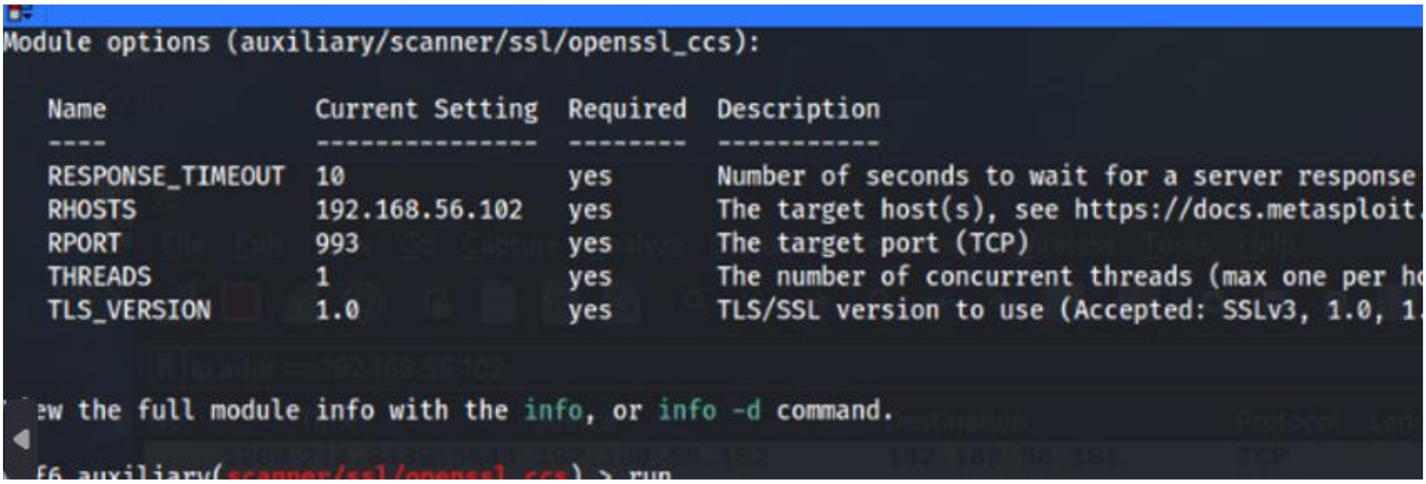
“alert tcp $HOME_NET ANY -> 192.168.56.102 995 (msg:”Heartbeat Request”; content “|18 03 01 00|” ; rawbytes; sid:100000)”

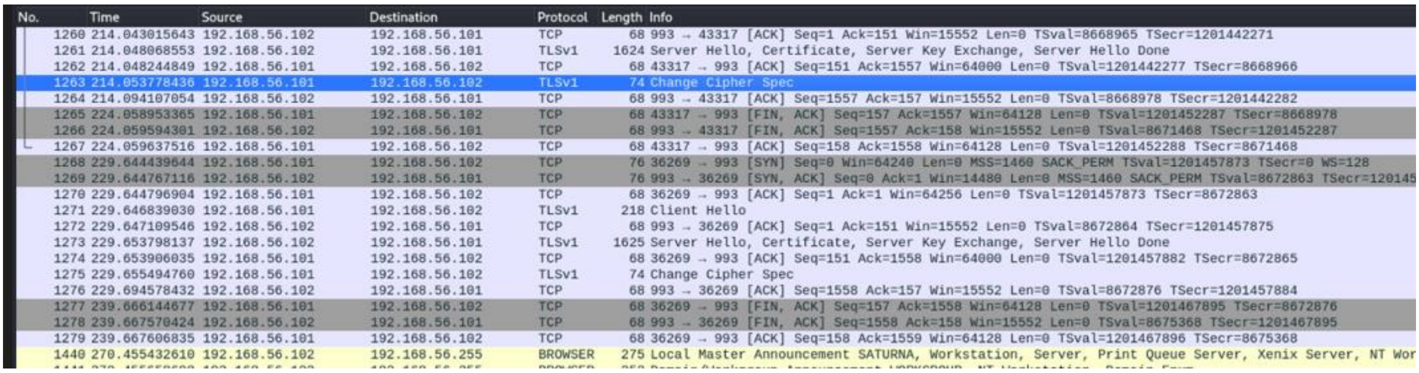
- alert tcp: This indicates that the rule is designed to trigger an alert for TCP traffic.
- $HOME_NET ANY -> 192.168.56.102: This specifies the source and destination IP addresses for the rule.
- $HOME_NET here represent the home network, since the attack is running locally, and it is defined in the subnet “192.168.56.0/24”
- 192.168.56.102 995 is the destination IP address, along with destination port 995
- msg:”Heartbeat Request”: This is the message associated with the rule, to make it easy to understand what the rule is doing.
- content “|18 03 01 00|”: This specifies the content that the rule is looking for in the payload of the TCP packet.
- rawbytes: This option instructs Snort to perform a raw byte search for the specified content, it looks for an exact sequence of bytes in the packet payload, treating the content as binary data.
- sid:100000: This is the Snort ID associated with the rule. Each rule has a unique Snort ID.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	01/24-10:34:39.298815	1	100000	0	Heartbeat request	TCP	192.168.56.101	34645	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0xBAB52C81	0x8C376ADC
2	01/24-10:34:52.502817	1	100000	0	Heartbeat request	TCP	192.168.56.101	44427	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0x8BE41042	0x2ED3E0E4
3	01/24-10:35:05.464247	1	100000	0	Heartbeat request	TCP	192.168.56.101	45867	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0xECF965B5	0xE74AEEA8
4	01/24-10:35:07.144974	100001	2000004	0	ICMP Connection		fe80::5f:3028:96f9:ea77		ff02::1:ff00:1		0A:00:27:00:00:06	33:33:FF:00:00:01	0x56			
5	01/24-10:35:18.362235	1	100000	0	Heartbeat request	TCP	192.168.56.101	42633	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0xF8E84996	0x64EEBCC4
6	01/24-10:35:31.396766	1	100000	0	Heartbeat request	TCP	192.168.56.101	32993	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0xDCA354AE	0xCE998A4B
7	01/24-10:35:44.887554	1	100000	0	Heartbeat request	TCP	192.168.56.101	39491	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0x7D8D66AC	0x8DBFBCCD
8	01/24-10:35:58.128645	1	100000	0	Heartbeat request	TCP	192.168.56.101	33031	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0xEF255E1C	0x3F906E91
9	01/24-10:36:08.145158	100001	2000004	0	ICMP Connection		fe80::5f:3028:96f9:ea77		ff02::1:ff00:1		0A:00:27:00:00:06	33:33:FF:00:00:01	0x56			
10	01/24-10:36:11.201950	1	100000	0	Heartbeat request	TCP	192.168.56.101	46601	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0x48D6BE8C	0x1E84830E
11	01/24-10:36:24.165006	1	100000	0	Heartbeat request	TCP	192.168.56.101	34925	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0x1C09EDA0	0xBF15908
12	01/24-10:36:37.051696	1	100000	0	Heartbeat request	TCP	192.168.56.101	41331	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0x519AAF9D	0xF4DDAA00
13	01/24-10:36:49.988339	1	100000	0	Heartbeat request	TCP	192.168.56.101	38941	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0xD28FD8CF	0xD96B39B8
14	01/24-10:37:02.950711	1	100000	0	Heartbeat request	TCP	192.168.56.101	45661	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0x2C4586C8	0x862F210
15	01/24-10:37:15.989652	1	100000	0	Heartbeat request	TCP	192.168.56.101	35691	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0x6A7DBD19	0x958A9B2
16	01/24-10:37:23.646096	100001	2000004	0	ICMP Connection		fe80::5f:3028:96f9:ea77		ff02::1:ff00:1		0A:00:27:00:00:06	33:33:FF:00:00:01	0x56			
17	01/24-10:37:28.980171	1	100000	0	Heartbeat request	TCP	192.168.56.101	37181	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0x9B1781A0	0xA7DF455C
18	01/24-10:37:41.877859	1	100000	0	Heartbeat request	TCP	192.168.56.101	34007	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0x48F5FA59	0xF6043539
19	01/24-10:37:54.945698	1	100000	0	Heartbeat request	TCP	192.168.56.101	37615	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0x897F600F	0x61F8D790
20	01/24-10:38:07.945803	1	100000	0	Heartbeat request	TCP	192.168.56.101	36781	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0x916E572D	0xCE95A507
21	01/24-10:38:20.849929	1	100000	0	Heartbeat request	TCP	192.168.56.101	39303	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0xFF25379	0x53EBA369
22	01/24-10:38:24.645954	100001	2000004	0	ICMP Connection		fe80::5f:3028:96f9:ea77		ff02::1:ff00:1		0A:00:27:00:00:06	33:33:FF:00:00:01	0x56			
23	01/24-10:38:33.770139	1	100000	0	Heartbeat request	TCP	192.168.56.101	33651	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0xFBF2687F	0xE85B4B18
24	01/24-10:38:46.742571	1	100000	0	Heartbeat request	TCP	192.168.56.101	45547	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0xD3F10AC0	0x8FC259BE
25	01/24-10:38:59.650319	1	100000	0	Heartbeat request	TCP	192.168.56.101	45465	192.168.56.102	995	08:00:27:74:8B:C2	08:00:27:C3:6B:1C	0x4A	***AP***	0x1689CBD9	0x46F8D49D

B. OpenSSL ‘ChangeCipherSpec’ MiTM Vulnerability

The same idea is applied her, exploit the machine with Metasploit, have WireShark, in the background and then create new rule, this is classified as a medium level vulnerability, and the way it works, it does send a CCS request and listens for a response to determine if the server is suspectable to attacks form this kind





Again, some packets to filters using the contents, and we came up with the rule below:

“alert tcp $HOME_NET any -> 192.168.56.102 993 (msg:”ChangeCipherSpec Mitm” ; content “|14 03 01 00| ; rawbytes; sid:1000001)”

- alert tcp: This indicates that the rule is designed to trigger an alert for TCP traffic.

- $HOME_NET ANY -> 192.168.56.102: This specifies the source and destination IP addresses for the rule.
- $HOME_NET here represent the home network, since the attack is running locally, and it is defined in the subnet “192.168.56.0/24”
- 192.168.56.102 995 is the destination IP address, along with destination port 995

- msg:”ChangeCipherSpec Mitm”: This is the message associated with the rule, to make it easy to understand what the rule is doing.

- content “|18 03 01 00|”: This specifies the content that the rule is looking for in the payload of the TCP packet.

- rawbytes: This option instructs Snort to perform a raw byte search for the specified content, it looks for an exact sequence of bytes in the packet payload, treating the content as binary data.

- sid:1000001: This is the Snort ID associated with the rule. Each rule has a unique Snort ID.

A	B	C	D	E	F	G	H	I	J	K	
1:39.629484	1	1000001	0	ChangeCipherSpec MITM Detected	TCP	192.168.56.101	40297	192.168.56.102	993	08:00:27:74:8B:C2	08:
2:00.702763	1	1000001	0	ChangeCipherSpec MITM Detected	TCP	192.168.56.101	45361	192.168.56.102	993	08:00:27:74:8B:C2	08:
2:14.195881	1	1000001	0	ChangeCipherSpec MITM Detected	TCP	192.168.56.101	43317	192.168.56.102	993	08:00:27:74:8B:C2	08:
2:28.297899	100001	2000004	0	ICMP Connection		fe80::5f:3028:96f9:ea77		ff02::1:ff00:1		0A:00:27:00:00:06	33:
2:29.790361	1	1000001	0	ChangeCipherSpec MITM Detected	TCP	192.168.56.101	36269	192.168.56.102	993	08:00:27:74:8B:C2	08:

```
asocrates@UBS16: ~ 237x51
GNU nano 2.5.3                File: /etc/snort/rules/local.rules

alert icmp any any -> any any (msg:"ICMP Connection";gid:100001; sid:2000004;)

alert tcp $HOME_NET any -> 192.168.56.102 995 (msg:"Heartbeat request"; content:"|18 03 01 00|"; rawbytes;sid:100000)

alert tcp $HOME_NET any -> 192.168.56.102 993 (msg:"ChangeCipherSpec MITM Detected"; content:"|14 03 01 00|"; rawbytes;sid:1000001)
```

4) Analysis for false positives and false negatives:

- We can say that in both rules, the risk of having a false positive is that is if legitimate apps that by chance uses the same byte sequence and same port, that could trigger a false positive.

- On the other hand, if the same attack generates a different byte sequence of if the request is obfuscated, the rule will then fail, generating a false negative.

Phase 2: Intrusion Detection (Iptables)

```
jpesci@saturna:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       icmp --  anywhere             anywhere
DROP       tcp  --  anywhere             192.168.56.102       tcp dpt:pop3s u32 "0x34=0x18030100:0x180301ff"
LOG        tcp  --  anywhere             192.168.56.102       tcp dpt:pop3s u32 "0x34=0x18030100:0x180301ff" LOG level warning prefix "Blocked:HeartBeat "
DROP       tcp  --  anywhere             192.168.56.102       tcp dpt:imaps u32 "0x34=0x14030100:0x1403ffff"
LOG        tcp  --  anywhere             192.168.56.102       tcp dpt:imaps u32 "0x34=0x14030100:0x1403ffff" LOG level warning prefix "Blocked:ChangeCipherSpec "

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
jpesci@saturna:~$
```

1) Define the IPTables rules

We wrote 2 rules for each vulnerability, 1 to log and detect the incoming packets and the other is to drop the packet and below we will explain each:

1. Log Heartbleed Attempt:

> “iptables -A INPUT -p tcp --dport 995 -d 192.168.56.102 -m u32 --u32 "52=0x18030000:0x1803FFFF" -j LOG --log-prefix "Blocked:Heartbeat "”

- -A INPUT: Appends the rule to the INPUT chain.
- -p tcp: Specifies the protocol as TCP.
- --dport 995: Filters packets destined for port 995 (POP3S).
- -d 192.168.56.102: Specifies the destination IP address.
- -m u32: Uses the u32 match module for more advanced packet matching.
- --u32 "52=0x18030000:0x1803FFFF": Matches packets where the 52nd byte of the TCP payload falls within the specified range, indicative of a Heartbleed attack, we opted to use the Hexadecimal notation
- -j LOG: Logs matching packets.
  - --log-prefix "Blocked:Heartbeat ": Adds a custom log prefix to identify the logged messages as related to Heartbleed attacks.

This rule records incoming TCP packets on port 995 to the given IP address (192.168.56.102) in the event that the payload's 52nd byte is inside the attack payload range for the Heartbleed vulnerability. The monitoring and identification of possible security issues is aided by logging.

2. Block Heartbleed Attack:

Same parameters as before, but the difference is we DORP(Block) the attack.

```
sudo iptables -A INPUT -p tcp --dport 995 -d 192.168.56.102 -m u32 --u32 "52=0x18030000:0x1803FFFF" -j DROP
```

3. Rule to Log CCS

```
iptables -A INPUT -p tcp --dport 993 -d 192.168.56.102 -m u32 --u32 "52=0x14030100x1403FFFF" -j LOG --log-prefix "Blocked:ChangeCiperSpec "
```

- -A INPUT: Appends the rule to the INPUT chain.
- -p tcp: Specifies the protocol as TCP.
- --dport 993: Specifies the destination port as 993 (IMAPS).
- -d 192.168.56.102: Specifies the destination IP address as 192.168.56.102.
- -m u32: Uses the u32 module for matching.
- --u32 "52=0x14030100x1403FFFF": Specifies a match condition based on the 52nd byte of the TCP payload, looking for the Heartbeat attack pattern.
- -j LOG --log-prefix "Blocked:Heartbeat ": Logs the matching packets with a prefix " Blocked:ChangeCiperSpec ".

This rule is designed to log packets that match the CSS attack pattern. The rule records incoming TCP packets on port 993 to the given IP address (192.168.56.102) in the event that the payload's 52nd byte is inside the attack payload range. The log prefix helps in identifying the specific type of attack.

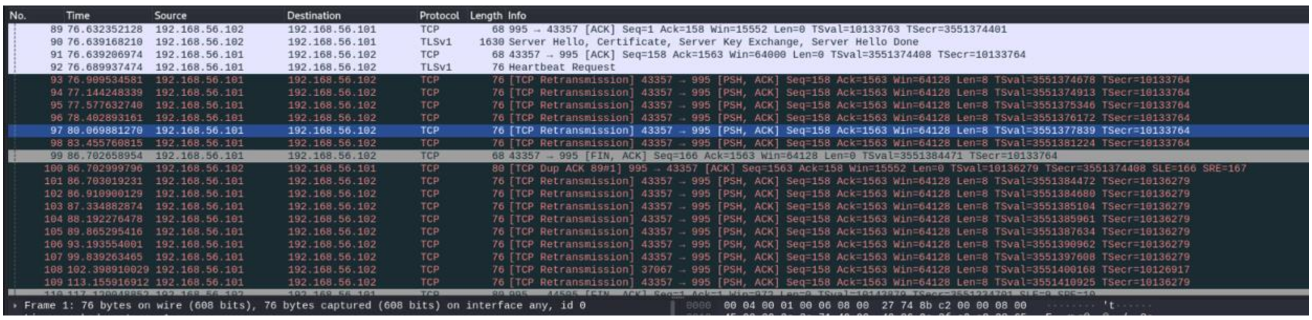
4. Blocking the CSS Attack

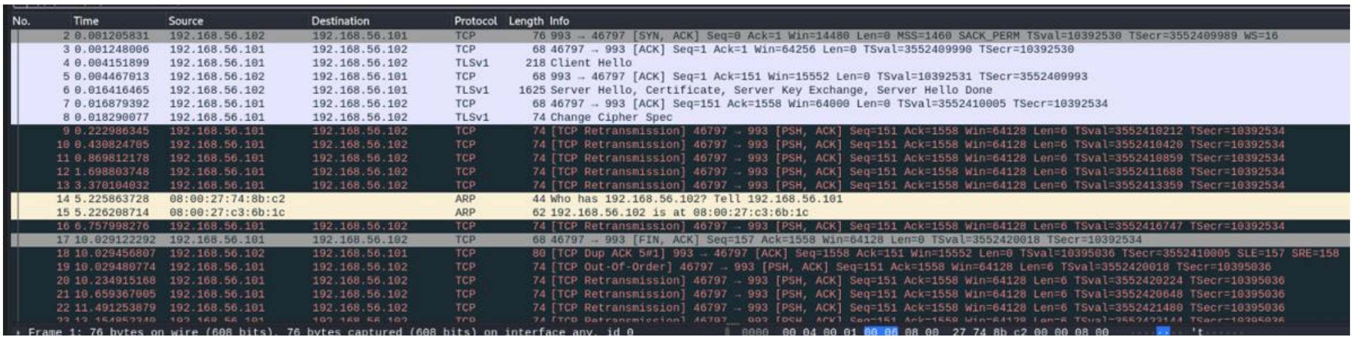
Same parameters as before, but the difference is we DORP(Block) the attack, instead of logging

```
iptables -A INPUT -p tcp --dport 993 -d 192.168.56.102 -m u32 --u32 "52=0x14030100x1403FFFF" -j DROP
```

2. Test the Iptable rules

We defined the rules and saved them and once again we launched the same 2 attacks and confirmed that the rules work and it does actually drop the incoming packets.





No.	Time	Source	Destination	Protocol	Length	Info
2	0.001205831	192.168.56.102	192.168.56.101	TCP	76	993 → 46797 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM TSval=10392530 TSecr=3552409989 WS=16
3	0.001248006	192.168.56.101	192.168.56.102	TCP	68	46797 → 993 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3552409990 TSecr=10392530
4	0.004151899	192.168.56.101	192.168.56.102	TLSv1	218	Client Hello
5	0.004467013	192.168.56.102	192.168.56.101	TCP	68	993 → 46797 [ACK] Seq=1 Ack=151 Win=15552 Len=0 TSval=10392531 TSecr=3552409993
6	0.016416465	192.168.56.102	192.168.56.101	TLSv1	1625	Server Hello, Certificate, Server Key Exchange, Server Hello Done
7	0.016879392	192.168.56.101	192.168.56.102	TCP	68	46797 → 993 [ACK] Seq=151 Ack=1558 Win=64000 Len=0 TSval=3552410005 TSecr=10392534
8	0.018290077	192.168.56.101	192.168.56.102	TLSv1	74	Change Cipher Spec
9	0.222986345	192.168.56.101	192.168.56.102	TCP	74	[TCP Retransmission] 46797 → 993 [PSH, ACK] Seq=151 Ack=1558 Win=64128 Len=6 TSval=3552410212 TSecr=10392534
10	0.430824705	192.168.56.101	192.168.56.102	TCP	74	[TCP Retransmission] 46797 → 993 [PSH, ACK] Seq=151 Ack=1558 Win=64128 Len=6 TSval=3552410420 TSecr=10392534
11	0.869812178	192.168.56.101	192.168.56.102	TCP	74	[TCP Retransmission] 46797 → 993 [PSH, ACK] Seq=151 Ack=1558 Win=64128 Len=6 TSval=3552410859 TSecr=10392534
12	1.698803748	192.168.56.101	192.168.56.102	TCP	74	[TCP Retransmission] 46797 → 993 [PSH, ACK] Seq=151 Ack=1558 Win=64128 Len=6 TSval=3552411688 TSecr=10392534
13	3.370104032	192.168.56.101	192.168.56.102	TCP	74	[TCP Retransmission] 46797 → 993 [PSH, ACK] Seq=151 Ack=1558 Win=64128 Len=6 TSval=3552413359 TSecr=10392534
14	5.225863728	08:00:27:74:8b:c2		ARP	44	Who has 192.168.56.102? Tell 192.168.56.101
15	5.226208714	08:00:27:c3:6b:1c		ARP	62	192.168.56.102 is at 08:00:27:c3:6b:1c
16	6.757998276	192.168.56.101	192.168.56.102	TCP	74	[TCP Retransmission] 46797 → 993 [PSH, ACK] Seq=151 Ack=1558 Win=64128 Len=6 TSval=3552416747 TSecr=10392534
17	10.029122292	192.168.56.101	192.168.56.102	TCP	68	46797 → 993 [FIN, ACK] Seq=157 Ack=1558 Win=64128 Len=0 TSval=3552420018 TSecr=10392534
18	10.029456807	192.168.56.102	192.168.56.101	TCP	80	[TCP Dup ACK 5#1] 993 → 46797 [ACK] Seq=1558 Ack=151 Win=15552 Len=0 TSval=10395036 TSecr=3552410005 SLE=157 SRE=158
19	10.029480774	192.168.56.101	192.168.56.102	TCP	74	[TCP Out-Of-Order] 46797 → 993 [PSH, ACK] Seq=151 Ack=1558 Win=64128 Len=6 TSval=3552420018 TSecr=10395036
20	10.234915168	192.168.56.101	192.168.56.102	TCP	74	[TCP Retransmission] 46797 → 993 [PSH, ACK] Seq=151 Ack=1558 Win=64128 Len=6 TSval=3552420224 TSecr=10395036
21	10.659367005	192.168.56.101	192.168.56.102	TCP	74	[TCP Retransmission] 46797 → 993 [PSH, ACK] Seq=151 Ack=1558 Win=64128 Len=6 TSval=3552420648 TSecr=10395036
22	11.491253879	192.168.56.101	192.168.56.102	TCP	74	[TCP Retransmission] 46797 → 993 [PSH, ACK] Seq=151 Ack=1558 Win=64128 Len=6 TSval=3552421480 TSecr=10395036

Frame 1: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface any, id 0

We can see here from the above images that the defines rules successfully blocked or dropped the incoming packets for the heartbeat request and the ChangeCipherSpec, proving the defined rules are working the ways they should be, but also just like the case with snort, there is a chance for blocking real legitimate traffic, resulting in false positives, or If an attack uses a different byte sequence or attacks is obfuscated, the rule may fail to detect the malicious activity, resulting in false negatives.

3. Additional defense strategy

Given the vulnerabilities discovered, a multi-layered defense strategy should be implemented to strengthen the target systems' security posture. First, intrusion detection and prevention systems (IDS/IPS) such as Snort can be used to monitor and filter network traffic based on signatures or behavioral anomalies. Configuring Snort signatures to recognize and stop certain attack patterns associated with vulnerabilities such as Heartbleed, SWEET32, Logjam, POODLE, SSLv2 and SSLv3 Protocol Detection, and so forth is possible. Snort's flexibility allows for real-time traffic analysis, allowing for quick reactions to potential threats.

Also, firewall rules established with iptables can strengthen the defense by restricting and filtering malicious traffic at the network level. For example, rules that block connections may be imposed using weak cryptographic techniques. Using iptables, a strong network perimeter defense can be created.

Additionally, using secure updates and setups for services like SSH and DNS servers is important. It is easy to enforce the use of strong cryptography settings and disable support for weak algorithms by modifying the configurations of SSH servers. DNS servers should be configured to lower the possibility of cache poisoning and eavesdropping. Regular updates and patches are necessary for software, like OpenSSL.

At last, a Web Application Firewall (WAF) can be used to prevent XSS attacks in light of the found web-based vulnerabilities. Web application firewalls gives additional security layer to websites by inspecting and filtering HTTP traffic, by blocking potentially harmful requests.