

Rapport S2.03

Belguebli Rayane, Ouhdda Anas, Desmee Nathan

Contents

1	Semaines 6 & 7	3
1.1	Que signifie “64-bit” dans “Debian 64-bit” ?	3
1.2	Quelle est la configuration réseau utilisée par défaut ? . .	3
1.3	Quel est le nom du fichier XML contenant la configuration de votre machine ?	4
1.4	Sauriez-vous le modifier directement ce fichier pour mettre 2 processeurs à votre machine ? Faites-le.	4
1.5	Qu’est-ce qu’un fichier iso bootable ?	5
1.6	Qu’est-ce que MATE ? GNOME ?	5
1.7	Qu’est-ce qu’un serveur web ?	5
1.8	Qu’est-ce qu’un serveur ssh ?	6
1.9	Qu’est-ce qu’un serveur mandataire ?	6
1.10	Comment peut-on savoir à quels groupes appartient l’utilisateur “user” ?	6
1.11	Quel est la version du noyau Linux utilisé par votre VM ?	7
1.12	À quoi servent les suppléments invités ? Donner 2 principales raisons de les installer.	7
1.13	À quoi sert la commande mount (dans notre cas de figure et dans le cas général) ?	8
1.14	Compte rendu de l’installation machine virtuel	9
1.14.1	Création de la machine virtuelle	9
1.14.2	Installation du système d’exploitation Debian 11 .	9
2	Semaine 9	15
2.1	Qu’est-ce que le Projet Debian ? D’où vient le nom Debian ?	15
2.2	Il existe 3 durées de prise en charge (support) de ces versions : la durée minimale, la durée en support long terme (LTS) et la durée en support long terme étendue (ELTS). Quelle sont les durées de ces prises en charge ?	15
2.3	Pendant combien de temps les mises à jour de sécurité seront-elles fournies ?	16

2.4	Combien de version au minimum sont activement maintenues par Debian ? Donnez leur nom générique (= les types de distribution).	16
2.5	Chaque distribution majeur possède un nom de code différent. Par exemple, la version majeur actuelle (Debian 11) se nomme Bullseye. D'où viennent les noms de code données aux distributions ?	16
2.6	L'un des atouts de Debian fut le nombre d'architecture officiellement prises en charge. Combien et lesquelles sont prises en charge par la version Bullseye ?	17
2.7	Première version avec un nom de code (date, numéro de version...)	17
2.8	Dernière nom de code attribué (dernier nom à ce jour, date, version...)	17
2.9	Installation préconfigurée de Debian 11.6	18
2.9.1	Préparation de la machine virtuelle	18
3	Semaine 10 & 11	23
3.1	Qu'est-ce que le logiciel git-gui ? Comment se lance-t-il ?	23
3.2	Qu'est-ce que gitk ?	23
3.3	Quelle sera la ligne de commande git pour utiliser par défaut le proxy de l'université sur tous vos projets git ?	24
3.4	Qu'est-ce que Gitea ?	24
3.5	Installation de Gitea	25

1 Semaines 6 & 7

1.1 Que signifie “64-bit” dans “Debian 64-bit” ?

- Un processeur 64 bits est un processeur dont **la largeur des registres est de 64 bits** sur les nombres entiers. Un processeur 64 bits **poura traiter plus d’information en même temps** qu’un processeur 32 bits. Dans notre cas **Debian 64-bit** signifie donc que cette version est compatible uniquement avec cette architecture.

Voir également :

- [32 ou 64 bits ? Comment savoir](#)
- [Windows 32 et 64 bits : Forum Aux Questions](#)

1.2 Quelle est la configuration réseau utilisée par défaut ?

- La configuration réseaux par défaut est une interface de type **Intel PRO/1000 MT Desktop (NAT)**, il s’agit d’un modèle de d’interface réseau de la marque Intel réputé.
- L’adresse MAC de notre machine est **08:00:27:e2:f5:a8**, nous l’avons récupéré dans les paramètres de VirtualBox (*voir Figure 1*) ou en utilisant la commande `ip link show` (*voir Figure 2*).

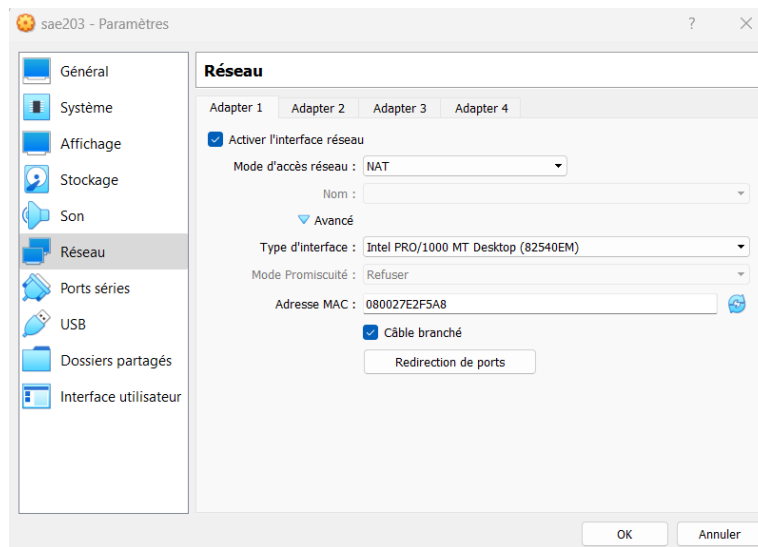


Figure 1: Paramètres de la VM

```

root@serveur:~# ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT group default qlen 1000
    link/ether 08:00:27:e2:f5:a8 brd ff:ff:ff:ff:ff:ff
root@serveur:~#

```

Figure 2: Affichage de l'adresse MAC

1.3 Quel est le nom du fichier XML contenant la configuration de votre machine ?

- Le fichier XML est en réalité le fichier `sae203.vbox`, avec une extension de fichier différente. Lorsque l'on ouvre celui-ci avec un éditeur de texte tel que [Codium](#), on remarque rapidement la syntaxe XML.
- XML est un métalangage informatique de balisage générique, sa syntaxe est dite « extensible » car elle permet de définir différents langages avec pour chacun son vocabulaire et sa grammaire, comme XHTML, XSLT, RSS, SVG... Elle est reconnaissable par son usage des chevrons (<, >) encadrant les noms des balises.

Voir également :

- [Extensible Markup Language](#)

1.4 Sauriez-vous le modifier directement ce fichier pour mettre 2 processeurs à votre machine ? Faites-le.

- On utilise l'éditeur de texte de notre choix (ici `nano`)

`nano sae203.vbox`

- On modifie la **balise** <CPU> on ajoute `count = 2`, qui aura pour effet de modifier le nombre de coeurs de notre machine.

```

<Hardware>
  <CPU count="2">
    <PAE enabled="false"/>
    <LongMode enabled="true"/>
    <X2APIC enabled="true"/>
    <HardwareVirtExLargePages enabled="true"/>
  </CPU>

```

Figure 3: Fichier *vbox*

- CPU est un acronyme anglais pour Central Processing Unit. Le CPU désigne un processeur ou microprocesseur principal d'un ordinateur, on parle aussi d'unité de traitement.

- Tandis qu'un coeur (en anglais, core) est un ensemble de circuits capables d'exécuter des programmes de façon autonome. Toutes les fonctionnalités nécessaires à l'exécution d'un programme sont présentes dans ces coeurs : compteur ordinal, registres, unités de calcul, etc.
- Ainsi au plus notre machine en disposera, au plus elle saura effectuer de calculs/opérations simultanément.

1.5 Qu'est-ce qu'un fichier iso bootable ?

- Un fichier ISO est, en termes simples, un format de fichier numérique reproduisant un CD ou un DVD. L'extension de fichier ISO ne se contente pas de stocker des fichiers et des dossiers : elle contient toutes les informations vitales du système de fichiers concernant la structure du disque.
- On dit d'un disque qu'il est « bootable » lorsqu'il contient les composants logiciels nécessaires pour être **démarré directement au chargement de l'ordinateur**, avant le chargement du système d'exploitation installé sur la machine.

Voir également :

- [Bootable : qu'est-ce que c'est ?](#)
- [Image disque \(ISO\)](#)

1.6 Qu'est-ce que MATE ? GNOME ?

- MATE est un fork de GNOME 2. Il fournit un **environnement de bureau** (apparence de l'ordinateur) attractif et intuitif en se basant sur les métaphores traditionnelles pour GNU/Linux et d'autres systèmes d'exploitation similaires à Unix.
- On précise qu'un fork signifie qu'un logiciel créé à partir du code source d'un autre.

Voir également :

- [Qu'est-ce MATE ?](#)
- [Qu'est-ce que GNOME](#)
- [Qu'est-ce qu'un environnement de bureau ?](#)

1.7 Qu'est-ce qu'un serveur web ?

- Un serveur web est un ordinateur qui **stocke, traite et livre des fichiers de sites web**. Il se compose d'un **côté matériel** et d'un **côté logiciel**, chacun jouant un rôle distinct dans le traitement des fichiers.

Voir également :

- [Qu'est-ce qu'un serveur web ?](#)

1.8 Qu'est-ce qu'un serveur ssh ?

- SSH (*acronyme de Secure SHell*) est un protocole réseau qui permet aux administrateurs **d'accéder à distance à un ordinateur**, en toute sécurité. SSH désigne également l'ensemble des utilitaires qui mettent en oeuvre le protocole.

Voir également :

- [Qu'est-ce qu'un serveur ssh ?](#)

1.9 Qu'est-ce qu'un serveur mandataire ?

- Un serveur mandataire ou proxy (de l'anglais) est un serveur informatique qui a pour fonction de **relayer des requêtes entre un poste client et un serveur**. Les serveurs mandataires sont notamment utilisés pour assurer les fonctions suivantes :
- mémoire cache
- la journalisation des requêtes ("logging")
- la sécurité du réseau local
- le filtrage et l'anonymat.

L'utilité des serveurs mandataires est importante, notamment dans le cadre de la sécurisation des systèmes d'information.

Voir également les différentes définitions d'un serveur mandataire :

- [Définition 1 \(techno-science\)](#)
- [Définition 2 \(futura-science\)](#)

1.10 Comment peut-on savoir à quels groupes appartient l'utilisateur "user" ?

- Nous pouvons visualiser les groupes auxquels appartient un utilisateur en lisant le fichier suivant à la ligne de l'utilisateur "*user*":

```
cat /etc/group | grep user
```

- Ou en utilisant directement la commande `groups`

```
groups user
```

```
root@serveur:~# groups user
user : user cdrom floppy sudo audio dip video plugdev netdev bluetooth lpadmin s
canner
root@serveur:~# █
```

Figure 4: Commande *groups*

Voir également :

- [Comment fonctionnent les groupes linux ?](#)

1.11 Quel est la version du noyau Linux utilisé par votre VM ?

- Il est avant tout important de bien saisir ce qu'est un noyau, dit Kernel. Il constitue le cœur du système qui exploite un ordinateur. Il établit la communication entre la partie matérielle et la partie logicielle de l'appareil. C'est ce qui permet à l'utilisateur d'interagir avec la machine.
- Ainsi pour en connaître la version il suffit d'utiliser la commande `uname`, on a ainsi le numéro de version : "5.10.0-21-amd64"

```
root@serveur:~# uname -r
5.10.0-21-amd64
root@serveur:~# uname -a
Linux serveur 5.10.0-21-amd64 #1 SMP Debian 5.10.162-1 (2023-01-21) x86_64 GNU/Linux
root@serveur:~#
```

Figure 5: Commande *uname*

- La première partie avant le - nous indiquant le numéro de version 5.10.0-21, et la seconde notre architecture amd64.

Voir également :

- [Comment voir la version du noyau linux ?](#)

1.12 À quoi servent les suppléments invités ? Donner 2 principales raisons de les installer.

- Après avoir [créé une machine virtuelle et installer l'OS sur Virtualbox](#), il convient d'installer les Virtualbox VirtualBox additions invités. (Guest Additions).
- Cela permet **d'améliorer les performances et intégrations entre la machine hôte et la machine invité** mais aussi apporter de nouvelles fonctionnalités.
- Par exemple, vous pouvez [activer le copier/coller entre la machine hôte et invité](#), ou encore profiter d'une résolution d'écran adaptative/automatique.

Voir également :

- [Que sont les suppléments invités ?](#)

1.13 À quoi sert la commande mount (dans notre cas de figure et dans le cas général) ?

- La commande mount permet de **demandeur au système d'exploitation de rendre un système de fichiers accessible**, à un emplacement spécifié (le point de montage).

Voir également :

- [Qu'est ce que le mount ?](#)

1.14 Compte rendu de l'installation machine virtuel

1.14.1 Création de la machine virtuelle

- Condition préalable : extension pack (VBoxGuestAddition.iso) présent sur la machine hôte
- Création de la machine :
 - Nom de la machine dans VirtualBox : sae203
 - Dossier de la machine : `/usr/local/virtual_machine/infoetu/login` avec notre login personnel.
 - Type : Linux
 - Version : Debian ou Debian 11 en [64-bit](#)

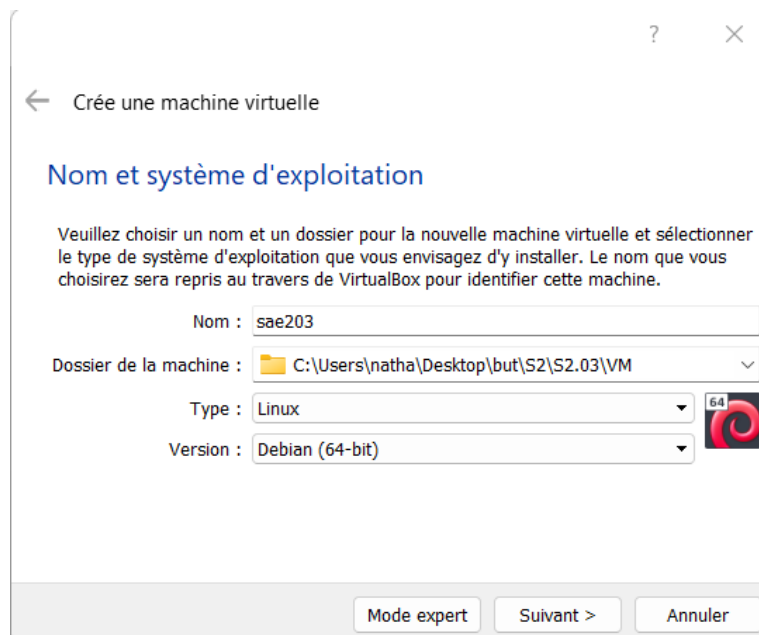


Figure 6: Création de la machine virtuelle

- Initialisation de la mémoire vive à 2048 Mo
- 20 Go de disque dur sur une seule partition

1.14.2 Installation du système d'exploitation Debian 11

- Insérer le fichier [iso bootable](#) pour l'installation dans le lecteur cd de la machine
- Pays/langue : France
- Nom de la machine : `serveur`

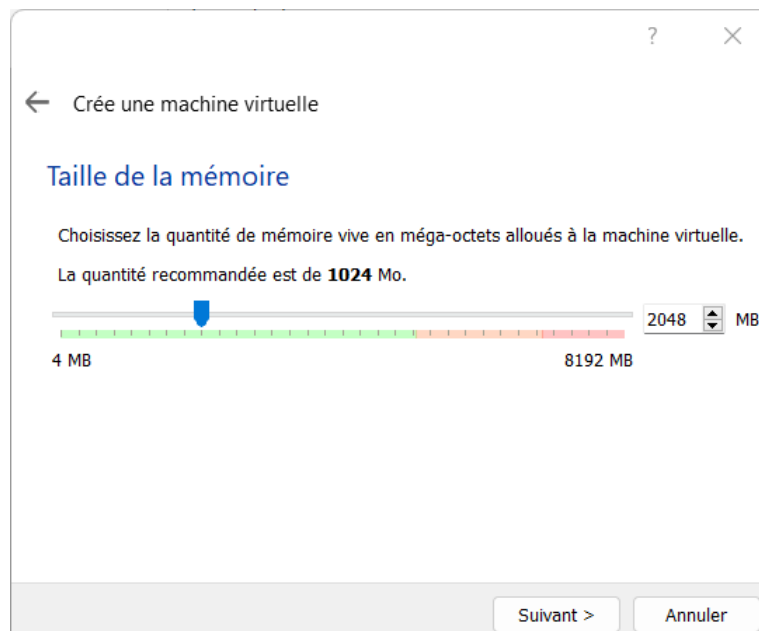


Figure 7: Allocation de la mémoire

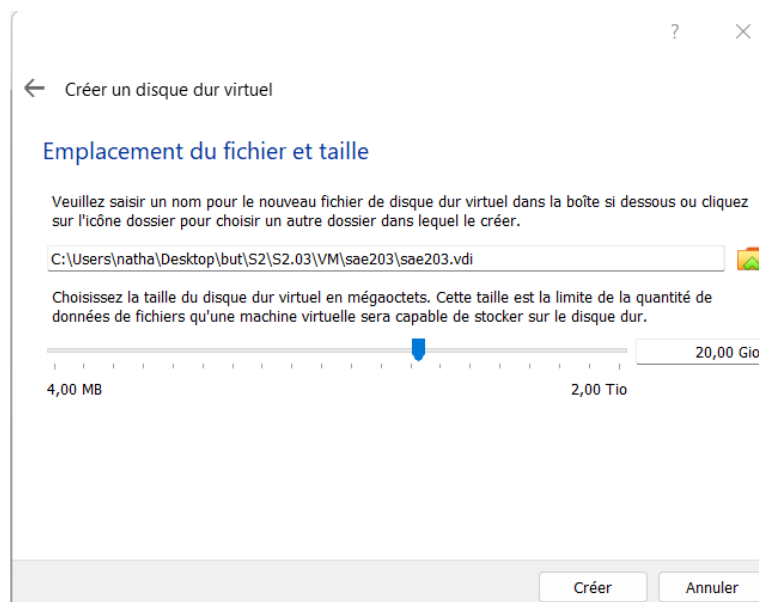


Figure 8: Allocation du stockage

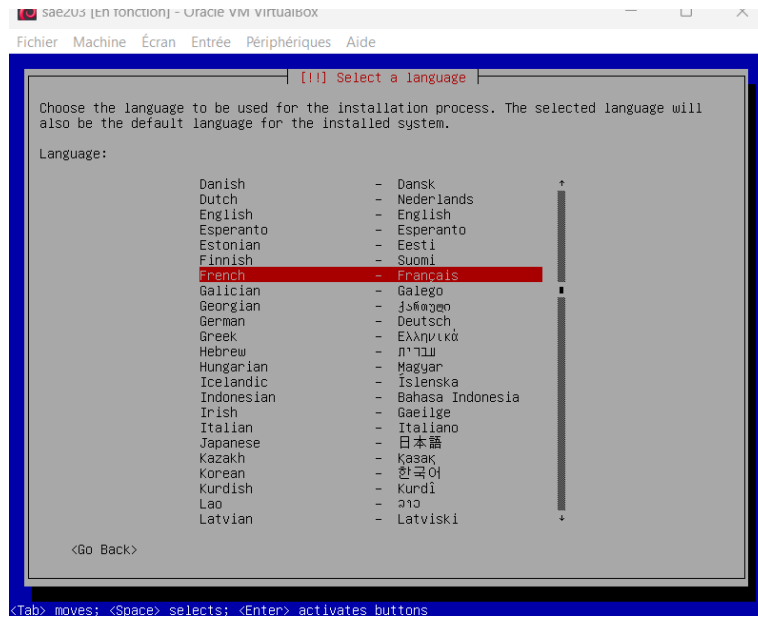


Figure 9: Choix de la langue

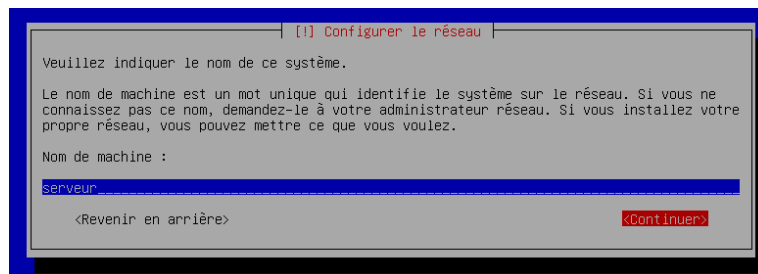


Figure 10: Nom de la machine

- Compte administrateur : **root:root**
- Un Compte utilisateur : **User:user**
- Partition : 1 seule partition recouvrant le disque entier suffit

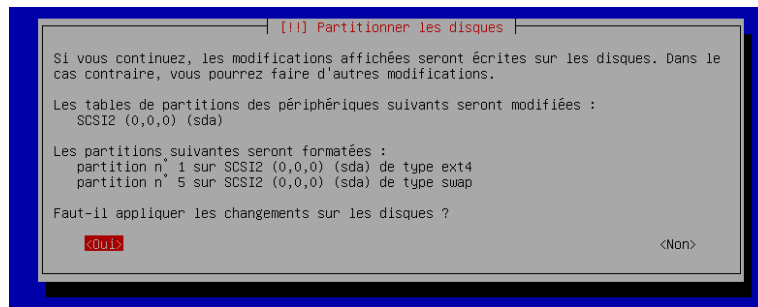


Figure 11: Partitionnement du disque

- Miroir : <http://debian.polytech-lille.fr>
 - Les miroirs sont des serveurs, répartis dans le monde, qui répliquent l'information des serveurs Debian originaux et permettent de répartir la charge et de raccourcir le temps d'accès à l'information. Il existe, une vingtaine de miroirs en France.

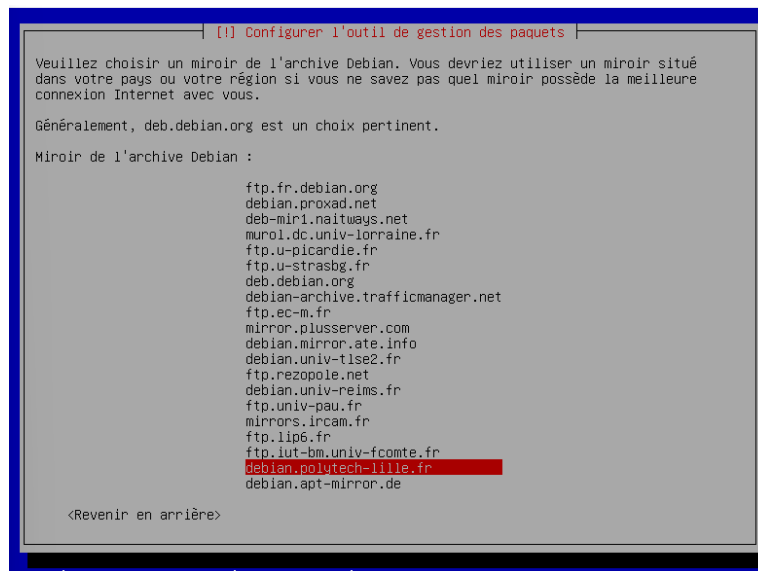


Figure 12: Outil de gestion des paquets

- Proxy : <http://cache.univ-lille.fr:3128>

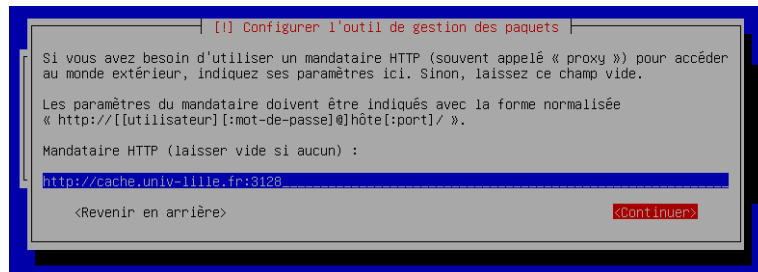


Figure 13: Serveur mandataire

- Sélection des logiciels de démarrage :
 - environnement de bureau Debian
 - MATE
 - serveur web
 - serveur ssh
- utilitaire usuels du système

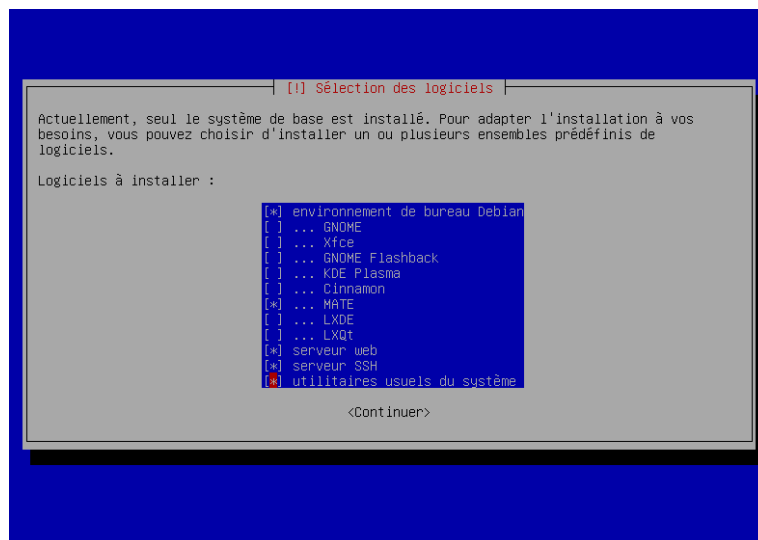


Figure 14: Logiciels à installer

- Après la fin de l'installation de la VM nous avons dû donner les droits sudo à notre user grâce à l'utilisateur root et à ces commandes :

```
usermod -aG sudo [login]
```

- Ensuite nous avons dû installer les suppléments invités et pour cela nous avons été dans : Périphériques › Insérer l'illustration CD des additions

invités...

- Et enfin nous avons utilisé ces commandes :
 - Monter le CD sudo : `mount /dev/cdrom /mnt`
 - Installer les suppléments : `sudo /mnt/VBoxLinuxAdditions.run`

```
user@serveur:~$ sudo mount /dev/cdrom /mnt
Nous espérons que vous avez reçu de votre administrateur système local
les consignes traditionnelles. Généralement, elles se concentrent sur ces trois éléments :

    #1) Respectez la vie privée des autres.
    #2) Réfléchissez avant d'utiliser le clavier.
    #3) De grands pouvoirs confèrent de grandes responsabilités.

[sudo] Mot de passe de user :
mount: /mnt: ATTENTION: source protégée en écriture, montée en lecture seule.
user@serveur:~$ sudo /mnt/VBoxLinuxAdditions.run
Verifying archive integrity... All good.
Uncompressing VirtualBox 6.1.38 Guest Additions for Linux.....
VirtualBox Guest Additions installer
Copying additional installer modules ...
Installing additional modules ...
```

Figure 15: Montage du disque + installation des suppléments

2 Semaine 9

2.1 Qu'est-ce que le Projet Debian ? D'où vient le nom Debian ?

Le **projet Debian** est un projet ayant pour but de créer un système d'exploitation exclusivement constitué de logiciels libres. Il est lancé en 1993 par **Ian Murdock**. Le nom “*Debian*” tire son origine des prénoms du créateur de Debian, *Ian Murdock*, et de son épouse, *Debra*.



Figure 16: Ian Murdock

- [Voir la présentation du projet Debian](#)
- [Qui est Ian Murdock ?](#)

2.2 Il existe 3 durées de prise en charge (support) de ces versions : la durée minimale, la durée en support long terme (LTS) et la durée en support long terme étendue (ELTS). Quelle sont les durées de ces prises en charge ?

- **LTS** : 5 ans (*en comptant le minimum*)
 - est un projet pour étendre la durée de vie de toutes les versions stables de Debian à (au moins) 5 ans. Debian LTS n'est pas gérée par l'équipe en charge de la sécurité de Debian, mais par un groupe distinct de bénévoles et sociétés intéressés pour en faire un succès.
- **ELTS** : 10 ans (*5 ans en plus après le LTS*)
 - est une offre commerciale pour prolonger la durée de vie des versions de Debian à 10 ans

- **Minimum : 3 ans**

Ces durées ne sont pas fixes et représentent le maximum.

- [Qu'est-ce que LTS ?](#)

2.3 Pendant combien de temps les mises à jour de sécurité seront-elles fournies ?

Les mises à jours de sécurité sont fournies pour une version si celle-ci est prise en charge.

- [Voir l'explication sur les différentes versions](#)

2.4 Combien de version au minimum sont activement maintenues par Debian ? Donnez leur nom générique (= les types de distribution).

- Debian 11 : bullseye (version stable)
- Debian 10: buster (version oldstable)
- Debian 9: stretch (version oldoldstable prise en charge LTS)
- Debian 8: jessie (version archivée avec prise en charge ELTS)
- [Voir le guide des versions sur le site officiel de Debian](#)

2.5 Chaque distribution majeur possède un nom de code différent. Par exemple, la version majeur actuelle (Debian 11) se nomme Bullseye. D'où viennent les noms de code données aux distributions ?

- Fait insolite, **Debian** tire ses noms de distributions de personnages des célèbres films [Toy Story](#)



Figure 17: Debian 11 *Bullseye*

- [Voir les références de Debian à ToyStory](#)

2.6 L'un des atouts de Debian fut le nombre d'architecture officiellement prises en charge. Combien et lesquelles sont prises en charge par la version Bullseye ?

- Si l'on en croit la page d'installation officielle de Debian, les architectures prises en charge sont au total de 9:
 - **amd64*** : l'architecture des premiers microprocesseurs 64 bits de la société Advanced Micro Devices
 - **arm64*** : architecture microélectronique développée par la société ARM. C'est la première architecture ARM 64 bits.
 - **armel**
 - **armhf**
 - **i386** : **Intel 80386**, est un microprocesseur 32 bits CISC fabriqué par Intel utilisé dans de nombreux ordinateurs personnels de 1986 à 1994.
 - **mips** : architecture de processeur de type [Reduced instruction set computer \(RISC\)](#) développée par la société [MIPS Technologies](#) (*alors appelée MIPS Computer Systems*).
 - **mipsel**
 - **ppc64el** : ppc64le est un pur mode [little-endian](#) qui a été introduit avec le [POWER8](#) comme cible principale des technologies fournies par la [Fondation OpenPOWER](#), visant à permettre le portage du logiciel basé sur Linux [x86](#) avec un minimum d'effort.
 - **s390x**

* *Les plus répandus*

Voir également :

- [Voir également la liste des architectures officiellement compatibles](#)
- [Différences entre ARM64, ARMel et ARMhf](#)

2.7 Première version avec un nom de code (date, numéro de version. . .)

- La première version [Debian “Buzz”](#) qui est la version 1.1, qui est sorti le *17 juin 1996*, contenait *474 paquets*.

2.8 Dernière nom de code attribué (dernier nom à ce jour, date, version. . .)

- La dernière version de debian publiée est [Debian “Bullseye”](#) qui est la version 11 qui est la version **actuelle** et **stable**. Elle a été annoncée le *6 juillet 2016* et publiée le *6 juillet 2019*. Debian est en train de développer

une nouvelle version avec comme nom de code “**Bookworm**” mais la date n’a pas encore été dévoilée, cette version est encore en phase de test.



Figure 18: Bookworm

2.9 Installation préconfigurée de **Debian 11.6**

2.9.1 Préparation de la machine virtuelle

On commence par créer la machine virtuelle sur **VirtualBox**, de la même manière qu’à la semaine précédente.

Création de la machine avec le nom sae203b, de type linux debian 64-bit, dans le fichier /usr/local/virtual_machine/infoetu/login, 2048 Mo de RAM et 20 GO de disque dur.

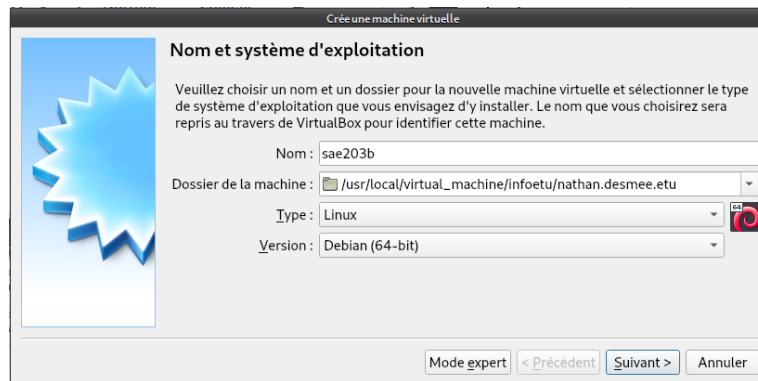


Figure 19: Création de la machine virtuelle 1.1

décompression du fichier autoinstall.zip au même endroit que la VM et faire la commande :

```
sed -i -E "s/(--iprt-iso-maker-file-marker-bourne-sh).*$/\1=$(cat /proc/sys/kernel/random/uuid)/"
```

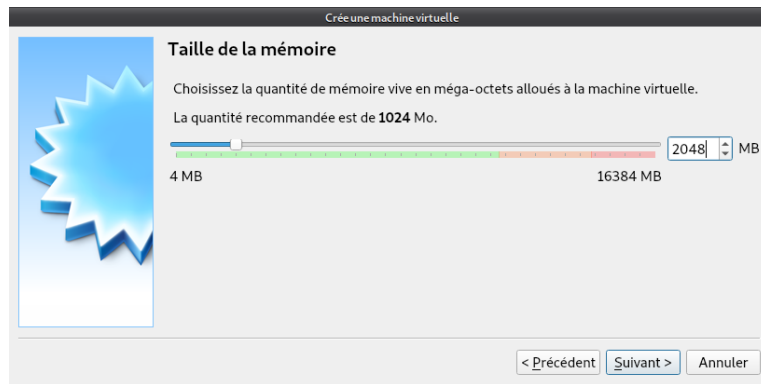


Figure 20: Création de la machine virtuelle 1.2

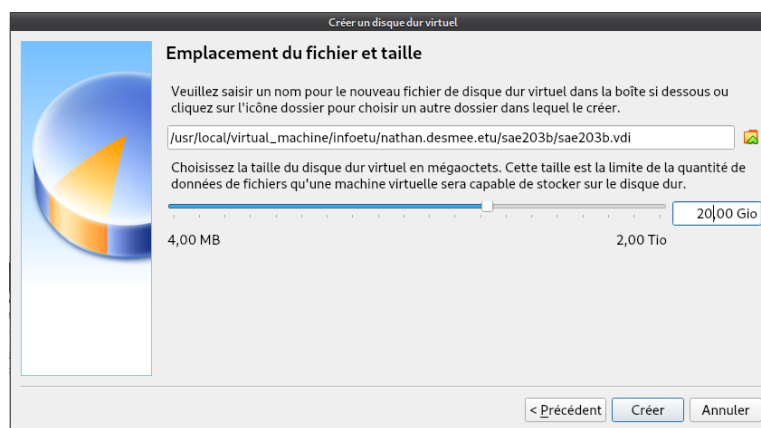


Figure 21: Création de la machine virtuelle 1.3

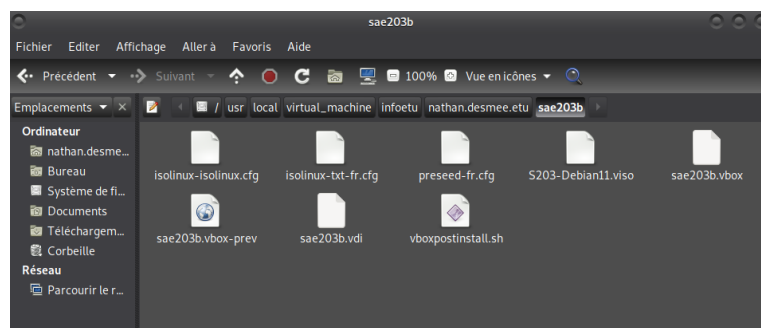


Figure 22: Création de la machine virtuelle 1.4

S203-Debian11.viso au même endroit pour remplacer la chaîne @@UUID@@ par un identifiant unique universel

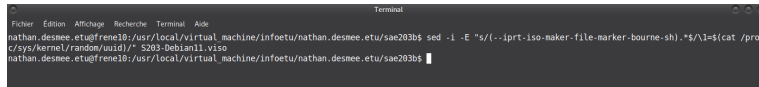


Figure 23: décompression du fichier autoinstall.zip

mise en place du fichier S203_Debian11.viso dans le lecteur optique de la machine virtuelle

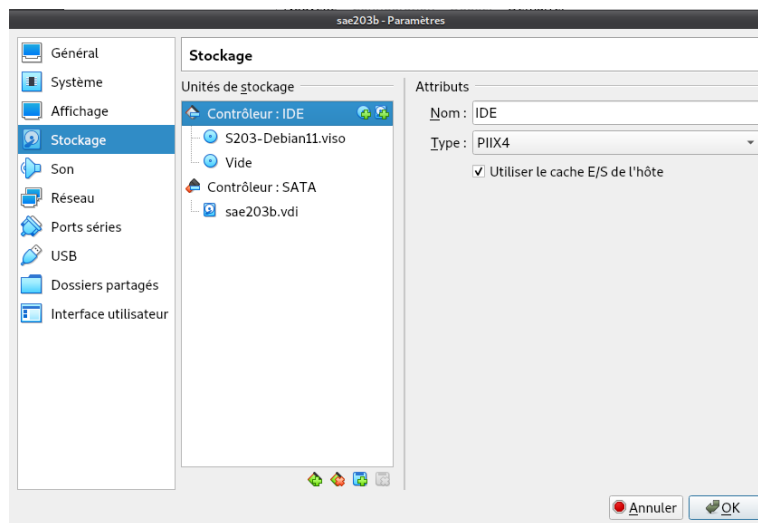


Figure 24: mise en place du fichier S203_Debian11.viso

démarrage de l'auto installation et test des ajouts invités

Ajout des droits sudo

Pour mettre les droits sudo à l'utilisateur, mettre MATE comme bureau et installer les paquets il faut :

- ajouter sudo aux groupes du user dans la partie Utilisateur Standard : `d-i passwd/user-default-groups string audio cdrom video sudo`
- ajouter tous les paquets à installer dans la partie Packages, Mirrors, Image : `d-i pkgssel/include string sudo git sqlite3 curl bash-completion neofetch`
- ajouter MATE comme style de bureau de base dans la partie installation de meta-paquetages : `tasksel tasksel/first multiselect standard ssh-server mate-desktop desktop`

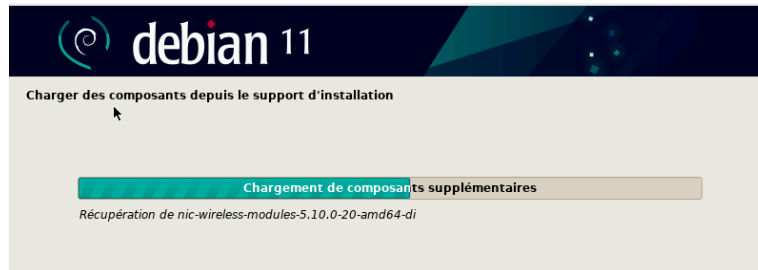


Figure 25: démarrage auto install 1.1

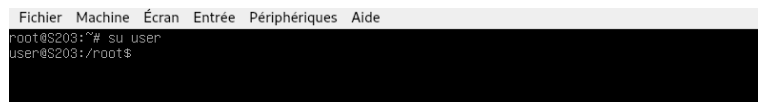


Figure 26: démarrage auto install 1.2

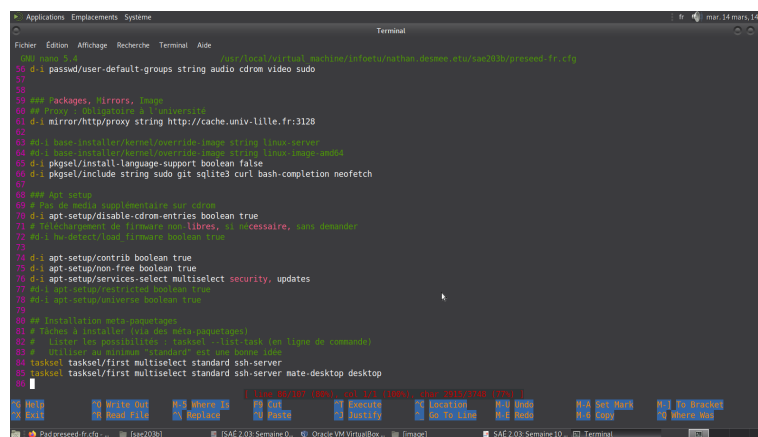


Figure 27: Ajout droit sudo

relancer l'installation et mettre 2 CPU pour que la machine puisse se lancer

L'installation est désormais terminée et prête à utilisation !

3 Semaine 10 & 11

3.1 Qu'est-ce que le logiciel git-gui ? Comment se lance-t-il ?

Git-gui dispose des mêmes fonctionnalités que Git, mais ayant pour avantage de posséder une interface graphique. On peut donc créer un nouveau dépôt Git en local, en indiquant le nom de celui-ci, cela est équivalent à la commande :

```
git init [mon_repo]
```

Git-gui permet aussi de cloner un dépôt distant en local en indiquant le chemin vers la source et le dossier de destination, ce qui équivaut à :

```
git clone [url_du_dépôt_source] [mon_dossier_local]
```

Il permet d'ouvrir le dépôt en local spécifié avec l'outil Git Gui, lister les derniers dépôts ouverts, menu dans lequel vous retrouverez l'ensemble des actions que nous venons de voir.

Il se lance grâce à ces commandes :

Pour l'installer :

```
sudo apt-get install git-gui
```

Pour le lancer :

```
git gui
```

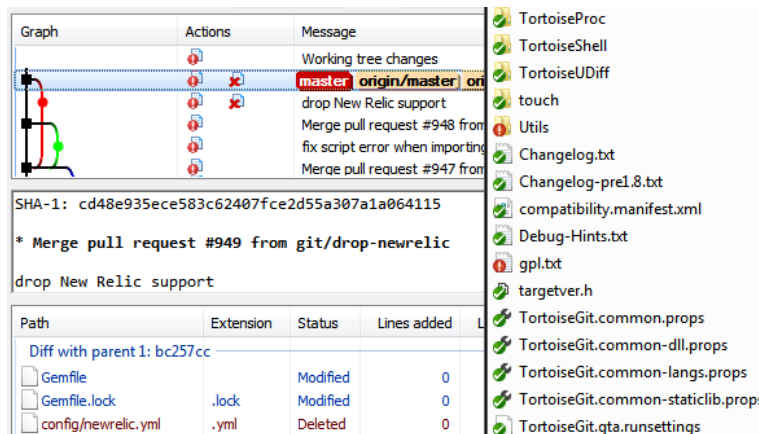


Figure 28: Interface graphique pour Git

3.2 Qu'est-ce que gitk ?

Gitk est un navigateur de dépôt graphique, le premier de son genre. Il peut être considéré comme un encapsuleur graphique pour `git log`. Il permet d'explorer

et de visualiser l'historique d'un dépôt. Il est écrit en [tcl/tk](#), ce qui le rend portable sur tous les systèmes d'exploitation. gitk est mis à jour par Paul Mackerras en tant que projet indépendant et distinct du noyau Git. Des versions stables sont distribuées dans le cadre de la suite Git pour faciliter la tâche des utilisateurs finaux. Gitk peut être un outil d'apprentissage utile pour les nouveaux venus dans Git.

On le lance avec cette commande : **gitk**.

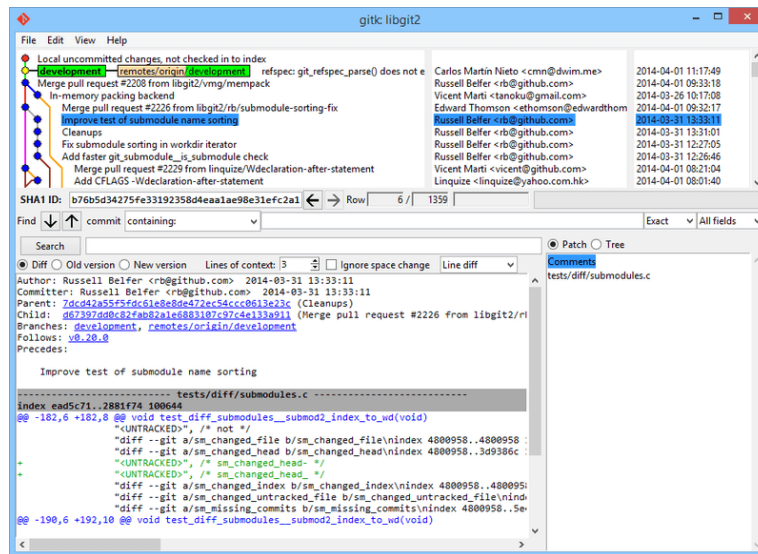


Figure 29: Navigateur gitk

3.3 Quelle sera la ligne de commande git pour utiliser par défaut le proxy de l'université sur tous vos projets git ?

La ligne de commande permettant de configurer de manière permanente un proxy à Git est la suivante :

```
git config --global http.proxy http://notreproxy.fr:notreport
```

3.4 Qu'est-ce que Gitea ?

[Gitea](#) est une forge logicielle libre, développée en [Golang](#) sous [licence MIT](#), pour l'hébergement de développement logiciel, basé sur le logiciel de gestion de versions Git pour la gestion du code source, comportant un système de suivi des bugs, un wiki, ainsi que des outils pour la relecture de code.

Pour information, **Go** est un langage de programmation compilé et concurrent

inspiré de **C** et **Pascal**. Ce langage a été développé par Google à partir d'un concept initial de Robert Griesemer, Rob Pike et Ken Thompson.

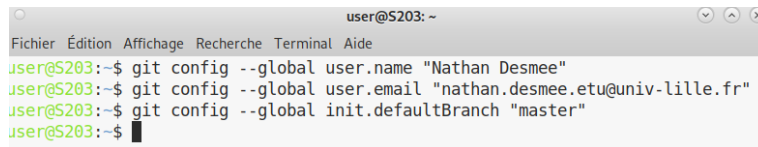
De plus la licence MIT est une licence de logiciel pour logiciels libres et open source, qui donne à toute personne recevant le logiciel le droit illimité de l'utiliser, le copier, le modifier, le fusionner, le publier, le distribuer, le vendre et de changer sa licence. La seule obligation est de mettre le nom des auteurs avec la notice de copyright.

Comme logiciel similaire à gitea nous pouvons retrouver [GitHub Desktop](#) ou encore [IBM Rational ClearCase](#).

3.5 Installation de Gitea

configuration des paramètres git :

```
git config --global user.name "Prénom Nom"
git config --global user.email "votre@email"
git config --global init.defaultBranch "master"
```

A screenshot of a terminal window titled 'user@S203: ~'. The window has a menu bar with 'Fichier', 'Édition', 'Affichage', 'Recherche', 'Terminal', and 'Aide'. The terminal shows the following commands and their outputs:

```
user@S203:~$ git config --global user.name "Nathan Desmee"
user@S203:~$ git config --global user.email "nathan.desmee.etu@univ-lille.fr"
user@S203:~$ git config --global init.defaultBranch "master"
user@S203:~$
```

Figure 30: configuration des paramètres git

installation de git-gui

```
sudo apt-get install git-gui
```

A screenshot of a terminal window titled 'user@S203: ~'. The window has a menu bar with 'Fichier', 'Édition', 'Affichage', 'Recherche', 'Terminal', and 'Aide'. The terminal shows the following command and its output:

```
user@S203:~$ sudo apt-get install git-gui
```

Figure 31: installation de git-gui

ensuite il faut rediriger les messages arrivant sur le port 3000 de la machine physique vers le port 3000 de la machine virtuelle pour cela configurer la machine virtuelle

installation de gitea 1.18.5

```
wget -O gitea https://dl.gitea.com/gitea/1.18.5/gitea-1.18.5-linux-amd64
```

: installation de gitea depuis un dépôt sur internet ne pas oublier les lignes :

```
export http_proxy=http://cache.univ-lille.fr:3128
export https_proxy=$http_proxy
```

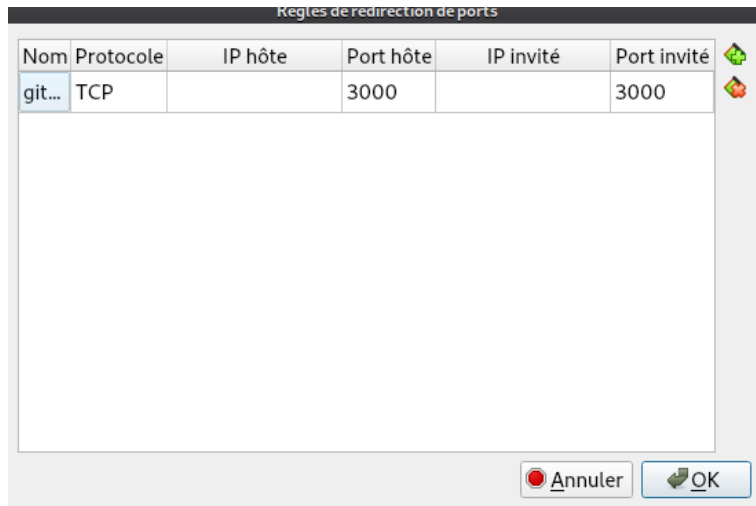


Figure 32: port 3000

```
user@S203: ~  
Fichier Édition Affichage Recherche Terminal Aide  
user@S203:~$ wget -O gitea https://dl.gitea.com/gitea/1.18.5/gitea-1.18.5-linux-  
x-amd64  
--2023-03-14 14:40:54-- https://dl.gitea.com/gitea/1.18.5/gitea-1.18.5-linux-  
amd64  
Résolution de cache.univ-lille.fr (cache.univ-lille.fr)... 193.51.59.104, 193.51.  
.59.105  
Connexion à cache.univ-lille.fr (cache.univ-lille.fr)|193.51.59.104|:3128... con  
necté.  
requête Proxy transmise, en attente de la réponse... 200 OK  
Taille : 116640224 (111M) [application/octet-stream]  
Sauvegarde en : « gitea »  
  
gitea 100%[=====>] 111,24M 97,1MB/s ds 1,1s  
2023-03-14 14:40:55 (97,1 MB/s) - « gitea » sauvegardé [116640224/116640224]
```

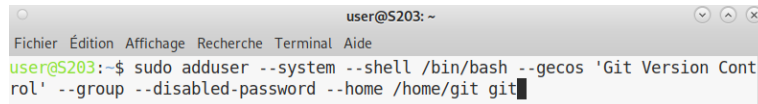
Figure 33: installation de gitea 1.18.5

afin d'être connecté à internet

`chmod +x gitea` pour ajouter les droits d'exécution sur le dossier gitea

`git --version` pour vérifier si on est bien sur git 2.0 au minimum

`sudo adduser --system --shell /bin/bash --gecos 'Git Version Control' --group --disabled-password --home /home/git git` pour créer l'utilisateur git pour exécuter gitea



```
user@S203: ~
Fichier Édition Affichage Recherche Terminal Aide
user@S203:~$ sudo adduser --system --shell /bin/bash --gecos 'Git Version Control' --group --disabled-password --home /home/git git
```

Figure 34: Git Version Control

Pour créer les fichiers nécessaires à gitea avec les droits et propriétaires :

`sudo mkdir -p /var/lib/gitea/{custom,data,log}`

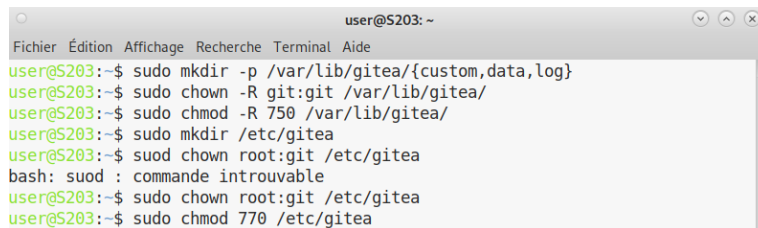
`sudo chown -R git:git /var/lib/gitea/`

`sudo chmod -R 750 /var/lib/gitea/`

`sudo mkdir /etc/gitea`

`sudo chown root:git /etc/gitea`

`sudo chmod 770 /etc/gitea`

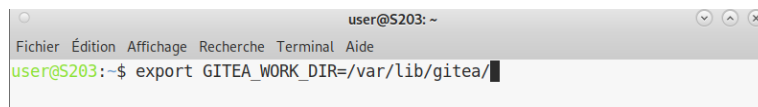


```
user@S203: ~
Fichier Édition Affichage Recherche Terminal Aide
user@S203:~$ sudo mkdir -p /var/lib/gitea/{custom,data,log}
user@S203:~$ sudo chown -R git:git /var/lib/gitea/
user@S203:~$ sudo chmod -R 750 /var/lib/gitea/
user@S203:~$ sudo mkdir /etc/gitea
user@S203:~$ sudo chown root:git /etc/gitea
bash: suod : commande introuvable
user@S203:~$ sudo chown root:git /etc/gitea
user@S203:~$ sudo chmod 770 /etc/gitea
```

Figure 35: création les fichiers nécessaires à gitea

pour définir une variable d'environnement pour que gitea utilise le bon répertoire de travail :

`export GITEA_WORK_DIR=/var/lib/gitea/`



```
user@S203: ~
Fichier Édition Affichage Recherche Terminal Aide
user@S203:~$ export GITEA_WORK_DIR=/var/lib/gitea/
```

Figure 36: définition d'une variable d'environnement

facultatif avec linux

copie du binaire de gitea au bon endroit :

```
sudo cp gitea /usr/local/bin/gitea
```

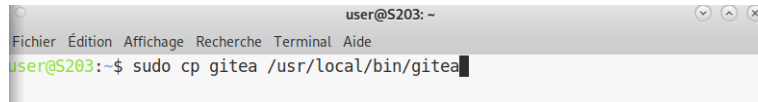


Figure 37: copie du binaire de gitea

Lancer gitea automatiquement :

créer un fichier `/etc/systemd/system/gitea.service` et coller le contenu de cette page dedans : <https://github.com/go-gitea/gitea/blob/main/contrib/systemd/gitea.service> pour lancer gitea faire :

```
sudo systemctl enable gitea --now
```

mettre cette URL dans un moteur de recherche (connexion avec gitea qui utilise le port 3000):

`http://localhost:3000`

paramétrer gitea avec la BDD SQLite3, user gitea password gitea et mail `git@localhost`

pour finir faire les commandes :

```
sudo chmod 750 /etc/gitea
```

```
sudo chmod 640 /etc/gitea/app.ini
```

afin de protéger les fichiers `/etc/gitea` et `/etc/gitea/app.ini`

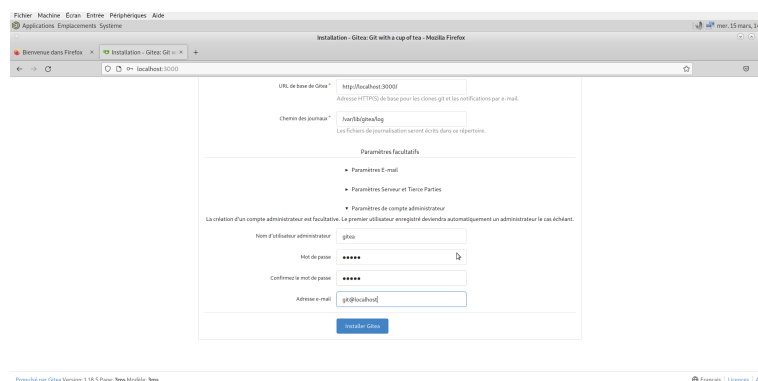


Figure 38: Gitea

Mise à jour

Pour cela, nous devons arrêter le service gitea, remplacer le binaire gitea de la version actuelle par la nouvelle version, et relancer le service. Pour faire ceci sans

reconfigurer soi-même, gitea a mit en place un script bash ([upgrade.sh](#)) pour automatiser tout cela, on précise uniquement la version souhaité avec l'option `-v`

```
sudo ./upgrade.sh -v 1.19
```

Nous allons maintenant ajouter quelques repos pour tester les fonctionnalités et le bon fonctionnement de Gitea. Depuis l'interface Gitea nous pouvons créer et enrichir directement le repos (comme vous pouvez le voir dans le screen avec le repos exemple). Mais également depuis la commande git:

Voici un exemple avec le repos 'dev-oo' dédié aux tp de dev poo et qualité dev créée par chaque étudiant dans son propre compte Gitlab

Tout d'abord on crée le repos 'dev-oo' depuis l'interface de Gitea, puis on clone le dépôt distant provenant de Gitlab, on change son origine pour lui attribuer celui de Gitea et ainsi commit puis push toutes les fichiers sur le repos 'dev-oo' dans Gitea.

```
git clone https://gitlab.univ-lille.fr/[login_etudiant]/dev-oo && cd dev-oo
git remote remove origin
git remote add origin http://localhost:3000/gitea/dev-oo
# La branche sur le repos git est main par défaut
# contrairement au paramétrage des préliminaires
git branch -m main master
# user: gitea, password: gitea
git push origin master
```

Également, pour les rapports de Saé 2.03, nous initialisons un repos git et on fait un premier commit distant avec tous les rapports:

```
git init
git remote add origin http://localhost:3000/gitea/s2.03
git add -A && git commit -m "Initial commit"
git push origin master
```

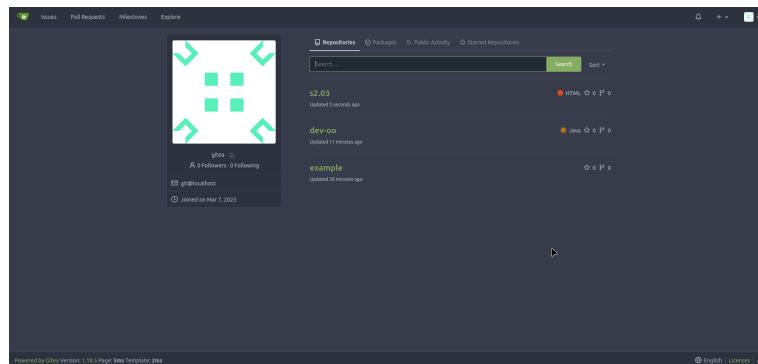


Figure 39: ajout gitea 1.1

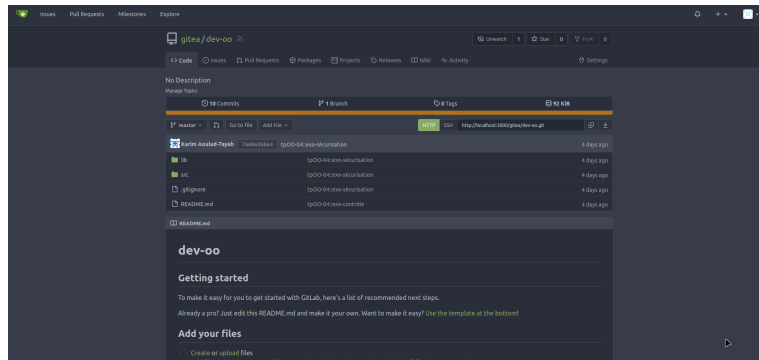


Figure 40: ajout gitea 1.2

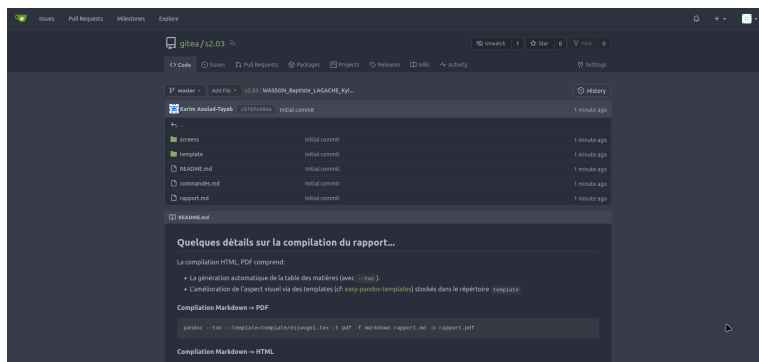


Figure 41: ajout gitea 1.3