Collège Lionel-Groulx Département d'informatique
Hiver 2018

Système exploitation 2

Cours: 420-KHC-LG

Travail Pratique intégration final

Compétences à développer

Ce laboratoire vous permettra de développer les compétences liées aux aspects suivants :

- Utiliser les fonctions disponibles des appels systèmes pour programmer des commandes Linux en langage C/C++,
- Utiliser les notions relatives aux scripts pour développer un script fonctionnel,
- Porter un programme qui fonctionne sous Windows vers le système Linux.

Travail à réaliser (équipe de 2 personnes maximum)

Ce travail est composé de trois (3) parties.

Partie 1 : Réalisation d'un script pour créer des comptes usagers automatiquement.

On vous demande de réaliser un script qui permettra de créer un certain nombre d'usager sur le système. Vous avez déjà réalisé un script qui permettait de créer un usager et un groupe. Vous pouvez aussi vous inspirer de votre script du calcul d'une facture pour aller lire les éléments d'un fichier.

La compagnie comporte plusieurs départements. En fait, il existe au total 10 départements pour lesquels on doit créer un certain nombre de compte usager. La compagnie vous donne le format du fichier qui contiendra les informations des comptes à créer. Ce fichier s'appelle « userfile.txt » et a le format suivant :

NomCompte:MotPasse:Groupe:Shell

À vous de créer ce fichier pour faire vos tests. Par contre, j'utiliserai exactement le même format de fichier mais avec plusieurs compte (au delà d'une cinquantaine) pour tester votre script.

On vous mentionne que le 4e champ (le shell) peut-être vide. Auquel cas, le shell « bash » sera alors le shell par défaut.

Ex. :	stef:stef:Developpement :	Ce compte utilisera donc le « bash ».
	jack:jack:Web:bash	Celui-ci utilisera le « bash ».
	joe:joe:Marketing:csh	Celui-ci utilisera le « csh ».

Dans l'exemple ci-dessus, un compte nommé « jack » sera créé avec le mot de passe « jack ». Cet usager sera dans le groupe d'usager nommé « Web » et aura le « /bin/bash » comme interpréteur de commande(shell). L'utilisateur « joe » aura, quant à lui, le shell « /bin/csh ».

On vous demande de créer un script qui permettra la création automatisée des usagers en lisant les données dans un fichier texte qui a le format ci-dessus.

Les répertoires « maison » (home directories) des usagers prendront la forme suivante :

« /home/nom_groupe/nom_Compte.

Ainsi l'usager « stef » ci-dessus aura le répertoire maison : « /home/Developpement/stef ».

Rappelez-vous de toutes les étapes pour créer un usager(Référence Module 6) Ce sont ces étapes que vous devrez automatiser en lisant les données qui proviennent du fichier texte.

Indice : Lors de la création d'un mot de passe, je vous suggère d'utiliser la commande « chpasswd » plutôt que « passwd ». Elle s'utilise de la façon suivante :

```
echo Nom_Utilisateur:Mot_de_passe | sudo chpasswd
```

Partie 2: Réalisation d'une commande Linux à l'aide du langage C/C++

Cette portion du travail consiste à réaliser une nouvelle commande Linux à l'aide du langage C++.

Au lieu de se rappeler les différents caractères à utiliser dans la commande chmod ou encore les valeurs octales, nous aimerions plutôt utiliser directement les caractères r w x dans la commande tel qu'on aimerait les voir afficher.

De façon plus spécifique:

- Il s'agit de permettre à l'usager de pouvoir changer les droits d'un fichier ou d'un dossier en utilisant la syntaxe rwx rwx rwx directement à la ligne de commande.

- Votre commande se nommera « monchmod » et effectuera une opération similaire à la commande « chmod » que vous connaissez.

Donc au lieu d'écrire chmod u+rwx,g+rx-w lefichier.txt
votre commande permettra d'écrire :

./monchmod rwxr-x--- lefichier.txt

Exemple d'utilisation :

./monchmod r-xr----- Dossier
Change les droits du dossier pour r-xr-----

./monchmod r-------- ex1.cpp
Change les droits du fichier ex1.cpp pour lecture à l'usager seulement.

Gestion des erreurs

- Si le fichier ou le dossier n'existe pas, un message d'erreur approprié est affiché

- Si il n'y a pas 9 caractères pour les droits, une erreur est affichée.
  Ex. : ./monchmod r-- r-- ex2.cpp → erreur minimum de 9 caractères non respectés
- Si le caractère n'existe pas, un message d'erreur est également affiché.
  Ex. : ./monchmod rpx –t r-- → Erreur caractère non reconnu.

- Si le caractère n'est pas au bon endroit, un message d'erreur est affiché.
  Ex. : ./monchmod wrxr--r-- → Mauvais caractère à la position 1 w

Partie 3: Porter un programme d'une plate-forme Windows vers une plate-forme Linux

La bataille des vaisseaux et des réparateurs dans un écran Linux près de chez vous!

Cette partie vous permettra de transformer un programme destiné à la plate-forme Windows vers une plate-forme Linux.

Plus précisément, vous devrez prendre un programme que vous avez fait lors de votre cours de « Programmation Orienté-Objet » sous Windows et le faire fonctionner sous Linux.

Le programme sera le même pour tous. Il s'agit de l'exercice des « vaisseaux et réparateurs » (exercice 6.2) tel que programmé dans votre cours d'orienté-objet.

Vous devrez donc porter ce programme pour qu'il fonctionne adéquatement sous la plate-forme Linux.

À remettre et à faire vérifier

Fonctionnement :

- Montrer le fonctionnement de la partie 1, 2 et 3 au plus tard le jeudi 17 mai 2018 au laboratoire.

Remise du code source :

- **Pour les 3 portions de ce travail, vous remettez le code source de chacun des projets ainsi que l'exécutable dans la boîte de remise Col.Net prévue à cet effet au plus tard le jeudi 17 mai 2018 avant 23:59.**

Barême et pointage

- Portion 1 Script : Fonctionnement 40%
- Portion 2 Programme : Fonctionnement 40%
- Portion 3 Windows->Linux: fonctionnement : 10%

- Commentaires et logique du programme : 10%