Gestion de mots de passe sécurisée sous Linux avec `pass`

Arnaud Tisserand

Décembre 2023

Licence

Objectifs de l'intervention

- Essayer d'aider à utiliser Linux dans un cadre professionnel individuel (auto-gestion machine) ou personnel
- Indiquer des recommandations officielles en terme de cybersécurité
- Expérimenter dans un environnement dédié
- Nous faire nous questionner sur nos pratiques

Limites de l'intervention

- Proposition individuelle de l'auteur sans lien avec son employeur, laboratoire de recherche ou établissement d'hébergement
- L'auteur n'est pas en mesure de faire des recommandations en terme de sécurité

Plan de la séance

1. Rappels : documents et référentiels importants, éléments de contexte, bonnes pratiques
2. Présentation rapide de l'outil `pass`
3. Démo de l'outil `pass` (auteur)
4. Partie pratique (chaque personne)

Rappels : documents importants

- ANSSI :
  - Recommandations relatives à l'authentification multifacteur et aux mots de passe
  - RGS Annexe B3 : Règles et recommandations concernant les mécanismes d'authentification
  - Guide d'hygiène informatique
- CNIL :
  - Mots de passe : une nouvelle recommandation pour maîtriser sa sécurité
  - Générer un mot de passe solide
  - De « azerty » à « pa$$word », une revue des pratiques de gestion des mots de passe
- Recommandations et guides de l'équipe de sécurité de votre distribution Linux

Rappels : bonnes pratiques (1/2)

Quelques recommandations ANSSI du guide Recommandations relatives à l'authentification multifacteur et aux mots de passe :

- Privilégier l'authentification multifacteur
- Générer les éléments aléatoires avec un générateur de nombres aléatoires robuste
- Mettre en place une politique de sécurité des mots de passe
- Imposer une longueur minimale pour les mots de passe
- Ne pas imposer de longueur maximale pour les mots de passe
- Mettre en œuvre des règles sur la complexité des mots de passe
- Limiter la durée de validité d'une session authentifiée
- Ne pas imposer par défaut de délai d'expiration sur les mots de passe des comptes non sensibles
- Imposer un délai d'expiration sur les mots de passe des comptes à privilèges

Rappels : bonnes pratiques (2/2)

- Révoquer immédiatement les mots de passe en cas de compromission suspectée ou avérée
- Mettre en place un contrôle de la robustesse des mots de passe lors de leur création ou de leur renouvellement
- Utiliser un sel aléatoire long
- Mettre à disposition un coffre-fort de mots de passe
- [Utilisateur] Utiliser des mots de passe robustes
- [Utilisateur] Utiliser un mot de passe différent pour chaque service
- [Utilisateur] Utiliser un coffre-fort de mots de passe
- [Utilisateur] Protéger ses mots de passe
- [Utilisateur] Utiliser un mot de passe robuste pour l'accès à sa messagerie électronique
- [Utilisateur] Choisir un mot de passe sans information personnelle
- [Utilisateur] Modifier les mots de passe par défaut

Différents types de mots de passe

- Accès physique uniquement $\Longrightarrow$ entrée manuelle $\Longrightarrow$ se souvenir de chaque mot de passe
- Accès possible depuis une machine de confiance $\Longrightarrow$ utilisation d'un **coffre fort de mots de passe**

Taille minimale d'un mot de passe

Source : Référentiel Général de Sécurité, version 1.0, Annexe B3, Authentification, Règles et recommandations concernant les mécanismes d'authentification, Version 1.0 du 13 janvier 2010, section "C.1. Mot de passe à usage unique", page 25.

Entropie minimale recommandée : $\geq$ 100 bits (longue durée $\geq$ 128 bits)

Caractéristiques du mot de passe	Nombres total de symboles	10 symboles (chiffres)			26 symboles (lettres)			62 symboles (chiffres, majuscules, minuscules)			90 symboles (jeu de caractères complet)		
	Nombre de symboles par mot de passe	4	7	10	8	10	16	8	10	16	8	10	16
Taille de clé équivalente (bits)		13	23	33	38	47	75	48	60	95	52	65	104
Cassage exhaustif possible		oui	oui	oui	oui	oui	?	oui	oui	non	oui	oui	non

Figure 1: Entropie d'un mot de passe

Outil `pass` : infos et principes

- Site : https://www.passwordstore.org/
- Script shell `bash` d'environ 700 lignes distribué en licence GPL v2
- Repose sur des fichiers chiffrés avec `gpg`
- Permet de générer, modifier, utiliser des mots de passe
- Interface en ligne de commande (lien avec presse-papier de l'interface graphique)
- Fonctionne sous Linux et MacOS

Démo

Objectif : montrer rapidement les principaux points abordés dans la partie pratique.

Objectifs de la partie pratique

- Expérimenter `pass`
- Sans (trop) impacter l'environnement habituel
- Afin d'aider à étudier la conduite à tenir par la suite éventuellement

Limites de la partie pratique

- Avoir une machine (éventuellement virtuelle) Linux fonctionnelle
- Manipulations en ligne de commande dans un interpréteur shell (p. ex. `bash`)
- Ce qui est présenté ici n'est **pas** une solution de sécurité mais une simple introduction à l'outil `pass`
- Comment intégrer `pass` dans chaque environnement et pour chaque usage est **hors** de porté de cette intervention

Partie pratique

C'est parti ! Mais **attention** à votre configuration actuelle !

Installation (par utilisateur avec privilèges)

```
sudo apt install pass

# utile seulement pour partie pratique en utilisateur
# dédié sans session graphique propre
sudo apt install tmux
```

Regardons `pass`

```
which pass

wc `which pass`

# XXX est le nom de votre éditeur (nano, gedit, ...)
XXX `which pass`
```

Création d'un utilisateur dédié

Avertissement : faire les manipulations `gpg` et `pass` avec un utilisateur dédié à la démo permet de pas perturber la configuration actuelle de `gpg` (dans répertoire `~/.gnupg`) !

```
sudo useradd -m -d /home/demo -s /usr/bin/bash demo
sudo passwd demo
```

Changement d'utilisateur :

- se déconnecter de l'interface graphique, puis se reconnecter en utilisateur `demo` (solution la plus simple)
- changer d'utilisateur dans un terminal (`su - demo`) mais attention aux conflits de droits d'accès à l'interface pour entrer la passephrase

Mise en place d'une arborescence isolée pour `pass` et son environnement

Avertissement : faire les manipulations `gpg` et `pass` dans un espace dédié évite de perturber d'autres utilisations de `gpg` et permet d'avoir toutes les données dans un même espace.

```
mkdir --mode=700 -p ~/demopass/gpg
cd ~/demopass
ll              # ls -lah
```

Génération d'une clé gpg dédiée au coffre-fort

Attention de ne pas oublier `--homedir ~/demopass/gpg` dans vos commandes

```
gpg --homedir ~/demopass/gpg --full-gen-key
```

Remarques :

- Mémoriser le nom utilisé pour créer la clé `gpg` (nécessaire pour initialiser `pass`)
- En cas d'utilisateur qui accède à un shell dans un terminal d'un autre utilisateur (`su - demo`), il y aura un message d'erreur (`gpg: agent_genkey failed: Permission denied`) car l'accès TTY pour entrer la phrase de passe n'est pas possible, alors utiliser un outil comme `tmux`.

  ```
  ls -l `tty`
  ```

Regardons le trousseau de clé(s)

```
gpg --homedir ~/demopass/gpg -k
gpg --homedir=~/demopass/gpg -k --keyid-format LONG
```

Changer la durée de vie du cache de passphrases gpg

```
man gpg-agent
```

Mettre dans `~/demopass/gpg` le fichier `gpg-agent.conf` suivant :

```
default-cache-ttl 3600
max-cache-ttl 7200
```

Remarque importante

En cas de souhait d'adopter `pass` (ou un autre coffre-fort de mots de passe) dans le futur, il faudra **au préalable** étudier comment organiser, protéger, valider et sauvegarder l'environnement associé.

Ces aspects ne sont **pas** traités dans cette intervention.

Configuration de `pass` dans shell `bash` (1/2)

Créer le fichier `~/demopass/pass-demo-setup.sh` contenant les lignes suivantes :

```
export GNUPGHOME=~/demopass/gpg
export PASSWORD_STORE_DIR=~/demopass/password-store
```

Configuration de pass dans shell bash (2/2)

Ajouter les lignes suivantes en fin du fichier ~/.bashrc :

```
if [[ -e "${HOME}/demopass/pass-demo-setup.sh" ]]; then
    pass-demo-setup () {
        source "${HOME}/demopass/pass-demo-setup.sh"
    }
fi
```

Lancer un nouveau shell/terminal.

Initialisation du STORE de pass

```
pass init <<KEY_NAME>>
```

A priori, l'auto-complétion du nom de la clé fonctionne dans bash (pass init puis touche TAB).

```
ls -alr password-store

pass
```

Documentation en ligne

```
pass --help

man pass
```

Remarque: a priori dans bash, l'auto-complétion des noms de mots de passe fonctionne par défaut.

Experimentations

```
pass ls
pass
pass insert MDP
pass generate MDP
pass show MPD
pass show -c MDP
pass MDP
pass -c MDP
pass insert -m MDP
pass edit MDP
...
```

En guise de conclusion

À vous de voir comment intégrer un coffre fort de mots de passe dans votre environnement (attention de bien sauvegarder tout ce qu'il faut) !

Ménage de fin de séance :

```
# récupérer les fichiers nécessaires
# NE PAS LAISSER D'UTILISATEUR DORMANT !
sudo userdel --remove demo
```