IUT de Lannion
Département INFO

Année universitaire 2014-2015
Première année

SYSTÈMES D'EXPLOITATION
Devoir Surveillé n°2

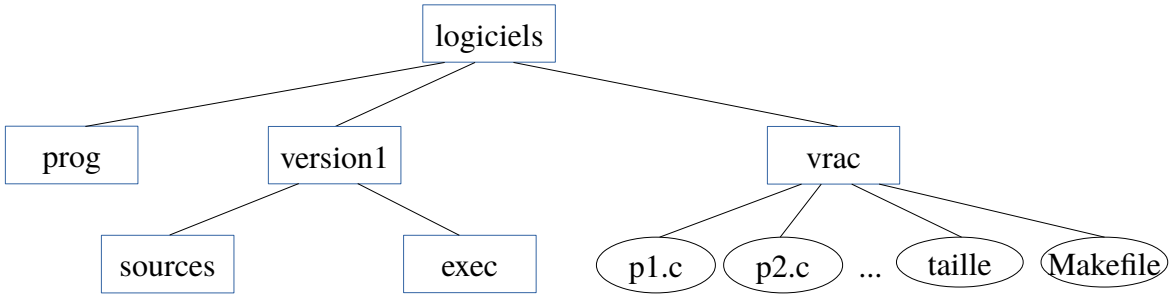
date : 28 janvier 2015 15h45
feuille A4 manuscrite personnelle autorisée.

durée : 1 heure
barème indicatif : P1/8, P2/12.

Lire les questions attentivement et complètement.

Partie 1: Protection des fichiers (24 minutes, 8 points)

Voici une partie de l'arborescence du système de fichiers d'une entreprise qui développe des logiciels (tous les dossiers et fichiers ne sont pa représentés).



A) droits des commandes sur les dossiers et les fichiers.

On se **place** initialement sur le dossier « **logiciels** ». On s'intéresse aux droits d'accès nécessaires pour pouvoir exécuter quelques commandes.
Répondez aux questions directement sur la feuille de réponses.

B) Configuration des droits.

Voici les noms des utilisateurs de l'entreprise et les groupes auxquels chacun appartient :

nom	groupe
Andrée	dev1
Carole	dev2
Jérôme	dev2
Marc	dev1
Sophie	resp

Dans cet exercice, pour ces utilisateurs, on souhaite autoriser ou interdire certaines opérations sur certains dossiers, et on vous demande de proposer une attribution des rôles et des droits associés sur ces dossiers.

Répondez aux questions directement sur la feuille de réponses.

Partie 2: Scripts Bash (36 minutes, 12 points)

Préparez vos réponses sur papier de brouillon avant de les recopier au propre. Au moins barrez proprement, avec une règle. Les copies sales seront pénalisées.

Proposez des scripts qui répondent le mieux possible à ce cahier des charges. Faites ce que vous pouvez. Tout élément écrit correct et pertinent rapporte des points.

On se trouve dans un dossier dans lequel il y a plusieurs fichiers source, certains en langage C, d'autres en Java. On voudrait écrire un script appelé `comp` qui compile automatiquement tous ces fichiers, si nécessaire.

Pour compiler un source appelé `prog.c`, il faut faire `cc -prog.c o prog -Wall`. Pour compiler un source appelé `prog.java`, il faut faire `javac prog`.

A) Compilation selon l'extension

Écrire le script `comp` qui parcourt les fichiers du répertoire courant et les compile selon leur extension `.c` ou `.java`. Réfléchissez bien car il existe des façons de faire plus simples que d'autres.

B) Compilation seulement si nécessaire

Maintenant, on se rend compte qu'à chaque fois qu'on lance le script `comp`, il recompile tous les sources, y compris ceux qui n'ont pas du tout changé. Toutefois, il est possible aussi qu'on ait rajouté de nouveaux sources depuis la précédente compilation – seulement ceux-là doivent être compilés.

La commande `stat -c %Y` *`fichier`* affiche la date de modification du fichier sous la forme d'un nombre entier. C'est le nombre de secondes écoulées depuis le 1er janvier 1970. On va se servir de ce nombre pour savoir s'il est nécessaire de recompiler un source. Qu'est-ce qu'on observe quand on compare la date d'un source (sous forme de ce nombre de secondes) et celle de l'exécutable correspondant quand ils sont à jour ?

1) Soit un fichier source `prog.c` ou `prog.java`. Sous quelles conditions est-il nécessaire de le recompiler ? Pensez à tous les cas de figure.

2) Écrire la nouvelle version du script `comp`.

C) Archivage des sources

En fait, tous ces sources font partie d'un grand projet. On voudrait tous les enregistrer dans une archive tar. Il s'agit d'un fichier contenant d'autres fichiers, comme une archive zip sur Windows. La commande pour créer une archive est : `tar cf archive.tar fichier1 fichier2…`

Ce qu'il faut, c'est appeler cette commande avec tous les sources `.c` et `.java`, mais seulement si c'est utile : il ne faut pas le faire si l'archive existe déjà et que aucun des fichiers n'est plus récent qu'elle.

Écrire le script `MkArc` qui crée l'archive seulement si nécessaire. Le nom de l'archive est fourni en paramètre du script `MkArc`. Par exemple, on tape `MkArc projet.tar`. Signalez une erreur si ce nom n'est pas fourni, ou si par erreur, ce n'est pas un nom en `.tar`.