Installation/Configuration diverses

Commande Screen

- Certains scripts lancés sur le serveur peuvent prendre plusieurs heures pour s'exécuter. Afin de pouvoir revenir sur l'état du script en cas de déconnexion du SSH, nous allons installer la commande `screen`. Elle permet de créer un session qui peut être récupéré à tout moment.
- Installer la commande : `apt install screen`
- Principe générale d'utilisation de `Screen` :
    1. Pour lancer une nouvelle session : `screen -S <nom-de-la-session>`
    2. Lancer la commande/script désiré
    3. Utiliser les séquences de touches suivante pour se détacher de la session : `Ctrl-a + Ctrl-d`
    4. Pour voir la liste des sessions en cours : `screen -ls`
        1. Le numéro de session correspond au nombre présent devant ".pts-0"
    5. Pour se connecter à nouveau à une session : `screen -r <numero-session>`
- Dans une session Screen, si l'on veut pouvoir remonter dans l'affichage de la console, il faut :
    - Appuyer sur les touches du clavier `Ctrl-A` puis appuyer sur `Esc`
    - Utiliser ensuite les touches flèches "Haut" ou "Bas" ou "Page Haut" and "Page Bas" pour faire défiler l'affichage dans la console.
    - Appuyer sur `Esc` pour sortir du mode défilement arrière.
- Autres commandes :
    - Détacher une session toujours attaché pour s'y attacher à nouveau : `screen -D <numero-session>.<nom-session>`
    - Quitter et supprimer une session en cours : `CTRL+a` et `:quit`
    - Pour supprimer une session : ''screen -X -S <number.session-name> quit ''

Installer Byobu et raccourcis utiles

- Byobu est un gestionnaire de sessions avancé basé sur Screen et Tmux, qui simplifie la gestion des terminaux persistants et ajoute une barre de statut pratique.
- Installation : `apt install byobu`
- Lancer Byobu : `byobu`
- Principaux raccourcis clavier utiles :
    1. `F2` : créer une nouvelle fenêtre
    2. `F3` / `F4` : naviguer entre les fenêtres précédentes/suivantes
    3. `F6` : détacher la session (équivalent à `Ctrl-a d`)
    4. `F8` : renommer la fenêtre courante
    5. `Maj + F2` : diviser la fenêtre en deux (split horizontal)
    6. `Maj + F3` / `Maj + F4` : se déplacer entre les splits
    7. `Ctrl + D` : fermer le split ou la fenêtre courante
    8. `Maj + F5` : réorganiser les splits
    9. `Maj + Flèche directionnelle` : redimensionner un split
    10. `F9` : menu de configuration Byobu
- **Notes** :
    - Byobu crée des fichiers temporaires dans `/dev/shm`. Pour éviter les faux positifs avec

RKHunter, voir la section dédiée du wiki.
    - Lister toutes les sessions en arrière plan : `byobu list-sessions`
    - Se rattacher à une session : lancer à nouveau byobu (qui listera les sessions pour en choisir une s'il en existe plusieurs)

Configurer le layout du clavier par défaut

Afin d'accéder à la console VNC de l'interface OVH avec une clavier Azerty, il est nécessaire de le configurer :

- `apt install keyboard-configuration` : choisir Français - latin 9 seulement, 105 touches.
    - S'il faut le reconfigurer : `dpkg-reconfigure keyboard-configuration`
- Pour que cela soit pris en compte dans la console VNC :
    - `apt install console-setup`
    - `service keyboard-setup restart`
- Tester sur l'interface d'OVH l'accès à la console VNC, assurer vous que le clavier Azerty fonctionne correctement.

Installer Liquidprompt

- Les utilisateurs des instances ayant besoins de clôner des dépôts Git peuvent installer Liquidprompt pour avoir des informations sur le dépôt visible dans la Console. C'est le cas des utilisateurs **geonat** et **admin** sur les instances *web-srv* et *pdb-srv*.
- Se connecter sur l'instance avec l'utilisateur concerné.
- Créer un nouveau dossier qui hébergera les applications locales à l'utilisateur : `mkdir ~/apps`
- Se placer dans ce nouveau dossier : `cd ~/apps`
- Clôner le dépôt Liquidprompt : `git clone https://github.com/nojhan/liquidprompt.git`
- Ajouter cette ligne au fichier *.bashrc* :`vi ~/.bashrc`

```
# Only load Liquid Prompt in interactive shells, not from a script or from scp
[[ $- = *i* ]] && source ~/apps/liquidprompt/liquidprompt
```

- Créer un fichier de configuration pour indiquer la coloration de l'utilisateur dans le prompt de la console : `mkdir ~/.config/; vi ~/.config/liquidpromptrc`

```
# Current user (root)
#LP_COLOR_USER_LOGGED="$RED"
# Current user (admin)
#LP_COLOR_USER_LOGGED="$YELLOW"
# Current user (geonat et autres utilisateurs)
#LP_COLOR_USER_LOGGED="$GREEN"
```

    - Décommenter la ligne *LP_COLOR_USER* correspondant à votre utilisateur et supprimer les autres.
- Relancer le Shell pour prendre en compte les changements : `source ~/.bashrc`

Configurer Git pour un utilisateur

- Créer un fichier *~/.gitconfig* et ajouter le contenu suivant : `vi ~/.gitconfig`

```
[core]
    editor = vim
[credential]
    helper = cache --timeout=86400
[color]
    ui = true
[push]
    default = simple
[pull]
    ff = only
[alias]
    co = checkout
    ci = commit
    st = status
    br = branch
    pfl = push --force-with-lease
    lg = log --color --graph --pretty=format:'%Cred%h%Creset -
%C(yellow)%d%Creset %s %Cgreen(%cr) %C(bold blue)<%an>%Creset' --
abbrev-commit --
```

- Ajouter les éléments suivant au fichier *~/.bash_aliases* : `vi ~/.bash_aliases`

```
# Alias pour Git
alias gs='git status '
alias gb='git branch '
alias gd='git diff'
```

Rechercher un mot de passe sur tout le système

- Ressource : Linux Password Mining — Extract Passwords from files and Memory Heaps – 22 novembre 2022
- Afin de retrouver tous les fichiers stockant un mot de passe spécifique, il est possible d'utiliser la commande :

```
grep --color=auto -rnw '/' -ie "<mot-de-passe>" --color=always
2>/dev/null
```

    - Il est possible de limiter la recherche à un dossier et ses sous-dossiers en changeant le chemin du paramètre ''-rnw '<chemin>' ''.
- La commande *find* peut aussi être employé en conjonction avec *grep* :

```
find '/' -type f -exec grep -i -I "<mot-de-passe>" {} /dev/null \;
```

Ajout d'une entré au script update-alternatives pour Python 2 et 3

Si pour une raison ou une autre, il est nécessaire d'installer les 2 versions majeures de Python, il est utile de créer une entrée dans l'outil Debian "*update-alternatives*". Exemple pour Python 2.7 et Python 3.7 :

- `python --version` : doit afficher "Python 2.7.16"
- `update-alternatives --list python` : doit afficher "update-alternatives: erreur: pas d'alternatives pour python"
- `update-alternatives --install /usr/bin/python python /usr/bin/python2.7 1` : ajoute l'aternative python2.7 et doit afficher "update-alternatives: utilisation de « /usr/bin/python2.7 » pour fournir « /usr/bin/python » (python) en mode automatique"
- `update-alternatives --install /usr/bin/python python /usr/bin/python3.7 2` : ajoute l'aternative python3.7 et doit afficher "update-alternatives: utilisation de « /usr/bin/python3.7 » pour fournir « /usr/bin/python » (python) en mode automatique"
- `python --version` : doit maintenant afficher "Python 3.7.3"
- `update-alternatives --list python` : doit afficher

```
/usr/bin/python2.7
/usr/bin/python3.7
```

- `update-alternatives --config python` : permet de configurer l'alternative.
- `update-alternatives --remove python2.7` : permet de supprimer une alternative (ici 2.7) dans le cas où le paquet n'est plus installé sur le système.



ATTENTION : la version de Pip n'est pas mise à jour en correspondance. Du coup, l'utilisation de `pip2` ou `pip` échoue quand la version 3 de python est utilisée... Seul `pip3` fonctionne. ⇒ Trouver une solution.

From:
https://wiki-sinp.cbn-alpin.fr/ - **CBNA SINP**

Permanent link:
https://wiki-sinp.cbn-alpin.fr/serveurs/installation/divers?rev=1790958241

Last update: **2026/10/02 16:24**