VLAN Hopping Attack

Haboob Team
Table of Contents

What is VLAN Hopping? .................................................................................................................. 2
What is DTP? .................................................................................................................................. 2
Attack Demonstration ..................................................................................................................... 3
Attack Demonstration Requirements .............................................................................................. 3
Step-by-step ...................................................................................................................................... 3
mitigation .......................................................................................................................................... 9
References ......................................................................................................................................... 10
**What is VLAN Hopping?**

VLAN hopping (virtual local area network hopping) is a method of attacking a network by sending packets to a port that is not normally accessible from a given end system. (A VLAN is a local area network with a definition that maps devices on some other basis than geographic location - for example, by department, type of user, or primary application.)

A VLAN hopping attack can occur in either of two ways. If a network switch is set for autotrunking, the attacker turns it into a switch that appears as if it has a constant need to trunk (that is, to access all the VLAN allowed on the trunk port).

**What is DTP?**

Dynamic Trunking Protocol (DTP) is a Cisco proprietary trunking protocol, which is used to automatically negotiate trunks between Cisco switches. Dynamic Trunking Protocol (DTP) can be used negotiate and form trunk connection between Cisco switches dynamically.

Dynamic Trunking Protocol (DTP) can operate in different trunking modes, as shown below.

<table>
<thead>
<tr>
<th>DTP Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dynamic desirable</td>
<td>A switch port configured as DTP dynamic desirable mode will actively try to convert the link to a trunk link using Dynamic Trunking Protocol (DTP). If the port connected to other port is capable to form a trunk, a trunk link will be formed. The interface which is configured as DTP dynamic desirable mode will generate DTP messages on the interface. If the switch receive DTP messages from the other side switch, it will assume that other side port is capable for handling tagged frames and a trunk link will be formed between two switches.</td>
</tr>
<tr>
<td>dynamic auto</td>
<td>A switch port configured as DTP dynamic auto is capable to form trunk link if the other side switch interface is configured to form a trunk interface and can negotiate with trunk using DTP. A switch interface which is configured as DTP dynamic auto mode will not generate DTP messages on the interface. DTP dynamic auto interface will only listen passively for DTP messages from other side switch's interface. If the DTP dynamic auto interface receives a DTP message from the interface of the other side switch, a trunk link will be formed.</td>
</tr>
<tr>
<td>trunk</td>
<td>A switch interface which is configured as trunk mode converts the switch's interface to pure trunking mode. A trunk mode interface can also negotiate with the other side switch interface to form a trunk link between two switches.</td>
</tr>
<tr>
<td>nonegotiate</td>
<td>The nonegotiate mode disables sending DTP packets from an interface. “nonegotiate” mode is possible only when the interface switchport mode is “access” or “trunk”. DTP is disabled.</td>
</tr>
<tr>
<td>access</td>
<td>A switch interface which is configured as access mode converts the switch's interface to access mode. “access” mode prevents the use of trunking and make the port as a pure access port. No frame tagging will happen in an access port. An access port belongs to a VLAN.</td>
</tr>
</tbody>
</table>

In order to make the attack successful, the switch mode has to be set on dynamic desirable, dynamic auto or trunk so the switches can be negotiating and sending DTP packets. By default, the Cisco switches are set to dynamic desirable.
ATTACK DEMONSTRATION

In this research paper, we are going to demonstrate an attack for VLAN Hopping and we will do step-by-step in order to understand the attack scenario.

ATTACK DEMONSTRATION REQUIREMENTS:
- GNS3 (For simulating the network).
- Kali Linux (Attacker).
- Virtual Host (Victim).
- Switch (Cisco-iosvl2).

STEP-BY-STEP

First of all, we made a scenario for a small network which has three clients (an attacker and two victims) in the same network and connected together in a switch. To understand the topology of the network, see the below design:

![Network Diagram]

We have the switch which in connected to the PC-1 (IP: 172.16.0.3), PC-2 (IP: 172.16.0.3) and Attacker (IP: 172.16.0.5). The table below explains the clients and the VLAN IDs:

<table>
<thead>
<tr>
<th>Name</th>
<th>IP</th>
<th>VLAN ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC-1</td>
<td>172.16.0.3</td>
<td>100</td>
</tr>
<tr>
<td>PC-2</td>
<td>10.0.0.4</td>
<td>200</td>
</tr>
<tr>
<td>Attacker</td>
<td>172.16.0.5</td>
<td>100</td>
</tr>
</tbody>
</table>
We supposed that the attacker got an access to a network and he is in a VLAN 100 along with the PC-1 that in VLAN 100 (same subnet and VLAN) which means that they can ping each other. The PC-2 which is in another subnet and has the VLAN 200 cannot ping the PC-1 and the attacker as well. Let’s do a ping from the PC-1 to the attacker and vice versa:

```
PC-1> ping 172.16.0.5
54 bytes from 172.16.0.5 icmp_seq=1 ttl=64 time=0.000 ms
54 bytes from 172.16.0.5 icmp_seq=2 ttl=64 time=0.000 ms
54 bytes from 172.16.0.5 icmp_seq=3 ttl=64 time=0.000 ms
54 bytes from 172.16.0.5 icmp_seq=4 ttl=64 time=0.000 ms
54 bytes from 172.16.0.5 icmp_seq=5 ttl=64 time=0.000 ms
```

From Kali to PC-1:

```
root@kali:-- ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
  inet 172.16.0.5 netmask 255.255.255.0 broadcast 172.16.0.255
  ether 00:0c:29:3f:e4:fa txqueuelen 1000 (Ethernet)
  RX packets 30 bytes 7545 (7.3 KBytes)
  TX packets 100 bytes 19040 (18.6 KBytes)
  RX errors 0 dropped 0 overruns 0 frame 0
  TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
lo: flags=73<LOOPBACK,UP,LOWER_UP> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0 broadcast 127.255.255.255
    loop txqueuelen 1000 (Local Loopback)
    RX packets 2176 bytes 175920 (171.7 KBytes)
    TX packets 2176 bytes 175920 (171.7 KBytes)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
root@kali:-- ping 172.16.0.3
PING 172.16.0.3 [172.16.0.3] 56(84) bytes of data.
64 bytes from 172.16.0.3: icmp_seq=1 ttl=64 time=37.9 ms
64 bytes from 172.16.0.3: icmp_seq=2 ttl=64 time=22.6 ms
64 bytes from 172.16.0.3: icmp_seq=3 ttl=64 time=25.0 ms
64 bytes from 172.16.0.3: icmp_seq=4 ttl=64 time=18.3 ms
--- 172.16.0.3 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3078ms
rtt min/avg/max/mdev = 18.387/26.263/37.967/7.283 ms
```

And let’s do a ping from Kali to PC-2 (in a different VLAN):

```
root@kali:-- ping 10.0.0.4
PING 10.0.0.4 [10.0.0.4] 56(84) bytes of data.
From 172.16.0.3 icmp_seq=1 Destination Host Unreachable
From 172.16.0.3 icmp_seq=2 Destination Host Unreachable
From 172.16.0.3 icmp_seq=3 Destination Host Unreachable
--- 10.0.0.4 ping statistics ---
4 packets transmitted, 0 received, +3 errors, 100% packet loss, time 3879ms
pipe 4
root@kali:--
```

It’s clearly that it will not ping because they are in a different VLAN.
And here is the VLAN table from the switch console:

<table>
<thead>
<tr>
<th>VLAN Name</th>
<th>Status</th>
<th>Ports</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>active</td>
<td>G1/0/1, G1/0/2</td>
</tr>
<tr>
<td>100 VLAN100</td>
<td>active</td>
<td>G1/0/1</td>
</tr>
<tr>
<td>200 VLAN200</td>
<td>active</td>
<td>G1/0/2</td>
</tr>
<tr>
<td>100 fddi-default</td>
<td>act/unsap</td>
<td></td>
</tr>
<tr>
<td>1000 tr0f-default</td>
<td>act/unsap</td>
<td></td>
</tr>
<tr>
<td>1004 fddo-default</td>
<td>act/unsap</td>
<td></td>
</tr>
<tr>
<td>1006 tibf-default</td>
<td>act/unsap</td>
<td></td>
</tr>
</tbody>
</table>

The interfaces (G0/0, G0/1) are assigned to VLAN 100 which are the (Kali and PC-1), and the interface (G0/2) is assigned to VLAN 200.

As we said previously, in order to make the attack successful, the switch has to be on default configuration (in Dynamic Desirable), let’s check the configuration of the attacker’s interface (G0/0):

Indeed the switch port is set on Dynamic Desirable thus the VLANs can be negotiated together.
Now we can run the tool (yersinia) in order to enable the TRUNK mode, but before we run the attack let’s see the status of the VLAN:

<table>
<thead>
<tr>
<th>Port</th>
<th>Name</th>
<th>Status</th>
<th>Vlan</th>
<th>Duplex</th>
<th>Speed</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>G1/0</td>
<td></td>
<td>connected</td>
<td>100</td>
<td>auto</td>
<td>auto</td>
<td>unknown</td>
</tr>
<tr>
<td>G1/1</td>
<td></td>
<td>connected</td>
<td>100</td>
<td>auto</td>
<td>auto</td>
<td>unknown</td>
</tr>
<tr>
<td>G1/2</td>
<td></td>
<td>connected</td>
<td>200</td>
<td>auto</td>
<td>auto</td>
<td>unknown</td>
</tr>
<tr>
<td>G1/3</td>
<td></td>
<td>connected</td>
<td>1</td>
<td>auto</td>
<td>auto</td>
<td>unknown</td>
</tr>
<tr>
<td>G1/4</td>
<td></td>
<td>connected</td>
<td>1</td>
<td>auto</td>
<td>auto</td>
<td>unknown</td>
</tr>
<tr>
<td>G1/5</td>
<td></td>
<td>connected</td>
<td>1</td>
<td>auto</td>
<td>auto</td>
<td>unknown</td>
</tr>
</tbody>
</table>

So the VLANs are set correctly and we will run the debug mode to see the incoming DTP packets.

Now we can run the tool (yersinia) and choose DTP and then launch attack:
Then choose “enabling trunking” and click OK:

Then we will go back to switch console and we can see that there are packets have been sent as shown below:

```
✈IOS-L2-01#
*Jul 17 00:16:28.850: DTP-event:Gi0/0:Received packet event ../dyntrk/dyntrk_process.c:2213
```

We will show the VLAN table:

```
✈IOS-L2-01#show interfaces status

<table>
<thead>
<tr>
<th>Port</th>
<th>Name</th>
<th>Status</th>
<th>VLAN</th>
<th>Dynamic</th>
<th>Speed</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gi0/0</td>
<td></td>
<td>connected</td>
<td>trunk</td>
<td>auto</td>
<td>auto unknown</td>
<td></td>
</tr>
<tr>
<td>Gi0/1</td>
<td></td>
<td>connected</td>
<td>100</td>
<td>auto</td>
<td>auto unknown</td>
<td></td>
</tr>
<tr>
<td>Gi0/2</td>
<td></td>
<td>connected</td>
<td>200</td>
<td>auto</td>
<td>auto unknown</td>
<td></td>
</tr>
<tr>
<td>Gi0/3</td>
<td></td>
<td>connected</td>
<td>1</td>
<td>auto</td>
<td>auto unknown</td>
<td></td>
</tr>
<tr>
<td>Gi1/0</td>
<td></td>
<td>connected</td>
<td>1</td>
<td>auto</td>
<td>auto unknown</td>
<td></td>
</tr>
<tr>
<td>Gi1/1</td>
<td></td>
<td>connected</td>
<td>1</td>
<td>auto</td>
<td>auto unknown</td>
<td></td>
</tr>
<tr>
<td>Gi1/2</td>
<td></td>
<td>connected</td>
<td>1</td>
<td>auto</td>
<td>auto unknown</td>
<td></td>
</tr>
<tr>
<td>Gi1/3</td>
<td></td>
<td>connected</td>
<td>1</td>
<td>auto</td>
<td>auto unknown</td>
<td></td>
</tr>
<tr>
<td>Gi2/0</td>
<td></td>
<td>connected</td>
<td>1</td>
<td>auto</td>
<td>auto unknown</td>
<td></td>
</tr>
<tr>
<td>Gi2/1</td>
<td></td>
<td>connected</td>
<td>1</td>
<td>auto</td>
<td>auto unknown</td>
<td></td>
</tr>
</tbody>
</table>
```

We can see that the interface (Gi0/0) is set on trunk which means that we can jump other VLANs!
And we can see that all the VLANS are allowed on interface (g0/0):

```
Port Mode Encapsulation Status Native vlan
G10/0 desirable n-802.1q trunking 1
```

On Kali, we will add the below commands:

```
root@kali:~

modprobe 8021q
vconfig add eth0 200
config add eth0 200
```

We added a new VLAN interface and we gave it the ID=200. Then we added a new IP and make it up then assign the new created VLAN interface to the eth0.200 interface and make up.

Finally, we can ping the PC-2 that were not accessible and on other VLAN.

```
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.
64 bytes from 10.0.0.4: icmp_seq=1 ttl=64 time=18.4 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=64 time=21.4 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=64 time=33.9 ms
64 bytes from 10.0.0.4: icmp_seq=4 ttl=64 time=24.9 ms
```

So we successfully jumped to the VLAN (200)!
MITIGATION

VLAN Hopping can only be exploited when interfaces are set to negotiate a trunk. To prevent the VLAN hopping from being exploited, we can do the below mitigations:

- Ensure that ports are not set to negotiate trunks automatically by disabling DTP:
  ```
  Switch(config-if)# switchport nonegotiate
  ```
- NEVER use VLAN 1 at all.
- Disable unused ports and put them in an unused VLAN
- Always use a dedicated VLAN ID for all trunk ports.
REFERENCES

- https://searchsecurity.techtarget.com/definition/VLAN-hopping