Introduction aux systèmes GNU/Linux

S23E03 inetdoc.net

UPSSITECH
ÉCOLE D'INGÉNIEURS

Philippe Latu / Université Toulouse 3

Document sous licence GNU FDL v1.3
http://www.gnu.org/licenses/fdl.html

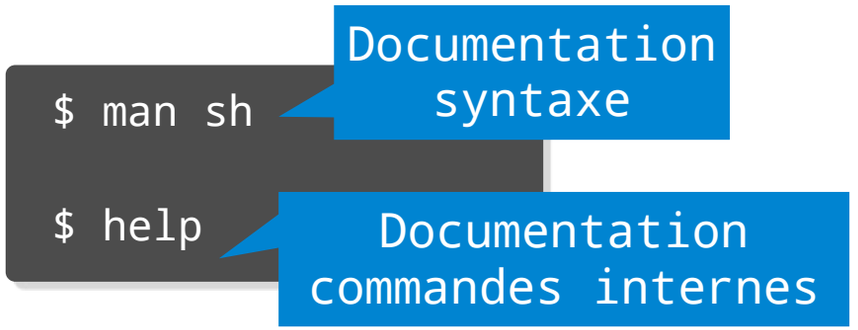
Objectifs de la séance

- Configurer un Système GNU/Linux
  - Utiliser les ressources du shell Bash
  - Identifier & gérer les processus
  - Gérer les permissions sur les fichiers et répertoires
  - Différencier les processus entre système hôte & conteneur

- Manipuler sur machine virtuelle & conteneur
  - Installer & configurer des services Internet
  - Repérer les propriétaires & les propriétés des processus correspondant

Bash → Shell et langage de commandes

- 1979 → SysV version 7 → Bourne Shell
- 1989 → GNU BASH → Bourne-again Shell → scripts
  - Interpréteur de commandes
    - IEEE POSIX P1003.2/ISO 9945.2 Shell and Tools standard
  - Environnement de développement
    - Algorithmique de base : conditions, boucles, etc.
    - Fonctions et alias
    - Arithmétique et tableaux
    - Manipulations de chaînes de caractères
  - Documentation shell Bash
    - Advanced Bash Scripting
      - http://www.tldp.org/LDP/abs/html/



Bash → séquencement entre les commandes

- Éditions et corrections en ligne de commande
  - Auto-correction → commande `shopt -s dirspell`
  - Auto-completion → touche tabulation
  - Historique et rappel des commandes antérieures
    - Commande `history`
    - Séquences de touches
      - Ctrl+R, Ctrl+A, Ctrl+E
      - Shift+PageUp, Shift+PageDown, flèches haut et bas

Bash → séquencement entre les commandes

- Séquencement entre les opérations
  - Tubes ou *pipes* → |
  - Redirections → > → >>
  - Enchaînements logiques → ; → && → ||
  - Gestionnaire de fenêtres ou de tâches console
    - Tmux, byobu, VSCode
  - Commandes internes → jobs, suspend

Bash → exemple de script

- Script checkup.sh : gestion des machines virtuelles

```
#!/bin/bash

set -e

GreenOnBlack='\E[32m'

echo "~> Check-up gestion des machines virtuelles"

# Arborescence
if [ ! -L ~/masters ]; then
    echo "Création du lien vers le catalogue des images de machines virtuelles."
    ln -s /var/cache/kvm/masters ~
fi

if [ ! -d ~/vm ]; then
    echo "Création du dossier des images de machines virtuelles."
    mkdir ~/vm
    ln -s ~/masters/scripts ~/vm/
fi

echo -e "${GreenOnBlack}~> Arborescence prête."
tput sgr0
```

Shebang

Affectation directe d'une variable

Test de la présence d'un lien symbolique

Test de la présence d'un dossier

Shell Bash

- Script checkup.sh : gestion des machines virtuelles

```
# Analyse des fichiers image
count=0
listFilename="$HOME/image-list-info.json"

echo -n "[" > $listFilename
imageList=$(find ~ -type f \( -iname \*.qcow2 -o -iname \*.raw \) -printf '%p ')
for file in $imageList
do
    echo "$((++count)) : $file"
    if [ -z "$(pgrep -u ${USER} | grep -o "${file##*/}\ ")" ]; then
        qemu-img info --output=json --backing-chain $file | tr -d '[]' >> $listFilename
        sed -i '/^$/d' $listFilename && sed -i '$s/$/\,/' $listFilename
    fi
done
sed -i '$ s/,/\n]/g' $listFilename

echo -e "${GreenOnBlack}~> $count fichier(s) image(s) présent(s)."
tput sgr0
```

Nom du fichier de rapport d'analyse

Boucle de traitement des fichiers image

Recherche du nom de fichier dans les processus actifs

Suppression ligne vide

Ajout virgule en fin de fichier

Shell Bash

- Script checkup.sh : gestion des machines virtuelles

```
# Liste des machines virtuelles actives
IFS=$'\n'
count=0

vmArray=( $(ps aux | grep ${USER} | grep 'qemu-system-x86_64\ ' || true) )
if [ ${#vmArray[@]} -gt 0 ]; then
    for vm in "${vmArray[@]}"
    do
        echo -n "$((++count)) : "
        echo $vm | grep -Po '(?<=-name\ ).*(?=-m)' | tr -d '\n'
        echo -n "utilise le cordon "
        echo $vm | grep -Po 'tap\d{1,3}'
    done
fi

echo -e "${GreenOnBlack}~> $count machine(s) virtuelle(s) active(s)."
tput sgr0

exit 0
```

Redéfinition du délimiteur de champs

Création d'un tableau pour chaque processus

Boucle de parcours des éléments du tableau des processus

Nom de machine virtuelle

Numéro du cordon de brassage

Bash → Exercices

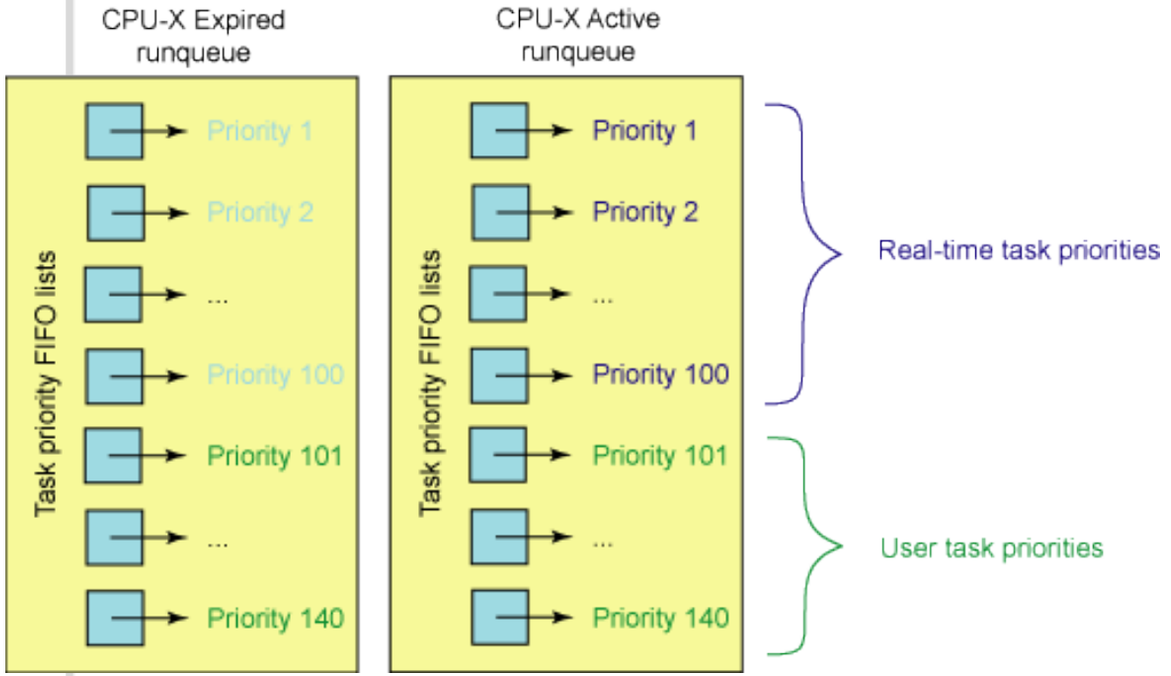
- Créer et tester le script checkup.sh
- Comment lister l'historique des commandes ?
- Quel est l'effet de la séquence de touche Ctrl+E ?
- Comment rappeler une commande dans l'historique ?
- Quel est l'effet de la séquence de touche Ctrl+D ?
- Comment accéder à la documentation de la commande intégrée shopt ?
- Comment activer l’auto-correction à l’ouverture d’un shell Bash ?
  - https://www.gnu.org/software/bash/manual/html_node/The-Shopt-Builtin.html
- Comment activer la personnalisation Oh My Bash ?
  - https://github.com/ohmybash/oh-my-bash

Processus → programme en cours d'exécution

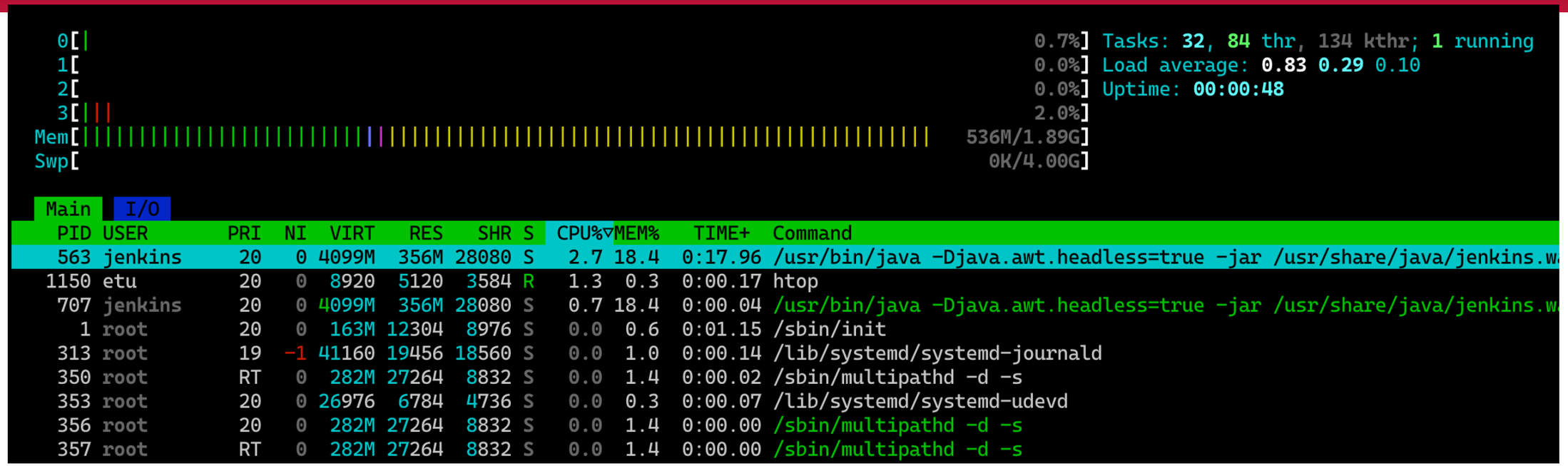
- Fonction Unix de base

  Partage des ressources du système entre différents programmes
- Ordonnanceur (*Scheduler*)

  Attribution des tranches de temps processeur en fonction du type de tâche
- Fonctions multi-tâches préemptives du noyau Linux
  - Planification de l'exécution des processus
  - Contrôle au début et à la fin de chaque tranche de temps processeur



Processus → quelles propriétés ?



- Gestion des processus
  - Qui est le propriétaire d'un processus ?
  - Quelles sont les ressources utilisées par un processus ?
  - Comment changer le niveau de priorité d'un processus ?
  - Comment tuer un processus défectueux ?

Processus → quelles commandes ?

- Commandes de gestion des processus
  - Qui, quoi, combien : ps, w, top, htop, iotop
  - Priorités : nice, renice, ionice
  - Signalisation et arrêt : kill, killall
  - Utilisation mémoire : free -m, vmstat -w -a -S M, vmstat -s

Processus → exercices

- Commande ps
  - Comment visualiser les processus, les propriétaires et les terminaux ?
  - Quelle est la signification des 4 options dans la commande ‘ps faux’ ?
- Commandes kill et killall
  - Quelle est la signification du terme signal ?
  - Comment relancer un processus ?
  - Comment tuer un processus «en force» ?
- Processus ou services inutiles
  - Comment supprimer un service inutile de façon permanente ?
  - Comment caractériser le gain en occupation mémoire correspondant ?

Permissions sur le système de fichiers

- Un masque de permissions pour tout fichier ou répertoire
  - Exemple : commande ls -lA

```
$ ls -lA
drwx------  2  etu etu    4096 Jan  1 00:04  mail
drwxrwxr-x  3  etu etu    4096 Nov 18 09:54  public_html
-rw-rw----  1  etu etu  136430 Feb  6 16:52  trash.file
```

permissions | owner | group | size | date & time | name

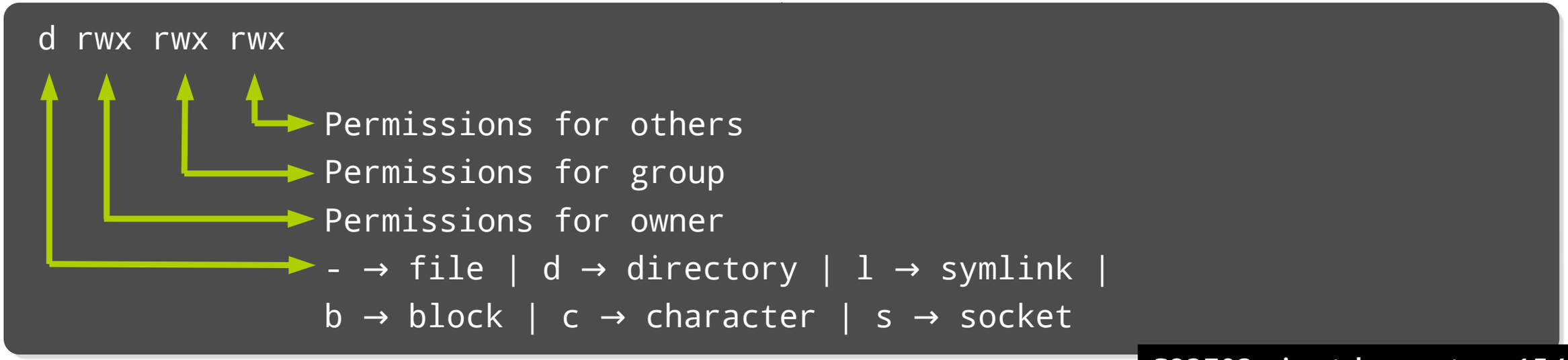
- Un utilisateur peut appartenir à plusieurs groupes
  - Exemple : commande id

```
$ id
uid=1000(etu) gid=1000(etu) groups=1000(etu),29(audio)
```

private uid & gid | public group

Permissions sur le système de fichiers

- Masque des permissions de base → 10 indicateurs
  - Partant de la gauche
    - Premier indicateur → nature de l'objet
      - fichier, répertoire, périphérique ou socket Unix
    - Autres indicateurs → droits
      - lecture, écriture, exécution
      - Propriétaire, groupe et autre



Permissions sur le système de fichiers

- Visualisation/Édition du masque des permissions
  - Commandes usuelles
    - `ls` → visualisation
    - `chown`, `chgrp` → changement de propriétaire ou de groupe
    - `chmod` → changement de masque
    - `umask` → masque utilisateur utilisé par défaut lors de la création d'objets
  - Codage des permissions
    - Notation littérale
      - r → read → droit de lecture
      - w → write → droit d'écriture
      - x → execute → droit d'exécution
    - Notation en octal
      - r → 2^2 → 4
      - w → 2^1 → 2
      - x → 2^0 → 1

```
$ touch emptyfile
$ ls -l emptyfile
-rw- r-- r-- 1 etu etu 0 mai   16 11:20 emptyfile
 420 400 400
   6   4   4 → 644
$ chmod +x emptyfile
-rwx r-x r-x 1 etu etu 0 mai   16 11:20 emptyfile
 421 401 401
   7   5   5 → 755 → chmod +x = chmod 755
```

Permissions sur le système de fichiers

- Applications
  - À l'aide de la commande 'ls', donner un exemple de :
    - programme exécutable
    - lien symbolique
    - périphérique en mode caractère
    - périphérique en mode bloc
    - socket Unix
  - Permissions sur les fichiers et répertoires
    - Donner les valeurs numériques des masques d'un fichier de données et d'un programme
    - Quel est l'effet de l'instruction 'umask 0027' ?
  - Shell script Bash «Hello, World!»
    - Créer le fichier script hello.sh et le rendre exécutable

```
#!/bin/bash

echo "Hello, World!"
```

Permissions sur le système de fichiers

- Masque étendu : 3 bits supplémentaires → extension des permissions
  - SUID : Set User ID bit
  - SGID : Set Group ID bit
  - directory Sticky bit
- Ces 3 bits prennent la place du bit d'exécution x
  - Pour le propriétaire du fichier
    - s indique qu'il a aussi le droit d'exécution
    - S indique qu'il n'a pas le droit d'exécution
  - Pour le groupe du fichier
    - s indique qu'il a aussi le droit d'exécution
    - S indique qu'il n'a pas le droit d'exécution
  - Directory Sticky bit
    - Utile pour les répertoires partagés
    - Un utilisateur ne peut effacer que les fichiers qu'il a créé

Permissions sur le système de fichiers

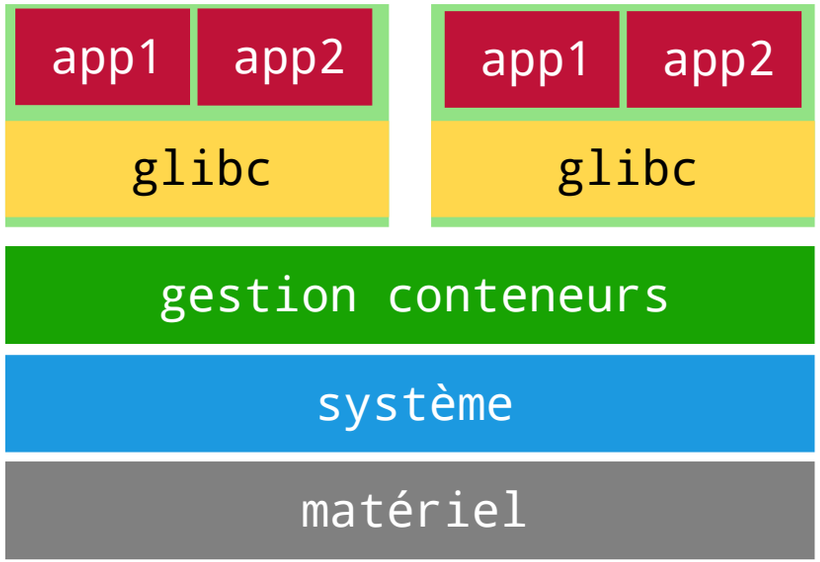
- Applications
  - Quel est le rôle du masque étendu pour les objets suivants ?

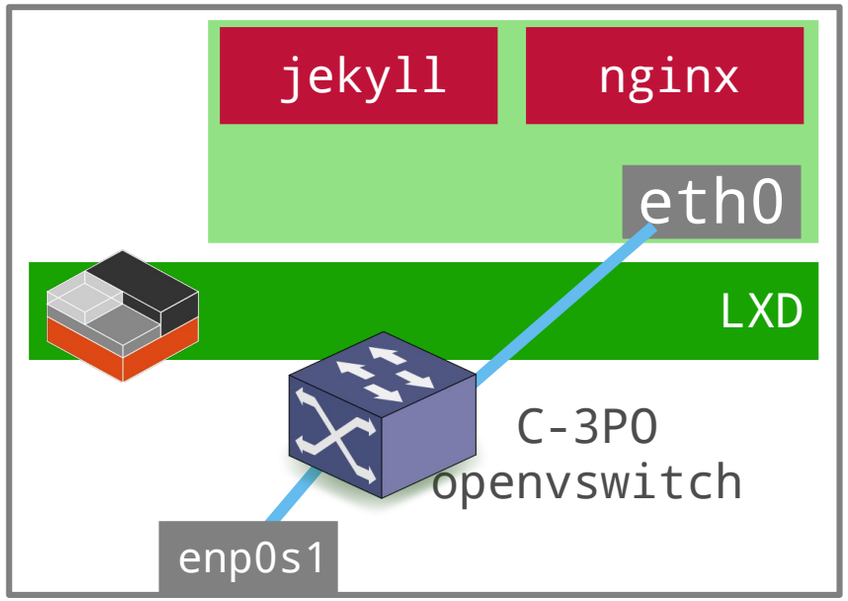
    /usr/bin/passwd et /tmp
  - Comment configurer un environnement de développement Web statique ?
    - Installer le paquet task-web-server
    - Fixer la valeur du masque utilisateur à 0027
    - Créer le répertoire /var/www/newhtml avec le masque rwxr-s---
      L'utilisateur propriétaire doit être etu et le groupe propriétaire www-data
    - Créer un fichier index.html dans ce nouveau répertoire
    - Ajouter une entrée newsite dans /etc/hosts avec l'adresse 127.0.0.2
    - Copier le fichier /etc/apache2/sites-available/000-default.conf en newsite.conf
    - Activer le nouveau site avec la commande a2ensite

      Recharger la configuration du service
    - Tester l'ouverture de la page Web et retrouver les traces dans les journaux du service Web

Application → Cas jekyll

- Objectifs
  - Transformer le système hôte en commutateur/routeur réseau
  - Installer de gestionnaire de conteneurs LXD
  - Créer un conteneur ‘jekyll’





Application → Cas jekyll

- Créer le conteneur et installer les paquets

```
incus launch images:debian/trixie jekyll
incus exec jekyll -- apt update
incus exec jekyll -- apt -y install nginx htop ruby-full build-essential
```

- Configurer le *reverse proxy* nginx

```
config=$(cat << EOF
server {
    listen 80;
    listen [::]:80;
    location / {
        proxy_pass http://127.0.0.1:4000;
    }
}
EOF
)
incus exec jekyll -- bash -c "echo \"${config}\" | tee /etc/nginx/sites-enabled/reverse.conf"
```

```
incus exec jekyll -- rm /etc/nginx/sites-enabled/default
incus exec jekyll -- systemctl restart nginx
incus exec jekyll -- ss -tan
```

Application → Cas jekyll

- Installer jekyll

```
incus exec jekyll -- gem install jekyll bundler
```

- Créer un premier site web

```
incus exec jekyll -- jekyll new myOwnBlOg
```

- Lancer le service

```
incus exec jekyll -- bash -c "cd myOwnBlOg && bundle exec jekyll serve"
```

!! Consulter la page web avant d’interrompre le service !!

Application → cas jekyll

- Accès au service Web via un tunnel SSH
  - Relever les adresses du conteneur jekyll

```
etu@vm0:~$ incus ls
+--------+---------+----------------------+-----------------------------------------------+-----------+-----------+
|  NAME  |  STATE  |         IPV4         |                     IPV6                      |   TYPE    | SNAPSHOTS |
+--------+---------+----------------------+-----------------------------------------------+-----------+-----------+
| jekyll | RUNNING | 198.18.28.212 (eth0) | 2001:678:3fc:1c:216:3eff:feaa:2b68 (eth0) | CONTAINER | 0         |
+--------+---------+----------------------+-----------------------------------------------+-----------+-----------+
```

  - Créer un tunnel via la connexion SSH à l'hyperviseur depuis son poste
    - Utiliser l'adresse IPv4 ou IPv6
    - Le nom « alice » est défini dans la configuration du client SSH

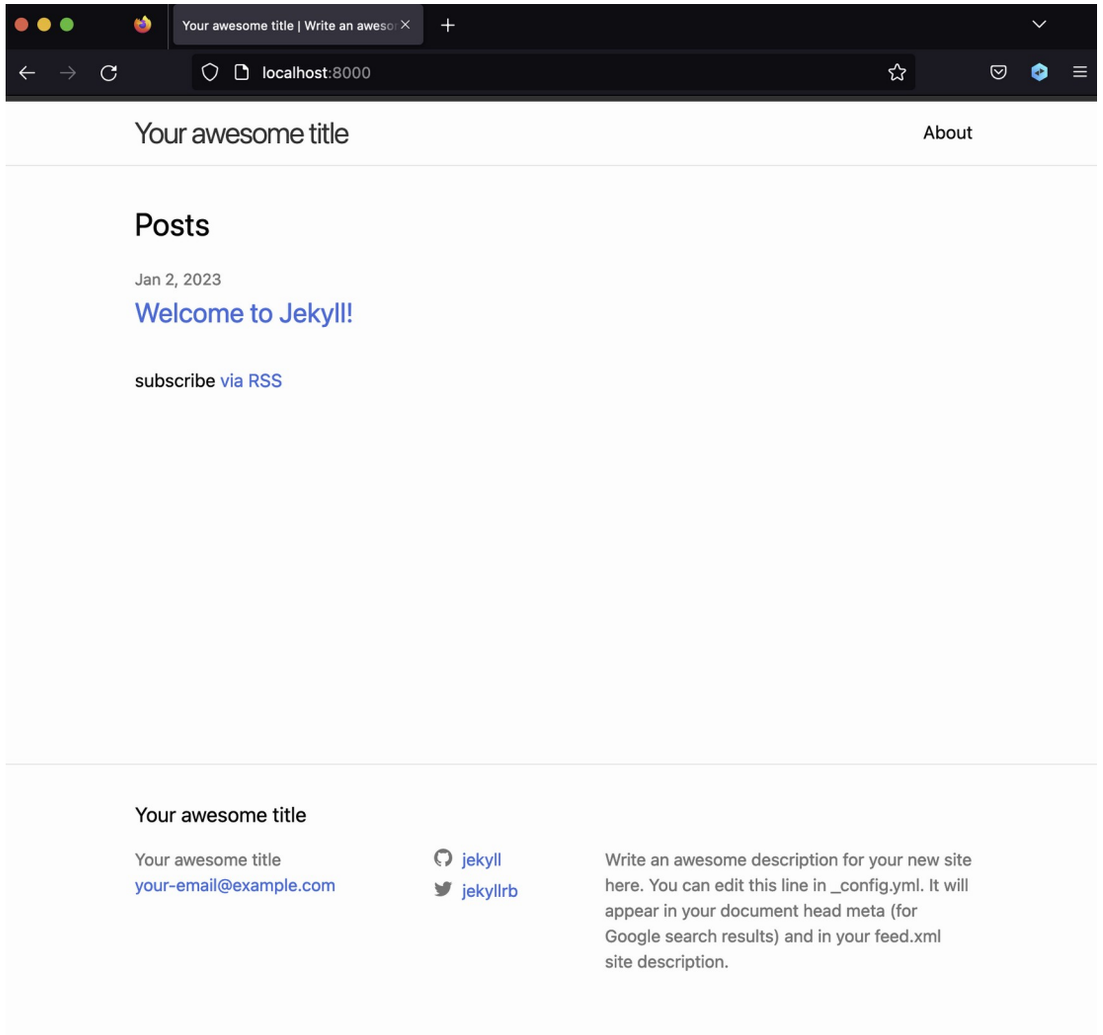
```
ssh -L 8000:198.18.28.212:80 alice
```

```
ssh -L "8000:[2001:678:3fc:1c:216:3eff:feaa:2b68]:80" alice
```

  - Identifier les processus nginx et jekyll avec htop dans le conteneur et sur le système hôte

Application → cas jekyll

- Accès au service Web via un tunnel SSH



- L'URL à consulter depuis le poste de travail est :

  http://localhost:8000

Bilan séance

- Shell Bash
  - Interpréteur de commandes aux fonctions étendues
  - Interface de base de l'administration système → contexte infrastructure
- Processus
  - Tout programme en cours d'exécution a l'«identité» de son propriétaire
- Droits sur le système de fichiers
  - Principes de gestion des droits Unix
  - Compromis efficacité/simplicité
- **Compétences essentielles en administration système**
  - Identifier les processus et leurs propriétaires
  - Différencier l’appartenance des processus entre système hôte et conteneur ou machine virtuelle

Défi pour la prochaine séance !

- Quelle est la relation entre uid/gid et subuid/subgid ?
  - Consulter la page à l'adresse ci-dessous
    - https://ubuntu.com/blog/custom-user-mappings-in-lxd-containers
    - Quel est le nom de l'utilisateur normal qui obtient la valeur subuid 100000 ?
    - À quel uid cette valeur correspond dans le conteneur ?