

Rapport - Projet Réseaux

Mathis Magnier mathis.magnier.etu@univ-lille.fr
Aymane Benafquir aymane.benfaquir.etu@univ-lille.fr
Milan Theron milan.theron.etu@univ-lille.fr

Contents

1	Résumé	1
2	Création d'une machine virtuelle	2
2.1	Création de la VM (virtual machine) à l'aide de VirtualBox	2
2.2	Installation de l'OS	2
2.2.1	Question que vous pourriez vous poser ?	3
2.3	Préparation du système	5
2.4	Suppléments invités	6
	Question 3	6
	Question 4	6
2.5	Apprentissage du markdown	7
	ceci est un titre de niveau 1	7
	ceci est un titre de niveau 2	7
	ceci est un titre de niveau 3	7
2.6	Exportation avec Pandoc	7
3	Études d'applications clientes	8
3.1	Configuration de Git	8
	Questions	8
4	Installation du service Gitea	9
4.1	Qu'est ce que Gitea?	10
4.2	À quels logiciels bien connus dans ce domaine peut-on le comparer ?	10
4.3	Installation de Gitea	10
4.3.1	Installation du binaire	11
4.3.2	Création du fichier de configuration Gitea	11
5	Les problème rencontrés	13
6	Conclusion	14

1 Résumé

Nous avons entamé la SAE 2.03 avec pour objectif l'automatisation de l'installation d'une machine virtuelle VirtualBox et l'apprentissage du langage Markdown ou AsciiDoc, en mettant en place un système d'exploitation Debian avec un environnement graphique MATE. La première semaine a été consacrée à la configuration matérielle dans VirtualBox, l'installation de l'OS, la préparation du système, et l'automatisation de cette installation. La deuxième a plutôt été orientée vers l'apprentissage du langage Markdown et l'utilisation de Pandoc. Et la troisième nous a servi à étudier des applications clientes comme avec l'exemple de Git.

2 Création d'une machine virtuelle

2.1 Création de la VM (virtual machine) à l'aide de VirtualBox

La configuration matérielle a commencé par la création d'une nouvelle machine virtuelle. Nous avons défini le nom de la machine ("serveur"), laissé le domaine vide, choisi le pays/langue (France), et configuré le miroir Debian (<http://debian.polytech-lille.fr>). Nous avons opté pour une seule partition de la taille du disque et sélectionné les logiciels de démarrage tels que l'environnement de bureau Debian MATE, un serveur web, un serveur SSH, et les utilitaires usuels du système.



Figure 1: L'image est introuvable

2.2 Installation de l'OS

L'installation de l'OS a impliqué la récupération d'une ISO bootable (*futur lien vers la définition d'iso bootable*) de Debian 11 ou 12 (Debian 64-bit) depuis le site officiel de Debian (<https://www.debian.org/index.fr.html>). Nous avons choisi une installation automatisée pour simplifier le processus.

Les caractéristiques de la machine virtuelle ont été configurées dans un fichier nommé **preseed-fr.cfg**, et une fois l'installation terminée, l'ISO d'installation a été retirée. Voici à quoi ressemble l'intérieur du fichier de configuration automatique :

```
$ vboxpostinstall.sh  preseed-fr.cfg x S203-Debian12.viso
preseed-fr.cfg
53 d-i passwd/username string user
54 d-i passwd/user-password password user
55 d-i passwd/user-password-again password user
56 d-i passwd/user-default-groups string audio cdrom video
57
58
59 ### Packages, Mirrors, Image
60
61
62 ## Proxy : Obligatoire à l'université
63 d-i mirror/http/proxy string http://cache.univ-lille.fr:3128
64
65 #d-i base-installer/kernel/override-image string linux-server
66 #d-i base-installer/kernel/override-image string linux-image-amd64
67 d-i pkgsel/install-language-support boolean false
68 d-i pkgsel/include string sqlite3 sudo curl bash-completion git neofetch
69 ### Apt setup
70 # Pas de media supplémentaire sur cdrom
71 d-i apt-setup/disable-cdrom-entries boolean true
72 # Téléchargement de firmware non-libres, si nécessaire, sans demander
73 #d-i hw-detect/load_firmware boolean true
74
75 d-i apt-setup/contrib boolean true
76 d-i apt-setup/non-free boolean true
77 d-i apt-setup/services-select multiselect security, updates
78 #d-i apt-setup/restricted boolean true
79 #d-i apt-setup/universe boolean true
80
81 ## Installation meta-paquetages
82 # Tâches à installer (via des méta-paquetages)
83 # Lister les possibilités : tasksel --list-task (en ligne de commande)
84 # Utiliser au minimum "standard" est une bonne idée
85 tasksel tasksel/first multiselect standard ssh-server, mate-desktop
86
87
88 ### Suivi statistiques paquets installés
89 popularity-contest popularity-contest/participate boolean false
90
91
```

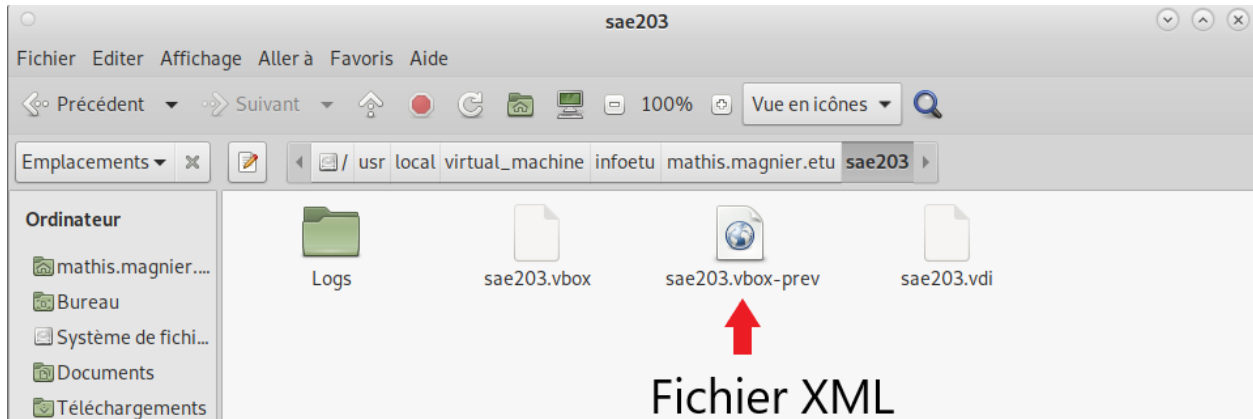
Comme vous pouvez le voir, nous avons apporté quelques modifications au fichier d'origine afin d'installer certain **paquets** par défaut sur la machine (sur la ligne **68**) notamment sqlite3, sudo neofetch ou encore Git. Et pour le plaisir visuel à la fin de la ligne **85** Dans le fichier preseed il nous faut rajouter “, **mate-desktop**” pour rajouter l'interface graphique Mate.

2.2.1 Question que vous pourriez vous poser ?

- Que signifie “64-bit” dans “Debian 64-bit”?
- Une architecture de 64 bits peut traiter des données par blocs de 64 bits à la fois, ce qui lui permet d'offrir beaucoup plus de mémoire qu'un système de 32 bits. Un système de 64 bits peut théoriquement traiter jusqu'à 18,4 millions de téraoctets (To) de mémoire. (<https://wiki.debian.org/fr/DebianInstall>)

Sur ce site, il y est expliqué ce qui est dit plus tôt, mais qu'il faut également un processeur capable de traiter des blocs de 64 bits à la fois pour que ce soit optimal.

- **Quelle est la configuration réseau utilisé par défaut par la machine?**
- La configuration réseau par défaut dans VirtualBox est souvent un adaptateur NAT.
- **Quel est le nom du fichier XML contenant la configuration de la machine?**
- Le nom du fichier XML est sae203.vbox



Afin de trouver le **fichier XML** il suffit de se rendre dans le dossier où l'on a créé la machine virtuelle et de trouver le fichier de type **XML**

- **Sauriez-vous le modifier directement pour mettre 2 processeurs à votre machine?**
- Oui, le fichier XML peut être modifié en ajoutant/modifiant l'élément `<vcpu>` pour spécifier le nombre de processeurs.

```

    <ExtraDataItem name="GUI/LastCloseAction" value="PowerOff"/>
    <ExtraDataItem name="GUI/LastGuestSizeHint" value="1920,963"/>
    <ExtraDataItem name="GUI/LastNormalWindowPosition" value="0,29,1920,991,max"/>
    <ExtraDataItem name="GUI/LastScaleWindowPosition" value="640,316,640,480"/>
  </ExtraData>
  <Snapshot uuid="{0630ec7e-34b7-469f-80ba-acc57093474f}" name="Instantan&#xE9; 1"
timeStamp="2023-10-12T05:16:07Z">
    <Hardware>
      <CPU>
        Ajouter Ici le <vcpu>
        <PAE enabled="false"/>
        <LongMode enabled="true"/>
        <X2APIC enabled="true"/>
        <HardwareVirtExLargePages enabled="true"/>
      </CPU>
      <Memory RAMSize="1024"/>
      <HID Pointing="USBTablet"/>
      <Display controller="VMSVGA" VRAMSize="16"/>
      <BIOS>
        <IOAPIC enabled="true"/>
        <SmbiosUuidLittleEndian enabled="true"/>
      </BIOS>
      <USB>
        <Controllers>
          <Controller name="OHCI" type="OHCI"/>
          <Controller name="EHCI" type="EHCI"/>
        </Controllers>

```

- Qu'est-ce qu'un fichier iso bootable?
- Un fichier ISO Bootable est conçu pour s'exécuter lorsque vous démarrez votre PC, il permet entre autres de démarrer votre ordinateur sans même qu'il y ait de système d'exploitation puisqu'il est bootable. Un fichier Iso est la copie conforme d'un disque optique, car il conserve une certaine structure comme sur un disque physique comme le dit le site Wikipedia (https://fr.wikipedia.org/wiki/Image__disque)
- Qu'est-ce qu'un serveur web?
- Un serveur web est un logiciel qui répond aux requêtes HTTP, permettant d'afficher des pages web sur un navigateur.

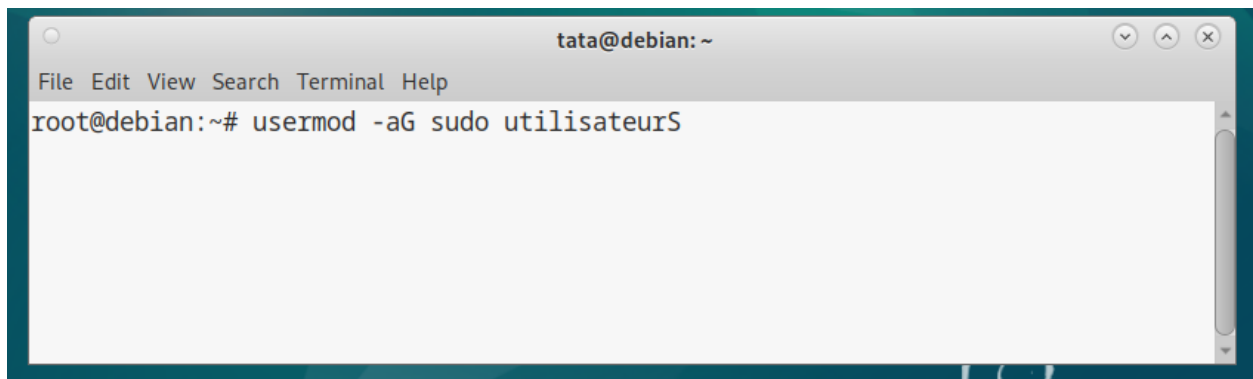
2.3 Préparation du système

La préparation du système a consisté à accorder l'accès sudo à l'utilisateur standard (l'accès sudo est l'équivalent du rôle administrateur de la machine). Les étapes incluaient le passage en mode console, la connexion en tant que root et l'ajout du groupe sudo à l'utilisateur principal.

Voici la page officiel debian qui documente **Sudo**.

Voici comment faire pour donner les droits Sudo à un utilisateur à partir d'un terminal : Tout d'abord, il faut ouvrir un terminal en mode Administrateur puis taper cette commande :

```
usermod -aG sudo utilisateur
```



et dans le **fichier automatique** de la VM

```

216
217 #|
218 # GAs
219 #
220
221 echo "-----" >> "${MY_LOGFILE}"
222 echo '** Installing VirtualBox Guest Additions...' | tee -a "${MY_LOGFILE}"
223 MY_IGNORE_EXITCODE=2 # returned if modules already loaded and reboot required.
224 log_command_in_target /bin/bash "${MY_CHROOT_CDROM}/vboxadditions/VBoxLinuxAdditions.run" --nox11
225 log_command_in_target /bin/bash -c "udevadm control --reload-rules" # GAs doesn't yet do this.
226 log_command_in_target /bin/bash -c "udevadm trigger" # (ditto)
227 MY_IGNORE_EXITCODE=
228 log_command_in_target usermod -a -G vboxsf "user"
229 log_command_in_target usermod -aG sudo "user"
230

```

2.4 Suppléments invités

Pour améliorer l'intégration, les suppléments invités ont été installés, impliquant le montage du CD des suppléments et l'exécution de l'installation. La version du noyau Linux utilisé par la VM a été vérifiée.

Question 3

- **Comment peut-on savoir à quels groupes appartient l'utilisateur user?**
- *Réponse:* La commande `groups user` permet de connaître les groupes auxquels appartient l'utilisateur user dans cet exemple.

Question 4

- **Quelle est la version du noyau Linux utilisé par votre VM?**
- Utiliser la commande `uname -r` pour obtenir la version du noyau.
- **À quoi servent les suppléments invités?**
- Les suppléments invités améliorent l'intégration entre la machine hôte et virtuelle, fournissant des fonctionnalités comme le redimensionnement automatique de la fenêtre.
- **À quoi sert la commande mount (dans notre cas de figure et dans le cas général)?**
- Dans notre cas, elle sert à monter le CD des suppléments invités. En général, la commande `mount` est utilisée pour attacher un système de fichiers à l'arborescence.

[1] # Markdown

2.5 Apprentissage du markdown

Pour commencer, comme aucun de nous n'avais jamais fait de Markdown nous somme allé sur un éditeur en ligne nommé **Dillinger** afin de découvrir les balises de base à connaître. En effet, le langage Markdown est un langage à balisage léger, il faut donc utiliser des balise afin d'apporter du traitement au texte.

Voici un tableau reprenant toute les balises de base du Markdown

Traitement	balise	exemple
mettre en italique	<code>* *</code>	<i>ceci est un texte en italique</i>
mettre en gras	<code>** **</code>	ceci est un texte en gras
mettre en italique + en gras	<code>*** **</code>	<i>ceci est un texte en italique + gras</i>
surligner	<code>== ==</code>	<u>ceci est un texte surligné</u>
Titre de niveau 1	<code>#</code>	les exemple pour les titres seront en dessous car ils ne fonctionnent pas en tableau
Titre de niveau 2	<code>##</code>	
Titre de niveau 3	<code>###</code>	
Titre de niveau 4	<code>####</code>	
Titre de niveau 5	<code>#####</code>	
Titre de niveau 6	<code>#####</code>	
mettre un lien	<code>texte</code>	exemple
mettre une image	<code>alt</code>	exemple
faire une liste	<code>-</code>	-exemple1

ceci est un titre de niveau 1

ceci est un titre de niveau 2

ceci est un titre de niveau 3

ceci est un titre de niveau 4

ceci est un titre de niveau 5 ceci est un titre de niveau 6

2.6 Exportation avec Pandoc

Pour exporter tout fichier vers un autre format avec l'application Pandoc, il suffit de rentrer la ligne de commande suivant dans un terminal dans le dossier qui contient le fichier en Markdown :

```
pandoc Titre.md -o Titre. Nouvelle_extension
```


La commande que nous avons utilisé nous pour exporter ce fichier markdown en pdf est celle ci-dessous :

```
pandoc -f markdown -t pdf -N -s Rapport.md metadata.yaml -o Rapport.pdf --toc -V geometry:margin=1in
```

les options:

Options	correspondance	Utilité
<code>-f</code>	from	permet de signifier le type du fichier de départ
<code>-t</code>	to	permet de signifier dans quel type nous souhaitons le fichier de sortie

Options	correspondance	Utilité
-N	Number	permet de numérotéer chaque partie du rapport pour différencier les points des autres
-s	standalone	Permet, si il y a des images ou des metadonnées au fichier de tout transporter en lui toute les images et metadonnés
-o	output	permet de choisir le nom que le fichier transformé portera
-toc	table of content	permet de créer automatiquement un sommaire pour le fichier lors de la conversion
-V		Permet d'ajouter des modification sur l'affichage, dans notre cas il permet de resuire les marge de chaque coté du fichier pdf

Biensur il existe bien d'autre option que vous pouvez utilisez avec pandoc que vous pouvez retrouver directement sur le manuel de pandoc ci-joint Manuel pandoc 

3 Études d'applications clientes

3.1 Configuration de Git

Maintenant que l'on a tout les packets d'installer nous pouvons configurer Git.
Tout d'abords, nous allons configurer le git à notre nom, mail, branche par défaut.

Configurer le nom	Configurer l'email	Configurer la branche par défaut
git config --global user.name "Prénom Nom"	git config --global user.email "votre@email"	git config --global init.defaultBranch "master"

Questions

Qu'est-ce que le logiciel gitk ? Comment se lance-t-il ? - Effectuer la commande "sudo apt install gitk" dans le terminal - Gitk est un outil graphique intégré à Git qui offre une vue historique interactive des changements dans un dépôt, permettant la visualisation des branches, des commits et des différences entre les versions. Il s'utilise généralement en ligne de commande avec la commande "gitk".

Voici sa documentation **Gitk**.

Qu'est-ce que le logiciel git-gui ? Comment se lance-t-il ? - Effectuer la commande "sudo apt-get install git-gui" dans le terminal - Git-GUI est une interface graphique pour Git, facilitant les opérations de contrôle de version via une interface utilisateur visuelle. Elle permet de créer des dépôts, gérer des branches, effectuer des commits, etc.

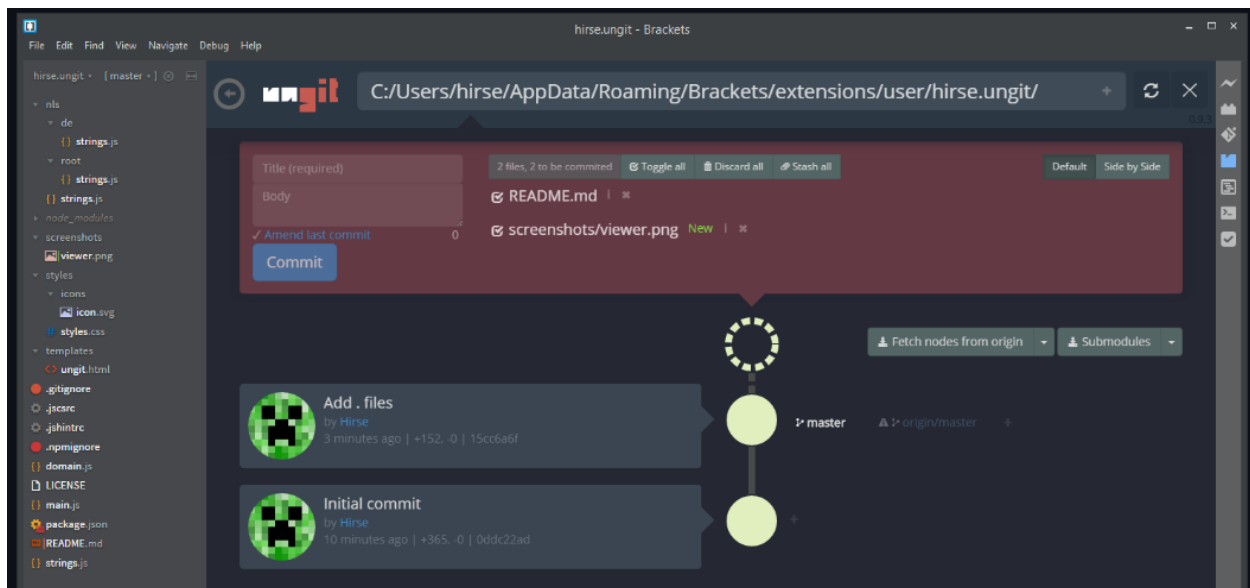
Comparons avec une autre interface

Nous devons d'abord aller sur le site git-scm.com puis choisir une interface pour comparer avec git-gui ou gitk.

Pour notre part, nous avons choisi "ungit", car celui-ci est simple d'utilisation et ergonomique. L'installation se fait en suivant ces commandes: - sudo apt install npm - npm install -g ungit

Ensuite, vous pouvez lancer ungit avec un simple "ungit" dans votre terminal.

Vous aurez une interface comme montrer ci-dessous



3.1.0.1 Comparaison de git-gui et gitk à ungit

gitk et git-gui :

Type : Outils graphiques intégrés à Git.

Utilité : gitk pour visualiser l'historique des commits, git gui pour des opérations administratives.

Lancement : Depuis la ligne de commande avec gitk ou git gui.

ungit :

Type : Interface graphique tierce.

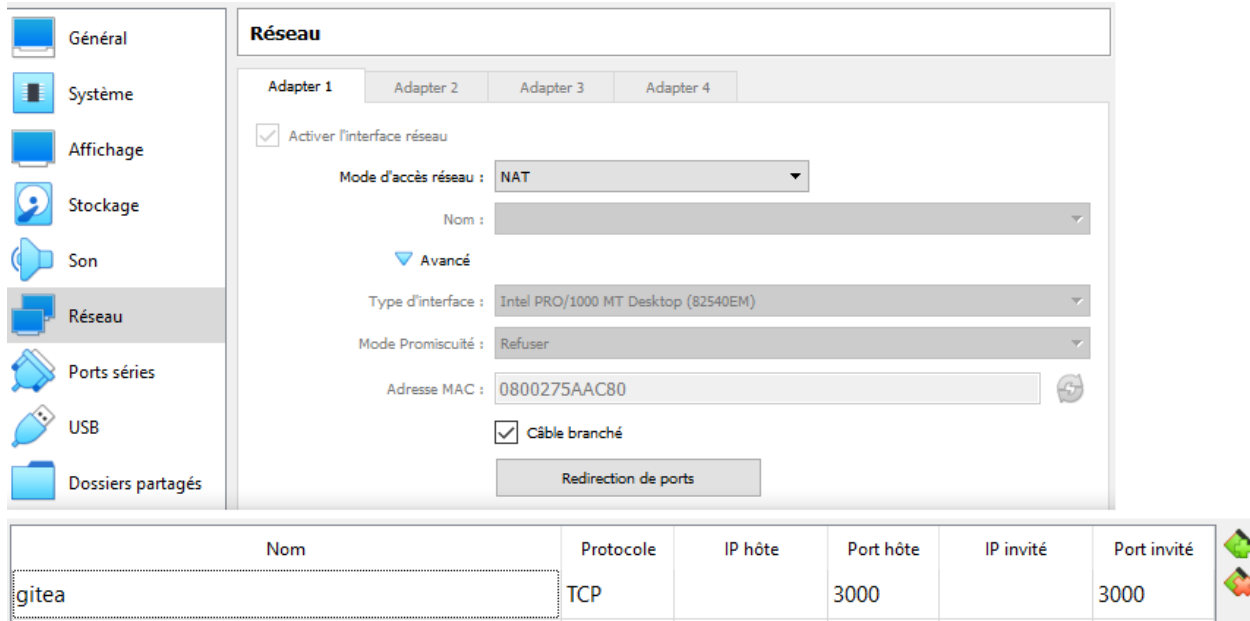
Utilité : Alternative visuellement agréable à Git, gestion intuitive.

Lancement : Application externe, installation séparée.

4 Installation du service Gitea

Premièrement, nous avons changé la direction de port de notre VM afin que le port de la machine hôte soit le même que celui de la VM. Cela a été fait dans le but d'utiliser gitea sans utiliser le terminal de la VM, mais en étant sur internet grâce au lien <http://localhost:300> sur la machine hôte.

Tous les détails se trouvent ici : Manuel VirtualBox“““



4.1 Qu'est ce que Gitea?

Gitea est un “forge software” permettant d'héberger des différents codes source. C'est le même principe que GitHub sauf qu'il a des atouts qui sont: - Il n'est pas basé sur le cloud, permettant donc aux entreprises ou aux utilisateurs d'avoir plus de contrôles sur leurs dépôts et leurs données. - Gestion des droits des utilisateurs

Voici sa documentation: [doc Gitea](#)

4.2 À quels logiciels bien connus dans ce domaine peut-on le comparer ?

On pourrait comparer le logiciel Gitea à GitHub, GitLab et Bitbucket, car ils offrent des fonctionnalités similaires telles que le suivi des problèmes, la gestion des versions, la collaboration et le contrôle d'accès.

4.3 Installation de Gitea

Passons à l'installation de Gitea sur la VM avec sqlite3 qui était préalablement installé. Nous nous sommes tout d'abord assuré que les paquets étaient à jour sur le serveur en utilisant les commandes `apt update` et `apt upgrade`.

Ensuite, il fallait créer un utilisateur système en utilisant cette commande: - `adduser` \ C'est la commande principale pour ajouter un nouvel utilisateur sur Linux.

- `--system` \ -- Cette option crée un utilisateur système plutôt qu'un utilisateur ordinaire.
- `--shell /bin/bash` \ -- Permet de définir le shell par défaut comme bash
- `--gecos 'Gitea Version Control'` \ -- Description de l'utilisateur.
- `--group` \ --" Crée un groupe du même nom que l'utilisateur, ici “git”
- `--disabled-password` \ -- Pas de mot de passe pour l'utilisateur.
- `--home /home/git` \ -- Définit le répertoire personnel de l'utilisateur.
- `git` -- Le nom de l'utilisateur créé.

4.3.1 Installation du binaire

Puis il a fallu installer le binaire depuis le site Gitea. On pouvait l'installer directement depuis le site ou bien en ligne de commandes. Nous avons privilégié directement depuis le terminal avec cette commande pour le mettre dans le répertoire `/usr/local/bin`:

```
wget https://dl.gitea.com/gitea/1.21.7/gitea-1.21.7-linux-amd64 -O ==/usr/local/bin/gitea==
```

A la suite, nous avons collé cette suite de commandes dans le terminal: `- mkdir -p /var/lib/gitea/{custom,data,log}`
-- Crée un répertoire principal '`var/lib/gitea`' ainsi que trois sous-répertoires `{custom,data,log}` pour stocker des fichiers de configuration, des données de l'applications et des fichiers de grande taille.

- `chown -R git:git /var/lib/gitea/` -- Change le propriétaire et le groupe du répertoire et de tous ses sous-répertoires et fichiers pour l'utilisateur et le groupe "git" pour que Gitea puisse accéder et modifier ces fichiers.
- `chmod -R 750 /var/lib/gitea/` -- cette commande définit les permissions sur le répertoire et ses sous-répertoires afin que l'utilisateur propriétaire puisse lire, écrire et exécuter. Les autres utilisateurs n'ont aucun droit.
- `mkdir /etc/gitea` -- Crée un répertoire pour stocker les fichiers de configurations de Gitea.
- `chown root:git /etc/gitea` -- Change le propriétaire du répertoire à root et au groupe git.
- `chmod 770 /etc/gitea` -- Donne les droits du propriétaire (root) d'exécuter, écrire et exécuter.

Voici la page d'aide d'installation de Gitea: [documentation installation Binary](#)

4.3.2 Création du fichier de configuration Gitea

S'en est suivi la création du fichier de configuration de Gitea avec la commande.

- `nano /etc/gitea/app.ini` et nous avons mis ces lignes dans:

- `DOMAIN = gitea.yourdomain.com` --

Définis le domaine sur lequel notre instance de Gitea va être accessible.

- `ROOT_URL = http://gitea.yourdomain.com/` --

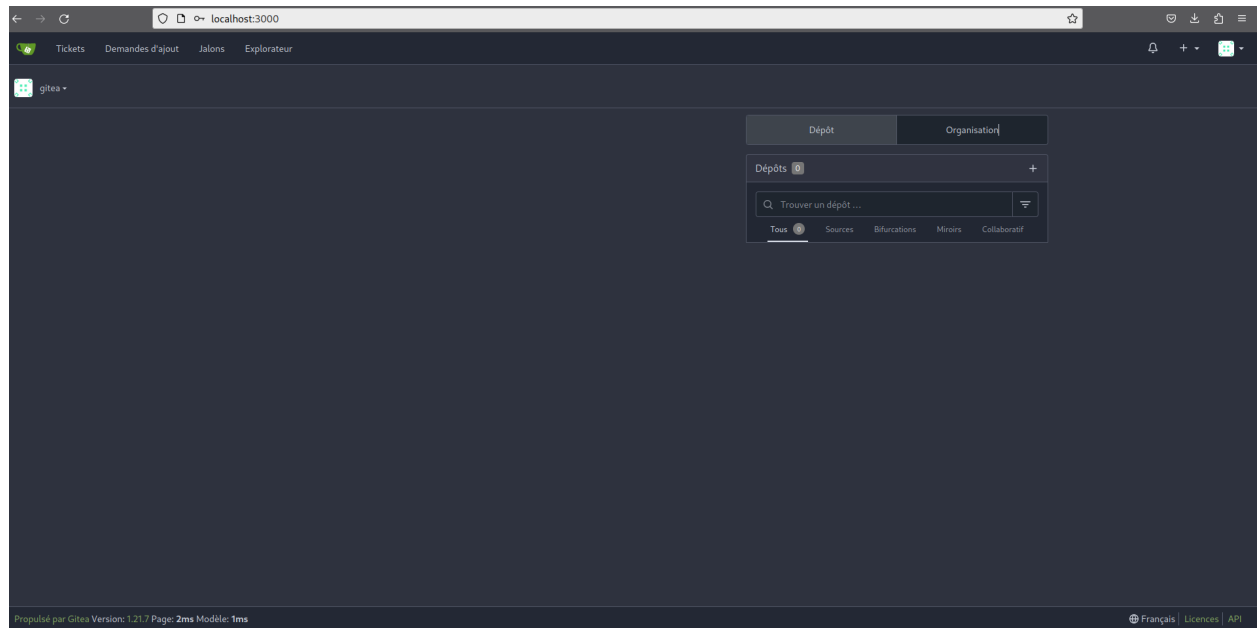
Enfin, il a fallu donner la propriété à l'utilisateur git. `chown git. /etc/gitea/app.ini`

L'installation de Gitea est maintenant terminée. Pour le lancer nous avons utilisé cette commande - `GITEA_WORK_DIR=/var/lib/gitea/ /usr/local/bin/gitea web -c /etc/gitea/app.ini`.

Le serveur se lance et il y a toutes ces informations qui s'affichent.

```
Applications Emplacements Systeme
user@S203:~$ ./gitea web kill
2024/03/14 10:44:02 cmd/web.go:242:runWeb() [I] Starting Gitea on PID: 2445
2024/03/14 10:44:02 cmd/web.go:111:showWebStartupMessage() [I] Gitea version: 1.21.7 built with GNU Make 4.3, go1.21.7 : bindata, sqlite, sqlite_unlock_notify
2024/03/14 10:44:02 cmd/web.go:112:showWebStartupMessage() [I] * RunMode: prod
2024/03/14 10:44:02 cmd/web.go:113:showWebStartupMessage() [I] * AppPath: /home/user/gitea
2024/03/14 10:44:02 cmd/web.go:114:showWebStartupMessage() [I] * WorkPath: /home/user
2024/03/14 10:44:02 cmd/web.go:115:showWebStartupMessage() [I] * CustomPath: /home/user/custom
2024/03/14 10:44:02 cmd/web.go:116:showWebStartupMessage() [I] * ConfigFile: /home/user/custom/conf/app.ini
2024/03/14 10:44:02 cmd/web.go:117:showWebStartupMessage() [I] Prepare to run web server
2024/03/14 10:44:02 routers/init.go:112:InitWebInstalled() [I] Git version: 2.39.2, Wire Protocol Version 2 Enabled (home: /home/user/data/home)
2024/03/14 10:44:02 ...les/setting/cache.go:75:loadCacheFrom() [I] Cache Service Enabled
2024/03/14 10:44:02 ...les/setting/cache.go:90:loadCacheFrom() [I] Last Commit Cache Service Enabled
2024/03/14 10:44:02 ...les/setting/session.go:74:loadSessionFrom() [I] Session Service Enabled
2024/03/14 10:44:02 ...s/storage/storage.go:176:initAttachments() [I] Initialising Attachment storage with type: local
2024/03/14 10:44:02 ...les/storage/local.go:33:NewLocalStorage() [I] Creating new Local Storage at /home/user/data/attachments
2024/03/14 10:44:02 ...s/storage/storage.go:166:initAvatars() [I] Initialising Avatar storage with type: local
2024/03/14 10:44:02 ...les/storage/local.go:33:NewLocalStorage() [I] Creating new Local Storage at /home/user/data/avatars
2024/03/14 10:44:02 ...s/storage/storage.go:192:initRepoAvatars() [I] Initialising Repository Avatar storage with type: local
2024/03/14 10:44:02 ...les/storage/local.go:33:NewLocalStorage() [I] Creating new Local Storage at /home/user/data/repo-avatars
2024/03/14 10:44:02 ...s/storage/storage.go:186:initLFS() [I] Initialising LFS storage with type: local
2024/03/14 10:44:02 ...les/storage/local.go:33:NewLocalStorage() [I] Creating new Local Storage at /home/user/data/lfs
2024/03/14 10:44:02 ...s/storage/storage.go:198:initRepoArchives() [I] Initialising Repository Archive storage with type: local
2024/03/14 10:44:02 ...les/storage/local.go:33:NewLocalStorage() [I] Creating new Local Storage at /home/user/data/repo-archive
2024/03/14 10:44:02 ...s/storage/storage.go:208:initPackages() [I] Initialising Packages storage with type: local
2024/03/14 10:44:02 ...les/storage/local.go:33:NewLocalStorage() [I] Creating new Local Storage at /home/user/data/packages
2024/03/14 10:44:02 ...s/storage/storage.go:219:initActions() [I] Initialising Actions storage with type: local
2024/03/14 10:44:02 ...les/storage/local.go:33:NewLocalStorage() [I] Creating new Local Storage at /home/user/data/actions_log
2024/03/14 10:44:02 ...s/storage/storage.go:223:initActions() [I] Initialising ActionsArtifacts storage with type: local
2024/03/14 10:44:02 ...les/storage/local.go:33:NewLocalStorage() [I] Creating new Local Storage at /home/user/data/actions_artifacts
2024/03/14 10:44:02 routers/init.go:131:InitWebInstalled() [I] SQLite3 support is enabled
2024/03/14 10:44:02 routers/common/db.go:23:InitDBEngine() [I] Beginning ORM engine initialization.
2024/03/14 10:44:02 routers/common/db.go:30:InitDBEngine() [I] ORM engine initialization attempt #1/10...
2024/03/14 10:44:02 cmd/web.go:194:serveInstalled() [I] PING DATABASE sqlite3
2024/03/14 10:44:02 cmd/web.go:104:serveInstalled() [I] Table custom_setting.Column version db default is ... struct default is 1
```

Suite à cela nous avons pu lancer une première fois le service Gitea en collant le lien <http://localhost:3000> sur Internet. Ce qui nous a amené à cette page où l'on a du configurer l'administrateur comme dans les consignes. Une fois fait, on est redirigé sur la page d'accueil ci-dessous.



Quelle version du binaire avez-vous installé ? Donnez la version et la commande permettant d'obtenir cette information.

Pour vérifier la version du binaire installé, nous avons utilisé cette commande:

- `/usr/local/bin/gitea --version`

Elle nous a bien renvoyé la version installée de Gitea qui était la 1.21.7 sous la forme:

Gitea version 1.21.7 built with GNU Make 4.3, go1.21.7 : bindata, sqlite, sqlite_unlock_notify.

Comment faire pour mettre à jour le binaire de votre service sans devoir tout reconfigurer ? Essayez en

mettant à jour vers la version 1.22-dev.

Pour mettre à jour sans devoir tout reconfigurer il a fallu installer le binaire souhaité, c'est à dire la version 1.22-dev. Avant de procéder à la mise à jour, nous avons dû nous assurer que le fichier binaire était exécutable avec la commande

- `chmod +x /Téléchargements/gitea`

Ensuite, nous avons arrêté le service Gitea avec la commande

- `killall gitea`

renommé l'ancien Gitea en Gitea.old avec `mv /usr/local/bin/gitea /usr/local/bin/gitea.old`. Enfin nous avons déplacé le nouveau binaire `mv /Téléchargements/gitea /usr/local/bin/gitea`. Puis on l'a exécuté avec la commande `./gitea`.

Enfin, nous avons créé notre premier projet Gitea comme suit.

L'accueil de ce projet se présente comme ceci.

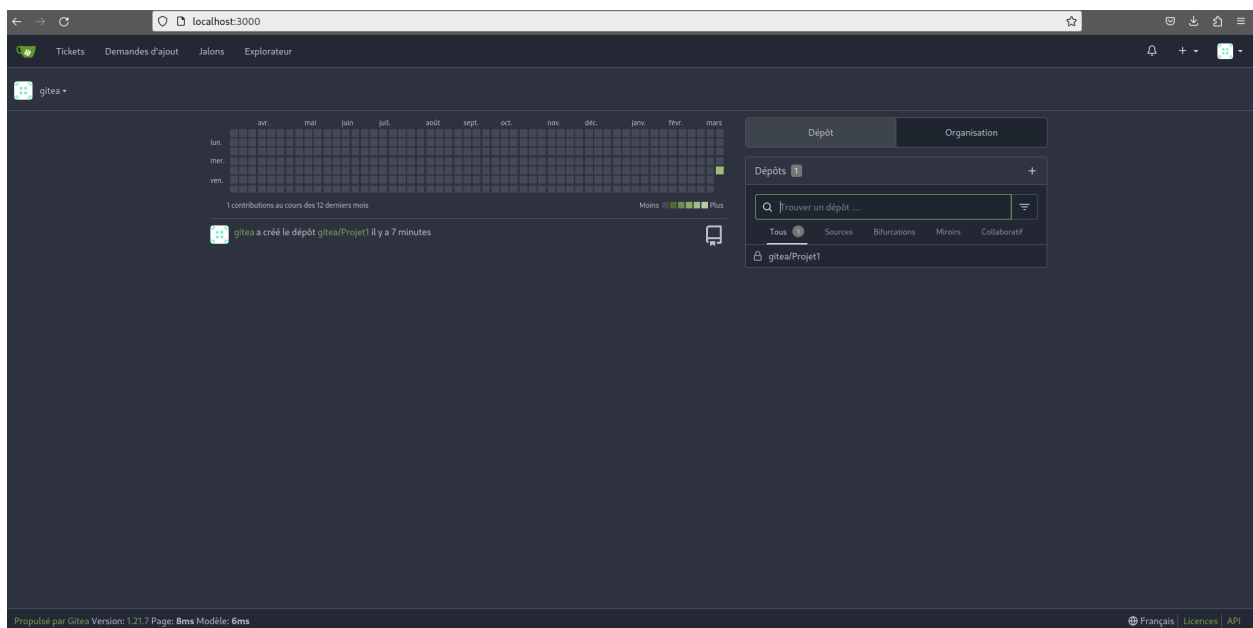


Figure 2: L'image est introuvable

Lorsqu'on dépose des fichiers on peut créer des dossiers etc.

5 Les problèmes rencontrés

Durant cette Saé, nous avons rencontré plusieurs problèmes plus ou moins problématiques.

- Au tout début de la Saé, au moment de l'installation des paquets pour notre machine virtuelle automatique nous nous étions trompé de fichier de pré configuration et nous avons mis l'installation des paquets dans le fichier config.sh, alors qu'il fallait les mettre dans le preseed-fr.cfg, ce qui nous a valu environ 30 minutes de recherche du problème afin de pouvoir le résoudre
- Ensuite avec le Markdown [^1], afin de pouvoir surligner le texte nous avons parcouru les sites d'apprentissage du markdown en ligne, car la seule balise que nous avions était `==texte==` mais elle ne fonctionnait pas à l'affichage et pas non plus dans le pdf ni l'html donc la seule solution que nous avons eu (si nous y arrivons) est d'utiliser une template afin de passer en fluo tous les textes dans ces balises

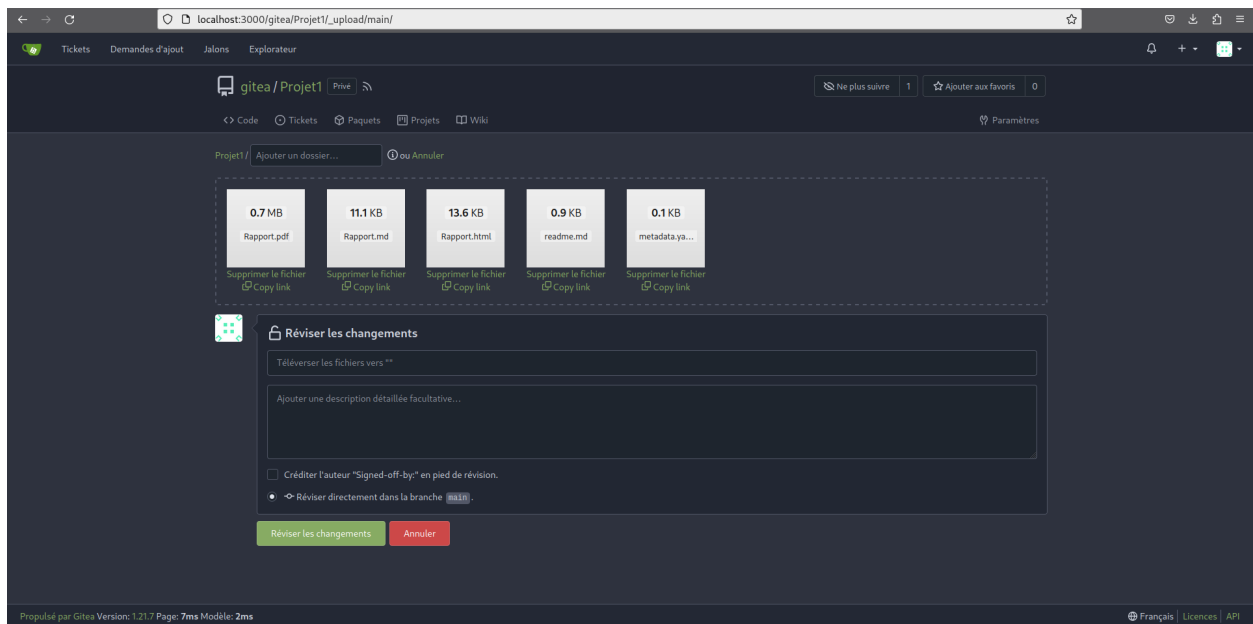


Figure 3: L'image est introuvable

- Et pour finir avec Gitea qui nous a posé le plus de problème et qui nous à pris le plus de temps car nous n'arrivions pas à démarrer le site à cause du fait que le fichier **app.ini** n'existait tout simplement pas, nous avons donc créé ce fichier et entré les informations qu'il devait y avoir à l'intérieur à l'aide d'un github

6 Conclusion

Pour conclure durant cette Saé nous avons vu:

- Comment installer et automatisé une machine virtuelle ? Pour ceci, nous avons modifié les fichiers de configuration qui servent de base pour la machine.
- Nous avons aussi vu l'utilisation de git, un logiciel permettant de stocker et d'échanger des fichiers/projets sur un cloud.
- L'utilisation d'application cliente telle que Gitea, git-gui, gitk fut aussi un des points importants de ce projet