DOCUMENTATION TECHNIQUE

Installation et configuration d’un bastion SSH

DOS SANTOS Sébastien

Compétence

Gérer le patrimoine informatique	**Recenser et identifier les ressources numériques**
Travailler en mode projet	**Planifier les activités**
Mettre à disposition des utilisateurs un service informatique	**Déployer un service**

Introduction

Un bastion SSH permet de sécuriser l'accès aux serveurs en ssh avec une machine qui permet d'empêcher toutes les machines de se connecter en ssh sauf un seul réseau, qui dans mon cas est le réseau administrateur.

Création du réseau administrateur

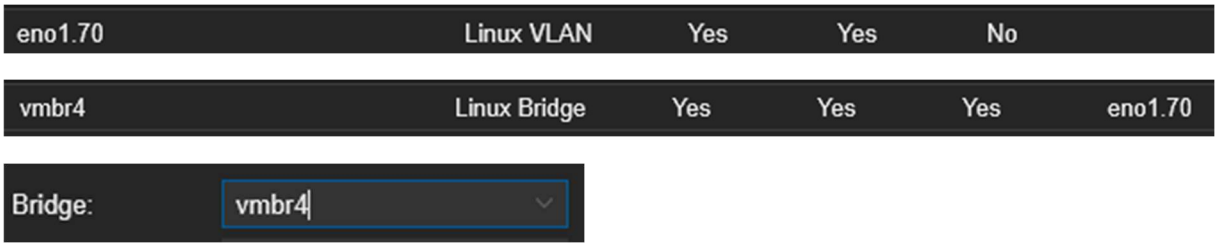
Dès le début j'ai commencé à créer le réseau administrateur en créant un vlan spécifique à ce réseau puis en ajoutant une sous interface à mon routeur.

```
sw1(config)#vlan 70
sw1(config-vlan)#name Admin
sw1(config-vlan)#exit
```

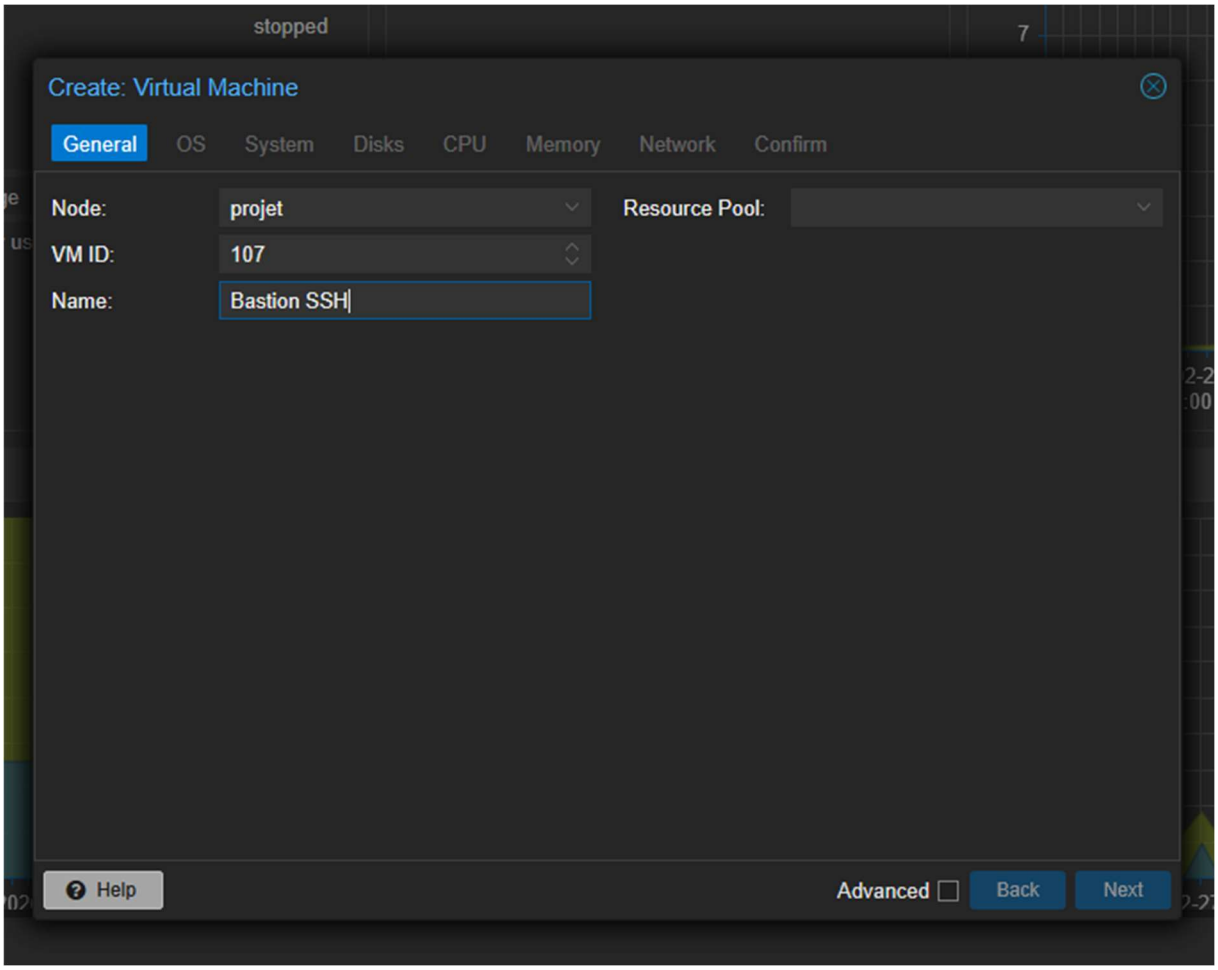
```
R1(config)#int fastEthernet 0/1.70
R1(config-subif)#encapsulation dot1Q 70
R1(config-subif)#ip address 192.168.200.1 255.255.255.0
R1(config-subif)#ip nat
R1(config-subif)#ip nat in
R1(config-subif)#ip nat inside
```

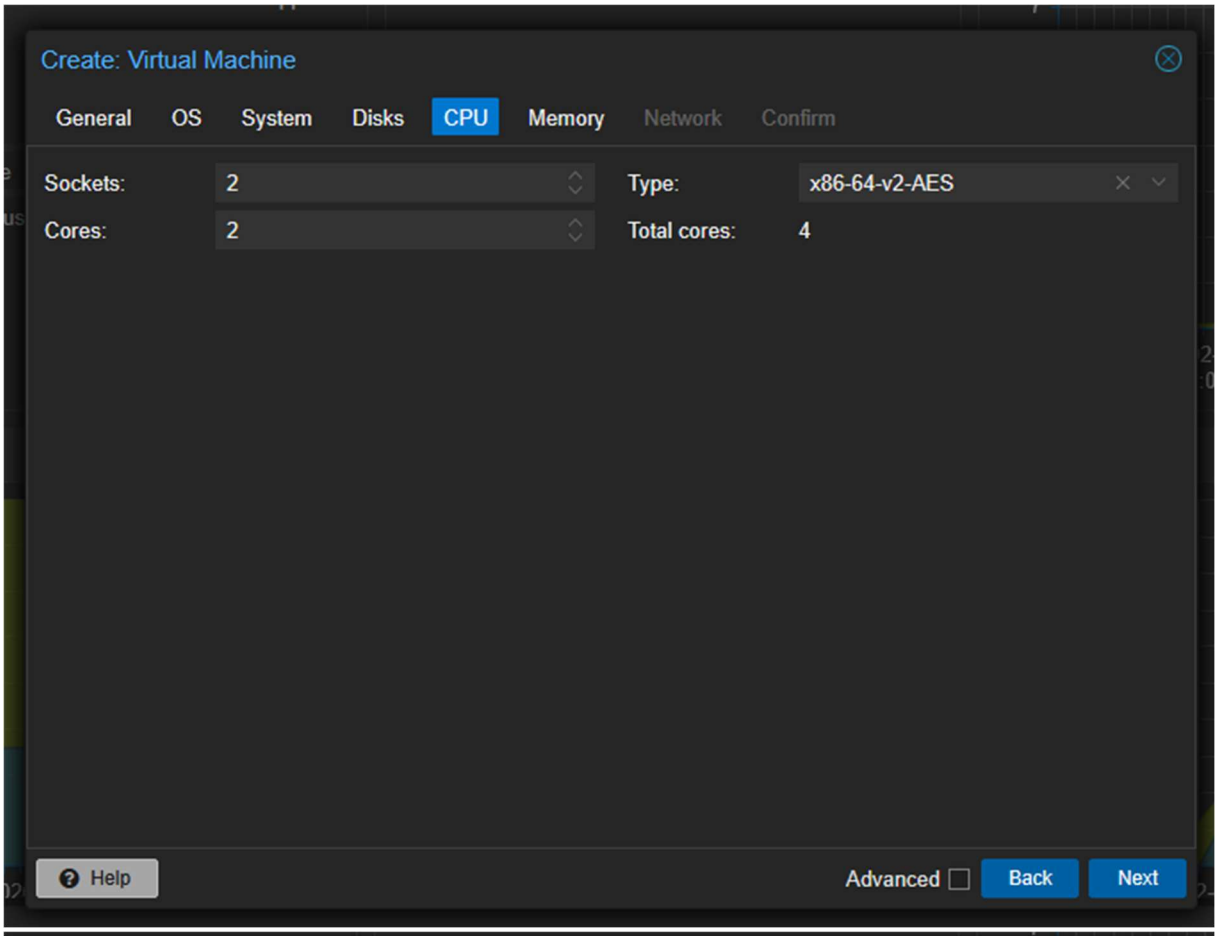
Création de la carte réseau et de la machine virtuelle

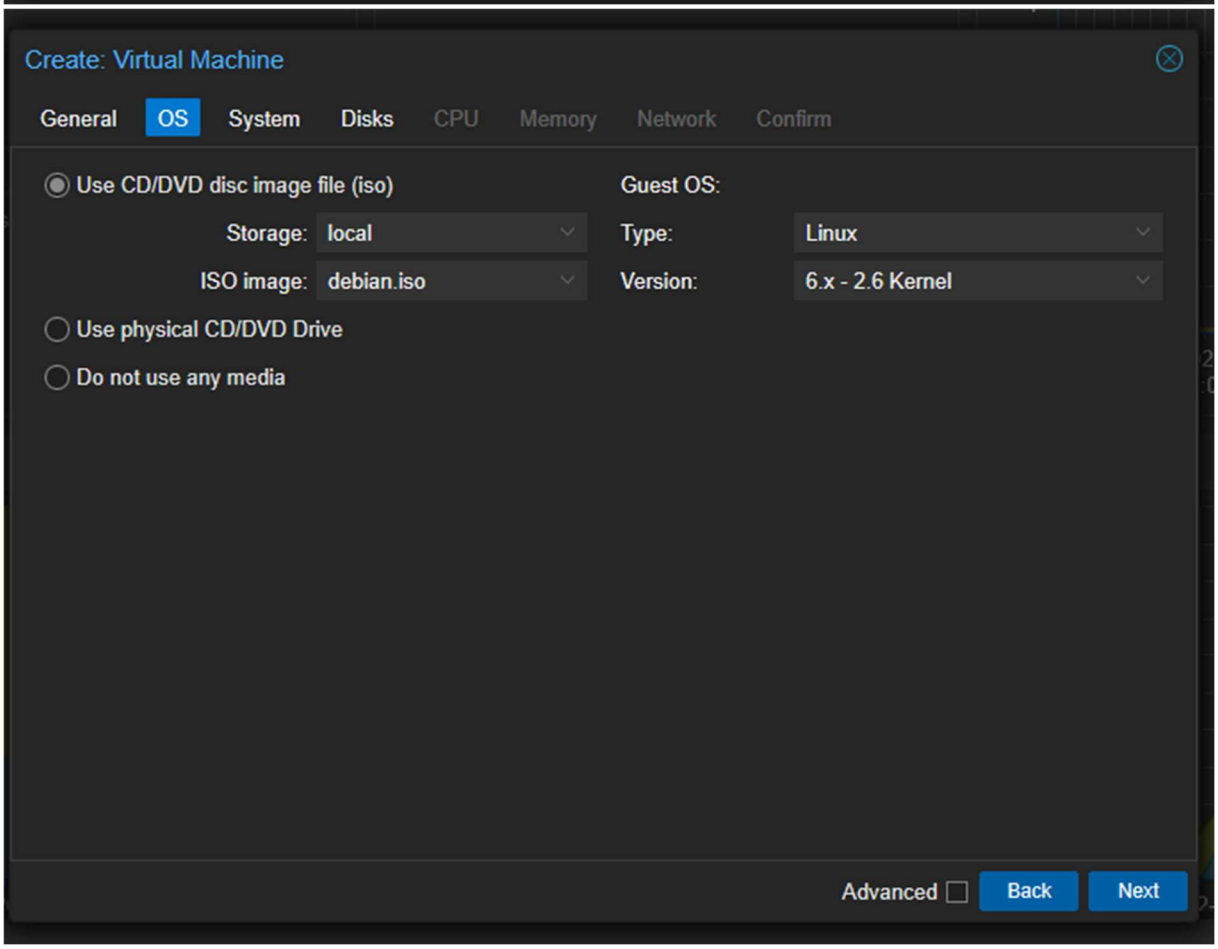
J’ai commencé par déclarer le vlan sur le serveur Proxmox et j’ai ensuite créer une carte réseau qui utilise le vlan 70.

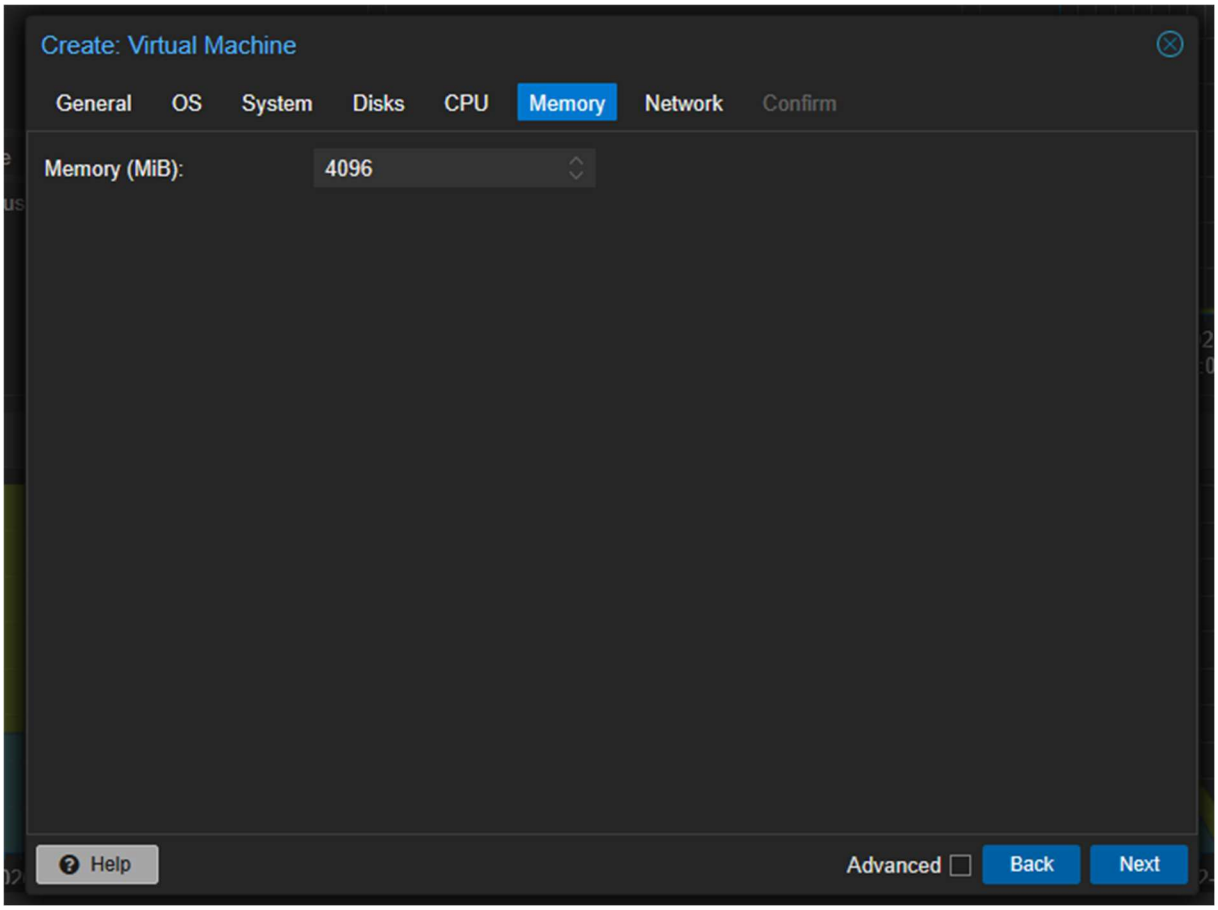


Je commence donc par créer la machine virtuelle en suivant les étapes suivantes :

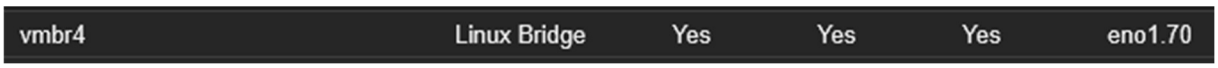



Create: Virtual Machine
General
OS
System
Disks
CPU
Memory
Network
Confirm
Sockets:
2
Cores:
2
Type:
x86-64-v2-AES
Total cores:
4
Help
Advanced
Back
Next


Create: Virtual Machine
General
OS
System
Disks
CPU
Memory
Network
Confirm
Use CD/DVD disc image file (iso)
Storage:
local
ISO image:
debian.iso
Use physical CD/DVD Drive
Do not use any media
Guest OS:
Type:
Linux
Version:
6.x - 2.6 Kernel
Advanced
Back
Next



J’ajoute ensuite la carte réseau qu’on vient de créer :



Ajout des règles d’accès sur le routeur

Pour filtrer les connexion SSH, je décide de mettre des règles qui bloquent les connexions des Vlan clients vers les serveurs et des serveurs entre eux :

```
Extended IP access list VLAN10_CLIENTS
    10 deny tcp 192.168.230.0 0.0.0.255 192.168.200.0 0.0.0.255 eq 22
    20 deny tcp 192.168.230.0 0.0.0.255 192.168.231.0 0.0.0.255 eq 22
    30 deny tcp 192.168.230.0 0.0.0.255 192.168.232.0 0.0.0.255 eq 22
    40 deny tcp 192.168.230.0 0.0.0.255 192.168.233.0 0.0.0.255 eq 22
    50 deny tcp 192.168.230.0 0.0.0.255 192.168.234.0 0.0.0.255 eq 22
    60 deny tcp 192.168.230.0 0.0.0.255 192.168.235.0 0.0.0.255 eq 22
    70 permit ip any any (118409 matches)
Extended IP access list VLAN20_SSH
    10 deny tcp 192.168.231.0 0.0.0.255 192.168.200.0 0.0.0.255 eq 22
    20 deny tcp 192.168.231.0 0.0.0.255 192.168.230.0 0.0.0.255 eq 22
    30 deny tcp 192.168.231.0 0.0.0.255 192.168.232.0 0.0.0.255 eq 22
    40 deny tcp 192.168.231.0 0.0.0.255 192.168.233.0 0.0.0.255 eq 22
    50 deny tcp 192.168.231.0 0.0.0.255 192.168.234.0 0.0.0.255 eq 22
    60 deny tcp 192.168.231.0 0.0.0.255 192.168.235.0 0.0.0.255 eq 22
    70 permit ip any any
Extended IP access list VLAN30_SSH
    10 deny tcp 192.168.232.0 0.0.0.255 192.168.200.0 0.0.0.255 eq 22
    20 deny tcp 192.168.232.0 0.0.0.255 192.168.230.0 0.0.0.255 eq 22
    30 deny tcp 192.168.232.0 0.0.0.255 192.168.231.0 0.0.0.255 eq 22
    40 deny tcp 192.168.232.0 0.0.0.255 192.168.233.0 0.0.0.255 eq 22
    50 deny tcp 192.168.232.0 0.0.0.255 192.168.234.0 0.0.0.255 eq 22
    60 deny tcp 192.168.232.0 0.0.0.255 192.168.235.0 0.0.0.255 eq 22
    70 permit ip any any
Extended IP access list VLAN40_SSH
    10 deny tcp 192.168.233.0 0.0.0.255 192.168.200.0 0.0.0.255 eq 22
    20 deny tcp 192.168.233.0 0.0.0.255 192.168.230.0 0.0.0.255 eq 22
    30 deny tcp 192.168.233.0 0.0.0.255 192.168.231.0 0.0.0.255 eq 22
    40 deny tcp 192.168.233.0 0.0.0.255 192.168.232.0 0.0.0.255 eq 22
    50 deny tcp 192.168.233.0 0.0.0.255 192.168.234.0 0.0.0.255 eq 22
    60 deny tcp 192.168.233.0 0.0.0.255 192.168.235.0 0.0.0.255 eq 22
```

```
    70 permit ip any any
Extended IP access list VLAN50_SSH
    10 deny tcp 192.168.234.0 0.0.0.255 192.168.200.0 0.0.0.255 eq 22
    20 deny tcp 192.168.234.0 0.0.0.255 192.168.230.0 0.0.0.255 eq 22
    30 deny tcp 192.168.234.0 0.0.0.255 192.168.231.0 0.0.0.255 eq 22
    40 deny tcp 192.168.234.0 0.0.0.255 192.168.232.0 0.0.0.255 eq 22
    50 deny tcp 192.168.234.0 0.0.0.255 192.168.233.0 0.0.0.255 eq 22
    60 deny tcp 192.168.234.0 0.0.0.255 192.168.235.0 0.0.0.255 eq 22
    70 permit ip any any
Extended IP access list VLAN60_SSH
    10 deny tcp 192.168.235.0 0.0.0.255 192.168.200.0 0.0.0.255 eq 22
    20 deny tcp 192.168.235.0 0.0.0.255 192.168.230.0 0.0.0.255 eq 22
    30 deny tcp 192.168.235.0 0.0.0.255 192.168.231.0 0.0.0.255 eq 22
```

Ajout d’un menu lors de la connexion au bastion

Ce script bash nommé « bastion-menu.sh » permet d’afficher un menu lors de la connexion au bastion permettant de choisir vers quelle machine se connecté, ce qui est plus intuitif que de taper la commande.

root@bastion:/# cat /usr/local/bin/bastion-menu.sh

```
#!/bin/bash


# ========================
#   CONFIGURATION COULEURS
# ========================
RED='\033[0;31m'
GREEN='\033[0;32m'
YELLOW='\033[1;33m'
BLUE='\033[1;34m'
CYAN='\033[1;36m'
MAGENTA='\033[1;35m'
WHITE='\033[1;37m'
BOLD='\033[1m'
DIM='\033[2m'
NC='\033[0m'


# ========================
#   CONFIGURATION CIBLES
# ========================
ROUTER_HOST="R1"
SWITCH_HOST="SW1"
PROXMOX_HOST="root@192.168.230.10"
ZABBIX_HOST="ZABBIX"
ASTERIX_HOST="Asterix"
GLPI_HOST="GLPI"
CLOUFLARE_HOST="CLOUDFLARE"


# ========================
#   FONCTIONS
# ========================
draw_line() {
```

```
    echo -e "${BLUE}============================================================${NC}"
}

pause_return() {
    echo
    read -p "Appuie sur Entrée pour revenir au menu..." _
}

show_header() {
    clear
    echo -e "${CYAN}${BOLD}"
    echo " ██████╗  ████╗ ███████╗████████╗██╗ ██████╗ ███╗   ██╗"
    echo " ██╔═══██╗██╔═══██╗██╔═══╝╚═══██╔═╝██╔═══██╗████╗  ██║"
    echo " ██████╔╝███████╔╝███████╗    ██╔  ██╔═══██╗ ██╔██╗ ██╔"
    echo " ██╔═══██╗██╔═══██╗╚═══██╗   ██╔  ██╔═══██╗ ██╔═╚═╗██╔"
    echo " ██████╔╝██╔  ██╔╝███████╔   ██╔  ██╔═══██╗ ██╔  ╚═╗██╔"
    echo " ╚═════╝ ╚═╝  ╚═╝╚═══════╝   ╚═╝  ╚═╝ ╚═══╝ ╚═╝  ╚═══╝"
    echo -e "${NC}"
    echo -e "${WHITE}${BOLD}              Bastion SSH - Console d'administration${NC}"
    echo -e "${DIM}        Accès centralisé aux équipements et serveurs${NC}"
    echo
}

show_footer() {
    echo
    draw_line
    echo -e "${DIM}Utilisateur : $(whoami)   |   Hôte : $(hostname)   |   Date : $(date
'+%d/%m/%Y %H:%M:%S')${NC}"
    draw_line
}

log_action() {
    logger "[BASTION-MENU] user=$(whoami) action=$1"
}

connect_host() {
    local label="$1"
    local host="$2"
```

```
    clear
    draw_line
    echo -e "${GREEN}${BOLD}Connexion à ${label}${NC}"
    draw_line
    echo -e "${DIM}Cible SSH : ${host}${NC}"
    echo

    log_action "connexion_${label}"
    ssh "$host"

    echo
    echo -e "${YELLOW}Retour au menu principal...${NC}"
    sleep 1
}

show_info_panel() {
    draw_line
    echo -e "${WHITE}${BOLD}MENU PRINCIPAL${NC}"
    draw_line
    echo -e "${GREEN}[1]${NC} Routeur"
    echo -e "${GREEN}[2]${NC} Switch"
    echo -e "${GREEN}[3]${NC} Proxmox"
    echo -e "${GREEN}[4]${NC} Serveurs Linux"
    echo -e "${YELLOW}[5]${NC} CLI du bastion"
    echo -e "${RED}[6]${NC} Quitter"
    draw_line
}

linux_menu() {
    while true; do
        show_header
        draw_line
        echo -e "${WHITE}${BOLD}SERVEURS LINUX${NC}"
        draw_line
        echo -e "${GREEN}[1]${NC} Zabbix"
        echo -e "${GREEN}[2]${NC} Asterix"
```

```
        echo -e "${GREEN}[3]${NC} Glpi"
        echo -e "${GREEN}[4]${NC} Cloudflare"
        echo -e "${YELLOW}[5]${NC} Retour"
        draw_line
        echo
        read -p "Choix : " srv

        case "$srv" in
            1)
                connect_host "Zabbix" "$ZABBIX_HOST"
                ;;
            2)
                connect_host "Asterix" "$ASTERIX_HOST"
                ;;
            3)
                connect_host "Glpi" "$GLPI_HOST"
                ;;
            4)
                connect_host "Cloudflare" "$CLOUDFLARE_HOST"
                ;;
            5)
                break
                ;;
            *)
                echo
                echo -e "${RED}Choix invalide.${NC}"
                sleep 1
                ;;
        esac
    done
}

open_bastion_cli() {
    clear
    draw_line
    echo -e "${MAGENTA}${BOLD}Accès au shell local du bastion${NC}"
    draw_line
```

```
    echo -e "${DIM}Tape 'exit' pour revenir ou fermer la session selon ton contexte.${NC}"
    echo
    log_action "cli_bastion"
    exec /bin/bash
}

# ========================
#   BOUCLE PRINCIPALE
# ========================
while true; do
    show_header
    show_info_panel
    show_footer
    echo
    read -p "Choix : " choix

    case "$choix" in
        1)
            connect_host "Routeur" "$ROUTER_HOST"
            ;;
        2)
            connect_host "Switch" "$SWITCH_HOST"
            ;;
        3)
            connect_host "Proxmox" "$PROXMOX_HOST"
            ;;
        4)
            linux_menu
            ;;
        5)
            open_bastion_cli
            ;;
        6)
            clear
            echo -e "${RED}${BOLD}Déconnexion du bastion...${NC}"
            log_action "deconnexion"
            exit 0
```

```
            ;;
        *)
            echo
            echo -e "${RED}Choix invalide. Merci de sélectionner une option valide.${NC}"
            sleep 1
            ;;
    esac
done
```

Il faut maintenant faire en sorte que ce script à chaque connexion, peu importe l'utilisateur qui se connecte, pour cela il faut d'abord attribuer les droits d'exécution au fichier avec cette commande : « chmod +x /usr/local/bin/bastion-menu.sh »

Une fois ceci fait on peut aller modifier le fichier de configuration de openssh nommé sshd_config en ajoutant cette ligne :

ForceCommand /usr/local/bin/bastion-menu.sh -> Lorsque l'on connecte en SSH au bastion cela force l'exécution du script pour afficher le menu.

Ajout de la connexion avec les comptes Active Directory

Maintenant j'ajoute la connexion au bastion avec les comptes Active Directory qui permet deux différents choses :

- Permet à chaque utilisateur d'avoir sa propre session ssh
- Permet de savoir plus facilement qui s'est connecté au bastion

On va d'abord lancer cette commande pour installer ces différents paquets :
apt install realmd sssd sssd-tools libnss-sss libpam-sss adcli samba-common-bin oddjob oddjob-mkhomedir packagekit -y

- realmd : Connexion au domaine
- ssd : Gestion des authentifications
- adcli : Intégration de l'active directory
- samba : Communication avec l'active directory

Avant de joindre le domaine, on vérifie qu'il est détecté.

/usr/sbin/realm discover projet.com

projet.com

type: kerberos

realm-name: PROJET.COM

domain-name: projet.com

configured: kerberos-member

server-software: active-directory

client-software: sssd

required-package: sssd-tools

required-package: sssd

required-package: libnss-sss

required-package: libpam-sss

required-package: adcli

required-package: samba-common-bin

login-formats: %U@projet.com

login-policy: allow-realm-logins

Ensuite on joint le domaine active directory

/usr/sbin/realm join projet.com -U Administrateur

Le mot de passe du compte Administrateur sera ensuite demandé, une fois que ceci est fait, si tout fonctionne la machine est maintenant membre du domaine.

Pour vérifier que la machine à bien réussit a joindre le domaine active directory on exécute cette commande :

```
root@bastion:/home/utec# /usr/sbin/realm list
projet.com
  type: kerberos
  realm-name: PROJET.COM
  domain-name: projet.com
  configured: kerberos-member
  server-software: active-directory
  client-software: sssd
  required-package: sssd-tools
  required-package: sssd
  required-package: libnss-sss
  required-package: libpam-sss
  required-package: adcli
  required-package: samba-common-bin
  login-formats: %U@projet.com
  login-policy: allow-realm-logins
```

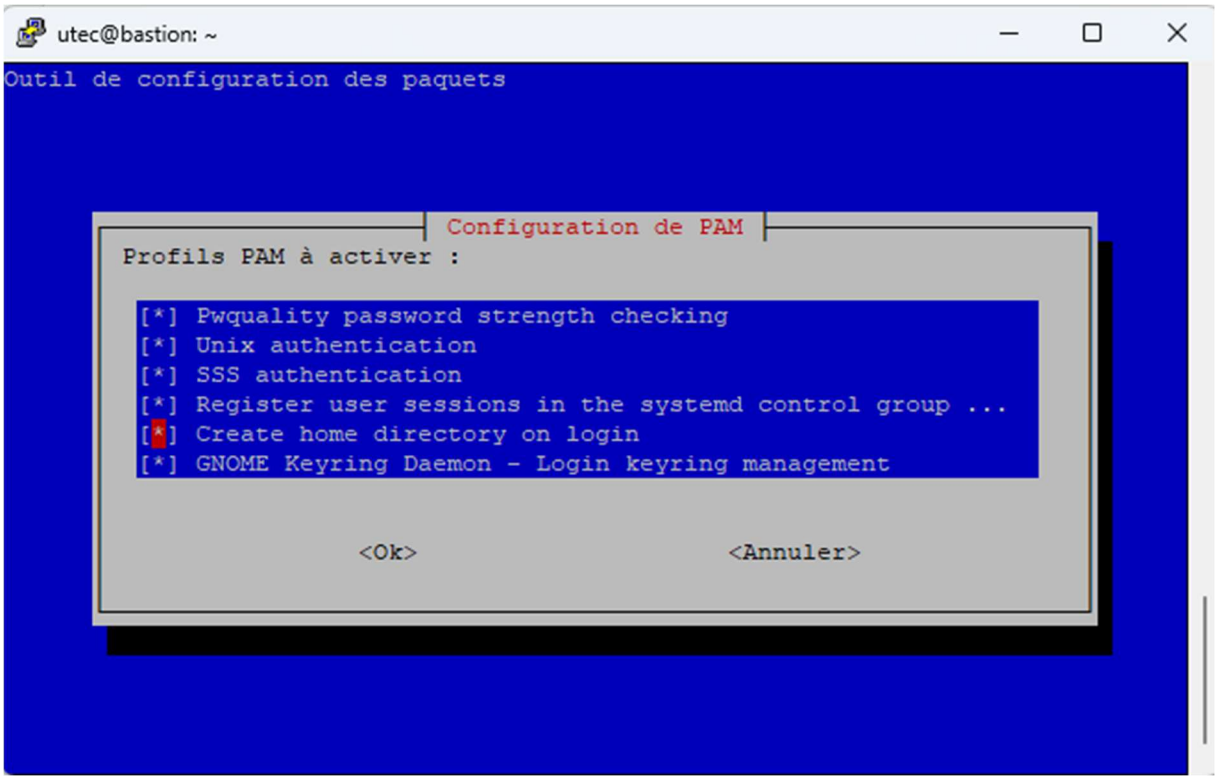
Maintenant on va faire en sorte qu'à chaque fois qu'un utilisateur active directory se connecte au bastion, il à un dossier utilisateur qui se créer. On doit d'abord télécharger ces paquets :

apt install libpam-runtime -y

Ensuite on exécute cette commande :

/usr/sbin/pam-auth-update

Elle nous fait atterrir sur ce menu et on coche la case « Create home directory on login »


utec@bastion: ~
Outil de configuration des paquets
Configuration de PAM
Profils PAM à activer :
[*] Pwquality password strength checking
[*] Unix authentication
[*] SSS authentication
[*] Register user sessions in the systemd control group ...
[*] Create home directory on login
[*] GNOME Keyring Daemon - Login keyring management
<Ok>
<Annuler>