

Compte rendu de travaux pratiques

TP1

Déployer un serveur DHCP et un serveur DNS

Nom : PACHOT

Prénom : Dorian

Classe : BTS SIO 2e année

Module : Linux avancé

Date de la séance : 7 septembre 2026

1. Objectif

Dans ce TP, je monte le réseau du domaine bts-sio.lan sur deux machines virtuelles Debian 12. Le serveur srv-linux distribue les adresses avec isc-dhcp-server. Il traduit aussi les noms en adresses avec bind9, dans les deux sens. La deuxième machine, cli-linux, est un clone du serveur. Elle sert de client de test : elle reçoit son adresse par DHCP et doit joindre les autres machines par leur nom.

La séance se termine par deux choses. D'abord je sécurise l'accès SSH au serveur. Ensuite un exercice de dépannage : trois fautes sont cachées dans ma configuration et je dois les retrouver seul.

Ces deux machines servent de base aux trois journées suivantes. Elles doivent rester en état à la fin du TP.

2. Environnement de travail

Machine	Système	Adresse IP	Rôle
poste-hote	Windows 11, VMware Workstation Pro 17.6.1	192.168.100.1 / 24 sur VMnet2	Poste étudiant, client de test
srv-linux	Debian 12 « bookworm », 2 vCPU, 2 Go de RAM, disque de 20 Go	ens33 : DHCP en NAT ; ens37 : 192.168.100.10 / 24 statique	Serveur DHCP, DNS et SSH
cli-linux	Debian 12, clone de srv-linux	ens37 : 192.168.100.50 par réservation DHCP	Client Linux de test

Le réseau pédagogique est VMnet2, en host-only sur 192.168.100.0/24. Le DHCP de VMware y est désactivé. C'est voulu : le seul serveur DHCP du réseau doit être le mien. La carte NAT de srv-linux sert juste à télécharger les paquets. Sur cli-linux, cette carte NAT n'est pas activée. Comme ça, le client ne reçoit sa passerelle et son DNS que de mon serveur.

La machine du TP0, sous Ubuntu Server 24.04, n'est pas réutilisée. Elle a déjà l'adresse 192.168.100.10 sur VMnet2. Je l'éteins donc avant de démarrer la nouvelle Debian.

3. Déroulé

Partie A. Préparer les deux VM Debian 12

Étape 1. Créer la VM et lancer l'installateur Debian

Avant de créer la machine, j'ai vérifié VMnet2 dans le Virtual Network Editor de VMware. Il est bien en host-only, sur 192.168.100.0/24. La case « Use local DHCP service to distribute IP addresses » est décochée.

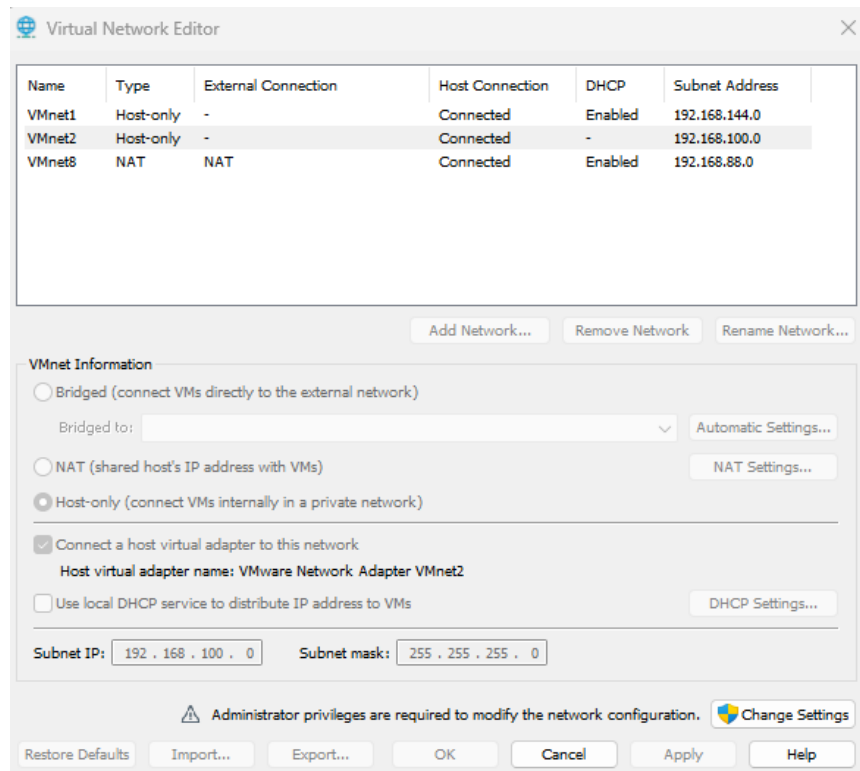


Figure 1 : Virtual Network Editor, VMnet2 en host-only, service DHCP de VMware désactivé

La colonne DHCP du tableau le montre bien : VMnet1 et VMnet8 affichent Enabled, VMnet2 affiche un tiret. Le service DHCP de VMware est actif sur les deux autres réseaux, et désactivé sur le mien. La case « Use local DHCP service to distribute IP addresses » est décochée, et le sous-réseau est bien 192.168.100.0 / 255.255.255.0.

J'ai ensuite créé la machine virtuelle en mode Custom (advanced), avec ces paramètres :

Paramètre	Valeur
Nom	srv-linux
Système invité déclaré	Linux, Debian 12.x 64-bit
Processeurs	1 processeur × 2 cœurs (2 vCPU)
Mémoire	2048 Mo
Disque	20 Go, SCSI, stocké en un seul fichier
Network Adapter	NAT, sortie vers Internet pour apt
Network Adapter 2	Custom : VMnet2, réseau pédagogique
Lecteur CD/DVD	debian-12.15.0-amd64-netinst.iso, connecté au démarrage
Firmware	BIOS
Périphériques retirés	imprimante, carte son, contrôleur USB

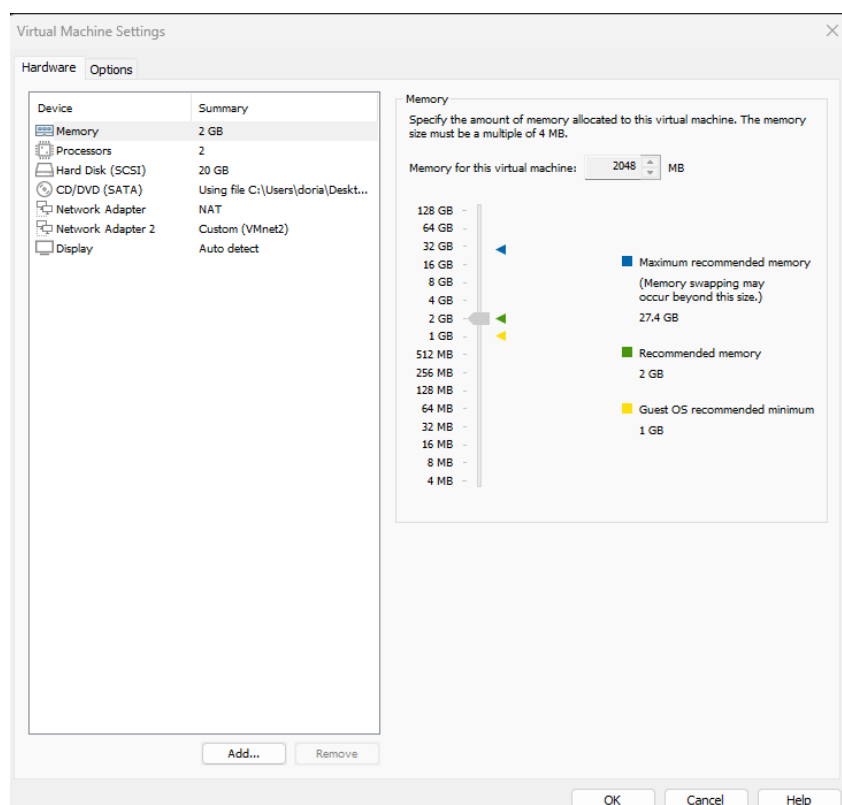


Figure 2 : paramètres de la VM srv-linux, deux cartes réseau et aucun périphérique inutile

La fenêtre confirme la configuration : 2 Go de mémoire, 2 processeurs, un disque SCSI de 20 Go, l'ISO Debian monté sur le lecteur CD/DVD, la carte NAT en premier et la carte Custom (VMnet2) en second. Aucune imprimante, aucune carte son, aucun contrôleur USB n'apparaît dans la liste.

Avant d'allumer la VM, j'ai éteint la machine Ubuntu du TP0 avec `sudo poweroff`. Elle a déjà 192.168.100.10 sur VMnet2. Deux machines avec la même adresse sur le même réseau ne peuvent pas fonctionner.

Installation de Debian 12 en mode texte

Au menu de démarrage j'ai choisi Install, en mode texte, et pas Graphical install.

Dans l'énoncé	Sur ma machine	Rôle
ens33	ens32	Carte NAT, accès aux dépôts APT
ens37	ens33	Carte VMnet2, réseau pédagogique

Voici ensuite les valeurs saisies :

Écran	Valeur saisie
Langue / pays / clavier	Français / France / Français
Nom de machine	srv-linux
Nom de domaine	bts-sio.lan
Mot de passe root	saisi (non recopié ici)
Nom complet et identifiant du compte	sio
Mot de passe du compte	saisi (non recopié ici)
Partitionnement	Assisté, disque entier, tout dans une seule partition
Analyser un autre CD / miroir	Non / deb.debian.org
Participer à l'étude statistique	Non
Sélection des logiciels	serveur SSH et utilitaires usuels du système uniquement
GRUB	installé sur /dev/sda

Premier démarrage et identification du système

Après le redémarrage, la console affiche « Debian GNU/Linux 12 srv-linux tty1 ». Je me connecte avec le compte sio.

hostnamectl

```
sio@srv-linux:~$ hostnamectl
Static hostname: srv-linux
Virtualization: vmware
Operating System: Debian GNU/Linux 12 (bookworm)
Kernel: Linux 6.1.0-52-amd64
Architecture: x86-64
```

Le nom de machine est bien srv-linux, celui que j'ai saisi pendant l'installation. Le système est Debian GNU/Linux 12 « bookworm ». Le noyau est le 6.1.0-52-amd64 et l'architecture x86-64. La ligne Virtualization: vmware confirme que c'est une machine virtuelle VMware.

ip -br a

```
sio@srv-linux:~$ ip -br a
lo                UNKNOWN          127.0.0.1/8 ::1/128
ens32             UP                192.168.88.135/24 fe80::20c:29ff:fec1:907f/64
ens33             DOWN
```

Étape 2. Découvrir l'absence de sudo et l'installer

Comme au TP0, je commence par mettre la machine à jour.

sudo apt update

```
sio@srv-linux:~$ sudo apt update
-bash: sudo : commande introuvable
```

Je corrige en passant par le compte root, qui lui est actif sur Debian :

```
sio@srv-linux:~$ su -
Mot de passe :
root@srv-linux:~# apt update
Atteint :1 http://security.debian.org/debian-security bookworm-security InRelease
[...]
Tous les paquets sont à jour.

root@srv-linux:~# apt install -y sudo
Les NOUVEAUX paquets suivants seront installés :
  sudo
0 mis à jour, 1 nouvellement installés, 0 à enlever et 0 non mis à jour.
[...]
root@srv-linux:~# usermod -aG sudo sio
root@srv-linux:~# exit
```

Un seul paquet à installer : sudo n'était vraiment pas là. L'option -a de usermod veut dire « ajouter ». Sans elle, -G remplacerait toute la liste des groupes de sio par le seul groupe sudo. Le compte perdrait ses huit autres groupes d'un coup.

Je ferme donc vraiment la session avec exit, jusqu'à revenir sur « srv-linux login: ». Puis je me reconnecte en sio.

```
srv-linux login: sio
Password:
Last login: Mon Sep  7 14:40:05 CEST 2026 on tty1
sio@srv-linux:~$ id
uid=1000(sio) gid=1000(sio) groupes=1000(sio),24(cdrom),25(floppy),27(sudo),
29(audio),30(dip),44(video),46(plugdev),100(users),106(netdev)
```

sudo apt update

```
sio@srv-linux:~$ sudo apt update
[sudo] Mot de passe de sio :
[...]
Tous les paquets sont à jour.
```

```
sudo apt install -y bind9-dnsutils net-tools nano
```

Vérifier qu'aucun environnement de bureau n'a été installé

```
dpkg -l | grep task
```

```
sio@srv-linux:~$ dpkg -l | grep task
ii task-french      3.73 all French environment
ii task-ssh-server 3.73 all SSH server
ii tasksel          3.73 all tool for selecting tasks for installation on Debian
systems
ii tasksel-data     3.73 all official tasks used for installation of Debian
systems
```

Deux tâches seulement sont installées : l'environnement français et le serveur SSH. C'est ce que j'avais coché. Aucun task-desktop ni task-gnome-desktop n'apparaît. La machine n'a donc pas d'environnement graphique.

Étape 3. Donner une adresse statique à srv-linux

Sur ce point, Debian ne se configure pas comme Ubuntu. Au TPO j'écrivais du YAML dans /etc/netplan/ et j'appliquais avec netplan apply. Ici il n'y a pas de netplan. Toute la configuration réseau tient dans un seul fichier texte, /etc/network/interfaces, géré par ifupdown.

```
sudo cp -p /etc/network/interfaces /etc/network/interfaces.bak
```

Le fichier livré par Debian ne déclarait que la carte NAT, et pas de la façon attendue :

```
# The primary network interface
allow-hotplug ens32
iface ens32 inet dhcp
```

J'ai remplacé allow-hotplug par auto. Les deux montent l'interface, mais pas de la même façon. Avec auto, c'est networking.service qui la monte au démarrage, et qui la remonte à chaque systemctl restart networking. Avec allow-hotplug, c'est udev qui la monte, quand le noyau détecte la carte. Le problème est là : un redémarrage du service réseau peut descendre l'interface sans la remonter, faute d'événement matériel. Je perdrais la carte NAT, donc l'accès aux dépôts, sans message d'erreur.

Le fichier complet est recopié en section 5.

Avant d'appliquer, je relis les déclarations :

```
grep -n "^auto\|^iface" /etc/network/interfaces
```

```
sio@srv-linux:~$ grep -n "^auto\|^iface" /etc/network/interfaces
7:auto lo
8:iface lo inet loopback
12:auto ens32
13:iface ens32 inet dhcp
17:auto ens33
18:iface ens33 inet static
```

Six lignes, et chaque interface déclarée une seule fois. Cette relecture sert à quelque chose : elle m'a fait voir une faute, racontée dans « Difficultés rencontrées ».

J'applique et je vérifie :

```
sio@srv-linux:~$ sudo systemctl restart networking
sio@srv-linux:~$ ip -br a
lo                UNKNOWN        127.0.0.1/8 ::1/128
ens32             UP             192.168.88.135/24 fe80::20c:29ff:fec1:907f/64
```

```
ens33          UP          192.168.100.10/24 fe80::20c:29ff:fec1:9089/64
```

ens33 est passée de DOWN à UP et porte bien 192.168.100.10/24. ens32 a gardé son adresse en DHCP.

```
sio@srv-linux:~$ ip r
default via 192.168.88.2 dev ens32
192.168.88.0/24 dev ens32 proto kernel scope link src 192.168.88.135
192.168.100.0/24 dev ens33 proto kernel scope link src 192.168.100.10

sio@srv-linux:~$ ping -c 2 192.168.100.1
64 bytes from 192.168.100.1: icmp_seq=1 ttl=128 time=1.41 ms
64 bytes from 192.168.100.1: icmp_seq=2 ttl=128 time=0.468 ms
2 packets transmitted, 2 received, 0% packet loss
```

Deux paquets envoyés, deux reçus, aucune perte. Le poste Windows est joignable sur VMnet2. Le ttl=128 indique un système Windows en face : une machine Linux répondrait avec ttl=64.

Enfin je vérifie le sens inverse, depuis PowerShell sur le poste Windows :

```
PS C:\windows\System32> ping 192.168.100.10
Reponse de 192.168.100.10 : octets=32 temps<1ms TTL=64
Reponse de 192.168.100.10 : octets=32 temps<1ms TTL=64
Paquets : envoyes = 4, recus = 4, perdus = 0 (perte 0%),
```

Quatre paquets envoyés, quatre reçus. La liaison marche dans les deux sens et l'étape 3 est finie. Au TP0, ce sens Windows vers Linux échouait à cause du pare-feu Windows. Ici il passe du premier coup : la règle ajoutée la semaine dernière est toujours là.

Étape 4. Cloner la VM pour créer cli-linux

Dans VMware : clic droit sur la VM, Manage, Clone. J'ai pris l'état courant comme source, puis Create a full clone, et j'ai nommé le clone cli-linux. Un linked clone prend moins de place mais reste lié à l'original : supprimer srv-linux casserait cli-linux. Je veux deux machines vraiment indépendantes, puisqu'elles doivent tenir jusqu'au jour 5.

Rendre le clone indépendant

sudo hostnamectl set-hostname cli-linux

Puis dans /etc/hosts, la ligne 127.0.1.1. Elle contenait, héritée du modèle :

```
sio@cli-linux:~$ cat /etc/hosts
127.0.0.1    localhost
127.0.1.1    cli-linux.bts-sio.lan cli-linux

sio@cli-linux:~$ cat /etc/hostname
cli-linux
```

La configuration réseau du clone est l'inverse de celle du serveur : la carte NAT est commentée, et ens33 passe en DHCP. Le fichier complet est en section 5.

grep -n "source|^auto|^iface" /etc/network/interfaces

```
1:source /etc/network/interfaces.d/*
3:auto lo
4:iface lo inet loopback
11:auto ens33
12:iface ens33 inet dhcp
```

Vérification du clone

Debian GNU/Linux 12 cli-linux tty1

```
cli-linux login: sio
sio@cli-linux:~$ hostname
cli-linux
```

ip link show ens33

```
sio@cli-linux:~$ ip link show ens33
3: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
    link/ether 00:0c:29:b1:8d:39 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
```

ip -br a

```
sio@cli-linux:~$ ip -br a
lo                UNKNOWN          127.0.0.1/8 ::1/128
ens32             DOWN
ens33             UP              fe80::20c:29ff:feb1:8d39/64
```

ens32 est DOWN, comme prévu. ens33 est UP mais n'a aucune adresse IPv4. On voit seulement une adresse IPv6 de lien local, que le noyau fabrique à partir de la MAC. C'est le résultat attendu : aucun serveur DHCP n'existe encore sur VMnet2. Le client est prêt, mais personne ne lui répond.

À ce stade j'ai pris un instantané VMware des deux machines, nommé J2 depart. C'est mon point de retour pour toute la journée.

Partie B. Le serveur DHCP

Étape 5. Installer isc-dhcp-server et choisir son interface

Toute cette partie se fait en SSH depuis le poste Windows. Ça me permet de copier les sorties en texte au lieu de les photographier.

sudo apt install -y isc-dhcp-server

L'installation se termine, mais le démarrage du service échoue immédiatement :

```
Generating /etc/default/isc-dhcp-server...
Job for isc-dhcp-server.service failed because the control process exited with error code.
invoke-rc.d: initscript isc-dhcp-server, action "start" failed.
```

sudo systemctl status isc-dhcp-server

```
× isc-dhcp-server.service - LSB: DHCP server
   Loaded: loaded (/etc/init.d/isc-dhcp-server; generated)
   Active: failed (Result: exit-code) since Mon 2026-09-07 16:12:13 CEST
     Docs: man:systemd-sysv-generator(8)
   Process: 1317 ExecStart=/etc/init.d/isc-dhcp-server start (code=exited, status=1/FAILURE)
```

sudo journalctl -u isc-dhcp-server -n 50 --no-pager

Étape 5 (suite). Choisir l'interface d'écoute

Je sauvegarde le fichier, puis je renseigne l'interface raccordée à VMnet2 :

sudo cp -p /etc/default/isc-dhcp-server /etc/default/isc-dhcp-server.bak

```
sio@srv-linux:~$ grep -v "^#" /etc/default/isc-dhcp-server | grep -v "^$"
INTERFACESv4="ens33"
INTERFACESv6=""
```

Étape 6. Écrire dhcpd.conf selon le cahier des charges

Ce que le TP demande :

Paramètre	Valeur imposée
Sous-réseau desservi	192.168.100.0 / 255.255.255.0
Plage dynamique	192.168.100.100 à 192.168.100.150
Passerelle annoncée	192.168.100.1
Serveur DNS annoncé	192.168.100.10
Nom de domaine annoncé	bts-sio.lan

Bail par défaut	600 secondes
Bail maximal	7200 secondes
Serveur faisant autorité	oui
Réservation	cli-linux, MAC 00:0c:29:b1:8d:39, adresse 192.168.100.50

Le fichier livré par Debian ne contient que des exemples commentés. Je l'ai gardé de côté sous le nom .orig et je l'ai réécrit entièrement. Le fichier complet est recopié en réponse à la question 7.

sudo cp -p /etc/dhcp/dhcpd.conf /etc/dhcp/dhcpd.conf.orig

La directive authoritative est importante. Elle autorise le serveur à répondre DHCPNAK, un refus clair, à un client qui demande une adresse d'un autre plan. C'est le cas d'un portable qui revient d'un autre réseau avec un bail encore valide. Sans authoritative, le serveur ne dit rien, et le client garde son ancienne adresse jusqu'à expiration sans pouvoir se connecter.

Je contrôle la syntaxe avant tout démarrage :

sudo dhcpd -t -cf /etc/dhcp/dhcpd.conf

```
sio@srv-linux:~$ sudo dhcpd -t -cf /etc/dhcp/dhcpd.conf
Internet Systems Consortium DHCP Server 4.4.3-P1
Config file: /etc/dhcp/dhcpd.conf
Database file: /var/lib/dhcp/dhcpd.leases
```

Étape 7. Contrôler, démarrer et activer le service

La syntaxe étant validée à l'étape précédente, je démarre le service et je le rends persistant :

```
sio@srv-linux:~$ sudo systemctl restart isc-dhcp-server
sio@srv-linux:~$ sudo systemctl enable isc-dhcp-server
isc-dhcp-server.service is not a native service, redirecting to systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable isc-dhcp-server
```

sudo systemctl status isc-dhcp-server

```
● isc-dhcp-server.service - LSB: DHCP server
   Active: active (running) since Tue 2026-09-08 08:53:40 CEST
   CGroup: /system.slice/isc-dhcp-server.service
           └─967 /usr/sbin/dhcpd -4 -q -cf /etc/dhcp/dhcpd.conf ens33

sept. 08 08:53:38 srv-linux isc-dhcp-server[955]: Launching IPv4 server only.
```

Le service est actif. Deux lignes prouvent que ma modification de /etc/default/isc-dhcp-server a été prise en compte.

La seconde est la ligne CGroup. Elle montre la commande vraiment lancée : /usr/sbin/dhcpd -4 -q -cf /etc/dhcp/dhcpd.conf ens33. Le nom de mon interface est passé en argument. C'est le script d'init qui a lu le fichier /etc/default et l'a transmis au démon. Le confinement n'est donc pas juste écrit dans un fichier : c'est un paramètre du processus qui tourne.

systemctl is-enabled isc-dhcp-server

```
sio@srv-linux:~$ systemctl is-enabled isc-dhcp-server
isc-dhcp-server.service is not a native service, redirecting to systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install is-enabled isc-dhcp-server
enabled
```

sudo ss -u | grep :67

```
sio@srv-linux:~$ sudo ss -u | grep :67
UNCONN 0 0 0.0.0.0:67 0.0.0.0:* users:(("dhcpd",pid=967,fd=8))
```

Étape 8. Tester depuis cli-linux et lire les journaux

J'ai travaillé avec deux fenêtres. Dans la première, une session SSH sur le serveur, mise à l'écoute du journal avant tout le reste :

sudo journalctl -u isc-dhcp-server -f

Le client demande une adresse dès son démarrage, puisque son interface ens33 est déclarée auto et en dhcp. Quatre lignes sont apparues :

```
sept. 08 09:06:16 srv-linux dhcpd[864]: DHCPDISCOVER from 00:0c:29:b1:8d:39 via ens33
sept. 08 09:06:16 srv-linux dhcpd[864]: DHCPOFFER on 192.168.100.50 to 00:0c:29:b1:8d:39
                                                    via ens33
sept. 08 09:06:16 srv-linux dhcpd[864]: DHCPREQUEST for 192.168.100.50 (192.168.100.10)
                                                    from 00:0c:29:b1:8d:39 via ens33
sept. 08 09:06:16 srv-linux dhcpd[864]: DHCPACK on 192.168.100.50 to 00:0c:29:b1:8d:39
                                                    via ens33
```

Ce que reçoit le client

```
sio@cli-linux:~$ ip -br a
ens32          DOWN
ens33          UP              192.168.100.50/24

sio@cli-linux:~$ ip r
default via 192.168.100.1 dev ens33

sio@cli-linux:~$ cat /etc/resolv.conf
domain bts-sio.lan
search bts-sio.lan
nameserver 192.168.100.10
```

Obtenir un vrai bail dynamique pour lire le registre

```
sio@srv-linux:~$ sudo cat /var/lib/dhcp/dhcpd.leases
# The format of this file is documented in the dhcpd.leases(5) manual page.
# This lease file was written by isc-dhcp-4.4.3-P1
authoring-byte-order little-endian;
server-uid "\000\001\000\00122l\365\000\014)\301\220\211";
```

Pour avoir quand même un bail à examiner, j'ai commenté le bloc host de dhcpd.conf, vérifié la syntaxe, redémarré le service, puis redemandé une adresse depuis le client. Le client est alors servi par la plage dynamique :

```
sio@cli-linux:~$ sudo dhclient -v ens33
DHCPDISCOVER on ens33 to 255.255.255.255 port 67 interval 5
DHCPOFFER of 192.168.100.100 from 192.168.100.10
DHCPREQUEST for 192.168.100.100 on ens33 to 255.255.255.255 port 67
DHCPACK of 192.168.100.100 from 192.168.100.10
bound to 192.168.100.100 -- renewal in 257 seconds.
```

La dernière ligne donne le minuteur T1 : renewal in 257 seconds. Le bail vaut 600 secondes, et le cours dit que T1 se déclenche à 50 %, soit 300 secondes. La valeur réelle est un peu plus basse parce que le client ajoute un petit écart aléatoire. Ça évite que des dizaines de postes démarrés en même temps redemandent tous leur bail à la même seconde.

Le registre contient alors un vrai bail, que j'analyse à la question 11. J'ai ensuite décommenté le bloc host, redémarré le service et renouvelé le bail du client. cli-linux est bien revenu en 192.168.100.50.

Partie C. Le serveur DNS, zone directe

Étape 9. Installer bind9 et écrire les options globales

```
sudo apt install -y bind9 bind9-utils bind9-dnsutils
```

```
sudo cp -p /etc/bind/named.conf.options /etc/bind/named.conf.options.bak
```

J'ai réécrit entièrement le bloc options. Il est recopié en réponse à la question 12.

```
sudo named-checkconf
```

La commande n'affiche rien, ce qui veut dire que la configuration est correcte.

```
sudo systemctl status named
```

```
● named.service - BIND Domain Name Server
   Loaded: loaded (/lib/systemd/system/named.service; enabled)
   Active: active (running) since Tue 2026-09-08 09:27:32 CEST

named[1276]: zone localhost/IN: loaded serial 2
named[1276]: all zones loaded
named[1276]: running
```

Les quatre zones chargées sont celles livrées par Debian : la boucle locale et ses zones inverses. Ma zone bts-sio.lan n'existe pas encore. Pour l'instant, le serveur sait seulement transmettre les questions à ses forwarders. Il est déjà résolveur récursif, mais pas encore autoritatif.

Étape 10. Déclarer la zone directe

La déclaration se fait dans /etc/bind/named.conf.local, le fichier réservé à mes zones. Il est recopié en section 5.

J'ai créé le fichier de zone à partir du modèle db.local de Debian, puis je l'ai réécrit entièrement. Il est recopié en réponse à la question 13.

Le numéro de série suit le format imposé AAAAMMJnn : 2026090801 pour la première modification du 8 septembre 2026. À chaque modification du fichier, il faudra l'incrémenter. Sinon un serveur esclave ne verrait jamais le changement.

Il reste les droits et les contrôles :

```
sio@srv-linux:~$ sudo chown root:bind /etc/bind/db.bts-sio.lan
sio@srv-linux:~$ sudo chmod 644 /etc/bind/db.bts-sio.lan
sio@srv-linux:~$ sudo named-checkconf
sio@srv-linux:~$ sudo named-checkzone bts-sio.lan /etc/bind/db.bts-sio.lan
zone bts-sio.lan/IN: loaded serial 2026090801
OK

sio@srv-linux:~$ sudo systemctl restart named
sio@srv-linux:~$ sudo systemctl enable named
Synchronizing state of named.service with SysV service script with
  /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable named
```

Étape 11. Interroger la zone avec dig

```
dig @192.168.100.10 srv-linux.bts-sio.lan
```

```
sio@srv-linux:~$ dig @192.168.100.10 srv-linux.bts-sio.lan
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 9334
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1

;; ANSWER SECTION:
srv-linux.bts-sio.lan. 604800 IN      A      192.168.100.10
```

Une seule réponse, l'enregistrement A attendu. Le 604800 en deuxième colonne est le TTL en secondes, soit sept jours. C'est la durée pendant laquelle un autre serveur peut garder cette réponse en cache.

dig @192.168.100.10 bts-sio.lan NS

```
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 2

;; ANSWER SECTION:
bts-sio.lan.          604800  IN      NS      ns1.bts-sio.lan.

;; ADDITIONAL SECTION:
ns1.bts-sio.lan.      604800  IN      A       192.168.100.10
```

dig @192.168.100.10 bts-sio.lan SOA

```
;; ANSWER SECTION:
bts-sio.lan. 604800 IN SOA ns1.bts-sio.lan. admin.bts-sio.lan.
                2026090801 604800 86400 2419200 604800
```

dig @192.168.100.10 nexistepas.bts-sio.lan

```
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 5676
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; AUTHORITY SECTION:
bts-sio.lan. 604800 IN SOA ns1.bts-sio.lan. admin.bts-sio.lan.
                2026090801 604800 86400 2419200 604800
```

Le statut NXDOMAIN est le résultat attendu, et c'est une bonne nouvelle. Le serveur ne dit pas « je ne sais pas », il dit « ce nom n'existe pas ». Seul un serveur autoritatif peut l'affirmer, et le drapeau aa est toujours là.

Partie D. La zone inverse

Étape 12. Créer et déclarer la zone 100.168.192.in-addr.arpa

La zone directe répond à la question « quelle adresse pour ce nom ». La zone inverse répond à l'inverse. Mais le DNS ne sait interroger que des noms. Il a donc fallu fabriquer un nom à partir d'une adresse. C'est le rôle du domaine spécial in-addr.arpa, sous lequel l'adresse est écrite à l'envers.

J'ajoute la déclaration de la zone inverse à la suite de la zone directe dans named.conf.local. Le fichier complet est en section 5.

Je crée ensuite le fichier à partir de db.127, le modèle de zone inverse livré par Debian. Il est recopié en réponse à la question 18.

```
sio@srv-linux:~$ sudo chown root:bind /etc/bind/db.192.168.100
sio@srv-linux:~$ sudo chmod 644 /etc/bind/db.192.168.100
sio@srv-linux:~$ sudo named-checkzone 100.168.192.in-addr.arpa
/etc/bind/db.192.168.100
zone 100.168.192.in-addr.arpa/IN: loaded serial 2026090801
OK
sio@srv-linux:~$ sudo rndc reload
server reload successful
```

J'ai utilisé rndc reload et pas systemctl restart named. La différence compte quand le serveur tourne pour de vrai. rndc reload demande au démon de relire ses zones sans s'arrêter : aucune coupure, et le cache est conservé. Un redémarrage coupe le service quelques instants et vide le cache. Le redémarrage complet reste nécessaire pour une modification de named.conf.options. Pour un simple changement de zone, rndc reload est la commande du quotidien.

Vérification :

```
sio@srv-linux:~$ dig @192.168.100.10 -x 192.168.100.10 +short
srv-linux.bts-sio.lan.
```

```
sio@srv-linux:~$ dig @192.168.100.10 -x 192.168.100.50 +short
cli-linux.bts-sio.lan.
```

Les deux adresses se résolvent en noms. La résolution inverse marche pour le serveur en .10 comme pour le client en .50. Ce dernier a reçu son adresse par la réservation DHCP de la partie B.

Partie E. Intégration DHCP et DNS côté client

Étape 13. Renouveler le bail et résoudre un nom court

Cette partie se fait entièrement sur cli-linux. C'est la seule partie où le client est la machine qu'on regarde, et pas juste l'outil de test. Elle vérifie que les deux services montés chacun de leur côté marchent ensemble.

```
sudo dhclient -r ens33 puis sudo dhclient -v ens33
```

```
DHCPDISCOVER on ens33 to 255.255.255.255 port 67 interval 3
DHCPOFFER of 192.168.100.50 from 192.168.100.10
DHCPREQUEST for 192.168.100.50 on ens33 to 255.255.255.255 port 67
DHCPACK of 192.168.100.50 from 192.168.100.10
bound to 192.168.100.50 -- renewal in 236 seconds.
```

Le client retrouve bien 192.168.100.50. La réservation a été remise en place après l'expérience de la question 11.

```
cat /etc/resolv.conf
```

```
domain bts-sio.lan
search bts-sio.lan
nameserver 192.168.100.10
```

```
ping -c 2 www
```

```
PING srv-linux.bts-sio.lan (192.168.100.10) 56(84) bytes of data:
64 bytes from srv-linux.bts-sio.lan (192.168.100.10): icmp_seq=1 ttl=64 time=0.393
ms
64 bytes from srv-linux.bts-sio.lan (192.168.100.10): icmp_seq=2 ttl=64 time=0.537
ms
--- srv-linux.bts-sio.lan ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
```

Les commandes ping intranet et ping srv-linux donnent le même résultat, aux temps de réponse près. Les trois noms mènent à 192.168.100.10.

```
dig www.bts-sio.lan +short
```

```
srv-linux.bts-sio.lan.
192.168.100.10
```

```
dig debian.org +short
```

```
151.101.66.132
151.101.194.132
151.101.2.132
151.101.130.132
```

Ce que devient une modification manuelle de resolv.conf

Pour répondre à la question 23, j'ai ajouté à la main une seconde ligne nameserver dans /etc/resolv.conf :

```
sio@cli-linux:~$ cat /etc/resolv.conf
domain bts-sio.lan
search bts-sio.lan
nameserver 192.168.100.10
nameserver 9.9.9.9
```

Puis j'ai renouvelé le bail et relu le fichier :

```
sio@cli-linux:~$ cat /etc/resolv.conf
domain bts-sio.lan
search bts-sio.lan
nameserver 192.168.100.10
```

Partie F. Durcissement SSH

Étape 14. Générer et déployer une paire de clés

Génération sur cli-linux :

```
sio@cli-linux:~$ ssh-keygen -t ed25519 -C "sio@cli-linux"
Your identification has been saved in /home/sio/.ssh/id_ed25519
Your public key has been saved in /home/sio/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:hdzXVvUj18BjCh7xS54zlg2Ixxh65RIwtLMQE6LB63UM sio@cli-linux

sio@cli-linux:~$ ls -l ~/.ssh/
-rw----- 1 sio sio 399  8 sept. 10:34 id_ed25519
-rw-r--r-- 1 sio sio  95  8 sept. 10:34 id_ed25519.pub
```

ssh-keygen met lui-même les bons droits : 600 sur la clé privée, que personne d'autre ne doit lire, et 644 sur la publique, qui n'est pas secrète.

Dépôt sur le serveur :

```
sio@cli-linux:~$ ssh-copy-id sio@192.168.100.10
INFO: 1 key(s) remain to be installed -- if you are prompted now it is to
install the new keys
sio@192.168.100.10's password:

Number of key(s) added: 1
```

Cette demande de mot de passe est normale, et l'outil le dit lui-même. C'est la dernière connexion classique, celle qui sert à déposer la clé. Le vrai test est la connexion suivante, qui doit passer sans rien demander.

Pour le poste Windows, ssh-copy-id n'existe pas dans l'OpenSSH intégré. J'avais déjà une paire de clés sur mon compte, créée en mai. Je ne l'ai pas régénérée. Écraser une clé privée la détruit définitivement, et je ne sais pas à quels autres services elle donne accès. Une clé publique se dépose sur autant de serveurs qu'on veut : il n'y a aucune raison d'en créer une par serveur.

J'ai donc envoyé la clé publique existante directement, sans passer par un copier-coller :

```
PS C:\Users\doria> type $env:USERPROFILE\.ssh\id_ed25519.pub |
ssh sio@192.168.100.10 "cat >> ~/.ssh/authorized_keys"
sio@192.168.100.10's password:
```

Étape 15. Durcir sshd_config

```
sio@srv-linux:~$ ls -l /etc/ssh/sshd_config.d/
total 0
```

J'ai sauvegardé le fichier livré par Debian, puis j'ai écrit ma configuration. Le fichier complet est recopié en section 5.

AllowUsers sio est la ligne dangereuse. C'est une liste blanche : tout compte absent est refusé, même avec la bonne clé. Une faute de frappe fermerait la porte à tout le monde, sans message clair.

J'ai contrôlé la syntaxe avant de redémarrer, et sans fermer ma session en cours :

```
sio@srv-linux:~$ sudo sshd -t
sio@srv-linux:~$ sudo systemctl restart ssh
```

sshd -t valide le fichier sans rien redémarrer, et ne dit rien s'il est correct. Le redémarrage, lui, ne coupe pas les sessions déjà ouvertes. La mienne est restée ouverte pendant tout le test. C'était ma seule garantie de pouvoir revenir en arrière.

Vérifier que tout marche encore

Trois essais depuis le poste Windows, puis depuis cli-linux.

```
PS C:Usersdoria> ssh sio@192.168.100.10
Linux srv-linux 6.1.0-52-amd64 ...
Last login: Tue Sep  8 10:48:49 2026 from 192.168.100.1
sio@srv-linux:~$

PS C:Usersdoria> ssh root@192.168.100.10
root@192.168.100.10: Permission denied (publickey).
```

La connexion par clé fonctionne toujours, sans mot de passe, et root est refusé.

```
sio@srv-linux:~$ ssh -o PubkeyAuthentication=no sio@192.168.100.10
Warning: Permanently added '192.168.100.10' (ED25519) to the list of known hosts.
ssh_dispatch_run_fatal: Connection to 192.168.100.10 port 22: Broken pipe
```

Étape 16. Observer les journaux d'authentification

L'énoncé demande de suivre /var/log/auth.log. Ce fichier n'existe pas sur ma machine :

```
sio@srv-linux:~$ sudo tail -f /var/log/auth.log
tail: impossible d'ouvrir '/var/log/auth.log' en lecture: Aucun fichier ou dossier
tail: aucun fichier restant
```

L'équivalent natif est donc :

sudo journalctl -u ssh -f

Partie G. Panne à diagnostiquer

Étape 17. Retrouver et corriger trois fautes de configuration

Avant de commencer, j'ai pris un instantané VMware des deux machines, nommé J2 avant panne. À cet instant tout fonctionne : les deux services sont actifs, les deux zones se chargent, et le client obtient son adresse et résout les noms.

Trois fautes ont ensuite été introduites dans ma configuration, sans qu'on me dise lesquelles ni où. J'ai appliqué la méthode dans l'ordre : constater, lire le message exact, contrôler la syntaxe, vérifier le comportement.

systemctl is-active isc-dhcp-server named

```
sio@srv-linux:~$ systemctl is-active isc-dhcp-server named
failed
failed
```

Les deux services sont tombés. Je lis les journaux avant de toucher à quoi que ce soit.

sudo dhcpcd -t -cf /etc/dhcp/dhcpd.conf

```
/etc/dhcp/dhcpd.conf line 27: semicolon expected.
    option
    ^
Configuration file errors encountered -- exiting
```

Le message donne le fichier, la ligne et le mot qui bloque. Mais l'erreur pointe la ligne 27, alors que la faute est à la ligne 26 : c'est elle qui a perdu son point-virgule. L'analyseur ne s'en aperçoit qu'en arrivant au mot suivant.

sudo named-checkconf

```
/etc/bind/named.conf.default-zones:2: unknown option 'zone'
/etc/bind/named.conf.default-zones:10: unknown option 'zone'
/etc/bind/named.conf.default-zones:15: unknown option 'zone'
/etc/bind/named.conf.default-zones:20: unknown option 'zone'
/etc/bind/named.conf.default-zones:25: unknown option 'zone'
/etc/bind/named.conf:12: '}' expected near end of file
```

Ici le message accuse deux fichiers livrés par Debian, auxquels je n'ai jamais touché. C'est la vraie difficulté de cette partie : le fichier nommé dans l'erreur n'est pas le fichier fautif.

L'explication est dans l'ordre des inclusions de named.conf : d'abord named.conf.options, puis named.conf.local, puis named.conf.default-zones. Une accolade jamais fermée dans mon fichier fait croire à l'analyseur qu'il est encore à l'intérieur d'un bloc quand il commence à lire le suivant. À l'intérieur d'un bloc, le mot zone n'a plus de sens : d'où les cinq plaintes sur un fichier parfaitement correct. Le coupable est donc le fichier inclus juste avant, named.conf.local.

Une fois les deux corrections faites, les services repartent. Mais les deux zones répondaient déjà OK au contrôle, donc la troisième faute est ailleurs. Je relis le journal de named en entier, et pas seulement les lignes marquées error :

```
sudo journalctl -u named --no-pager | grep -i serial
```

```
sept. 08 14:04:09 srv-linux named[1133]: zone bts-sio.lan/IN: zone serial
(2026090801)
    unchanged. zone may fail to transfer to secondaries.
sept. 08 14:04:09 srv-linux named[1133]: zone bts-sio.lan/IN: loaded serial
2026090801
```

C'est un avertissement, pas une erreur. Le fichier de zone a été modifié mais son numéro de série n'a pas bougé. En le relisant, je trouve bien un enregistrement que je n'ai jamais écrit :

```
sio@srv-linux:~$ grep -n "imprimante|Serial" /etc/bind/db.bts-sio.lan
7:                2026090801                ; Serial - A INCREMENTER
20:imprimante      IN      A      192.168.100.30
```

Je corrige en incrémentant le serial, puis je recharge :

```
sio@srv-linux:~$ sudo named-checkzone bts-sio.lan /etc/bind/db.bts-sio.lan
zone bts-sio.lan/IN: loaded serial 2026090802
OK
```

Vérification finale, une fois les trois fautes corrigées :

```
sio@srv-linux:~$ systemctl is-active isc-dhcp-server named
active
active

sio@cli-linux:~$ sudo dhclient -v ens33
DHCPACK of 192.168.100.50 from 192.168.100.10
bound to 192.168.100.50 -- renewal in 272 seconds.

sio@cli-linux:~$ ping -c 2 www
PING srv-linux.bts-sio.lan (192.168.100.10) 56(84) bytes of data.
64 bytes from srv-linux.bts-sio.lan (192.168.100.10): icmp_seq=1 ttl=64 time=0.283
ms
2 packets transmitted, 2 received, 0% packet loss
```

Les deux services sont actifs, le client reçoit son adresse réservée, et ping www affiche srv-linux.bts-sio.lan : le CNAME, la zone directe et la zone inverse fonctionnent toutes les trois. Tout est revenu comme avant la panne.

4. Réponses aux questions

Partie A. Préparation des VM

Q1. *Que répond la machine à `sudo apt update` ? Expliquez en deux phrases pourquoi ce comportement diffère d'Ubuntu Server 24.04, en vous appuyant sur ce qui a été saisi pendant l'installation.*

La machine répond :

```
sio@srv-linux:~$ sudo apt update
-bash: sudo : commande introuvable
```

Le message vient de bash, pas de sudo. Le shell cherche un programme de ce nom dans le PATH et n'en trouve aucun. Le paquet sudo n'est pas installé.

La cause est un écran de l'installation. Debian m'a demandé un mot de passe pour root et je lui en ai donné un. Le compte root est donc actif et fait office d'administrateur. sudo devient facultatif, et l'installateur ne le pose pas. Ubuntu Server 24.04 ne pose jamais cette question. Il laisse root verrouillé, installe sudo d'office et met le premier compte créé dans le groupe sudo. Sans ça, plus personne ne pourrait administrer la machine.

Q2. *Pourquoi faut-il fermer la session et en rouvrir une pour que l'appartenance au groupe sudo soit prise en compte ? Que montre la sortie de `id` avant et après ?*

Les groupes d'un utilisateur sont lus une seule fois, à l'ouverture de la session. Le programme de connexion lit /etc/group, construit la liste et l'attache au processus du shell. Tout ce que le shell lance ensuite hérite de cette liste, et personne ne la relit. usermod modifie le fichier sur le disque, pas les processus déjà lancés. Mon shell avait été ouvert avant la modification : il travaillait avec l'ancienne liste.

Avant le usermod, et de nouveau après le usermod mais dans la même session, id renvoie exactement la même chose :

```
uid=1000(sio) gid=1000(sio) groupes=1000(sio),24(cdrom),25(floppy),29(audio),
30(dip),44(video),46(plugdev),100(users),106(netdev)
```

Après déconnexion complète et reconnexion :

```
uid=1000(sio) gid=1000(sio) groupes=1000(sio),24(cdrom),25(floppy),27(sudo),
29(audio),30(dip),44(video),46(plugdev),100(users),106(netdev)
```

Le groupe 27(sudo) est le seul ajout. Il s'est glissé entre 25(floppy) et 29(audio). La liste est triée par numéro de groupe, elle a donc été reconstruite entièrement à l'ouverture de session.

Q3. *Relevez le nom exact de vos deux interfaces et l'adresse obtenue par la première. Comment savez-vous laquelle est raccordée à VMnet2 ?*

Mes deux interfaces s'appellent ens32 et ens33. L'énoncé donnait ens33 et ens37. Ces noms ne se recopient jamais : ils dépendent de la position des cartes sur le bus PCI de chaque machine.

```
sio@srv-linux:~$ ip -br a
lo                UNKNOWN          127.0.0.1/8  ::1/128
ens32             UP               192.168.88.135/24  fe80::20c:29ff:fec1:907f/64
ens33             DOWN
```

ens32 a obtenu 192.168.88.135/24. C'est une adresse du réseau NAT de VMware, donnée par le DHCP de l'hyperviseur. Elle peut changer d'un démarrage à l'autre, c'est le principe d'un bail DHCP.

C'est comme ça que je sais laquelle est branchée sur VMnet2 : c'est celle qui ne reçoit rien. Le DHCP de VMware est désactivé sur ce réseau, et le mien n'existe pas encore. Aucune adresse ne peut donc

y être distribuée. La seule carte qui en reçoit une est forcément celle du NAT. On peut aussi vérifier côté VMware, en comparant la MAC affichée par `ip link show` avec celle de chaque carte dans les paramètres de la VM.

Q4. Relevez l'adresse MAC de `ens37` sur `cli-linux`. Pourquoi faut-il régénérer les adresses MAC d'un clone ? Que se passerait-il si les deux VM partageaient la même MAC sur VMnet2 ?

Sur ma machine l'interface s'appelle `ens33` et non `ens37`. Son adresse MAC est :

```
sio@cli-linux:~$ ip link show ens33
3: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
    link/ether 00:0c:29:b1:8d:39 brd ff:ff:ff:ff:ff:ff
```

Il faut régénérer ces adresses parce qu'un clone complet copie le fichier de description de la VM, MAC comprises. Sans régénération, deux cartes différentes auraient le même identifiant de couche 2 sur le même réseau. Or une MAC doit être unique sur un segment Ethernet : c'est elle, et pas l'adresse IP, qui sert à livrer les trames.

Si les deux VM avaient la même MAC sur VMnet2, plusieurs choses casseraient en même temps. Le commutateur virtuel, qui associe une MAC à un port, enverrait les réponses tantôt à l'une tantôt à l'autre, et les connexions tomberaient par moments. Le cache ARP des autres machines deviendrait faux. Et surtout le serveur DHCP, qui reconnaît ses clients par leur MAC, croirait n'en voir qu'une seule : il donnerait le même bail aux deux, et ma réservation 192.168.100.50 s'appliquerait à l'une ou à l'autre au hasard.

Partie B. Le serveur DHCP

Q5. Le service est en échec juste après l'installation. Relevez le message exact affiché par `systemctl status` et expliquez pourquoi cet échec est normal à ce stade.

`systemctl status` affiche :

```
x isc-dhcp-server.service - LSB: DHCP server
   Active: failed (Result: exit-code) since Mon 2026-09-07 16:12:13 CEST
   Process: 1317 ExecStart=/etc/init.d/isc-dhcp-server start (code=exited,
   status=1/FAILURE)
```

Mais le message utile est dans le journal, et il tient en une ligne :

```
dhcpcd[1329]: Not configured to listen on any interfaces!
dhcpcd[1329]: exiting.
```

Cet échec est normal parce que rien n'est encore configuré. Installer un paquet pose des fichiers et un service, mais ça ne devine pas ce que l'administrateur veut distribuer. Le `dhcpcd.conf` livré par Debian ne contient que des exemples commentés. Aucune déclaration subnet ne correspond au réseau de mes interfaces. Le démon examine chaque carte, ne trouve la description d'aucun segment, et se retrouve sans rien à servir.

Q6. Expliquez pourquoi la ligne `INTERFACESv4` est indispensable et en quoi elle constitue une mesure de sécurité. Que se passerait-il si vous laissiez la valeur vide sur cette machine, qui possède aussi une carte en NAT ?

Cette ligne indique les interfaces sur lesquelles le démon a le droit de répondre. Laisser vide, elle ne veut pas dire « aucune » : elle veut dire « toutes ». Le journal de la question 5 le montre en direct, avant même que j'aie touché au fichier :

```
No subnet declaration for ens33 (192.168.100.10).
No subnet declaration for ens32 (192.168.88.134).
```

C'est pour ça que cette ligne est une mesure de sécurité, et pas une simple formalité. Elle enferme le service sur un réseau choisi, quel que soit le contenu de `dhcpcd.conf`. Sans elle, il suffit qu'une

déclaration subnet corresponde par hasard au réseau d'une autre carte pour que le serveur se mette à y répondre.

Sur cette machine, le risque est réel. La carte ens32 est branchée sur le NAT de VMware, donc sur le réseau de l'établissement. Si je laissais la valeur vide et qu'une déclaration couvrirait ce réseau, mon serveur répondrait aux DISCOVER des postes voisins. Or un client garde la première offre reçue. Les machines de mes camarades prendraient des adresses de mon plan, avec ma passerelle et mon DNS, qui n'existent pas pour elles. Je deviendrais un serveur DHCP pirate et je mettrais la salle en panne sans le vouloir.

J'écris donc `INTERFACESv4="ens33"`, le vrai nom de ma carte VMnet2 relevé avec `ip -br a`. `INTERFACESv6` reste vide, le TP ne demande pas d'IPv6.

Q7. Recopiez le contenu complet de votre `/etc/dhcp/dhcpd.conf`, commentaires compris.

```
# -----
# Serveur DHCP - BTS SIO - reseau pedagogique 192.168.100.0/24
# Machine : srv-linux (192.168.100.10), interface d'ecoute ens33
# -----

# Nom de domaine annonce aux clients (option 15). Il alimente la ligne
# « search » de /etc/resolv.conf chez le client.
option domain-name "bts-sio.lan";

# Serveur(s) DNS annonces (option 6). Ici, notre propre bind9.
option domain-name-servers 192.168.100.10;

# Duree de bail par default, en secondes, si le client ne demande rien.
default-lease-time 600;

# Duree maximale accordee, meme si le client en demande davantage.
max-lease-time 7200;

# Ce serveur fait autorite sur les reseaux qu'il declare : il repond par
# un DHCPNAK a un client qui reclame une adresse hors plan.
authoritative;

# Declaration du sous-reseau desservi.
subnet 192.168.100.0 netmask 255.255.255.0 {
    range 192.168.100.100 192.168.100.150; # plage dynamique
    option routers 192.168.100.1; # passerelle (option 3)
    option subnet-mask 255.255.255.0; # masque (option 1)
    option broadcast-address 192.168.100.255;
    option domain-name-servers 192.168.100.10;
    option domain-name "bts-sio.lan";
}

# Reservation : cli-linux recoit toujours 192.168.100.50.
host cli-linux {
    hardware ethernet 00:0c:29:b1:8d:39;
    fixed-address 192.168.100.50;
}
```

Q8. Pourquoi la réservation 192.168.100.50 est-elle placée en dehors de la plage 100-150 ? Quel serait le risque, en exploitation, de la placer à l'intérieur ?

Une réservation et une plage dynamique sont deux mécanismes différents, mais ils puisent dans les mêmes adresses. La plage est un stock que le serveur distribue au premier qui demande. La réservation est une promesse faite à une machine précise, reconnue par sa MAC. Rien n'empêche de mettre l'une dans l'autre. C'est justement le problème.

Le risque vient de l'ordre d'arrivée. Si 192.168.100.50 était dans la plage, le serveur pourrait la donner à n'importe quelle machine, simplement parce qu'elle a demandé la première, avant que cli-

linux soit allumé. Quand cli-linux se présenterait ensuite, son adresse serait déjà prise par un bail actif. Le serveur lui donnerait autre chose, ou essaierait de lui donner une adresse occupée. Résultat : deux machines sur la même adresse, une réservation qui ne réserve rien, et une panne qui n'apparaît que selon l'ordre d'allumage. C'est le pire type de panne, parce qu'elle disparaît quand on essaie de la reproduire.

Q9. *Quelle est la différence entre `systemctl start` et `systemctl enable` ? Laquelle des deux manque-t-il si le service ne repart pas après un redémarrage de la VM ?*

Les deux commandes n'agissent pas sur la même chose, et aucune ne remplace l'autre.

`start` agit maintenant : il lance le service tout de suite. Son effet disparaît au prochain arrêt de la machine, et il ne modifie aucun fichier sur le disque.

`enable` agit sur le démarrage : il inscrit le service dans la liste de ceux que `systemd` doit lancer au prochain amorçage, en créant un lien symbolique dans `/etc/systemd/system`. Il écrit donc sur le disque, mais il ne démarre rien tout de suite.

Si le service ne repart pas après un redémarrage alors qu'il marchait avant, c'est `enable` qui manque. Le symptôme est typique : tout marche pendant la séance, et plus rien le lendemain, sans qu'aucune configuration ait changé. `systemctl is-enabled` répond alors `disabled`.

```
sio@srv-linux:~$ systemctl is-enabled isc-dhcp-server
enabled
```

Q10. *Recopiez les quatre lignes du journal correspondant à l'échange DORA, puis expliquez chacune : qui émet, vers quelle destination, et pourquoi le message circule en diffusion ou en unicast.*

```
sept. 08 09:06:16 srv-linux dhcpcd[864]: DHCPDISCOVER from 00:0c:29:b1:8d:39 via ens33
sept. 08 09:06:16 srv-linux dhcpcd[864]: DHCPOFFER on 192.168.100.50 to 00:0c:29:b1:8d:39
                                                    via ens33
sept. 08 09:06:16 srv-linux dhcpcd[864]: DHCPREQUEST for 192.168.100.50 (192.168.100.10)
                                                    from 00:0c:29:b1:8d:39 via ens33
sept. 08 09:06:16 srv-linux dhcpcd[864]: DHCPACK on 192.168.100.50 to 00:0c:29:b1:8d:39
                                                    via ens33
```

DHCPDISCOVER. Il est émis par le client cli-linux, en diffusion. Il part de l'adresse 0.0.0.0 vers 255.255.255.255 sur le port 67, et en couche 2 vers `ff:ff:ff:ff:ff:ff`. La diffusion est obligatoire pour deux raisons. Le client n'a aucune adresse IP, donc il ne peut pas émettre en unicast. Et il ne connaît pas l'adresse du serveur, donc il ne saurait pas à qui écrire. Le journal ne montre que sa MAC : c'est sa seule identité à ce moment-là.

DHCPOFFER. Il est émis par le serveur srv-linux, en diffusion aussi. Il propose 192.168.100.50 avec tous les paramètres du bail : masque, passerelle, DNS, nom de domaine et durée. Le serveur diffuse parce que le client n'a toujours pas posé d'adresse sur sa carte. Lui répondre en unicast supposerait de pouvoir le joindre en IP, ce qui n'est pas encore possible.

DHCPREQUEST. Il est émis par le client, en diffusion, et c'est le point qui surprend le plus. Le client a choisi une offre, la première reçue. On pourrait croire qu'il répond directement au serveur retenu. Il diffuse pourtant, et le nom du serveur choisi est écrit dans le message : c'est la valeur entre parenthèses dans le journal, (192.168.100.10), qui correspond à l'option `server-identifier`. La raison est que les autres serveurs du réseau doivent apprendre qu'ils ne sont pas retenus, pour libérer l'adresse qu'ils avaient mise de côté en faisant leur offre. Sans cette diffusion, chaque demande bloquerait une adresse chez tous les serveurs.

DHCPACK. Il est émis par le serveur. Il confirme l'attribution et inscrit le bail dans son registre. Le client pose alors l'adresse sur sa carte et réécrit /etc/resolv.conf. C'est ce message qui rend l'attribution définitive : tant qu'il n'est pas arrivé, rien n'est acquis.

Q11. Relevez dans `dhcpd.leases` les champs `starts`, `ends`, `binding state` et `hardware ethernet`. Pourquoi les dates ne correspondent-elles pas à l'heure de votre montre, et à quelle heure ce bail expire-t-il réellement ?

Première chose que je n'attendais pas. Tant que cli-linux recevait son adresse par réservation, le fichier ne contenait aucun bail. Une adresse donnée par `fixed-address` n'est pas empruntée à un stock : elle est écrite dans la configuration. Le serveur n'a donc rien à retenir. Il relit sa configuration à chaque demande et retrouve la même réponse. Le fichier `dhcpd.leases` est le registre des adresses prêtées à partir de la plage dynamique, et 192.168.100.50 n'en fait pas partie. C'est aussi ce qu'expliquait le (not found) du DHCPRELEASE : le client a fait chercher au serveur un bail qui n'avait jamais existé.

J'ai donc commenté la réservation pour que le client soit servi par la plage, et le registre s'est rempli :

```
sio@srv-linux:~$ sudo cat /var/lib/dhcp/dhcpd.leases
lease 192.168.100.100 {
  starts 2 2026/09/08 07:16:39;
  ends 2 2026/09/08 07:26:39;
  cltt 2 2026/09/08 07:16:39;
  binding state active;
  next binding state free;
  rewind binding state free;
  hardware ethernet 00:0c:29:b1:8d:39;
  client-hostname "cli-linux";
}
```

Voici comment se lisent les champs demandés. `starts` est la date d'attribution du bail, `ends` sa date d'expiration. L'écart entre les deux vaut exactement 600 secondes, soit le `default-lease-time` que j'ai déclaré. `binding state active` veut dire que le bail court et que l'adresse est prise ; `free` voudrait dire qu'elle est revenue dans le stock. La ligne `next binding state free` annonce l'état vers lequel le bail basculera à son expiration. `hardware ethernet` donne la MAC de la machine qui a le bail, 00:0c:29:b1:8d:39, qui est bien celle de ens33 sur cli-linux : c'est le seul lien entre une adresse IP et une machine. Le champ `cltt`, pour client last transaction time, garde la date du dernier échange. Et `client-hostname` est le nom que la machine a annoncé elle-même.

```
sio@srv-linux:~$ date
mar. 08 sept. 2026 09:17:41 CEST
```

Il est 09:17:41 en heure locale alors que le bail a démarré à 07:16:39 dans le fichier. Deux heures d'écart, ce qui correspond au décalage de la France en été : le fuseau CEST vaut UTC+2. C'est normal. Un serveur DHCP peut servir des clients dans plusieurs fuseaux, et son registre doit rester comparable d'un site à l'autre. Écrire en heure locale rendrait aussi le fichier incohérent deux fois par an, au changement d'heure.

Ce bail expire donc à 07:26:39 UTC, soit 09:26:39 heure locale, dix minutes après son attribution. Le client n'attendra pas jusque-là. Il aura demandé un renouvellement bien avant, au déclenchement du minuteur T1, annoncé à 257 secondes dans sa propre sortie.

Partie C. Le serveur DNS, zone directe

Q12. Recopiez votre bloc `options` complet. Expliquez pourquoi `allow-recursion` limité au réseau local est une mesure de sécurité, et non une simple optimisation.

```
options {
```

```

directory "/var/cache/bind";

// Interfaces d'ecoute : boucle locale et reseau pedagogique.
// Sans cette ligne, named repondrait aussi sur la carte NAT.
listen-on { 127.0.0.1; 192.168.100.10; };
listen-on-v6 { none; };

// Qui a le droit de poser des questions a ce serveur.
allow-query { localhost; 192.168.100.0/24; };

// Recursion autorisee, mais pour notre reseau seulement.
recursion yes;
allow-recursion { localhost; 192.168.100.0/24; };

// Aucun transfert de zone : pas d'esclave dans ce montage.
allow-transfer { none; };

// Ce que l'on ne sait pas resoudre soi-meme part vers ces resolveurs.
forwarders {
    8.8.8.8;
    1.1.1.1;
};
forward first;

// Validation DNSSEC desactivee : le TLD .lan n'existe pas dans la
// racine publique. En production, on laisse « auto ».
dnssec-validation no;
};

```

Un serveur qui accepte la récursion de n'importe qui s'appelle un résolveur ouvert. Le danger n'est pas la charge qu'il subirait, mais l'usage qu'un attaquant peut en faire contre quelqu'un d'autre.

Ça s'appelle une attaque par amplification. Le DNS marche en UDP, un protocole sans connexion. Rien n'oblige donc celui qui pose la question à être vraiment celui qu'il dit être. Un attaquant envoie au résolveur ouvert une question courte, mais en mettant l'adresse de sa victime comme adresse source. Le serveur envoie alors une réponse beaucoup plus grosse que la question, parfois cinquante fois plus, à une victime qui n'a rien demandé. Multiplié par des milliers de résolveurs ouverts, ce trafic suffit à saturer une liaison. L'attaquant dépense peu, la victime encaisse tout, et ce sont les résolveurs mal configurés qui font le travail.

En limitant allow-recursion à localhost et 192.168.100.0/24, mon serveur ne rend ce service qu'aux machines de mon réseau. Une question récursive venue d'ailleurs reçoit un REFUSED. Il ne peut donc plus servir d'amplificateur.

Q13. Recopiez le contenu complet de `/etc/bind/db.bts-sio.lan` et la sortie exacte de `named-checkzone`.

```

;
; Zone directe de bts-sio.lan
; Serveur maitre : ns1.bts-sio.lan (192.168.100.10)
;
$TTL      604800
@         IN      SOA      ns1.bts-sio.lan. admin.bts-sio.lan. (
                        2026090802      ; Serial - A INCREMENTER
                        604800          ; Refresh
                        86400           ; Retry
                        2419200         ; Expire
                        604800 )        ; Negative Cache TTL
;
; --- Serveur de noms de la zone ---
@         IN      NS       ns1.bts-sio.lan.
; --- Enregistrements d'hotes (A) ---
ns1       IN      A        192.168.100.10

```

```

srv-linux      IN      A      192.168.100.10
cli-linux      IN      A      192.168.100.50
imprimante     IN      A      192.168.100.30

; --- Alias (CNAME) : ils pointent vers un NOM, jamais vers une adresse ---
www            IN      CNAME   srv-linux.bts-sio.lan.
intranet       IN      CNAME   srv-linux.bts-sio.lan.

```

Sortie du contrôle :

```

sio@srv-linux:~$ sudo named-checkzone bts-sio.lan /etc/bind/db.bts-sio.lan
zone bts-sio.lan/IN: loaded serial 2026090801
OK

```

Q14. *named-checkconf n'affiche rien quand la configuration est correcte, alors que named-checkzone affiche OK. Est-ce un problème ? Justifiez.*

Ce n'est pas un problème. Les deux comportements sont voulus, parce que les deux outils n'ont pas la même chose à dire.

named-checkconf suit la règle du silence, l'habitude des outils UNIX : un programme qui n'a rien à signaler ne dit rien, et toute sortie est un problème. C'est pratique. Dans un script, il suffit de tester le code de retour. Et à l'écran, une sortie vide se repère tout de suite. Son travail est de vérifier une syntaxe : elle est correcte ou elle ne l'est pas, il n'y a rien à ajouter.

named-checkzone fait plus. Il charge vraiment la zone, l'analyse, et en tire des informations utiles. Sa sortie n'est donc pas un simple « tout va bien », c'est un compte rendu :

```

zone bts-sio.lan/IN: loaded serial 2026090801
OK

```

Q15. *Dans la sortie de dig @192.168.100.10 www.bts-sio.lan, relevez le status, les flags, le nombre d'enregistrements en ANSWER et la ligne SERVER. Expliquez la présence du drapeau aa, et son absence sur un nom externe.*

Élément	Valeur relevée
status	NOERROR
flags	qr aa rd ra
ANSWER	2 enregistrements
SERVER	192.168.100.10#53(192.168.100.10) (UDP)

Voici comment se lisent les quatre drapeaux. qr, pour query response, indique que c'est une réponse et pas une question. rd, recursion desired, est posé par le client : dig demande au serveur de chercher pour lui s'il ne sait pas. ra, recursion available, est la réponse du serveur : il accepte de le faire, ce qui correspond à mon recursion yes limité par allow-recursion. Et aa, authoritative answer, est celui qui nous intéresse.

Le drapeau aa est présent parce que ce serveur détient la zone bts-sio.lan. Sa réponse ne vient pas d'un cache ni d'un autre serveur, mais du fichier /etc/bind/db.bts-sio.lan chargé au démarrage. C'est lui qui détient l'information pour ce domaine, et il le dit.

Sur un nom externe comme debian.org, ce drapeau disparaît, parce que le rôle du serveur change. Il ne détient pas cette zone. Il transmet la question à un forwarder, reçoit la réponse et me la donne. Il ne fait que relayer, et une réponse recopiée ne peut pas être déclarée autoritative. Seul le serveur qui détient vraiment la zone debian.org peut poser ce drapeau.

Q16. *Pourquoi dig www.bts-sio.lan +short renvoie-t-il deux lignes alors que dig srv-linux.bts-sio.lan +short n'en renvoie qu'une ?*

www porte un CNAME, qui associe un alias à un autre nom, et pas à une adresse. Répondre seulement srv-linux.bts-sio.lan. ne donnerait pas au client ce qu'il cherche vraiment. Il devrait poser une seconde question pour obtenir l'adresse de ce nom. Le serveur suit donc l'alias tout seul et renvoie les deux enregistrements dans la même réponse :

L'option +short enlève tout l'habillage et garde juste les valeurs. D'où les deux lignes : d'abord la cible de l'alias, ensuite son adresse. Ce n'est donc pas une anomalie. C'est même la preuve que la chaîne d'alias a bien été suivie.

La réponse arrive, avec le statut NOERROR et quatre adresses. Pourtant mon serveur ne détient aucune information sur debian.org : il n'a chargé que la zone `bts-sio.lan` et les zones techniques de Debian.

Le drapeau aa est absent des deux réponses, contrairement aux questions sur ma propre zone. C'est logique : mon serveur ne fait pas autorité sur `debian.org`, il relaie seulement une réponse obtenue ailleurs.

Requête	Query time	Origine de la réponse
première	24 msec	interrogation réelle du forwarder 8.8.8.8
seconde	0 msec	cache local du serveur

Partie D. La zone inverse

[illegible]

```
@      IN      NS      ns1.bts-sio.lan.

; Le nom court est le DERNIER octet de l'adresse.
; La cible est un FQDN : point final obligatoire.
10     IN      PTR      srv-linux.bts-sio.lan.
50     IN      PTR      cli-linux.bts-sio.lan.
```

Sortie complète de l'interrogation :

```
sio@srv-linux:~$ dig @192.168.100.10 -x 192.168.100.50

;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 30461
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; QUESTION SECTION:
;50.100.168.192.in-addr.arpa.      IN      PTR

;; ANSWER SECTION:
50.100.168.192.in-addr.arpa. 604800 IN  PTR      cli-linux.bts-sio.lan.

;; Query time: 0 msec
;; SERVER: 192.168.100.10#53(192.168.100.10) (UDP)
```

La QUESTION SECTION est intéressante. J'ai tapé -x 192.168.100.50, mais la vraie question posée porte sur 50.100.168.192.in-addr.arpa, de type PTR. L'option -x est juste un raccourci de dig : elle retourne les octets et ajoute le suffixe à ma place. Le protocole, lui, n'interroge que des noms.

Le drapeau aa est présent : mon serveur fait autorité sur la zone inverse comme sur la zone directe.

Q19. La zone s'appelle 100.168.192.in-addr.arpa et non 192.168.100.in-addr.arpa. Expliquez pourquoi les octets sont inversés, en vous appuyant sur le sens de lecture d'un nom DNS et sur le mécanisme de délégation.

L'inversion vient du fait que les deux notations se lisent dans des sens opposés.

Une adresse IP se lit du plus général au plus précis, de gauche à droite. Dans 192.168.100.10, la partie réseau est à gauche et la machine à droite. Un nom DNS fait l'inverse : dans www.bts-sio.lan., la machine est à gauche et la racine à droite. Chaque niveau vers la droite est plus général que le précédent.

```
192 . 168 . 100 . 10
└── du plus general au plus precis ──>

www . bts-sio . lan .
<── du plus general au plus precis ──┘
```

Écrire l'adresse telle quelle sous in-addr.arpa mettrait donc la partie réseau du mauvais côté du nom, là où le DNS attend la partie la plus précise. Il faut la retourner pour que les deux sens se rejoignent.

C'est la délégation qui rend cette contrainte obligatoire. Le DNS délègue toujours de la droite vers la gauche : la racine délègue lan, qui délègue bts-sio, qui me laisse gérer ce qu'il y a en dessous. Pour qu'un opérateur puisse déléguer à son client le réseau 192.168.100.0/24, il faut que ce réseau soit la partie droite du nom, du côté de la racine. Les machines se retrouvent alors à gauche, du côté que le client gère librement.

Q20. Citez deux usages concrets de la résolution inverse en production, et expliquez pour chacun ce qui se passe si le PTR est absent ou incohérent.

Premier usage : le courrier électronique. Quand une connexion s'ouvre, presque tous les serveurs de messagerie regardent si l'adresse IP en face a un PTR. Ils vérifient aussi que le nom trouvé renvoie bien vers cette même adresse. Ce double contrôle s'appelle FCrDNS. Il est efficace parce qu'un

expéditeur légitime maîtrise ses adresses et peut y poser un PTR, alors qu'une machine piratée chez un particulier n'en a pas.

Si le PTR est absent, le message est marqué comme indésirable, retardé, ou refusé directement. Si le PTR ne colle pas, c'est souvent pire : un nom qui pointe ailleurs, ça ressemble à une machine qui se fait passer pour une autre. Le cas classique, c'est un serveur de messagerie tout neuf dont les messages n'arrivent jamais, sans erreur visible côté expéditeur, parce que le rejet a lieu chez le destinataire.

Deuxième usage : les contrôles d'accès par nom. Certains services autorisent un client par son nom plutôt que par son adresse. Ils doivent donc résoudre l'adresse entrante avant de décider. C'est le cas de NFS, que nous verrons au jour 4 : un partage exporté vers cli-linux.bts-sio.lan oblige le serveur à retrouver ce nom à partir de l'adresse du client.

Sans PTR, le serveur ne peut pas faire la correspondance et refuse l'accès. Avec un PTR qui ne colle pas, c'est plus embêtant : l'accès est donné à la mauvaise machine, ou refusé à la bonne. Et le message d'erreur parle de permissions alors que le problème est dans le DNS. On cherche alors longtemps du mauvais côté.

Partie E. Intégration DHCP et DNS côté client

Q21. D'où viennent exactement les deux lignes de `/etc/resolv.conf` sur cli-linux ? Nommez le fichier de configuration, la machine et les deux directives qui les produisent.

Le fichier obtenu sur le client est le suivant :

```
domain bts-sio.lan
search bts-sio.lan
nameserver 192.168.100.10
```

Rien n'a été saisi sur cli-linux. Tout vient du fichier `/etc/dhcp/dhcdp.conf`, sur la machine srv-linux, par deux directives.

Directive dans dhcdp.conf	Option DHCP	Ligne produite chez le client
option domain-name-servers 192.168.100.10;	6	nameserver 192.168.100.10
option domain-name "bts-sio.lan";	15	domain et search bts-sio.lan

Q22. ping www fonctionne alors que www n'est pas un nom complet. Détaillez la suite des opérations réalisées par le client, depuis la frappe de la commande jusqu'à la réponse du serveur.

Voici ce qui se passe, dans l'ordre.

1. ping demande au système de traduire www en adresse. Le système consulte d'abord `/etc/nsswitch.conf`, qui contient `hosts: files dns` : le fichier `/etc/hosts` est lu en premier, le DNS ensuite. `/etc/hosts` ne contient aucune entrée www, la recherche passe donc au DNS.
2. Le résolveur lit `/etc/resolv.conf`. Il y trouve `nameserver 192.168.100.10`, à qui poser la question, et `search bts-sio.lan`, le suffixe à essayer. Il interroge d'abord le nom tel quel : le serveur répond NXDOMAIN, car www n'existe pas à la racine du DNS. Il ajoute alors le suffixe et repose la question pour `www.bts-sio.lan`.
3. Le serveur, autoritatif sur `bts-sio.lan`, trouve un CNAME vers `srv-linux.bts-sio.lan`. Il suit l'alias lui-même et renvoie l'adresse `192.168.100.10`. ping peut alors envoyer ses paquets. Mais avant d'afficher quoi que ce soit, il fait une résolution inverse sur cette adresse pour la transformer en nom. D'où l'affichage `PING srv-linux.bts-sio.lan (192.168.100.10)`, qui n'est ni le nom tapé, ni une adresse brute.

Cette sortie toute courte fait donc marcher ensemble quatre choses montées séparément dans la journée : la ligne search venue du DHCP, la zone directe, l'alias CNAME et la zone inverse. Si l'un des quatre manquait, le symptôme serait différent. Sans search, échec net. Sans zone inverse, l'adresse s'afficherait à la place du nom. C'est donc un bon test pour vérifier que tout tient.

Q23. Vous éditez `/etc/resolv.conf` sur cli-linux pour y ajouter un second nameserver, puis vous relancez `dhclient`. Que devient votre modification, et où fallait-il intervenir ?

La modification disparaît complètement. Je l'ai vérifié en comparant le fichier avant et après le renouvellement du bail :

Avant	Après
domain bts-sio.lan	domain bts-sio.lan
search bts-sio.lan	search bts-sio.lan
nameserver 192.168.100.10	nameserver 192.168.100.10
nameserver 9.9.9.9	(ligne disparue)

La raison est simple. Sur une machine en DHCP, `/etc/resolv.conf` n'est pas un fichier de configuration : c'est un fichier de résultat. À chaque bail, `dhclient` appelle le script `/sbin/dhclient-script`, qui reconstruit ce fichier à partir des options reçues. Tout ce qui ne vient pas du serveur est écrasé, sans avertissement ni sauvegarde.

Il fallait donc corriger à la source, sur `srv-linux`, dans `/etc/dhcp/dhcpd.conf`. L'option 6 accepte une liste ordonnée : il suffit d'y ajouter le second serveur.

```
option domain-name-servers 192.168.100.10, 9.9.9.9;
```

Puis de redémarrer le service et de renouveler le bail côté client :

```
sio@srv-linux:~$ sudo dhcpd -t -cf /etc/dhcp/dhcpd.conf
sio@srv-linux:~$ sudo systemctl restart isc-dhcp-server
sio@cli-linux:~$ sudo dhclient -r ens33 ; sudo dhclient -v ens33
```

Partie F. Durcissement SSH

Q24. Combien de lignes contient `authorized_keys` après les deux dépôts, et pourquoi ? Quelles sont les permissions exactes de `~/.ssh` et de `~/.ssh/authorized_keys`, et que fait le serveur si elles sont trop larges ?

Le fichier contient deux lignes :

```
sio@srv-linux:~$ wc -l ~/.ssh/authorized_keys
2 /home/sio/.ssh/authorized_keys

sio@srv-linux:~$ cut -d" " -f3 ~/.ssh/authorized_keys
sio@cli-linux
dorianpachot@gmail.com
```

Une ligne par clé publique déposée, et chaque ligne autorise une origine. J'ai deux postes clients, cli-linux et le poste Windows, donc deux clés pour un seul compte sur le serveur. C'est le fonctionnement normal : on n'ajoute pas un compte par machine, on ajoute une clé par machine autorisée. Retirer l'accès d'un poste revient alors à supprimer une ligne, sans toucher aux autres ni changer de mot de passe.

Les permissions relevées sont les suivantes :

Chemin	Octal	Affichage
/home/sio	700	drwx-----
/home/sio/.ssh	700	drwx-----
/home/sio/.ssh/authorized_keys	600	-rw-----

Le dossier .ssh doit être en 700 et le fichier en 600 : accessibles au seul propriétaire. Le dossier personnel ne doit pas être accessible en écriture au groupe ni aux autres. 755 ou 750 conviennent ; le mien est en 700.

Si ces permissions sont trop larges, le serveur refuse la clé, et il le fait sans rien dire. Il n'envoie aucune explication au client : il écarte cette méthode et propose la suivante, en général le mot de passe. Côté utilisateur, on ne comprend pas ce qui se passe. La clé est bien déposée, elle est correcte, et le serveur redemande quand même un mot de passe.

L'explication figure en clair côté serveur, dans /var/log/auth.log :

```
Authentication refused: bad ownership or modes for directory /home/sio/.ssh
```

Q25. Recopiez les messages d'erreur des deux dernières commandes. Expliquez ce qui les distingue : dans un cas le compte est refusé, dans l'autre c'est la méthode d'authentification.

Les deux messages obtenus :

```
PS C:\Users\doria> ssh root@192.168.100.10
root@192.168.100.10: Permission denied (publickey).

sio@srv-linux:~$ ssh -o PubkeyAuthentication=no sio@192.168.100.10
ssh_dispatch_run_fatal: Connection to 192.168.100.10 port 22: Broken pipe
```

Ils sont différents parce que les deux échecs n'arrivent pas au même moment de l'échange.

Dans le premier cas, le client a une clé et la propose. Le serveur accepte cette méthode, donc l'échange va jusqu'à la tentative d'authentification. C'est le compte qui est refusé : root n'est pas dans AllowUsers. Le client reçoit un vrai refus, Permission denied, et la parenthèse (publickey) liste les méthodes que le serveur accepterait encore. Le rejet porte sur qui demande à entrer.

Dans le second cas, le compte sio est valide et autorisé. Mais j'ai désactivé la clé côté client, et le serveur refuse le mot de passe. Il ne reste donc aucune méthode commune aux deux. Aucune tentative d'authentification n'a lieu, l'échange s'arrête avant, et le serveur ferme la connexion. Le client ne reçoit pas un refus mais une coupure, d'où ce message bizarre. Le rejet porte sur comment on demande à entrer.

La distinction n'est lisible que côté serveur, dans le journal, où les deux cas donnent deux lignes bien claires :

```
User root from 192.168.100.50 not allowed because not listed in AllowUsers
Connection closed by authenticating user sio 192.168.100.50 port 45304 [preauth]
```

La première ligne donne le motif et la directive. La seconde parle d'un authenticating user, et pas d'un invalid user : le compte est reconnu, seule la méthode a manqué. C'est la règle quand on dépanne du SSH. Le message du client dit qu'il y a un problème, le journal du serveur dit lequel.

Q26. Recopiez trois lignes différentes de /var/log/auth.log correspondant à ces tentatives et expliquez chacune. Laquelle prouve que la directive AllowUsers est bien active ?

Le fichier /var/log/auth.log n'existe pas sur cette installation, parce que Debian 12 minimale n'installe plus rsyslog. J'ai relevé les mêmes messages dans le journal de systemd, avec sudo journalctl -u ssh -f. Le contenu est identique.

Première ligne, une connexion réussie :

```
sshd[1668]: Accepted publickey for sio from 192.168.100.1 port 50226 ssh2:
ED25519 SHA256:fwRImSnax132SXIPW+R7BxE4PMv5IN0+W0+tPtuhl9o
```

Le mot Accepted indique la réussite, et publickey la méthode employée, donc pas un mot de passe. L'empreinte de la clé utilisée est notée dans le journal. Sur un serveur qui autorise plusieurs clés, ça

permet de savoir laquelle a servi, donc depuis quel poste vient la connexion. Ici, l'adresse 192.168.100.1 montre que c'est le poste Windows.

Deuxième ligne, un compte existant mais non autorisé :

```
sshd[1690]: User root from 192.168.100.50 not allowed because not listed in AllowUsers
```

C'est cette ligne qui prouve que la directive AllowUsers est active, et elle le dit clairement. Le compte root existe bien sur la machine, c'est même lui qui l'administre. Mais il n'est pas dans ma liste blanche, et le serveur s'arrête là.

```
sshd[1692]: Invalid user admin from 192.168.100.50 port 45302
```

Le terme Invalid user veut dire que le compte n'existe pas du tout sur la machine. C'est la marque d'un robot d'attaque, qui essaie les noms les plus courants : root, admin, test, ubuntu. Il ne sait pas ce qui existe sur la machine. Compter ces lignes donne une idée du nombre d'attaques : `sudo journalctl -u ssh | grep -c "Invalid user"` donne le nombre de tentatives. Si le résultat se compte en milliers, la machine est attaquée.

Il y a une quatrième ligne, celle de mon troisième test :

```
sshd[1694]: Connection closed by authenticating user sio 192.168.100.50 port 45304 [preauth]
```

Le compte sio est valide et autorisé : le message parle d'un authenticating user, pas d'un invalid user. Mais aucune méthode d'authentification commune n'a abouti, puisque j'avais désactivé la clé côté client et que le mot de passe est refusé côté serveur. La mention [preauth] veut dire que la connexion s'est terminée avant toute authentification.

Q27. *MaxAuthTries 3 est en place. Décrivez ce qui se passe côté client au bout de trois échecs, et expliquez pourquoi cette limite complique le travail d'un robot d'attaque sans le rendre impossible. Quel outil, vu au jour 4, complète cette protection ?*

Au bout de trois essais ratés sur une même connexion, le serveur coupe. Le client reçoit un message du type Too many authentication failures et la session se termine. Il revient à son invite locale sans avoir obtenu d'accès. Le journal du serveur garde la trace.

Pour un robot d'attaque, cette limite ne bloque rien, elle coûte cher. Sans elle, il pourrait enchaîner des centaines de mots de passe dans une seule session TCP. Avec elle, il doit rouvrir une connexion complète toutes les trois tentatives, donc refaire une poignée de main TCP puis tout l'échange de clés. C'est bien plus lourd que l'essai lui-même. Son débit s'effondre, et chaque nouvelle connexion laisse une ligne dans le journal, ce qui le rend visible.

L'outil qui complète cette protection est fail2ban, au programme du jour 4. Il marche autrement. Il surveille le journal d'authentification, compte les échecs par adresse IP, et bannit temporairement au pare-feu celles qui dépassent un seuil. MaxAuthTries limite ce qu'on peut faire dans une connexion ; fail2ban empêche d'en ouvrir de nouvelles.

Partie G. Panne à diagnostiquer

Q28. *Pour chacune des trois fautes, indiquez dans un tableau : le symptôme observé, la commande qui vous a mis sur la piste, le message exact obtenu, le fichier et la ligne fautifs, et la correction appliquée. Précisez pour la faute qui ne produit aucune erreur comment vous l'avez tout de même identifiée.*

Faute	Symptôme observé	Message exact obtenu	Fichier et ligne	Correction
Point-virgule manquant	isc-dhcp-server en échec (failed) au	/etc/dhcp/dhcpd.conf line 27: semicolon expected.	/etc/dhcp/dhcpd.conf, ligne	Point-virgule remis en fin de

	démarrage		26 (option routers)	ligne, puis dhcpd -t et restart
Accolade non fermée	named en échec, redémarrages répétés puis abandon	/etc/bind/named.conf:12: '}' expected near end of file	/etc/bind/named.conf.local, dernière ligne du bloc de zone inverse	Accolade }; rajoutée, puis named-checkconf et restart
Serial non incrémenté	Aucun. Les deux services tournent, tous les contrôles répondent OK	zone bts-sio.lan/IN: zone serial (2026090801) unchanged. zone may fail to transfer to secondaries.	/etc/bind/db.bts-sio.lan, ligne 7 (le serial), après un ajout ligne 20	Serial passé de 2026090801 à 2026090802, puis rndc reload

La commande qui m'a mis sur la piste est différente pour chacune : `sudo dhcpd -t -cf` pour la première, `sudo named-checkconf` pour la deuxième, et la lecture complète du journal de named pour la troisième.

Je m'arrête sur la deuxième faute. Le message accuse `/etc/bind/named.conf` et `/etc/bind/named.conf.default-zones`, deux fichiers livrés par Debian que je n'ai jamais ouverts. Le fichier nommé dans l'erreur n'est donc pas le fichier fautif. L'explication tient à l'ordre des inclusions : `named.conf` inclut d'abord `named.conf.options`, puis `named.conf.local`, puis `named.conf.default-zones`. Une accolade jamais fermée dans mon fichier fait croire à l'analyseur qu'il est encore dans un bloc quand il commence le suivant, où le mot `zone` n'a alors plus de sens. Le coupable est le fichier inclus juste avant celui qui proteste.

La troisième faute est celle qui ne produit aucune erreur, et voici comment je l'ai tout de même trouvée. Les deux services étaient repartis, `named-checkconf` restait muet et les deux `named-checkzone` répondaient OK : de ce côté, tout était propre. Mais l'énoncé annonçait trois fautes et je n'en avais que deux. J'ai donc relu le journal de named en entier, sans filtrer sur le mot `error`, et j'y ai trouvé un avertissement : le numéro de série de la zone n'avait pas changé alors que le fichier avait été modifié. En relisant le fichier, j'ai découvert un enregistrement imprimante que je n'avais jamais écrit, et un serial resté à 2026090801.

Cette faute est la plus dangereuse des trois sur un vrai serveur. Sur ce serveur, tout fonctionne : la modification est bien chargée et servie. Mais un serveur secondaire ne compare que les numéros de série. Voyant le même numéro qu'avant, il en conclut que rien n'a changé et ne demande aucun transfert. Il continue donc de servir l'ancienne version de la zone, indéfiniment. Le résultat est une incohérence entre deux serveurs qui se disent tous les deux autoritatifs, et rien ne prévient.

5. Fichiers de configuration

`/etc/network/interfaces` (sur `srv-linux`)

```
# 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

# ens32 : carte NAT, adresse obtenue automatiquement (accès aux dépôts APT).
# C'est elle qui fournit la route par défaut vers Internet.
auto ens32
iface ens32 inet dhcp

# ens33 : réseau pédagogique VMnet2, adresse fixe.
```

```
# Pas de ligne "gateway" ici : la route par default vient deja d'ens32.
auto ens33
iface ens33 inet static
    address 192.168.100.10
    netmask 255.255.255.0
    dns-nameservers 192.168.100.10
    dns-search bts-sio.lan
```

[/etc/network/interfaces \(sur cli-linux\)](#)

```
source /etc/network/interfaces.d/*

auto lo
iface lo inet loopback

# La carte NAT n'est PAS declaree "auto" : elle reste eteinte, pour que
# le client ne recoive ni route par default ni DNS venant du NAT VMware.
# iface ens32 inet dhcp

# Reseau pedagogique VMnet2, en DHCP : c'est notre serveur qui repondra.
auto ens33
iface ens33 inet dhcp
```

[/etc/default/isc-dhcp-server](#)

Hors commentaires, le fichier se réduit à deux lignes :

```
INTERFACESv4="ens33"
INTERFACESv6=""
```

[/etc/dhcp/dhcpd.conf](#)

Recopié en réponse à la question 7.

[/etc/bind/named.conf.options](#)

Recopié en réponse à la question 12.

[/etc/bind/named.conf.local](#)

```
//
// Zones du reseau pedagogique BTS SIO
//

// Zone directe : nom -> adresse
zone "bts-sio.lan" {
    type master;
    file "/etc/bind/db.bts-sio.lan";
};

// Zone inverse : adresse -> nom, pour 192.168.100.0/24
zone "100.168.192.in-addr.arpa" {
    type master;
    file "/etc/bind/db.192.168.100";
};
```

[/etc/bind/db.bts-sio.lan](#)

Recopié en réponse à la question 13.

[/etc/bind/db.192.168.100](#)

Recopié en réponse à la question 18.

/etc/ssh/sshd_config (extrait durci)

```
Include /etc/ssh/sshd_config.d/*.conf
Port 22

PermitRootLogin no
AllowUsers sio

PubkeyAuthentication yes
PasswordAuthentication no
PermitEmptyPasswords no
KbdInteractiveAuthentication no
UsePAM yes

MaxAuthTries 3
LoginGraceTime 30
X11Forwarding no
ClientAliveInterval 300
ClientAliveCountMax 2

PrintMotd no
AcceptEnv LANG LC_*
Subsystem      sftp    /usr/lib/openssh/sftp-server
```

6. Difficultés rencontrées

Une interface déclarée deux fois dans /etc/network/interfaces

En écrivant le fichier à l'étape 3, j'ai tapé auto ens33 au-dessus de la ligne iface ens32 inet dhcp, au lieu de auto ens32. Le fichier déclarait donc ens33 deux fois en auto, et ens32 plus du tout.

La syntaxe du fichier restait bonne : aucune erreur ne serait apparue. Mais la carte NAT n'aurait plus été montée au démarrage. J'aurais perdu l'accès aux dépôts au prochain redémarrage, sans aucun message pour m'expliquer pourquoi.

Je l'ai vue en relisant le fichier avant d'appliquer, avec `grep -n "^auto\|^iface"`. La correction a demandé de changer un seul caractère. Ce que j'en retiens : une faute de configuration ne produit pas toujours une erreur, et relire les lignes qui comptent prend dix secondes.

Le nom complet oublié dans /etc/hosts du clone

En rendant cli-linux indépendant, j'ai remplacé srv-linux par cli-linux sur la ligne 127.0.1.1 de /etc/hosts, mais seulement le nom court. La ligne est restée un moment comme ça :

```
127.0.1.1    srv-linux.bts-sio.lan cli-linux
```

Aucune erreur n'apparaît, et la machine marche. Mais /etc/nsswitch.conf contient hosts: files dns : le fichier hosts est lu avant le DNS, et sa réponse gagne. Le client aurait donc cru que srv-linux.bts-sio.lan se trouve en 127.0.1.1, donc sur lui-même.

C'est le genre de panne où on cherche longtemps du mauvais côté. dig srv-linux.bts-sio.lan aurait répondu correctement 192.168.100.10, parce que dig interroge le serveur DNS sans passer par nsswitch. Mais ping et ssh seraient partis sur la boucle locale. J'aurais cherché la panne dans bind9 alors que mon DNS était bien configuré.

J'ai corrigé en écrivant cli-linux.bts-sio.lan cli-linux. Ce que j'en retiens : quand dig et ping ne sont pas d'accord, le coupable est /etc/hosts, pas le serveur DNS.

7. Conclusion

Les deux machines marchent et tout ce qui était demandé est en place. srv-linux distribue les adresses du réseau 192.168.100.0/24, avec une plage dynamique et une réservation par adresse MAC pour cli-linux. Il résout les noms du domaine bts-sio.lan dans les deux sens, et transmet le reste à des résolveurs publics. Le client reçoit son adresse, sa passerelle et son DNS sans aucune configuration locale, et il joint les autres machines par leur nom court. L'accès SSH au serveur ne se fait plus que par clé, pour le seul compte sio.

Les trois fautes de la partie G ont été retrouvées et corrigées, et la vérification finale repasse au vert. Les deux instantanés J2 depart et J2 avant panne sont conservés, et les deux machines restent en état pour les journées suivantes.

Ce que je retiens de ce TP tient en trois points. D'abord, un nom d'interface ne se recopie jamais depuis un énoncé : chez moi les cartes s'appelaient ens32 et ens33, et transposer ces noms partout était la condition pour que tout le reste fonctionne. Ensuite, une faute de configuration ne produit pas toujours une erreur : le auto ens33 écrit deux fois, le nom oublié dans /etc/hosts et le serial non incrémenté auraient tous les trois laissé la machine dans un état apparemment sain. Enfin, le fichier nommé dans un message d'erreur n'est pas forcément le fichier fautif, comme l'a montré l'accolade manquante accusée sur un fichier de Debian.