Réalisation d'un programme d'envoi de messages sécurisé entre les utilisateurs d'une même machine.

Présentation

Conditionnement : Binômes de 2 ou 3 personnes.

Restrictions : Pas de copier/coller entre les groupes. Google autorisé (évidemment).

Rendu : Code propre, mini-rapport (quelques pages) détaillant la mise en oeuvre, à rendre avant le 17 décembre à 23:00, sur ARCHE. Un point en moins par heure de retard.

Questions : guillaume@netlor.fr avant le 10 décembre. 0,5 points en moins par question après cette date, sauf question pertinente ou tentative de corruption.

Le but de ce projet est de réaliser un système d'échange de messages entre les utilisateurs d'une même machine. Pour rappel, il est possible d'avoir un nombre illimité d'utilisateurs sur une machine Linux, qui peuvent être connectés physiquement dessus, ou par SSH.

Ce logiciel de messagerie doit pouvoir :

- **Permettre l'échange de messages texte entre utilisateurs d'une même machine :** Via une interface, un utilisateur doit pouvoir envoyer des messages à un ou plusieurs utilisateurs de la machine. Les utilisateurs seront idéalement listés dans une fenêtre (commande "dialog"). Les messages contiennent des destinataires, un objet, une date d'envoi, et un contenu texte.
- **Il peut y avoir un ou plusieurs destinataires au messages :** Un même message peut être envoyé à un ou plusieurs utilisateurs
- **Un utilisateur non-destinataire ne doit pas pouvoir le lire, modifier, ou le supprimer :** Un utilisateur de la machine qui n'a pas reçu un message, ne doit pas pouvoir le lire en accédant au fichier, ni le supprimer, ni le modifier. On utilisera pour cela une bonne

gestion des permissions, et un chiffrement GPG. Notez qu'un unique fichier peut être chiffré pour plusieurs personnes d'un coup.

- **Un message supprimé par un destinataire ne doit pas être supprimé pour les autres :** Si un message est supprimé par un destinataire, cela ne doit pas affecter les autres utilisateurs. Deux solutions sont envisageables : faire une copie du message (chiffré !) pour chaque utilisateur, ou retirer l'utilisateur du chiffrement.
- **Compresser les messages anciens :** Les messages datés de plus de 24 heures doivent être compressés (avec la commande "gzip") pour économiser de l'espace sur le disque. Lorsqu'on veut lire le message, il faudra donc décompresser le fichier avant la lecture.
- **Bonus : Possibilité d'ajouter des pièces jointes :** Même principe, la pièce jointe est chiffrée et ne peut pas être lue par des non-destinataires.

Gestion des permissions

Ce projet part du principe que les utilisateurs lancent le script en tant que leur utilisateur, et qu'il n'y a pas d'accès root direct. Afin de pouvoir modifier correctement les permissions (propriétaire, etc), il faudra donc utiliser la commande "sudo". Cependant, par défaut, la commande "sudo" permet à tout utilisateur du groupe "sudoers" de lancer n'importe quelle commande en root, ce qui n'est pas le comportement souhaité.

En modifiant le fichier /etc/sudoers, il est possible d'autoriser l'accès à une commande ou un script en particulier, en bloquant toujours les autres. Pour cela, modifiez (en root) le fichier /etc/sudoers, et ajoutez la ligne suivante :

```
%group  ALL=NOPASSWD: /bin/rmdir
```

Où *%group* correspond au groupe pouvant lancer la commande (on partira du principe que tous les utilisateurs sont dans un groupe "emails" par exemple, donc on mettra "%emails"), *ALL=NOPASSWD* permet d'autoriser sudo sans mot de passe, et */bin/rmdir* correspond à la commande à autoriser.
Dans le cadre du projet, il suffit juste d'autoriser le fichier principal de votre script à s'exécuter en tant que root. On peut imaginer avoir un fichier "/usr/local/bin/monmail.sh" qui ne fait qu'appeler "sudo /usr/local/sbin/monvraimail.sh" afin de ne pas demander à l'utilisateur de

taper systématiquement “sudo monmail.sh”, mais simplement “monmail.sh”. Dans le fichier sudoers, on autorisera alors “/usr/local/sbin/monvraimail.sh”.