PARIS ynov CAMPUS

YDAYS 2024-2025

Rapport de projet : Audit Linux 2

“AUDIX”

Chef de projet :

JRIDI Dine

Encadrant:

NTUMBA Marc

Table des matières

1 - Explication du projet

Ce projet a pour vocation de mettre en place un système minimaliste pour une tâche bien précise, ainsi qu'une explication de ce système, sur la manière dont il est plus résistant qu'un système d'exploitation classique.

Il est ici question d'un système minimaliste, mais que signifie-t-il exactement ?
Il s'agit d'un système d'exploitation (ex : Windows, Mac, Linux) ne comportant que les services essentiels, qui serviront au bon fonctionnement d'une tâche précise et définie au préalable...
Effectivement, dans nos vies personnelles, nous n'utilisons pas toutes les fonctionnalités mises à disposition par le système d'exploitation.
Le but de ce projet est de créer une version personnalisée (que l'on désignera comme distribution) d'un système d'exploitation (qu'on appellera OS pour "Operating System") afin d'éviter les installations superflues.
Concernant la défense de l'OS, on s'intéressera aux flux de données entrants et sortants, en particulier les trames réseau, en les comparants avec d'autres OS plus complexes. Dans l'optique de remarquer les différences et surtout de lister les différents points d'attaques possibles.

2 - Problématique

Les méthodes pour mener à bien ce projet sont légion, mais laquelle choisir ? Quelle tâche attribuer à la distribution ? Sur quel support informatique devrait-on porter le projet ? Comment assurer sa portabilité et sa distribution ?
Énormément de questions se posaient, et leurs résolutions n'allaient pas être évidentes. Le projet a nécessité une approche progressive, avec parfois des ajustements pour améliorer la démarche. En somme, ce fut une tâche ardue.
Dans un premier temps, il était essentiel de définir les objectifs de cette distribution. Après mûres réflexions, l'idée retenue était la suivante : "Permettre à Monsieur et Madame tout le monde de pouvoir jouir des fonctionnalités de base".
Pour respecter cette idée il faudra que le projet respecte plusieurs points :

- **Environnement de bureau** : qui permettra à l'utilisateur d'avoir un retour visuel sur ses actions, avec tout ce que cela implique…. horodatage, gestionnaire de fichiers etc.

- **Navigateur web** : Accès à internet

- **Logiciel de traitement de texte** : Édition et modification de texte

Une fois cette problématique résolue, quel support de travail fallait-il déployer pour le projet ? Une machine virtuelle ? Une machine physique ?
Les deux options comportent des avantages et inconvénients, commençons par la machine virtuelle.
Une machine virtuelle (on parlera de VM pour Virtual Machine) est une émulation (simulation) d'un ordinateur sur notre propre ordinateur. Elle utilisera donc les ressources matérielles de l'ordinateur physique (inconvénient), ce qui peut être très handicapant car on sera obligé de partager la puissance de l'ordinateur (ram, stockage, cpu). La VM étant un ordinateur à part entière, est complètement indépendante du système hôte (avantage) ce qui empêche toutes interactions entre elle et l'hôte. Évitant ainsi, toutes corruptions du système. Il est possible de créer plusieurs VM (avantage), permettant de disposer de points de sauvegarde qui serviront au besoin pour revenir à une ou plusieurs versions précédentes.

Une machine physique, n'est pas une émulation, c'est un ordinateur que l'on peut toucher directement (d'où le mot physique...), il utilisera ses propres ressources matérielles (avantage) mais en contrepartie si l'un problème survient l'ordinateur peut-être inutilisable (inconvénient).

Pour un projet comme celui-ci et même en général, le choix de la VM est souvent celui de la sécurité et de la prudence. Personnellement, j'ai choisi d'allier les deux ensembles. En effet, en utilisant une machine physique sur laquelle on se connecte depuis une autre machine physique, (via SSH) cela permet de pouvoir utiliser la puissance de la machine sans avoir à la partager (avantage machine physique) tout en étant protégé si le système se casse (avantage VM).

3 - OS minimal

Depuis le début, le mot distribution revient souvent mais qu'est-ce qui la caractérise ? Il est crucial de bien comprendre cette notion pour pouvoir avancer dans la construction d' une distribution conforme à cette définition. Linux est composé de plusieurs éléments :

- **Le chargeur d'amorçage** : Cette partie du logiciel gère le mode de démarrage de l'ordinateur. La plupart du temps, cela est considéré par les utilisateurs comme l'écran de démarrage qui apparaît avant de céder lorsque le système d'exploitation prend le relais.

- **Le noyau** : Le noyau est le niveau le plus élémentaire du système d'exploitation. C'est également ce que l'on appelle « Linux ». Le noyau est l'élément central du système d'exploitation, il gère le processeur, la mémoire et les périphériques.

- **Shell** : À l'instar de l'invite DOS présent dans le système d'exploitation Windows, il s'agit

du processus de commande qui permet à un utilisateur de contrôler l'ordinateur en saisissant des commandes tapées dans une interface texte. Celui-ci est hébergé par le terminal dans Ubuntu.

- **Processus** : Les services en arrière-plan démarrent soit au démarrage du système, soit dès que vous y avez accès.
- **Serveur graphique** : Sous-système qui affiche les graphismes sur votre écran.
- **Environnement de bureau** : La partie avec laquelle nous interagissons normalement. Cela inclut les applications intégrées.
- **Logiciels additionnels** : Ensemble des logiciels installés pour le plein fonctionnement de la distribution

4 - Net install

Pour créer notre propre distribution, deux choix se sont donc distingués. Utiliser une distribution déjà existante et installer son image ISO, ou, partir de rien et faire du LFS (Linux From Scratch). Pour rendre LFS possible, il fallait impérativement qu'un OS soit déjà présent sur la machine. LFS consiste à, comme son nom l'indique, créer un environnement Linux en partant de rien (**cf. 1**).
Pour être tout à fait honnête, un de mes premiers choix se tourna vers LFS, mais en atteignant la partie sur le chroot (changement de racine du système), j'ai décidé de me tourner vers une distribution déjà existante. Ce choix est la conséquence directe du manque de membres dans le projet, qui m'oblige à prendre des précautions afin de ne pas risquer de mettre en péril le projet pour faute de temps.

Il était nécessaire de bien choisir la distribution qui servira de base pour l'ISO finale. Deux choix se proposaient naturellement, le premier, prendre une distribution "générale" et retirer des fonctionnalités bout par bout. Le système protège ses paquets essentiels, rendant la minimalisation partielle, et donc cette méthode impossible.
En revanche, l'autre choix, qui quant à lui est réalisable, correspond davantage aux attentes du projet. Il s'agit de prendre une image comprenant uniquement les outils de base d'une distribution, puis d'y ajouter les logiciels choisis pour obtenir notre propre distribution. Le choix du support d'installation s'est tourné vers l'image ISO Debian netinst amd64 (**cf. 2**). La distribution liée à cette image est Bookworm qui est la version stable de Debian. L'intérêt

d’utiliser une telle distribution est qu’elle permet d'avoir une base Debian pour pouvoir construire une image ISO de type Debian, ce qui permettra in fine de pouvoir la flasher sur une clef USB pour la rendre portable et bootable.

5 - Installation Bookworm

Nous avons entre entre les mains, avec une clef flashée comportant Bookworm (**cf. “Manuel d’utilisation”** pour flasher une clef USB), il est temps maintenant de voir ce que comporte cette clef… Une fois dans le BIOS du système, on va boot sur la clef et lancer l’ISO. Plusieurs questions aussi diverses que variées nous sont proposées, les choses à retenir ici sont que :

- Installation terminal
- Installation UNIQUEMENT de SSH
- Mettre en place les différents mot de passe et comptes

Prenons chaque point un par un.
L’installation en mode terminal (ou l’absence d’installation graphique) est là juste dans un souci de minimalisation. Indéniablement, le graphique n’est en aucun cas utile.
L’installation de SSH (**cf. 3**) n’est pas utile en elle-même (il sera d’ailleurs supprimé ultérieurement), sa plus-value se dégage de son utilisation. Il offre la possibilité de se connecter à une autre machine physique, pour effectuer toutes les batteries de test à réaliser (il sera d’ailleurs coupler à la commande “screen” **cf. 4**), dans le but d’avoir en produit final une ISO saine, fait de SSH un outil indispensable.

La mise en place des utilisateurs est cruciale pour le bon fonctionnement du système, dans les étapes suivantes, il sera obligatoire d'avoir des droits de super-utilisateur (admin) afin de passer outre certaines interdiction que pourrait rencontrer un utilisateur lambda.

6 - Justification

Il existe une multitude de paquets susceptibles de répondre aux besoins du projet. Il fallait impérativement trouver un moyen de trier et éliminer cette masse de paquets.
Pour répondre à cette problématique, il est important de comprendre ce qui est demandé. Peu importe quels choix sont faits, il faudra rendre dans tous les cas une image ISO pour prouver sa portabilité.
Par souci de cohérence, la liste des paquets choisis, a elle aussi dû répondre à cette demande.

Prenons le cas de l'environnement de bureau LXDE, qui ne possède que 3 dépendances (**cf. 5**) et son sigle signifie "Lightweight X11 Desktop Environment", ce qui en fait un élément de choix pour sa légèreté.
Pour ce qui est d'Abiword (qui sera utilisé comme logiciel de traitement de texte) lui aussi est un logiciel léger, il est notamment utilisé lors de la mise en place d'une distribution LFS (**cf. 6**, **cf. 7**).
À l'instar des autres, le navigateur web a nécessité de longues réflexions pour trouver lequel se démarquera plus des autres. Le paquet LXDE-core qui est donc le bureau graphique, recommandait d'installer Firefox.
Dans un souci de recommandation et de bon fonctionnement, le choix s'est naturellement porté sur Firefox. Bien que le navigateur soit installé, cela ne suffira pas à lui permettre d'accéder à internet via le wifi. À contrario des autres paquets, il aura besoin d'installer des paquets supplémentaires (qui ne font pas partie de ses dépendances), les "network-manager" (**cf. 8**). Network-manager, est le paquet installé par défaut dans la plupart des distributions, il faut donc l'ajouter à la nôtre. Network-manager a besoin d'une interface pour marcher correctement, ici le choix s'est porté sur network-manager-gnome vu qu'il est recommandé par Ubuntu et un environnement très utilisé dans Debian en général.

Depuis le commencement, la recherche a toujours été au cœur du projet, que ça soit dans un but informatif ou pour régler des problématiques, la recherche a toujours fait partie de ce projet. Elle en représente à elle seule plus de 90 %. En toute logique, cette partie du travail à elle aussi sa place dans ce rapport.

Distinguons bien que la recherche peut-être le meilleur et le pire des atouts selon la façon dont on l'utilise, prenons par exemple l'environnement de bureau. Le rendu graphique est une étape clef du projet, choisir le bon n'est pas une mince affaire. En effet, certains ne sont pas compatibles, d'autres trop compliqués à mettre en place, ou cumulant les deux problèmes ! Finalement, il aura fallu plusieurs tests d'environnement de bureau (Enlightenment, Xmonad, KDE, Gnome et enfin LXDE) pour enfin en trouver un qui correspondait aux attentes du projet. Tout cela n'a été possible qu'après un travail de recherche minutieux, accompagné d'une batterie de tests (et d'échecs), ce qui a permis à la recherche de débloquer cette situation (point positif).
Malheureusement tout n'est pas aussi simple, étant loin d'être un expert en la matière, il m'est arrivé plusieurs fois de prendre de mauvaises directions et de devoir revenir en arrière de plusieurs heures voir jours de travail. L'exemple qui est le plus flagrant est LFS. LFS est un projet extrêmement chronophage, l'inclure dans ce projet n'était pas clairement pas le meilleur des choix à faire (bien qu'il m'a énormément appris). Entre, les problèmes de résolutions, la difficulté de compréhension, s'approprier la logique du contenu et j'en passe... En arrivant sur la partie du Chroot, c'est à ce moment là que je me suis rendu compte que la tâche était gigantesque, et que j'allais peut-être dans la mauvaise direction. à cause de mes recherches (et de mon manque d'évaluation de la difficulté), j'ai "perdu" énormément de temps sur cette partie

là.

Tous ces exemples ne sont que quelques-uns parmi tant d'autres (on pourrait aussi parler notamment du pilote graphique qui a été aussi une vraie calamité), au final, peu importe la fonctionnalité, il était impossible de choisir "bêtement" un paquet sans le comparer. On peut donc extrapoler cette démarche à l'entièreté du projet et remarquer à quel point ça a pu être long et fastidieux de tout vérifier et comparer.

7 - Live-build

La dernière pièce et non des moindres de ce projet est la construction d'une image ISO de notre distribution. Réciproquement, que ce soit des paquets, logiciels ou des fichiers ISO, il existe toujours plusieurs méthodes pour arriver à ses fins. Il est par conséquent essentiel d'avoir un critère de sélection permettant de choisir une méthode par rapport à une autre. Et c'est à ce moment-là que l'essence du projet intervient. Car oui, dans un premier plan comme précisé dans l'introduction, le but est de permettre à des utilisateurs d'avoir accès au stricte minimum, mais soyons honnêtes, qui serait intéressé par une telle distribution ? Les retraités et néophytes sont souvent ceux qui bénéficieraient le plus d'une telle distribution. La plupart de ces personnes là possèdent des ordinateurs qui sont complètement dépassés. Permettre à ces vieux ordinateurs de pouvoir être fonctionnels à nouveau et d'avoir une seconde vie constitue un enjeu majeur pour ce projet. C'est là qu'intervient notre critère de sélection, il faut donc optimiser la démarche pour que peu importe le système, peu importe la personne, tout soit fait de façon à ce qu'en dépit du cadre, l'image ISO soit faite automatiquement. C'est avec cette volonté que, durant les recherches, la méthode retenue a été le live-build.

Le live-build permet une automatisation des processus et de créer une image ISO en personnalisant les différentes composantes de celle-ci. Il est en revanche nécessaire d'ajouter un fichier contenant les paquets à installer mais ensuite, rien n'est obligatoire. Bien évidemment plusieurs modifications ont été mis en place afin d'avoir l'installation la plus optimale et agréable possible (**cf. Manuel d'utilisation**).
Poussons l'automatisation encore plus loin. Le live-build implique de savoir : les commandes; aller dans les bons dossiers et fichiers; gérer son cache. Globalement, ce n'est pas le plus simple, surtout que la documentation dessus n'aide en rien (**cf. 9**). C'est pourquoi j'ai pris la peine d'ajouter un script.sh qui permet d'automatiser le live-build. Cette automatisation est là pour simplifier la tâche des utilisateurs pour que tout soit lancé en une seule et unique commande. Cette centralisation du code permet ainsi une meilleure visualisation et compréhension des paquets installés.

8 - Audit

Pour illustrer la pertinence de cette distribution, nous la comparons avec une distribution “générale”. La comparaison sera faite entre Mint (Linux Mint 22.1) et la distribution de l’audit que l’on appellera “AUDIX”. Toutes deux ont été installées en live évitant ainsi l’écriture des logs ou la journalisation, permettant de maintenir cette approche de minimalisation. Le parallèle entre ces deux distributions se fera sur 5 points clefs. Le nombre de paquets, les services actifs au démarrage, l’environnement graphique, la taille occupée sur le disque et pour finir l’étude des trames réseaux

En premier lieu, nous avons choisi le nombre total de paquets comme critère de comparaison. Ce critère permet notamment de mettre en évidence l’impact d’une distribution “générale” face à celle d’une distribution “dédiée”. Grâce à la commande

$ ls /var/lib/dpkg/info/*.list | wc -l

Le système nous affiche le nombre de paquets qu’il a dû installer pour son bon fonctionnement. AUDIX possède un peu plus de 800 paquets (816 **cf. nombre_paquets**), quant à lui, Mint en a plus du double (1987 **cf. nombre_paquetsLM**). Cet écart considérable met en évidence l’impact qu’a la généralisation d’une distribution sur son nombre de paquets.

On vient de voir le nombre de paquets installés sur les deux distributions, enchaînons maintenant sur les services actifs au démarrage du système. Pour éviter de biaiser les résultats, après le démarrage sur le live, on exécute directement la commande suivante

$ ps -x | wc -l

La commande nous donne les chiffres suivants : 33 services (**cf. services_actifs**) pour AUDIX et 63 (**cf. services_actifsLM**) pour Mint. Encore une fois Mint a un nombre deux fois plus élevé qu’AUDIX, ce résultat s’inscrit dans la continuité du précédent.

Nous en avons maintenant terminé avec l’analyse des paquets et des services. Passons à la partie graphique pour en examiner les résultats. Bien qu’il ait été mentionné à plusieurs reprises, l’environnement graphique d’AUDIX est LXDE, tandis que celui de Mint est Cinnamon. Deux aspects principaux seront mis en avant. Leur consommation, et l’intérêt de leur mise en service.
LXDE n’utilise que très peu de RAM, ce qui implique qu’il est un environnement graphique ultra-léger (moins de 200 Mo RAM **cf. 12**). En revanche, Cinnamon nécessite 400 à 800 Mo de RAM, soit 2 à 4 fois plus.
Cette augmentation de la consommation de ressources se définit par l’idée derrière la création de ces environnements. LXDE est avant tout conçu pour de vieux PC et/ou à faibles ressources. Au contraire Cinnamon recherche de la performance et du confort. Le public visé n’étant pas le même, il est logique que la consommation soit si différente. À titre informatif,

LXDE, qui est basé sur Openbox est un environnement léger, ne consommant qu'environ 1 % du CPU. Cinnamon, un environnement beaucoup plus complet et ergonomique, nécessite davantage de ressources pour fonctionner correctement. On estime ses besoins entre 2 et 5 % du CPU.

Les parties précédentes ont pu mettre l'accent sur l'intérêt de la minimalisation. Examinons maintenant l'élément final pour confirmer la véracité des données. L'élément final pour comparer ces deux distributions est la taille sur le disque. Un moyen d'obtenir l'espace occupé est la commande

$ df -h

En sortie on obtient qu'AUDIX ne prend que 765 Mo (**cf. taille_disque**) contre 2,8 Go (**cf. taille_disqueLM**) pour Mint. Il aurait été surprenant que Mint soit plus léger qu'AUDIX, mais une telle différence est d'autant plus marquante lorsqu'on considère les points ultérieurs.

L'analyse des trames réseau s'appuiera sur deux images (**cf. wireshark_audix**, **cf. wireshark_mint**). L'objectif est de déterminer les différences réseau entre l'installation d'une distribution "générale" et celle d'AUDIX, voici un tableau récapitulatif des cents premiers protocoles appelés :

	AUDIX	MINT
ARP	98	42
ICMPv6	2	40
MDNS	0	14
IGMPv3	0	2
NTP	0	2
Temps d'exécution	106s	47s

Après une brève analyse, nous constatons qu'AUDIX ne réalise que très peu d'appels de protocole. Sur une centaine d'appels, la plupart sont des protocoles de communications internes (ARP) et le peu qui reste sont des protocoles de gestion de réseau. Pour le moment, cette information n'est pas très pertinente, mais en la comparant avec la trame de Mint, il sera possible d'en tirer des conclusions. Contrairement à AUDIX, la trame de Mint est remplie de protocoles. Ces protocoles sont réparties en cinq types :

- **ARP (Address Resolution Protocol)** : Protocole qui associe une adresse IP à une adresse MAC afin de permettre la communication dans le réseau local
- **ICMP (Internet Control Message Protocol)** : Protocole utilisé pour la gestion d'erreur
- **MDNS (Multicast Domain Name System)** : Protocole utilisé pour la résolution des noms dans les petits réseaux
- **IGMP (Internet Group Management Protocol)** : Protocole qui partage une même adresse IP à tous les appareils du réseau
- **NTP (Network Time Protocol)** : Protocole qui synchronise l'horloge locale

Nombreux sont les protocoles qui communiquent avec le réseau. Par exemple, NTP, qui permet la synchronisation de l'heure avec un serveur distant pour mettre à jour l'horloge locale ou encore IGMP qui permet que tout appareil communiquant avec le même réseau partage la même adresse IP.
Parmi ces protocoles, ARP est le seul à ne pas communiquer avec l'extérieur.
En faisant une brève comparaison, Mint appelle énormément de protocoles qui interagissent avec le réseau (interne et externe) mais surtout il fait des appels beaucoup plus fréquemment que AUDIX. 47 s pour Mint, tandis que AUDIX fait le même nombre d'appels en 106s. À titre de comparaison, si on utilise un produit en croix, AUDIX fait seulement 44 appels de protocole pour 47 s. AUDIX effectue deux fois moins d'appels de protocoles que Mint, et se distingue par sa faible communication avec le réseau externe.
Il est raisonnable de conclure qu'AUDIX présente potentiellement une meilleure sécurité que Mint, en raison de sa moindre diversité de protocoles et de sa fréquence d'appel réduite.

9 - Conclusion

Depuis le commencement de ce projet, le fil directeur a toujours été de créer une distribution qui soit à la fois minimaliste, légère et simple, afin de simplifier l'expérience utilisateur tout en garantissant des performances optimales. Maintenant que le projet est terminé, il est temps de faire un bilan de tout cela. En termes d'enrichissement personnel, ce projet est parfait. Avoir la chance de maîtriser la structure de Linux, maîtriser son fonctionnement, créer sa propre distribution, approcher le monde des trames réseaux… La liste des compétences acquises est longue mais elle témoigne de l'intérêt de cet audit. Bien sûr, ce parcours n'a pas été sans embûches. À maintes reprises je me suis trompé, j'ai dû ajuster ou

même recommencer. On pourrait voir ces erreurs comme des pertes de temps mais au final, elles m’ont permis d’apprendre car elles font partie intégrante de l’apprentissage et de la progression. Si tout s’était passé sans accroc, le projet aurait été fini bien plus tôt. Toutefois, je n'aurais pas pu apprendre autant de choses, et surtout, le sentiment d’accomplissement lors de la première compilation de l’ISO ou encore la mise en service de la distribution n’aurait pas été aussi intense, si ces difficultés n’avaient pas été là. Pour autant, AUDIX n’est pas parfait. Il pourrait être perfectionné en sécurisant les protocoles qu’il appelle.

10 - Bibliographie

1.LFS : http://fr.linuxfromscratch.org/view/lfs-systemd-stable/
2.Image ISO Debian : https://www.debian.org/download.fr.html
3.Commande SSH : https://doc.ubuntu-fr.org/ssh
4.Commande screen : https://doc.ubuntu-fr.org/screen
5.Paquet lxde-core : https://packages.debian.org/fr/sid/lxde-core
6.Paquet abiword : https://packages.debian.org/fr/source/sid/abiword
7.LFS abiword : https://fr.linuxfromscratch.org/view/blfs-12.0-fr/xsoft/AbiWord.html
8.Network-manager : https://wiki.debian.org/fr/NetworkManager
9.Live-build : https://debian-facile.org/doc:install:live-build
10.Wireshark : https://www.wireshark.org/docs/
11.Liste application légères : https://doc.ubuntu-fr.org/liste_applications_legeres
12.LXDE :

- ★ Selon le site officiel, après le démarrage de X11 et LXDE, l'utilisation totale de la mémoire est d'environ 45 Mo sur des machines i386. lxde.sourceforge.net
- ★ Une analyse de Phoronix indique que LXDE consomme moins de mémoire que des environnements de bureau plus lourds, tels que KDE 4.4.1, qui utilise 67 % de mémoire en plus que LXDE. phoronix.com
- ★ Une comparaison des environnements de bureau montre que LXDE utilise environ 87 Mo de mémoire, le classant parmi les environnements les plus légers. wimpysworld.com

13.Cinnamon :

- ☑ Dans la même comparaison, Cinnamon utilise environ 176,3 Mo de mémoire, ce qui est plus élevé que LXDE mais reste raisonnable pour un environnement de bureau moderne. wimpysworld.com

- ☑ Un utilisateur a rapporté que Cinnamon consommait environ 5 % de RAM en moins en mode veille par rapport à XFCE, et environ 10 % de CPU en moins, bien que ces chiffres puissent varier en fonction de la configuration matérielle. reddit.com

IMAGE nombre_paquets :

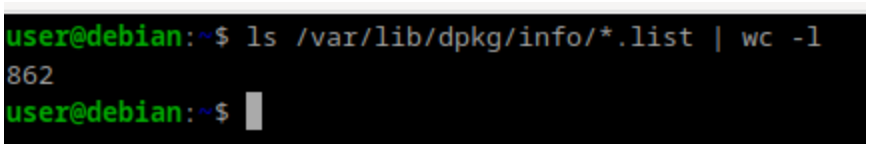

IMAGE nombre_paquetsLM :

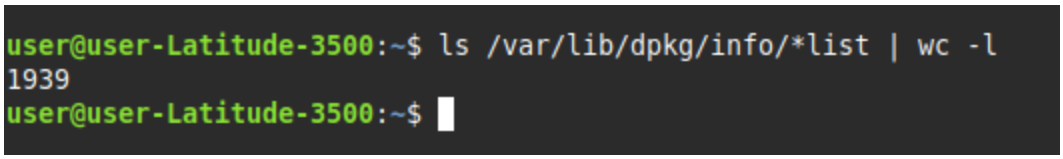

IMAGE services_actifs :

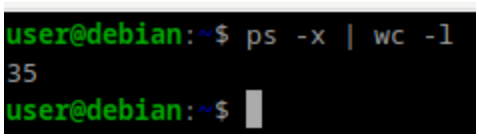

IMAGE services_actifsLM :

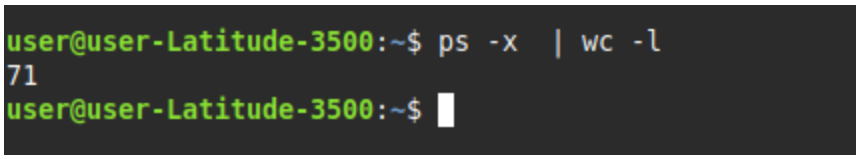

IMAGE taille_disque :

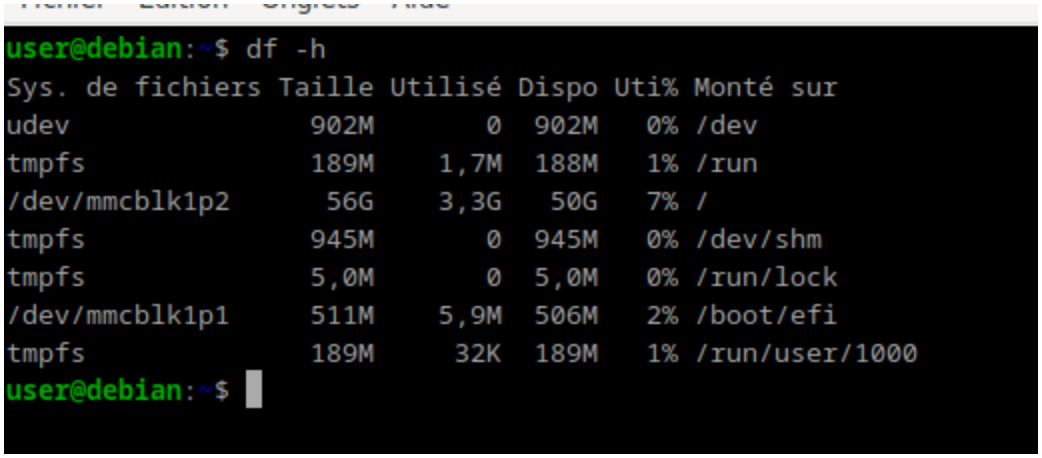

IMAGE taille_disqueLM :

```
user@user-Latitude-3500:~$ df -h
Sys. de fichiers Taille Utilisé Dispo Uti% Monté sur
tmpfs              782M    1,8M  780M   1% /run
efivarfs           374K    145K  225K  40% /sys/firmware/efi/efivars
/dev/nvme0n1p2     234G    9,5G  212G   5% /
tmpfs              3,9G       0  3,9G   0% /dev/shm
tmpfs              5,0M    8,0K  5,0M   1% /run/lock
/dev/nvme0n1p1     511M     12M  499M   3% /boot/efi
tmpfs              782M    208K  781M   1% /run/user/1000
user@user-Latitude-3500:~$
```

IMAGE wireshark_audix :

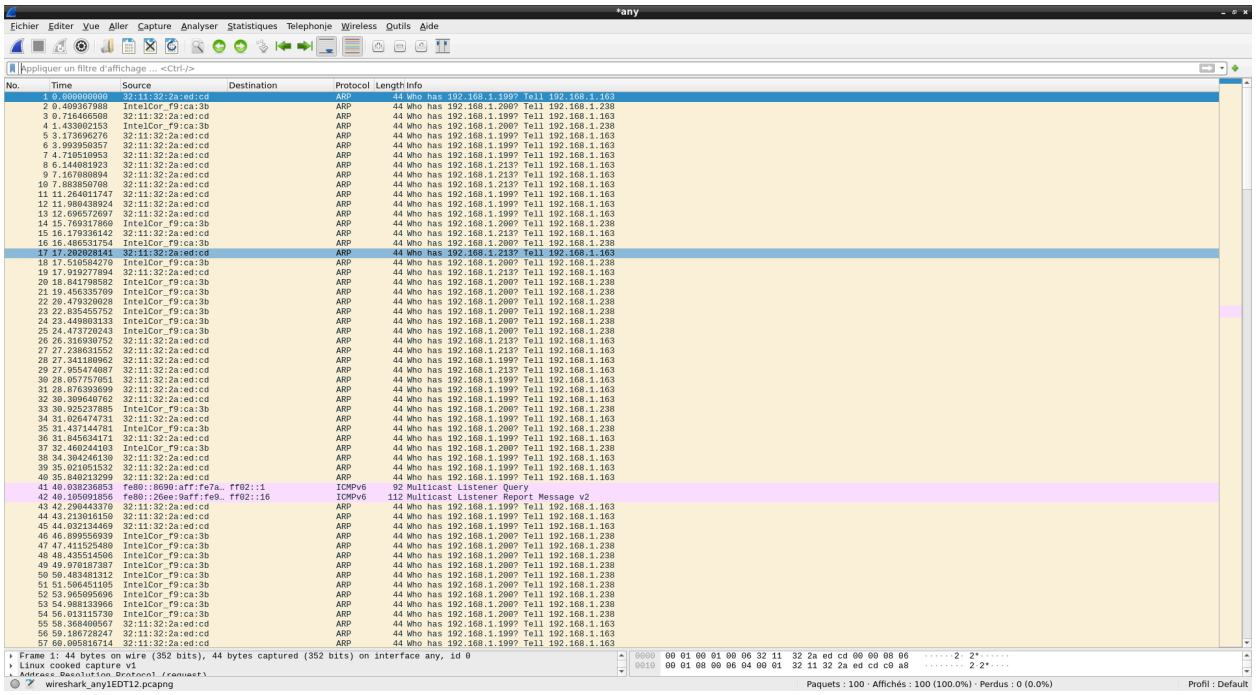

IMAGE wireshark_mint :