

Script

Téléchargez le script au format txt, renommez-le et rendez-le exécutable :

```
[trainee@centos8 ~]$ wget https://www.dropbox.com/s/oxzeb4aslhnuqn4/script.txt
[trainee@centos8 ~]$ mv script.txt script.sh
[trainee@centos8 ~]$ chmod u+x script.sh
```

Installez ksh

```
[trainee@centos8 ~]$ su -
Password: fenestros
[root@centos8 ~]# dnf install ksh -y
```

En cas de problèmes :

```
[root@centos8 ~]# sed -i 's/^mirrorlist/#mirrorlist/g' /etc/yum.repos.d/CentOS-*
[root@centos8 ~]# sed -i 's|#baseurl=http://mirror.centos.org|baseurl=http://vault.centos.org|g'
/etc/yum.repos.d/CentOS-*
[root@centos8 ~]# dnf install ksh -y
```

Déplacez et exécutez le script :

```
[root@centos8 ~]# mv /home/trainee/script.sh .
[root@centos8 ~]# ./script.sh
```

```
#!/bin/ksh

# Ce script effectue des taches d'administration
#
# Declaration de variables utilisees dans le script
```

```
PASSWD="/etc/passwd"
GROUP="/etc/group"
HOME="/home"
OPASSWD="/etc/passwd"
BINPATH="/usr/sbin"

# Cette fonction genere une pause ecran
function pause {
    printf "\nAppuyer sur la touche \"Entree\" ou \"Return\" pour continuer...\n"
    read x
}

function existe {
    while getopts "ug" option
    do
        case "$option" in
            u)      grep -i "^$2:" $PASSWD > /dev/null && return 0
                    return 1
                    ;;
            g)      grep -i "^$2:" $GROUP > /dev/null && return 0
                    return 1
                    ;;
            *)      echo "Option incorrecte."
                    ;;
        esac
    done
}

function saisie {
    while getopts "ug" option
    do
        case $option in
            u)      printf "\nNom de l'utilisateur : "
                    read user
                    ;;
```

```
        print
        ;;
    g) printf "\nNom du groupe : "
        read groupe
        print
        ;;
    *) echo "Option incorrecte."
        ;;
esac
done
}

#####
#          GESTION DES GROUPES
#####
#
# Cette fonction cree un groupe
function cree_group
{
    while (true) ; do
        # Saisie du nom du groupe
        saisie -g
        # Verifier que le groupe n'existe pas
        if ! existe -g $groupe ; then
            # Saisie securisee du numero du groupe (GID)
            while(true) ; do
                printf "\nNumero GID : "
                read gid
                expr ":$gid:" : ':[0-9]\{2,5\}:' > /dev/null
                if (( $? != 0 )) ; then
                    print "Mauvaise saisie. Recommencer"
                else
                    # Verifier que le GID n'existe pas dans /etc/group
                    grep ".*:x:$gid:" $GROUP > /dev/null
                fi
            done
        else
            print "Groupe existe. Recommencer"
        fi
    done
}
```

```
        if      (( $? == 0 ))
        then
                print "$gid existe dans $GROUP"
                print "Saisir un autre numero."
        else
                #su -l root -s /bin/bash -c \
                $BINPATH/groupadd -g $gid $groupe > /dev/null 2>&1
                if (( $? == 0 )) ; then
                        printf "\nLe groupe $groupe a bien ete cree"
                else
                        printf "\nEchec de creation de $groupe"
                fi
                break
        fi
        done
        break
    else
        printf "\n$groupe existe dans $GROUP"
        printf "\nSaisir un autre nom\n"
    fi
done
}

# Cette fonction modifie un groupe
function modif_group
{
    while (true) ; do
        # Saisie du nom du groupe
        saisie -g
        # Verifier que le groupe existe
        if existe -g $groupe ; then
            ligne=`grep -i "^\$groupe:" $GROUP`  

            IFS=:
```

```
set $ligne
printf "\nListe des champs a modifier \
\n\t 1. Nom du groupe : $1\
\n\t 2. Numero du groupe: $3\n"
while (true) ; do
    printf "\nSaisir votre choix : "
    read choix
    expr "$choix" : '[12]\{1\}' > /dev/null 2>&1
    if (( $? != 0 )) ; then
        printf "Saisie incorrecte. Recommencer."
    else
        if (( $choix == 1 )) ; then
            saisie -g
            $BINPATH/groupmod -n $groupe $1 > /dev/null
            if (( $? != 0 )) ; then
                printf "\nEchec de modification.Recommencer"
            else
                printf "\nModification reussie"
                break
            fi
        elif (( $choix == 2 )) ; then
            printf "\nSaisir le numero de groupe : "
            read gid
            $BINPATH/groupmod -g $gid $1
            if (( $? != 0 )) ; then
                printf "\nEchec de modification."
            else
                printf "\nModification reussie"
                break
            fi
        fi
    done
    break
```

```
        else
            printf "\n$groupe n'existe pas."
        fi
    done
}

# Cette fonction supprime un groupe
function delete_group {
    while (true) ; do
        # Saisie du nom du groupe
        saisie -g
        # Vérifier que le groupe existe
        if existe -g $groupe ; then
            $BINPATH/groupdel $groupe
            if (( $? == 0 )) ; then
                printf "\n$groupe a été supprimé avec succès"
                break
            else
                printf "\nEchec de suppression du groupe. Recommencer"
            fi
        else
            printf "\n$groupe n'existe pas dans $GROUP"
            pause
        fi
    done
}

# Cette fonction affiche des informations sur un groupe
function affiche_group {
    while (true) ; do
        # Saisie du nom du groupe
        saisie -g
        # Vérifier que le groupe existe
        if existe -g $groupe ; then
```

```
        ligne=`grep -i "^\$groupe:" $GROUP`  
        IFS=:  
        set $ligne  
        printf "\nNom du groupe : $1"  
        printf "\nNumero du groupe : $3"  
        printf "\nListe des membres du groupe : \n"  
        grep ".*:x:[0-9]*:$3:" $PASSWD > membres  
        gawk -F: '{print $1}' membres  
        rm membres  
        break  
    else  
        printf "\n$groupe n'existe pas dans $GROUP"  
    fi  
done  
}  
  
#####  
#          GESTION DES UTILISATEURS          #  
#####  
  
# Cette fonction affiche les informations sur un compte  
function affiche_user  
{  
    # Saisie du nom du compte  
    saisie -u  
    # Vérifier que le compte existe  
    if ! existe -u $user ; then  
        printf "\n$user n'existe pas dans /etc/passwd"  
    else  
        ligne=`grep -i "^\$user:" $PASSWD`  
        printf "\nInformations sur le compte $user\n"  
        IFS=:  
        set $ligne  
        printf "Nom de connexion : $1\n"
```

```
        printf "Numero de l'utilisateur : $3\n"
        printf "Numero du groupe : $4\n"
        printf "Nom du shell de connexion : $7\n"
        printf "Nom du repertoire de connexion : $6\n"
    fi
}

# Cette fonction cree un compte utilisateur
function cree_user
{
    while(true) ; do
        # Saisie du nom du compte
        saisie -u
        # Verifier que le compte n'existe pas deja
        if ! existe -u $user ; then
            # Saisie securisee du numero du compte (UID)
            while(true) ; do
                printf "\nNumero UID : "
                read uid
                expr ":$uid:" : ':[0-9]\{3,5\}:' > /dev/null
                if [ $? != 0 ] ; then
                    print "Saisie incorrecte. Recommencer"
                else
                    # Verifier que l'UID n'existe pas dans /etc/passwd
                    grep "^.+:x:$uid:" $PASSWD > /dev/null
                    if      (( $? == 0 ))
                    then
                        print "$uid existe dans $PASSWD"
                        print "Saisir un autre numero."
                    else
                        break
                    fi
                fi
            done
        fi
    done
}
```

```
# Saisie du numero du groupe (GID)
while (true) ; do
    printf "\nNumero GID : "
    read gid
    expr "$gid" : ':[0-9]\{3,5\}:' > /dev/null
    if (( $? != 0 )) ; then
        print "Saisie incorrecte. Recommencer"
    else
        # Vérifier que le GID existe dans /etc/group
        # Sinon, le créer
        grep ".*:$gid:" $GROUP > /dev/null
        if (( $? != 0 )) ; then
            print "$gid n'existe pas dans $GROUP"
            print "Création de $gid"
            # Appel de cree_group
            cree_group
        fi
        break
    fi
done
# Nom du répertoire de connexion
rep="$HOME/$user"

# Saisie du shell
while (true) ; do
    printf "\nNom du shell parmi la liste suivante :
"
    print "$(cat /etc/shells)"
    print
    printf "\nVotre choix : "
    read shell
    grep "^$shell$" /etc/shells > /dev/null
    if (( $? != 0 )) ; then
        print "Saisie incorrecte. Recommencer"
```

```
        else
            break
        fi
    done

    # Mot de passe
    printf "\nMot de passe : "
    read mdp
    # Rajouter le nouvel utilisateur
    #su -l root -c "
    $BINPATH/useradd -u $uid -g $gid -d $rep -m -s $shell \
-p $mdp $user > /dev/null
    if [ $? == 0 ] ; then
        printf "\n$user a bien ete cree "
    else
        printf "\nEchec de creation de $user"
    fi
    break
else
    print "$user existe dans $PASSWD"
    print "Saisir un autre nom."
    print
fi
done
printf "\nRetour au menu precedent"
}

# Cette fonction modifie les informations d'un compte
function modif_user {
while (true) ; do
    # Saisie du nom du compte
    saisie -u

    # Vérifier que le compte existe
```

```
if ! existe -u $user ; then
    printf "\n$user n'existe pas. Recommencer\n"
else
    ligne=`grep -i "^$user:" $PASSWD`
    IFS=:
    set $ligne
    printf "\nChamps a modifier : \
\nt1. Nom de connexion : $1\
\nt2. Numero UID : $3\
\nt3. Numero GID : $4\
\nt4. Shell de connexion : $7\
\nt5. Repertoire de connexion : $6\n"
    printf "\nSaisir votre choix : "
    read choix
    while (( $choix < 1 || $choix > 5 )) ; do
        printf "\nChoix incorrect. Saisir un autre choix : "
        read choix
    done
    case $choix in
        1)      while(true) ; do
                    saisie -u
                    $BINPATH/usermod -l $user $1 > /dev/null
                    if (( $? == 0 )) ; then
                        printf "$1 a ete modifie avec succes\n"
                        break
                    else
                        printf "\nEchec de modification. Recommencer"
                    fi
                done
                ;;
        2)      while (true) ; do
                    printf "\nSaisir le nouveau numero UID : "
                    read uid
                    $BINPATH/usermod -u $uid $user >/dev/null
                done
    esac
```

```
                if (( $? == 0 )) ; then
                    printf "$user a ete modifie avec succes.\n"
                    break
                else
                    printf "Echec de modification. Recommencer"
                fi
            done
        ;;
    3)   while (true) ; do
            printf "\nSaisir le nouveau numero GID : "
            read gid
            $BINPATH/usermod -g $gid $user >/dev/null
            if (( $? == 0 )) ; then
                printf "$user a ete modifie avec succes.\n"
                break
            else
                printf "Echec de modification. Recommencer"
            fi
        done
    ;;
    4)   while (true) ; do
            printf "\nSaisir le shell dans la liste:\n"
            cat /etc/shells
            printf "\nNouveau shell : "
            read shell
            $BINPATH/usermod -s $shell $user > /dev/null
            if (( $? == 0 )) ; then
                printf "$user a ete modifie avec succes.\n"
                break
            else
                printf "Echec de modification. Recommencer"
            fi
        done
    ;;

```

```
5)      while (true) ; do
            printf "\nSaisir le repertoire de connexion : "
            read rep
            $BINPATH/usermod -d $rep -m $user > /dev/null
            if (( $? == 0 )) ; then
                printf "$user a ete modifie avec succes.\n"
                break
            else
                printf "Echec de modification. Recommencer"
            fi
        done
        ;;
    esac
    break
fi
done
}

# Cette fonction supprime un compte
function delete_user {
    while(true) ; do
        # Saisie du nom du compte
        saisie -u
        # Verifier que le compte existe
        if ! existe -u $user ; then
            printf "\n$user n'existe pas dans $PASSWD. Recommencer"
        else
            $BINPATH/userdel -r $user > /dev/null
            if (( $? == 0 )) ; then
                printf "\n$user a ete supprime avec succes."
                break
            else
                printf "\nEchec de suppression. Recommencer"
            fi
    done
}
```

```
        fi
    done
}

# Cette fonction cree une liste d'utilisateurs qui se trouvent dans un \
# fichier. Ce dernier contient egalement les informations necessaires . \
# la creation des comptes
# Chaque ligne du fichier contient les donnees suivantes :
#     prenom nom nom_groupe nom_shell
# Les champs sont separes par des espaces

function cree_liste_user {
    UID_DEB=`cat /etc/passwd | cut -d: -f 3 | sort -n | tail -2 | head -1`
    UID=`expr $UID_DEB + 1`
    printf "\nSaisir le nom de la base qui contient les comptes a creer : "
    read base

    # Verifier l'existence du fichier
    if [ ! -f $base ] ; then
        printf "\n $base n'existe pas \n"
    else
        while read prenom nom groupe shell
        do
            $BINPATH/useradd -u $UID -g $groupe -d /home/$nom -m \
            -s /bin/$shell -p $nom $nom
            if [ $? == 0 ] ; then
                printf "\n$nom a ete cree avec succes\n"
                ((UID+=1))
            else
                printf "\nEchec de creation de $nom\n"
            fi
        done < $base
    fi
}
```

```
#####
# Sauvegarde et archivage du systeme #
#####

# Cette fonction archive un repertoire
#
function archive_rep
{
    printf "\nFonctionnalite à créer.\n"
}

# Cette fonction procede a l'extraction d'une archive
function restaure_rep
{
    printf "\nFonctionnalite à créer.\n"
}

# Cette fonction affiche le contenu d'une archive
function affiche_archive
{
    printf "\nFonctionnalite à créer.\n"
}

# Cette fonction compresse une archive a l'aide de gzip
function compress_archive
{
    printf "\nFonctionnalite à créer.\n"
}

# Cette fonction decomprime une archive compressée par gzip
function decompress_archive
{
    printf "\nFonctionnalite à créer.\n"
}
```

```
# Affichage du menu
clear
printf "\t\t\t MENU \n\n"
print
    PS3="Quel est votre choix ? "

select item in "- Creer un compte utilisateur " \
"- Modifier un compte utilisateur " \
"- Supprimer un compte utilisateur " \
"- Afficher un compte utilisateur " \
"- Creer une liste d'utilisateurs" \
"- Creer un groupe" \
"- Modifier un groupe " \
"- Supprimer un groupe" \
"- Afficher un groupe" \
"- Creer une archive d'un repertoire " \
"- Restaurer une archive d'un repertoire " \
"- Visualiser le contenu d'une archive " \
"- Compresser une archive a l'aide de gzip " \
"- Decompresser une archive a l'aide de gunzip " \
"- Quitter"
do
    case "$REPLY" in
        1)      cree_user ;;
        2)      modif_user ;;
        3)      delete_user ;;
        4)      affiche_user ;;
        5)      cree_liste_user ;;
        6)      cree_group ;;
        7)      modif_group ;;
        8)      delete_group ;;
        9)      affiche_group ;;
       10)     archive_rep ;;
       11)     restaure_rep ;;
```

```
12)    affiche_archive ;;
13)    compress_archive ;;
14)    decompress_archive ;;
15)    printf "\nFin de traitement\n\n"
        break ;;
*)    print "Mauvais choix." ;;

esac
print
pause
done
exit 0
```

From:

<https://www.ittraining.team/> - **www.ittraining.team**



Permanent link:

<https://www.ittraining.team/doku.php?id=elearning:workbooks:script>

Last update: **2023/05/04 09:10**