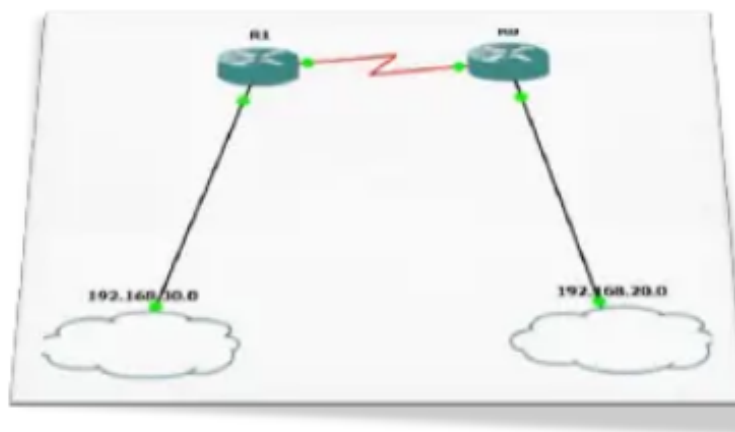


Institut Universitaire Technologique
réseaux et télécoms
Cybersécurité

Projet en SAE Cybersécurité :

Concevoir un réseau sécurisé à l'aide de GNS3



Réalisé par :

HONORINE Kylian
GRONDIN Angélique

Encadré par :

SITAL DAHONE Aloïs

Année Universitaire : 2025/2026

TABLE DES MATIÈRES

Introduction :	3
Configurations Switchs et Routeurs :	3
I. VLAN.....	3
Pour SA1.....	3
Pour SA2.....	4
Pour SA3.....	5
II. Routage et routage inter-vlan.....	5
Pour le switch L3 SD1,.....	5
Pour le switch L3 SD2.....	7
III. HSRP (Hot Standby Router Protocol).....	8
Pour SD1.....	8
Pour SD2.....	8
IV. PVST+ (Per VLAN Spanning Tree Plus).....	9
Pour SD1 :	9
Pour SD2 :	11
V. ACLS.....	12
ACL RH (Sur SD1).....	12
ACL serveur de fichier (Sur SD1).....	13
PRE VPN MPLS :	14
Sur les SDx :	14
Sur les PEx :	14
VPN MPLS.....	15
PE1 :	15
PE2 :	16
PE3 :	17
PE :	18
Schéma final :	20
Conclusion :	21
ANNEXE :	23
Configuration GNS3 image debian et installation FTP debian.....	23

Introduction :

Dans le cadre de la SAE 3.Cyber 03, nous avons pour mission de concevoir une infrastructure réseau sécurisée pour une entreprise multi-sites. Ce projet vise à interconnecter trois sites distants via un VPN MPLS tout en garantissant la sécurité, la redondance et la fiabilité du réseau.

Nous mettrons en œuvre des VLANs, du routage inter-VLAN, des ACL pour le contrôle d'accès, ainsi que des protocoles comme STP/PVST+ et HSRP pour éviter les boucles et assurer la redondance. La validation et la simulation seront réalisées avec Packet Tracer et GNS3, en respectant les recommandations de l'ANSSI.

Ce projet nous permettra de combiner compétences réseau et cybersécurité pour répondre aux besoins d'une infrastructure moderne et sécurisée.

Configurations Switchs et Routeurs :

I. VLAN

Pour SA1

```
vlan 10
name RH
vlan 20
name Vente
vlan 30
name AdminReseau
vlan 40
name serverfichier
vlan 50
name clientwifi
```

```
interface Ethernet0/1
switchport access vlan 10
switchport mode access
```

```
interface Ethernet0/2
switchport access vlan 20
switchport mode access
```

```
interface Ethernet0/3
switchport access vlan 30
switchport mode access
```

```
interface Ethernet1/0
switchport access vlan 40
switchport mode access
```

```
interface Ethernet1/1
switchport access vlan 50
switchport mode access
```

```
interface Ethernet0/0
switchport trunk allowed vlan 10,20,30,40,50,60,99
switchport trunk encapsulation dot1q
switchport mode trunk
```

Pour SA2

Ce switch L2 contient seulement la vlan 60 pour les VPCS.

```
vlan 60
name other
```

```
interface Ethernet0/1
switchport access vlan 60
```

```
interface Ethernet0/2
switchport access vlan 60
```

```
interface Ethernet0/3
switchport trunk allowed vlan 60
switchport trunk encapsulation dot1q
switchport mode trunk
```

Pour SA3

Nous utilisons la même configuration que pour SA2.

```
vlan 60  
name other
```

```
interface Ethernet0/3  
  switchport access vlan 60  
  switchport mode access
```

```
interface Ethernet1/0  
  switchport access vlan 60
```

```
interface Ethernet1/1  
  switchport access vlan 60
```

```
interface Ethernet0/0  
  switchport trunk allowed vlan 60  
  switchport trunk encapsulation dot1q  
  switchport mode trunk
```

II. Routage et routage inter-vlan

Les switches de niveau 3 sont configurés pour assurer le routage inter-VLAN, permettant ainsi aux différents services de communiquer tout en conservant une isolation optimale.

Pour le switch L3 SD1,

Nous avons configuré les commandes pour les vlans de 10 à 50.

```
interface Vlan10
```

```
ip address 192.168.10.1 255.255.255.0
```

```
interface Vlan20
```

```
ip address 192.168.20.1 255.255.255.0
```

```
interface Vlan30
```

```
ip address 192.168.30.1 255.255.255.0
```

```
interface Vlan40
```

```
ip address 192.168.40.1 255.255.255.0
```

```
interface Vlan50
```

```
ip address 192.168.50.1 255.255.255.0
```

```
interface Vlan99
```

```
ip address 192.168.99.1 255.255.255.0
```

```
interface gi0/0
```

```
switchport access vlan 99
```

```
interface gi0/1
```

```
switchport trunk allowed vlan 10,20,30,40,50
```

Ping d'un des VPCS sur sa passerelle :

```
PC8> ip 192.168.10.2 255.255.255.0 192.168.10.1
Checking for duplicate address...
PC8 : 192.168.10.2 255.255.255.0 gateway 192.168.10.1

PC8> ping 192.168.10.1

84 bytes from 192.168.10.1 icmp_seq=1 ttl=255 time=27.363 ms
84 bytes from 192.168.10.1 icmp_seq=2 ttl=255 time=2.696 ms
^C
PC8> █
```

Table de routage SD1 :

```

      192.168.10.0/24 is variably subnetted, 2 subnets, 2 masks
C       192.168.10.0/24 is directly connected, Vlan10
L       192.168.10.1/32 is directly connected, Vlan10
      192.168.20.0/24 is variably subnetted, 2 subnets, 2 masks
C       192.168.20.0/24 is directly connected, Vlan20
L       192.168.20.1/32 is directly connected, Vlan20
      192.168.30.0/24 is variably subnetted, 2 subnets, 2 masks
C       192.168.30.0/24 is directly connected, Vlan30
L       192.168.30.1/32 is directly connected, Vlan30
      192.168.40.0/24 is variably subnetted, 2 subnets, 2 masks
C       192.168.40.0/24 is directly connected, Vlan40
L       192.168.40.1/32 is directly connected, Vlan40
      192.168.50.0/24 is variably subnetted, 2 subnets, 2 masks
C       192.168.50.0/24 is directly connected, Vlan50
L       192.168.50.1/32 is directly connected, Vlan50
SD1#
```

Test de ping d'un pc à un autre :

```
PC6> ping 192.168.50.1

84 bytes from 192.168.50.1 icmp_seq=1 ttl=255 time=1.917 ms
^C
PC6> ping 192.168.40.1

84 bytes from 192.168.40.1 icmp_seq=1 ttl=255 time=2.274 ms
^C
PC6> ping 192.168.40.2

84 bytes from 192.168.40.2 icmp_seq=1 ttl=63 time=9.285 ms
84 bytes from 192.168.40.2 icmp_seq=2 ttl=63 time=2.517 ms
84 bytes from 192.168.40.2 icmp_seq=3 ttl=63 time=2.753 ms
84 bytes from 192.168.40.2 icmp_seq=4 ttl=63 time=2.189 ms
84 bytes from 192.168.40.2 icmp_seq=5 ttl=63 time=3.135 ms
^C
PC6> □
```

Pour le switch L3 SD2

Nous avons une seule interface VLAN (60) qui met les VPCS dans celle-ci.

```
interface Vlan60
```

```
ip address 192.168.60.1 255.255.255.0
```

```
int vlan 99
```

```
ip address 192.168.99.2 255.255.255.0
```

```
int gi0/1
```

```
switchport trunk allowed vlan 60
```

```
int gi0/0
```

```
switchport a vlan 99
```

Jusqu'ici, tout est interconnecté et le ping fonctionne.

III. HSRP (Hot Standby Router Protocol)

Le protocole HSRP est mis en place pour assurer la redondance des routeurs en cas de panne.

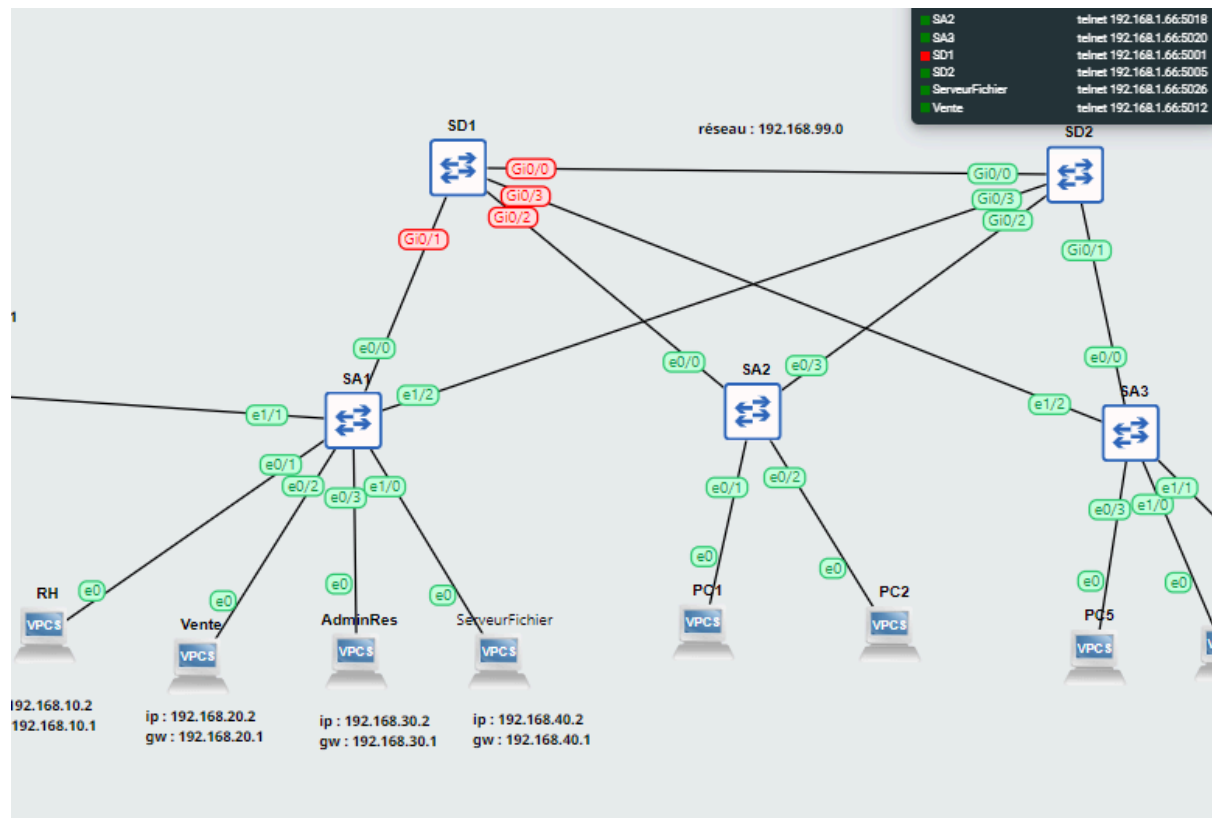
Pour SD1

```
spanning-tree mode pvst
spanning-tree extend system-id
spanning-tree vlan 10 priority 4096
```

Pour SD2

```
spanning-tree mode pvst
spanning-tree extend system-id
spanning-tree vlan 20 priority 4096
```

Nous observons une redondance lorsque SD1 cesse de fonctionner.




```
192.168.20.1 icmp_seq=1 timeout
192.168.20.1 icmp_seq=2 timeout
^C
PC7> ping 192.168.20.1

192.168.20.1 icmp_seq=1 timeout
84 bytes from 192.168.20.1 icmp_seq=2 ttl=255 time=3.032 ms
84 bytes from 192.168.20.1 icmp_seq=3 ttl=255 time=3.122 ms

PC7> ^Cping 192.168.20.1

192.168.20.1 icmp_seq=1 timeout
84 bytes from 192.168.20.1 icmp_seq=2 ttl=255 time=2.379 ms
84 bytes from 192.168.20.1 icmp_seq=3 ttl=255 time=3.654 ms
^C
PC7> ping 192.168.20.1

84 bytes from 192.168.20.1 icmp_seq=1 ttl=255 time=2.675 ms
84 bytes from 192.168.20.1 icmp_seq=2 ttl=255 time=2.734 ms
84 bytes from 192.168.20.1 icmp_seq=3 ttl=255 time=2.084 ms
^C
PC7> ping 192.168.20.1

84 bytes from 192.168.20.1 icmp_seq=1 ttl=255 time=2.058 ms
^C
PC7> []
```

IV. PVST+ (Per VLAN Spanning Tree Plus)

Le protocole PVST+ permet d'optimiser l'équilibrage de charge et d'éviter les boucles de commutation.

Pour SD1 :

```
interface Vlan10
ip address 192.168.10.2 255.255.255.0
standby 0 priority 110
standby 10 ip 192.168.10.1
standby 10 preempt
```

```
interface Vlan20
ip address 192.168.20.2 255.255.255.0
standby 20 ip 192.168.20.1
standby 20 priority 110
standby 20 preempt
```

```
interface Vlan30
ip address 192.168.30.2 255.255.255.0
standby 30 ip 192.168.30.1
standby 30 priority 110
standby 30 preempt
```

```
interface Vlan40
ip address 192.168.40.2 255.255.255.0
standby 40 ip 192.168.40.1
standby 40 priority 110
standby 40 preempt
```

```
interface Vlan50
ip address 192.168.50.2 255.255.255.0
standby 50 ip 192.168.50.1
standby 50 priority 110
standby 50 preempt
```

```
interface Vlan60
ip address 192.168.60.2 255.255.255.0
standby 60 ip 192.168.60.1
standby 60 priority 110
standby 60 preempt
```

!

```
interface Vlan99
ip address 192.168.99.2 255.255.255.0
standby 99 ip 192.168.99.1
standby 99 priority 110
standby 99 preempt
```

```
SD1(config-if)#
SD1(config-if)#int vlan 30
SD1(config-if)#ip addr 192.168.30.2 255.255.255.0
SD1(config-if)#stda
SD1(config-if)#sta
SD1(config-if)#standby 30 pr
SD1(config-if)#standby 30 pri
SD1(config-if)#standby 30 priority 110
SD1(config-if)#stdan
SD1(config-if)#st
SD1(config-if)#standby 30 pree
SD1(config-if)#standby 30 preempt
SD1(config-if)#int vlan 40
SD1(config-if)#ip addr 192.168.40.2 255.255.255.0
SD1(config-if)#stan
SD1(config-if)#standby 40 pri
SD1(config-if)#standby 40 priority 110
SD1(config-if)#stda
SD1(config-if)#sta
SD1(config-if)#standby 40 pree
SD1(config-if)#standby 40 preempt
SD1(config-if)#int vlan 50
SD1(config-if)#ip addr 192.168.50.2 255.255.255.0
SD1(config-if)#sta
SD1(config-if)#standby 50 ip 192.168.50.1
SD1(config-if)#sta
SD1(config-if)#standby 50 pr
SD1(config-if)#standby 50 pri110
^
% Invalid input detected at '^' marker.

SD1(config-if)#standby 50 pr
SD1(config-if)#standby 50 pri
SD1(config-if)#standby 50 priority 110
SD1(config-if)#stand
```

Cette capture d'écran nous permet de constater que l'adresse IP de nos interfaces physiques sont bien en .2

Vlan10	192.168.10.2	YES	NVRAM	up	up
Vlan20	192.168.20.2	YES	NVRAM	up	up
Vlan30	192.168.30.2	YES	NVRAM	up	up
Vlan40	192.168.40.2	YES	NVRAM	up	up
Vlan50	192.168.50.2	YES	NVRAM	up	up
Vlan60	192.168.60.2	YES	NVRAM	down	down
Vlan99	192.168.99.2	YES	NVRAM	up	up

Cette capture d'écran nous permet de constater que l'adresse IP de nos interfaces virtuelles sont bien en .1

```
SD1#sh standby br
          P indicates configured to preempt.
          |
Interface  Grp  Pri  P State   Active        Standby        Virtual IP
Vl10       10   100 P Standby 192.168.10.3  local          192.168.10.1
Vl20       20   110 P Active  local         192.168.20.3  192.168.20.1
Vl30       30   110 P Active  local         192.168.30.3  192.168.30.1
Vl40       40   110 P Active  local         192.168.40.3  192.168.40.1
Vl50       50   110 P Active  local         192.168.50.3  192.168.50.1
Vl60       60   110 P Init   unknown       unknown        192.168.60.1
Vl99       99   110 P Active  local         192.168.99.3  192.168.99.1
SD1#
```

Pour SD2 :

```
interface Vlan10
ip address 192.168.10.3 255.255.255.0
standby 10 ip 192.168.10.1
standby 10 preempt
```

```
interface Vlan20
ip address 192.168.20.3 255.255.255.0
standby 20 ip 192.168.20.1
standby 20 preempt
```

```
interface Vlan30
ip address 192.168.30.3 255.255.255.0
standby 30 ip 192.168.30.1
standby 30 preempt
```

```
interface Vlan40
ip address 192.168.40.3 255.255.255.0
standby 40 ip 192.168.40.1
standby 40 preempt
```

```
interface Vlan50
ip address 192.168.50.3 255.255.255.0
```

```
standby 50 ip 192.168.50.1
standby 50 preempt
```

```
interface Vlan60
ip address 192.168.60.3 255.255.255.0
standby 60 ip 192.168.60.1
standby 60 preempt
```

```
interface Vlan99
ip address 192.168.99.3 255.255.255.0
standby 99 ip 192.168.99.1
standby 99 preempt
```

Vlan10	192.168.10.3	YES	NVRAM	up	up
Vlan20	192.168.20.3	YES	NVRAM	up	up
Vlan30	192.168.30.3	YES	NVRAM	up	up
Vlan40	192.168.40.3	YES	NVRAM	up	up
Vlan50	192.168.50.3	YES	NVRAM	up	up
Vlan60	192.168.60.3	YES	NVRAM	up	up
Vlan99	192.168.99.3	YES	NVRAM	up	up

Interface	Grp	Pri	P	State	Active	Standby	Virtual IP
Vl10	10	100	P	Active	local	192.168.10.2	192.168.10.1
Vl20	20	100	P	Standby	192.168.20.2	local	192.168.20.1
Vl30	30	100	P	Standby	192.168.30.2	local	192.168.30.1
Vl40	40	100	P	Standby	192.168.40.2	local	192.168.40.1
Vl50	50	100	P	Standby	192.168.50.2	local	192.168.50.1
Vl60	60	100	P	Active	local	unknown	192.168.60.1
Vl99	99	100	P	Standby	192.168.99.2	local	192.168.99.1

Dû au protocole HSRP, nous avons observé le changement des adresses IP physique sur les ports vlan et mis une virtuelle.

V. ACLs

Les ACLs permettent de restreindre les accès entre différents segments du réseau.

ACL RH (Sur SD1)

```
ip access-list extended ACL_RH
permit tcp 192.168.20.0 0.0.0.255 host 192.168.10.10 eq 443
permit tcp host 192.168.30.30 host 192.168.10.10 eq 443
deny tcp any host 192.168.10.10 eq 443
permit ip any host 192.168.10.10
interface vlan 10
ip access-group ACL_RH in
```

```
SD1#sh access-lists ACL_RH
Extended IP access list ACL_RH
 10 permit tcp 192.168.20.0 0.0.0.255 host 192.168.10.10 eq 443
 20 permit tcp host 192.168.30.30 host 192.168.10.10 eq 443
 30 deny tcp any host 192.168.10.10 eq 443
 40 permit ip any host 192.168.10.10
SD1#
```

ACL serveur de fichier (Sur SD1)

```
ip access-list extended ACL_FTP
permit tcp 192.168.20.0 0.0.0.255 host 192.168.40.40 eq 21
permit tcp 192.168.20.0 0.0.0.255 host 192.168.40.40 eq 20
permit tcp host 192.168.30.30 host 192.168.40.40 eq 21
permit tcp host 192.168.30.30 host 192.168.40.40 eq 20
deny tcp any host 192.168.40.40 eq 21
deny tcp any host 192.168.40.40 eq 20
permit ip any host 192.168.40.40
```

```
SD1#sh access-lists ACL_FTP
Extended IP access list ACL_FTP
 10 permit tcp 192.168.20.0 0.0.0.255 host 192.168.40.40 eq ftp
 20 permit tcp 192.168.20.0 0.0.0.255 host 192.168.40.40 eq ftp-data
 30 permit tcp host 192.168.30.30 host 192.168.40.40 eq ftp
 40 permit tcp host 192.168.30.30 host 192.168.40.40 eq ftp-data
 50 deny tcp any host 192.168.40.40 eq ftp
 60 deny tcp any host 192.168.40.40 eq ftp-data
 70 permit ip any host 192.168.40.40
SD1#
```

Cela empêche les réseaux **10, 20, 30, 40, 50** d'accéder à **192.168.30.30**, sauf si la source est le réseau **192.168.99.0/24**.

```
access-list 104 permit ip 192.168.99.0 0.0.0.255 host 192.168.30.30
access-list 104 deny ip 192.168.10.0 0.0.0.255 host 192.168.30.30
access-list 104 deny ip 192.168.20.0 0.0.0.255 host 192.168.30.30
access-list 104 deny ip 192.168.30.0 0.0.0.255 host 192.168.30.30
access-list 104 deny ip 192.168.40.0 0.0.0.255 host 192.168.30.30
access-list 104 deny ip 192.168.50.0 0.0.0.255 host 192.168.30.30
access-list 104 permit ip any any
```

PRE VPN MPLS :

Pour le VPN MPLS, il fallait que les routeurs bridges PEx connaissent les routes des switch L3 SDx. Le routeur étant un routeur, il ne peut pas faire de “switchport mode access”, on a donc opté pour un pont vlan entre le port du routeur bridge relié au SDx et le port SDx relié au routeur.

Sur les SDx :

```
int vlan 1xx (100,101,102 etc..)
ip addr 192.168.1xx.1 255.255.255.252 (252 car nous faisons uniquement passer les réseaux par cette ip)
```

```
interface GigabitEthernet1/0
switchport access vlan 100
switchport trunk allowed vlan 10,20,30,40,50,60,99,100
switchport trunk encapsulation dot1q
switchport mode access
```

```
router ospf 1
network 192.168.10.0 0.0.0.255 area 0
network 192.168.20.0 0.0.0.255 area 0
network 192.168.30.0 0.0.0.255 area 0
etc jusqu'à 50 ou 60
network 192.168.1xx.0 0.0.0.3 area 0
```

Sur les PEx :

```
int vlan 1xx
ip addr 192.168.1xx.2 255.255.255.252
```

```
router ospf 1
network 192.168.1xx.0 0.0.0.3 area 0
```

Nous executons ceci sur les PE et les SD de chaque site pour obtenir les réseau des switch L3 sur les PE

VPN MPLS

Une fois que les PE connaissent les réseaux des SD, on peut faire marché MPLS :

PE1 :

```
hostname PE1
mpls ldp router-id Loopback0 force
ip vrf site1sd1
rd 100:1
route-target export 100:1
route-target import 100:1
ip vrf site1sd2
rd 100:2
route-target export 100:2
route-target import 100:2
interface Loopback0
ip address 1.1.1.1 255.255.255.255
interface GigabitEthernet0/0
ip vrf forwarding site1sd1
ip address 192.168.99.1 255.255.255.0
mpls ip
no shutdown
interface GigabitEthernet0/1
ip vrf forwarding site1sd2
ip address 192.168.99.2 255.255.255.0
mpls ip
no shutdown
interface GigabitEthernet0/2
ip address 10.1.1.1 255.255.255.252
mpls ip
no shutdown
router bgp 100
bgp router-id 1.1.1.1
bgp log-neighbor-changes
neighbor 10.1.1.4 remote-as 100
neighbor 10.1.1.4 update-source Loopback0
address-family vpnv4
neighbor 10.1.1.4 activate
neighbor 10.1.1.4 send-community both
exit-address-family
```

PE2 :

```
hostname PE2
mpls ldp router-id Loopback0 force
ip vrf site2sd3
rd 100:3
route-target export 100:3
route-target import 100:3
ip vrf site2sd4
rd 100:4
route-target export 100:4
route-target import 100:4
interface Loopback0
ip address 2.2.2.2 255.255.255.255
interface GigabitEthernet0/0
ip vrf forwarding site2sd3
ip address 192.168.99.3 255.255.255.0
mpls ip
no shutdown
interface GigabitEthernet0/1
ip vrf forwarding site2sd4
ip address 192.168.99.4 255.255.255.0
mpls ip
no shutdown
interface GigabitEthernet0/2
ip address 10.1.2.1 255.255.255.252
mpls ip
no shutdown
router bgp 100
bgp router-id 2.2.2.2
bgp log-neighbor-changes
neighbor 10.1.2.4 remote-as 100
neighbor 10.1.2.4 update-source Loopback0
address-family vpnv4
neighbor 10.1.2.4 activate
neighbor 10.1.2.4 send-community both
exit-address-family
```


PE3 :

```
hostname PE3
mpls ldp router-id Loopback0 force
ip vrf site3sd5
rd 100:5
route-target export 100:5
route-target import 100:5
ip vrf site3sd6
rd 100:6
route-target export 100:6
route-target import 100:6
interface Loopback0
ip address 3.3.3.3 255.255.255.255
interface GigabitEthernet0/0
ip vrf forwarding site3sd5
ip address 192.168.99.5 255.255.255.0
mpls ip
no shutdown
interface GigabitEthernet0/1
ip vrf forwarding site3sd6
ip address 192.168.99.6 255.255.255.0
mpls ip
no shutdown
interface GigabitEthernet0/2
ip address 10.1.3.1 255.255.255.252
mpls ip
no shutdown
router bgp 100
bgp router-id 3.3.3.3
bgp log-neighbor-changes
neighbor 10.1.3.4 remote-as 100
neighbor 10.1.3.4 update-source Loopback0
address-family vpnv4
neighbor 10.1.3.4 activate
neighbor 10.1.3.4 send-community both
exit-address-family
```

PE :

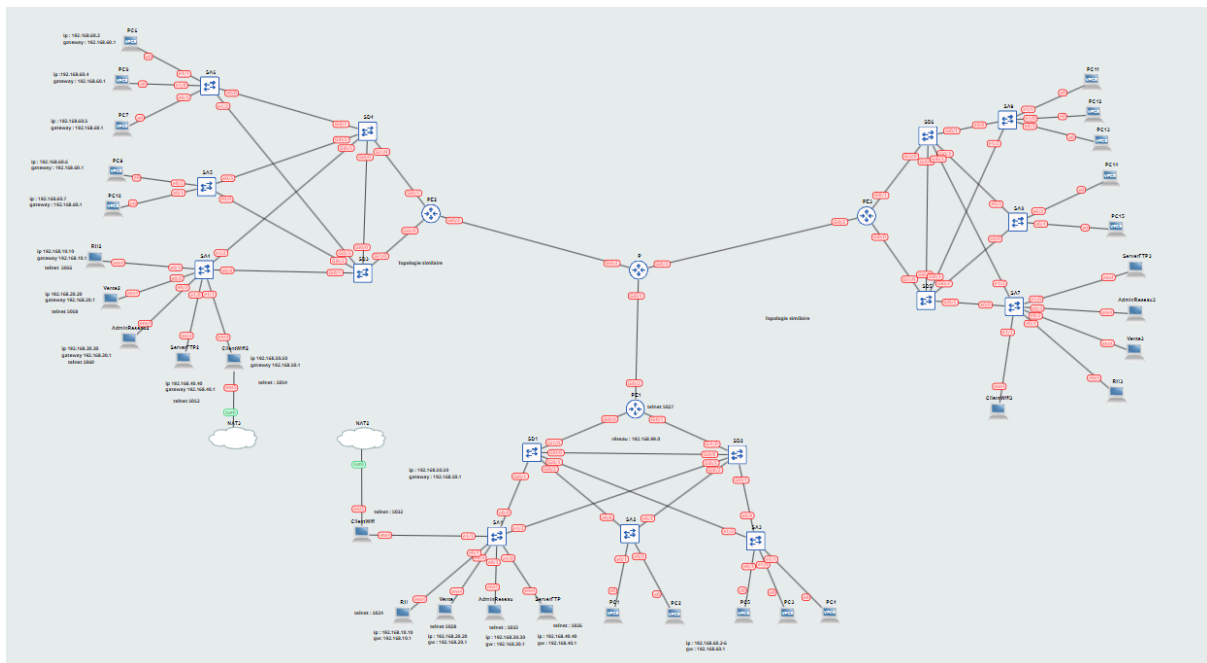
```
hostname PE
mpls ldp router-id Loopback0 force
ip vrf site1sd1
```

```
rd 100:1
route-target export 100:1
route-target import 100:1
ip vrf site1sd2
rd 100:2
route-target export 100:2
route-target import 100:2
ip vrf site2sd3
rd 100:3
route-target export 100:3
route-target import 100:3
ip vrf site2sd4
rd 100:4
route-target export 100:4
route-target import 100:4
ip vrf site3sd5
rd 100:5
route-target export 100:5
route-target import 100:5
ip vrf site3sd6
rd 100:6
route-target export 100:6
route-target import 100:6
interface Loopback0
ip address 4.4.4.4 255.255.255.255
interface GigabitEthernet0/0
ip address 10.1.1.4 255.255.255.252
mpls ip
no shutdown
interface GigabitEthernet0/1
ip address 10.1.2.4 255.255.255.252
mpls ip
no shutdown
interface GigabitEthernet0/2
ip address 10.1.3.4 255.255.255.252
mpls ip
no shutdown
router ospf 1
router-id 4.4.4.4
network 10.1.1.0 0.0.0.3 area 0
network 10.1.2.0 0.0.0.3 area 0
network 10.1.3.0 0.0.0.3 area 0
router bgp 100
bgp router-id 4.4.4.4
bgp log-neighbor-changes
neighbor 10.1.1.1 remote-as 100
```

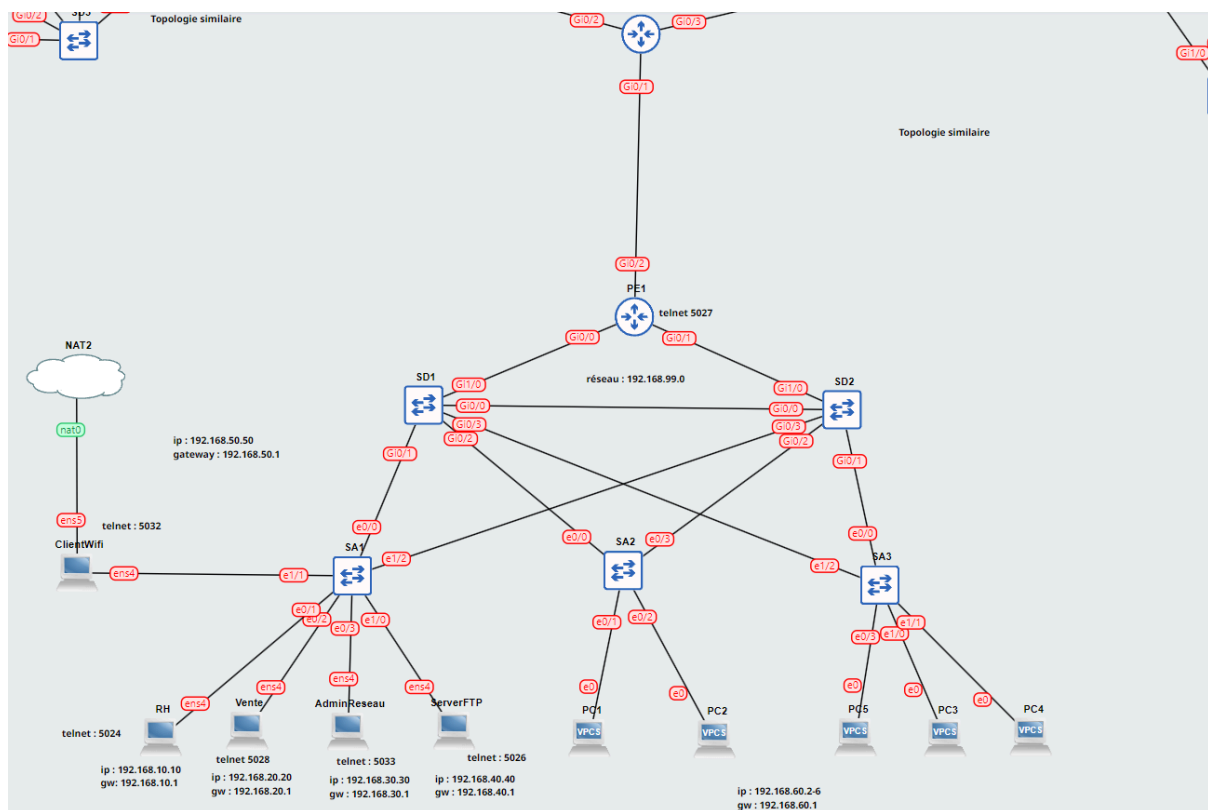
```
neighbor 10.1.1.1 update-source Loopback0
neighbor 10.1.2.1 remote-as 100
neighbor 10.1.2.1 update-source Loopback0
neighbor 10.1.3.1 remote-as 100
neighbor 10.1.3.1 update-source Loopback0
address-family vpnv4
neighbor 10.1.1.1 activate
neighbor 10.1.1.1 send-community both
neighbor 10.1.2.1 activate
neighbor 10.1.2.1 send-community both
neighbor 10.1.3.1 activate
neighbor 10.1.3.1 send-community both
exit-address-family
```

Schéma final :

Ici vous pouvez retrouver notre topologie finale de notre réseau. Chaque site possède la même configuration sauf pour les pc vlan 6 ou les ips ne sont pas tout à fait les mêmes.



Ici nous nous focalisons sur 1 seul site en particulier, le site qui nous a apporté des difficultés notamment pour importer les pc debian. (Voir ci dessous dans l'[ANNEXE](#))



Conclusion :

Dans le cadre de ce projet, nous avons mis en place une infrastructure MPLS VPN permettant l'interconnexion sécurisée de plusieurs sites via un PE central. En utilisant MP-BGP pour l'échange des routes VPNv4 et MPLS LDP pour assurer l'acheminement des paquets, nous avons pu garantir une segmentation efficace du réseau à travers des VRFs spécifiques à chaque site.

L'architecture retenue avec un PE centralisé offre plusieurs avantages :

- Simplicité de gestion : La centralisation des échanges via PE permet un meilleur contrôle du routage et des flux inter-sites.
- Scalabilité : De nouveaux sites peuvent être facilement intégrés en ajoutant des VRFs et des sessions BGP sur le PE.
- Sécurité et isolation : L'utilisation des VRFs empêche toute communication non souhaitée entre les clients, tout en garantissant un acheminement efficace des données.

Enfin, les tests effectués, tels que la vérification des sessions MPLS LDP, BGP VPNv4, et des routes VRF, ont confirmé le bon fonctionnement de l'architecture. Cette solution pourrait être renforcée en ajoutant des mécanismes de redondance (VRRP, BGP multi-path) ou des mesures de durcissement pour améliorer la résilience et la sécurité globale du réseau.

Ce projet nous a permis de mieux comprendre la mise en œuvre d'une infrastructure MPLS VPN opérateur, tout en appliquant des concepts avancés de routage, virtualisation et interconnexion de sites distants dans un cadre sécurisé et optimisé.

ANNEXE :

Configuration GNS3 image debian et installation FTP debian

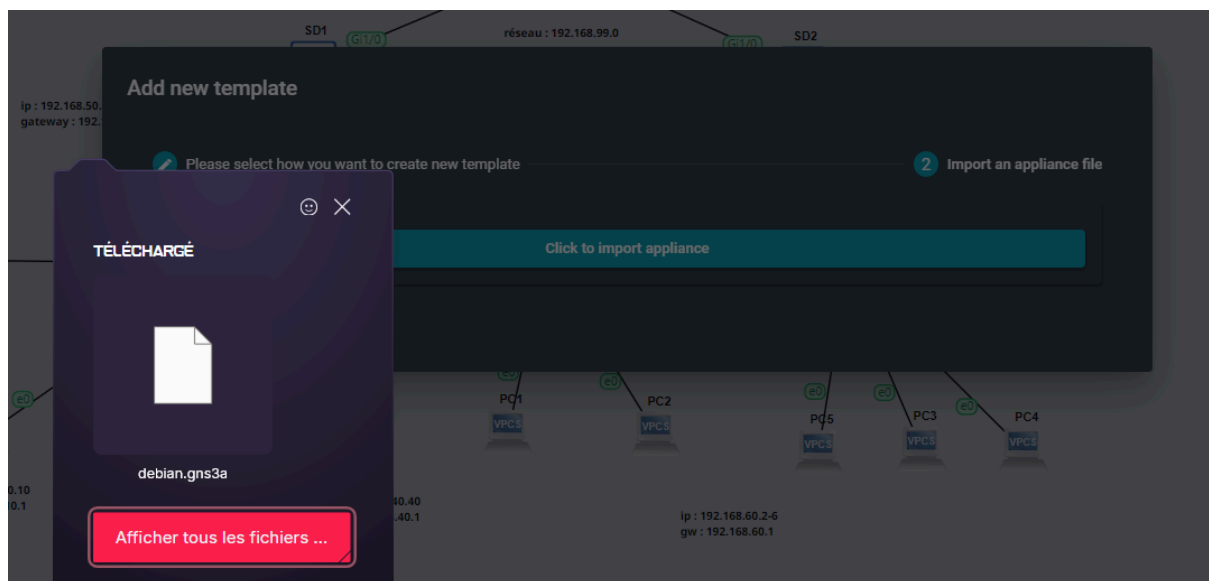
Pour mettre un pc debian sur gns3 j'ai du télécharger l'image debian gns3a sur ce site :

<https://gns3.com/marketplace/appliances/debian-2>

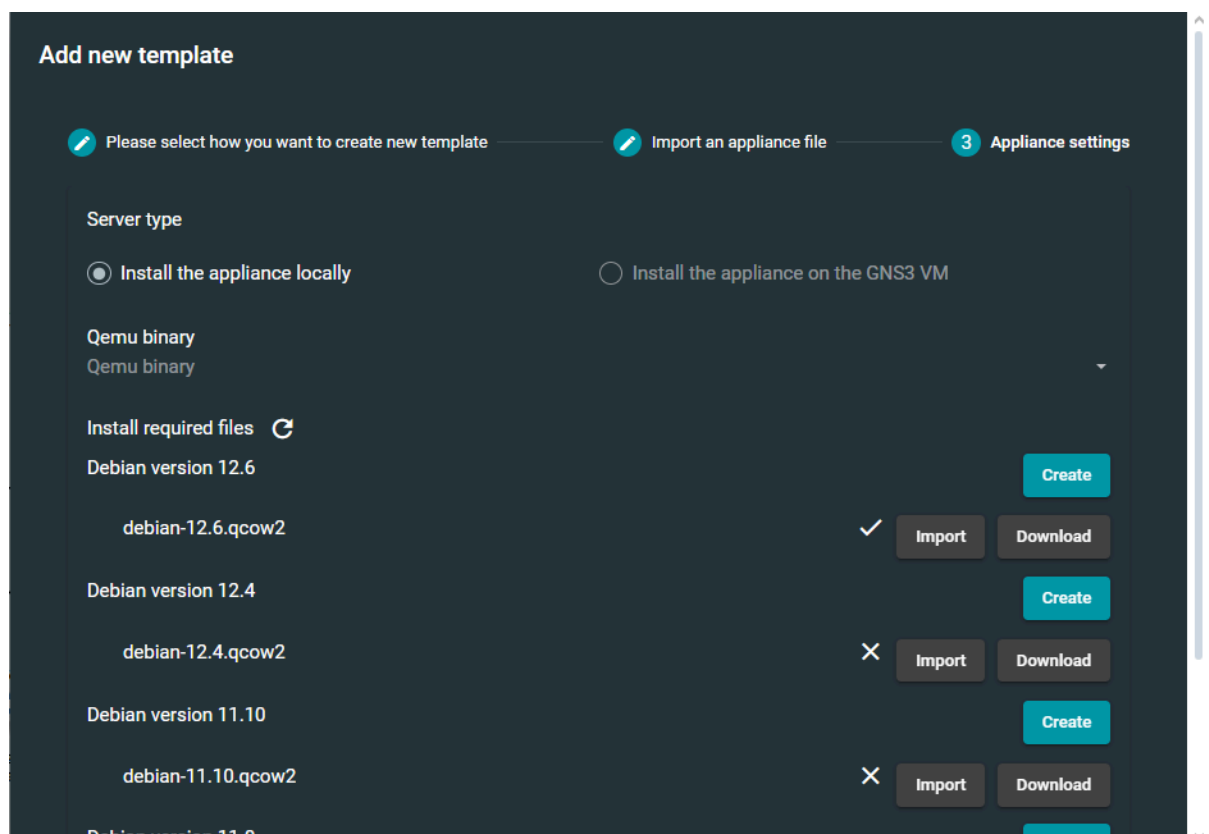
Ensuite de télécharger le fichier qcow2 de debian sur ce site :

<https://sourceforge.net/projects/gns-3/files/Qemu%20Appliances/debian-12.6.qcow2/download>

Une fois téléchargé, il faut créer un QEMU debian, pour ça on fait new template on importe un nouveau fichier et on met la debian gns3a



Une fois fait on arrive sur cette page ou on demande d'importer l'image qcow2



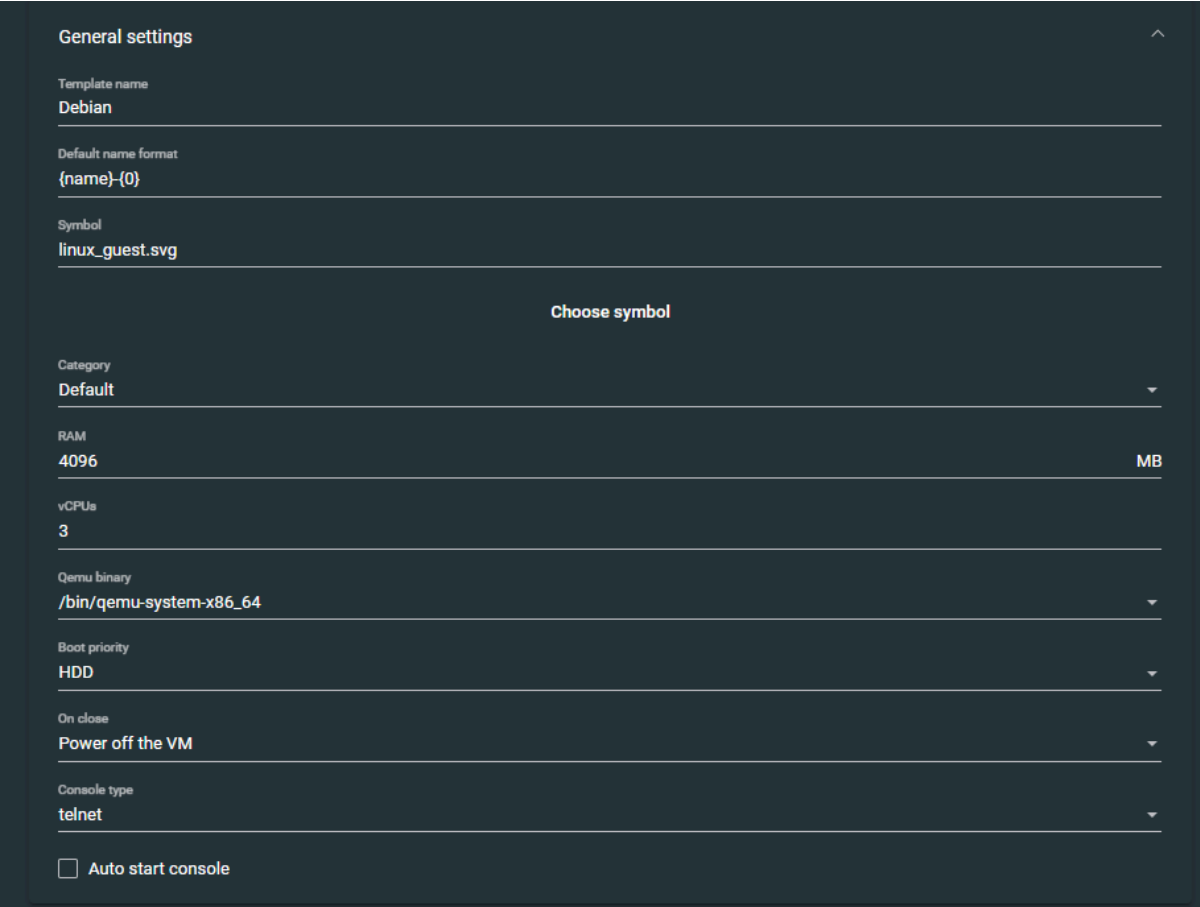
On fait “import” et on valide. Une fois fini on aura le coche sur import et on fera create, cela nous créer une debian.

La debian était installé marche pas encore, il faut configurer son QEMU pour s’assurer que la machine ne plante pas au démarrage, il faut la paramétrer.

Donc dans le menu projet de votre gns3 web on clique sur préférence, QEMU, tout en bas on trouve la debian



On clique dessus et on va dans général:



The screenshot shows the 'General settings' window for a virtual machine template. The settings are as follows:

- Template name: Debian
- Default name format: {name}-{0}
- Symbol: linux_guest.svg
- Category: Default
- RAM: 4096 MB
- vCPUs: 3
- Qemu binary: /bin/qemu-system-x86_64
- Boot priority: HDD
- On close: Power off the VM
- Console type: telnet
- ☐ Auto start console

Il faut augmenter la mémoire RAM entre 2049 Mo et 4096 Mo pour garantir des performances optimales, ainsi que d'allouer 2 à 3 cœurs de CPU afin d'améliorer la réactivité du système.

La Debian est désormais prête à l'emploi : elle peut être lancée et configurée.

Attribution d'une adresse IP fixe sur Debian :

```
vim /etc/network/interfaces
```

```
# DHCP config for ens4
```

```
auto ens5
```

```
iface ens5 inet dhcp
```

```
# Static config for ens4
```

```
auto ens4
```

```
iface ens4 inet static
```

```
address 192.168.40.40
```

```
netmask 255.255.255.0
```

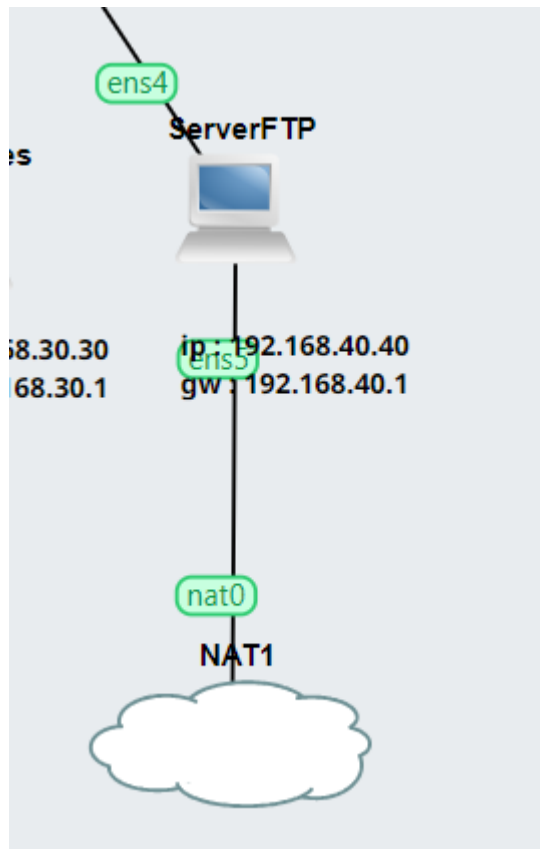
```
gateway 192.168.40.1
dns-nameservers 192.168.1.1
```

```
valid_lft forever preferred_lft forever
2: ens4: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group
p default qlen 1000
    link/ether 0c:e8:88:71:00:00 brd ff:ff:ff:ff:ff:ff
    altname enp0s4
    inet 192.168.40.24/24 brd 192.168.40.255 scope global ens4
        valid_lft forever preferred_lft forever
    inet6 fe80::ee8:88ff:fe71:0/64 scope link
        valid_lft forever preferred_lft forever
3: ens5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group
p default qlen 1000
    link/ether 0c:e8:88:71:00:01 brd ff:ff:ff:ff:ff:ff
    altname enp0s5
    inet 192.168.122.238/24 brd 192.168.122.255 scope global dynamic ens5
        valid_lft 3510sec preferred_lft 3510sec
    inet6 fe80::ee8:88ff:fe71:1/64 scope link
        valid_lft forever preferred_lft forever
debian@debian:~$
```

L'interface ens5 est connectée à ma carte réseau GNS3 VM qui elle-même est connectée à ma machine hôte qui est reliée à la boxe pour l'accès à internet, pour télécharger les paquets ftp, vsftpd et autre.

Elle est évidemment connectée à un NAT dans la topologie.

```
valid_lft forever preferred_lft forever
4: virbr0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500
    link/ether 52:54:00:6f:81:c9 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1/24 brd 192.168.122.255 scope global
        valid_lft forever preferred_lft forever
```



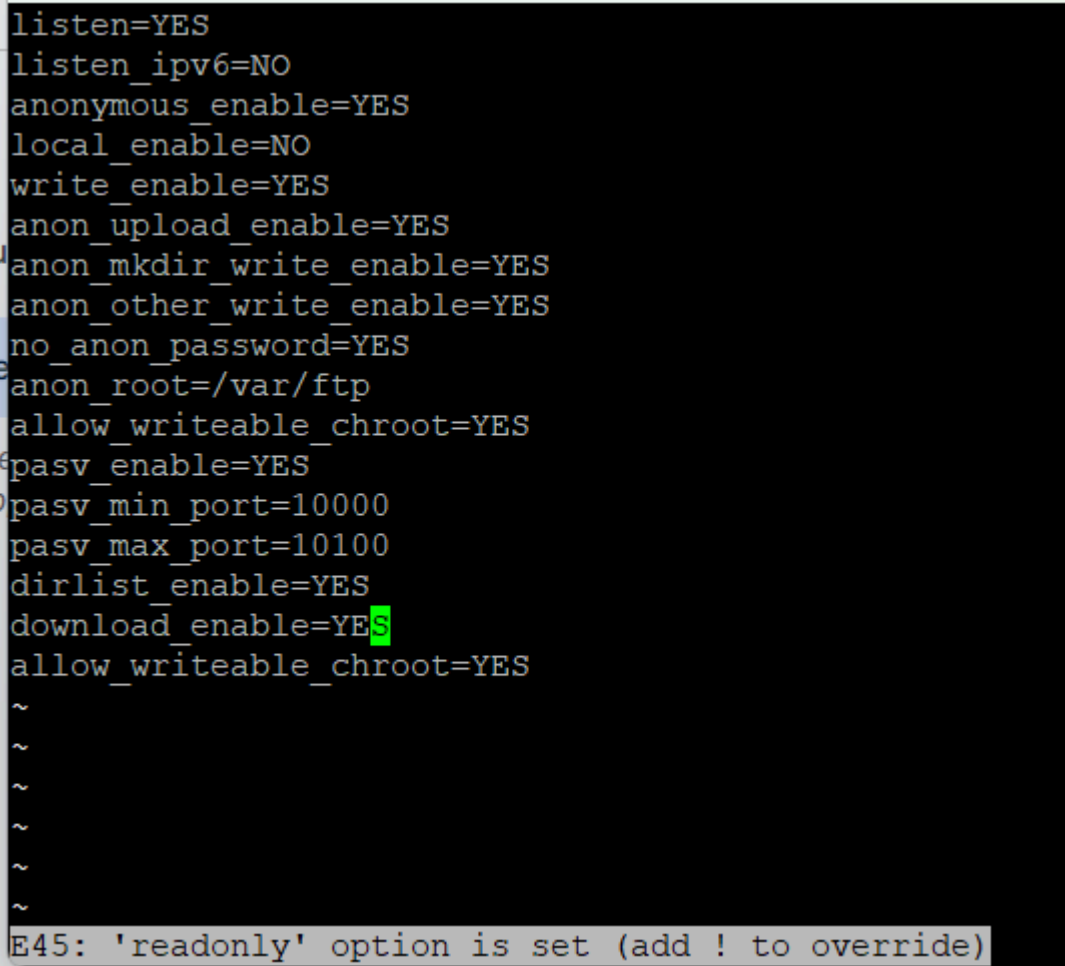
Maintenant qu'on a accès à internet on peut commencer le serveur de partage de fichier (ftp)

Installation serveur FTP sur le poste serverfichier ip 192.168.40.40

```
apt upgrade
apt update
apt install vsftpd ftp
```

```
vim /etc/vsftpd.conf
debian@debian:~$ cat /etc/vsftpd.conf
listen=YES
listen_ipv6=NO
anonymous_enable=YES
local_enable=NO
write_enable=YES
anon_upload_enable=YES
anon_mkdir_write_enable=YES
anon_other_write_enable=YES
no_anon_password=YES
anon_root=/var/ftp
allow_writeable_chroot=YES
pasv_enable=YES
pasv_min_port=10000
```

```
pasv_max_port=10100
dirlist_enable=YES
download_enable=YES
allow_writeable_chroot=YES
debian@debian:~$
```



```
listen=YES
listen_ipv6=NO
anonymous_enable=YES
local_enable=NO
write_enable=YES
anon_upload_enable=YES
anon_mkdir_write_enable=YES
anon_other_write_enable=YES
no_anon_password=YES
anon_root=/var/ftp
allow_writeable_chroot=YES
pasv_enable=YES
pasv_min_port=10000
pasv_max_port=10100
dirlist_enable=YES
download_enable=YES
allow_writeable_chroot=YES
~
~
~
~
~
~
E45: 'readonly' option is set (add ! to override)
```

```
sudo systemctl restart vsftpd
```

```
mkdir -p /var/ftp
chmod 755 -R /var/
chmod a-w /var/ftp
```

```
ftp localhost
user : anonymous
password : null
```

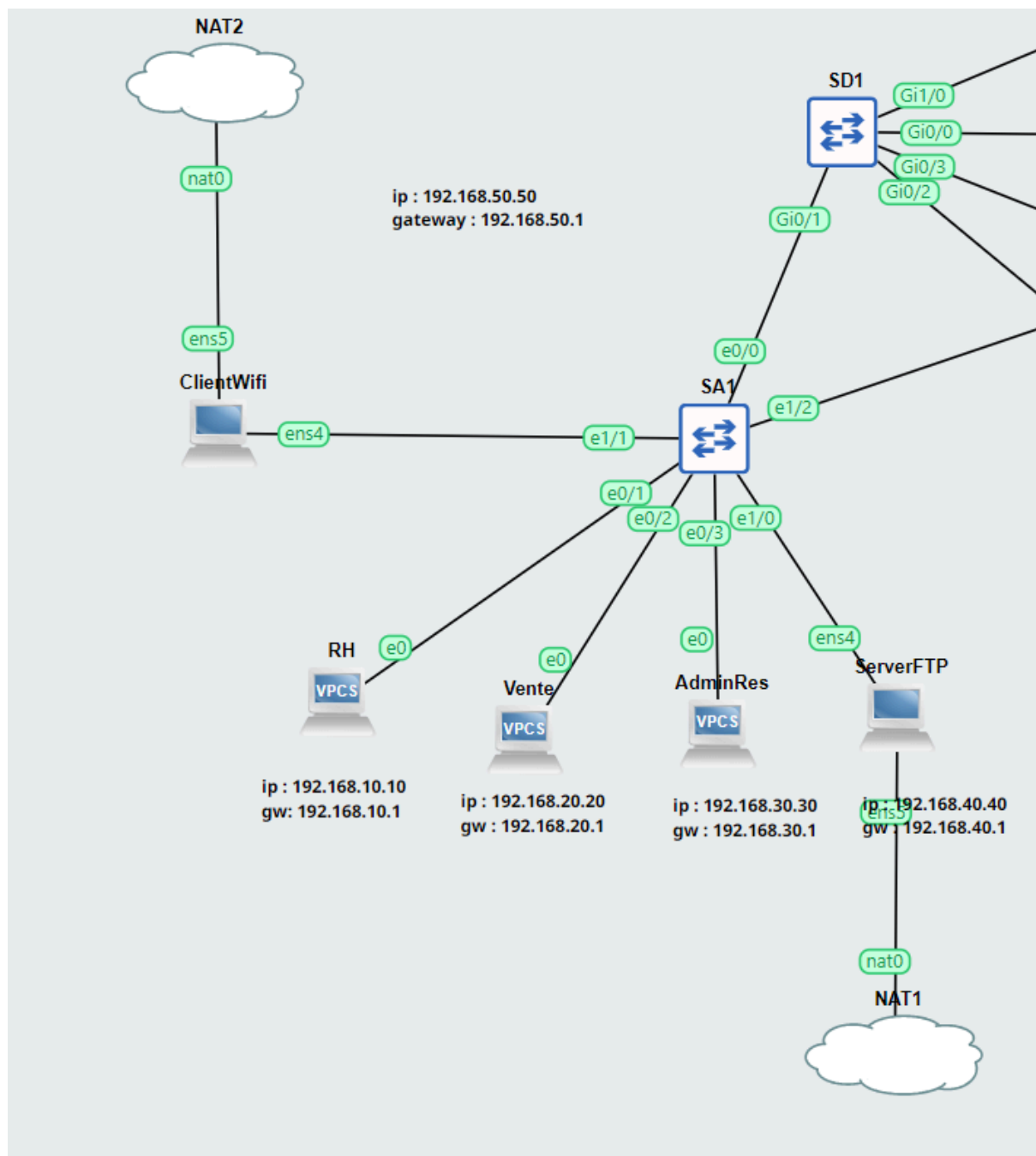
```
debian@debian:~$ ftp localhost
Trying [::1]:21 ...
ftp: Can't connect to `[::1]:21': Connection refused
Trying 127.0.0.1:21 ...
Connected to localhost.
220 (vsFTPD 3.0.3)
Name (localhost:debian): anonymous
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls -al
229 Entering Extended Passive Mode (||||10075|)
150 Here comes the directory listing.
dr-xr-xr-x    2 0      0          4096 Jan 09 12:31 .
dr-xr-xr-x    2 0      0          4096 Jan 09 12:31 ..
-rw-r--r--    1 0      0           0 Jan 09 12:31 coucou.txt
226 Directory send OK.
ftp> bye
221 Goodbye.
debian@debian:~$ cd /var/ftp
debian@debian:/var/ftp$ ls
coucou.txt
debian@debian:/var/ftp$ █
```

Via cette capture d'écran nous observons que notre serveur ftp est totalement opérationnel.

Pour permettre la communication entre un autre PC et cette machine sur le même réseau, nous devons lui attribuer une adresse IP fixe. Nous prendrons l'adresse **192.168.50.50**, en suivant les mêmes étapes que celles utilisées pour la configuration d'un serveur FTP.

1. **Attribuer une adresse IP statique** à la machine cible afin d'assurer une connectivité stable.
2. **Configurer les paramètres réseau** pour qu'ils correspondent au sous-réseau utilisé par les autres machines.
3. **Vérifier la connectivité** en testant la communication entre les appareils à l'aide de commandes telles que **ping** ou **arp**.
4. **Configurer les règles de pare-feu**, si nécessaire, afin d'autoriser le trafic entre les machines sur le réseau local.

En appliquant cette configuration, la machine sera joignable de manière fiable sous l'adresse **192.168.50.50**, facilitant ainsi l'échange de données et la communication avec les autres équipements du réseau.



```
vim /etc/network/interfaces
```

```
auto ens5
```

```
iface ens5 inet dhcp
```

```
# Static config for ens4
```

```
auto ens4
```

```
iface ens4 inet static
```

```
    address 192.168.50.50
```

```
    netmask 255.255.255.0
```

```
    gateway 192.168.50.1
```

```
debian@debian:~$ cat /etc/network/interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# DHCP config for ens4
auto ens5
iface ens5 inet dhcp

# Static config for ens4
auto ens4
iface ens4 inet static
    address 192.168.50.50
    netmask 255.255.255.0
    gateway 192.168.50.1
debian@debian:~$
```

Voici la configuration de la machine client wifi qui possède une ip en 192.168.50.50

```
valid_lft forever preferred_lft forever
2: ens4: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_c
p default qlen 1000
    link/ether 0c:f6:b5:36:00:00 brd ff:ff:ff:ff:ff:ff
    altname enp0s4
    inet 192.168.50.50/24 brd 192.168.50.255 scope global ens4
        valid_lft forever preferred_lft forever
    inet6 fe80::ef6:b5ff:fe36:0/64 scope link
        valid_lft forever preferred_lft forever
3: ens5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_c
p default qlen 1000
    link/ether 0c:f6:b5:36:00:01 brd ff:ff:ff:ff:ff:ff
    altname enp0s5
    inet 192.168.122.230/24 brd 192.168.122.255 scope global d
        valid_lft 2900sec preferred_lft 2900sec
    inet6 fe80::ef6:b5ff:fe36:1/64 scope link
        valid_lft forever preferred_lft forever
debian@debian:~$
```

Le PC (192.168.50.50) peut **pinguer son routeur**, mais ne peut pas communiquer avec les autres réseaux car **il ne possède pas de routes**. Bien que le **switch L3 connaisse les routes**, le PC ne les a pas définies localement.

Il faut ajouter une **route par défaut** pour permettre la communication avec les autres réseaux ip route add default via 192.168.50.1 dev ens4

Cette configuration permet au PC d'envoyer ses paquets vers le routeur, qui se charge du routage vers les autres réseaux.

```
debian@debian:~$ ping 192.168.50.1
PING 192.168.50.1 (192.168.50.1) 56(84) bytes of data.
64 bytes from 192.168.50.1: icmp_seq=1 ttl=255 time=13.7 ms
64 bytes from 192.168.50.1: icmp_seq=2 ttl=255 time=4.80 ms
64 bytes from 192.168.50.1: icmp_seq=3 ttl=255 time=5.65 ms
^C
--- 192.168.50.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2007ms
rtt min/avg/max/mdev = 4.799/8.034/13.652/3.987 ms
debian@debian:~$
```



```
SD1#ping 192.168.50.50
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.50.50, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/4/8 ms
SD1#
```

Il faut donc créer des routes par défaut sur les pc pour qu'il puissent communiquer :
pc client wifi

```
root@debian:/home/debian# ip route
192.168.50.0/24 dev ens4 proto kernel scope link src 192.168.50.50
192.168.122.0/24 dev ens5 proto kernel scope link src 192.168.122.230
root@debian:/home/debian#
```

ip route add default via 192.168.50.1 dev ens4

```
root@debian:/home/debian# ip route
default via 192.168.50.1 dev ens4
192.168.50.0/24 dev ens4 proto kernel scope link src 192.168.50.50
192.168.122.0/24 dev ens5 proto kernel scope link src 192.168.122.230
root@debian:/home/debian#
```

On peut ping les routeurs des autres réseaux avec cette route. Mais en revanche on a plus internet car les paquets sont redirigés vers le routeur localement (SD1) pour remettre il faudrait supprimer cette route par défaut et mettre celle en 192.168.122.1 mais dans ce cas nous pouvons plus sortir du réseaux en 192.168.50.0.

```
PING 192.168.40.1 (192.168.40.1) 56(84) bytes of data.
64 bytes from 192.168.40.1: icmp_seq=1 ttl=255 time=4.56 ms
64 bytes from 192.168.40.1: icmp_seq=2 ttl=255 time=3.93 ms
64 bytes from 192.168.40.1: icmp_seq=3 ttl=255 time=5.33 ms
^C
--- 192.168.40.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2010ms
rtt min/avg/max/mdev = 3.933/4.607/5.332/0.572 ms
root@debian:/home/debian# ping 192.168.10.1
PING 192.168.10.1 (192.168.10.1) 56(84) bytes of data.
64 bytes from 192.168.10.1: icmp_seq=2 ttl=255 time=12.2 ms
64 bytes from 192.168.10.1: icmp_seq=3 ttl=255 time=8.94 ms
64 bytes from 192.168.10.1: icmp_seq=4 ttl=255 time=6.12 ms
^C
--- 192.168.10.1 ping statistics ---
4 packets transmitted, 3 received, 25% packet loss, time 3545ms
rtt min/avg/max/mdev = 6.123/9.070/12.150/2.462 ms
root@debian:/home/debian# ping 192.168.40.40
PING 192.168.40.40 (192.168.40.40) 56(84) bytes of data.
^C
--- 192.168.40.40 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2031ms
```

Pour le teste du serveur ftp nous mettra la route par défaut du gateway.

Il est donc nécessaire d'appliquer la même configuration sur le serveur FTP afin de garantir son accessibilité depuis l'extérieur.

```
default via 192.168.122.1 dev ens5
192.168.40.0/24 dev ens4 proto kernel scope link src 192.168.40.40
192.168.122.0/24 dev ens5 proto kernel scope link src 192.168.122.238
debian@debian:/var/ftp$
```

```
root@debian:/var/ftp# ip route add default via 192.168.40.1 dev ens4
root@debian:/var/ftp# ip route
default via 192.168.40.1 dev ens4
192.168.40.0/24 dev ens4 proto kernel scope link src 192.168.40.40
192.168.122.0/24 dev ens5 proto kernel scope link src 192.168.122.238
root@debian:/var/ftp# ping 192.168.50.50
PING 192.168.50.50 (192.168.50.50) 56(84) bytes of data.
64 bytes from 192.168.50.50: icmp_seq=1 ttl=63 time=6.57 ms
64 bytes from 192.168.50.50: icmp_seq=2 ttl=63 time=5.89 ms
64 bytes from 192.168.50.50: icmp_seq=3 ttl=63 time=6.30 ms
^C
--- 192.168.50.50 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2007ms
rtt min/avg/max/mdev = 5.894/6.257/6.573/0.279 ms
root@debian:/var/ftp# ping 192.168.10.10
PING 192.168.10.10 (192.168.10.10) 56(84) bytes of data.
64 bytes from 192.168.10.10: icmp_seq=2 ttl=63 time=18.9 ms
64 bytes from 192.168.10.10: icmp_seq=3 ttl=63 time=10.1 ms
64 bytes from 192.168.10.10: icmp_seq=4 ttl=63 time=7.15 ms
^C
--- 192.168.10.10 ping statistics ---
4 packets transmitted, 3 received, 25% packet loss, time 3033ms
rtt min/avg/max/mdev = 7.151/12.052/18.941/5.014 ms
root@debian:/var/ftp#
```

Avec la route par défaut configurée, nous pouvons désormais pinguer les machines des autres réseaux, et celles-ci peuvent également atteindre le serveur FTP. Nous procédons maintenant à un test depuis la machine client Wi-Fi.

```
root@debian:/home/debian# ip -c a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: ens4: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 0c:f6:b5:36:00:00 brd ff:ff:ff:ff:ff:ff
    altname enp0s4
    inet 192.168.50.50/24 brd 192.168.50.255 scope global ens4
        valid_lft forever preferred_lft forever
    inet6 fe80::ef6:b5ff:fe36:0/64 scope link
        valid_lft forever preferred_lft forever
3: ens5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 0c:f6:b5:36:00:01 brd ff:ff:ff:ff:ff:ff
    altname enp0s5
    inet 192.168.122.230/24 brd 192.168.122.255 scope global dynamic ens5
        valid_lft 2285sec preferred_lft 2285sec
    inet6 fe80::ef6:b5ff:fe36:1/64 scope link
        valid_lft forever preferred_lft forever
root@debian:/home/debian# ftp 192.168.40.40
Connected to 192.168.40.40.
220 (vsFTPD 3.0.3)
Name (192.168.40.40:debian): anonymous
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls -al
229 Entering Extended Passive Mode (|||10060|)
150 Here comes the directory listing.
dr-xr-xr-x   2 0       0          4096 Jan 09 12:31 .
dr-xr-xr-x   2 0       0          4096 Jan 09 12:31 ..
-rw-r--r--   1 0       0           0 Jan 09 12:31 coucou.txt
226 Directory send OK.
ftp> █
```

La configuration est opérationnelle, nos PC peuvent désormais interagir avec le serveur FTP sans restriction.

De plus, la communication avec les VPCs du réseau 192.168.60.0 (de 2 à 6) est pleinement fonctionnelle.

```
root@debian:/home/debian# ping 192.168.60.1
PING 192.168.60.1 (192.168.60.1) 56(84) bytes of data.
64 bytes from 192.168.60.1: icmp_seq=1 ttl=255 time=16.8 ms
^C^[A
--- 192.168.60.1 ping statistics ---
2 packets transmitted, 1 received, 50% packet loss, time 1005ms
rtt min/avg/max/mdev = 16.753/16.753/16.753/0.000 ms
root@debian:/home/debian# ping 192.168.60.6
PING 192.168.60.6 (192.168.60.6) 56(84) bytes of data.
64 bytes from 192.168.60.6: icmp_seq=2 ttl=63 time=28.9 ms
64 bytes from 192.168.60.6: icmp_seq=3 ttl=63 time=22.5 ms
64 bytes from 192.168.60.6: icmp_seq=4 ttl=63 time=10.8 ms
^C
--- 192.168.60.6 ping statistics ---
4 packets transmitted, 3 received, 25% packet loss, time 3987ms
rtt min/avg/max/mdev = 10.842/20.770/28.923/7.487 ms
root@debian:/home/debian# █
```