Sécurité des Systèmes d'Information
Authentification et problèmes de sécurité

Martin Peres

Doctorant de Francine Krief au LaBRI

October 7, 2013

Sommaire

1. I - Définitions
   - Notions générales
2. II - Sécurité par le “je possède”
3. III - Je connais: les mots de passe

Identification

Procédure permettant de connaître l'identité d'une personne ou d'un service. Quelques méthodes d'identification:

- Nom prénom + date et lieu de naissance
- numéro sécurité sociale
- login

Authentification

Procédure permettant de vérifier l'identité d'une personne.

- signes distinctifs (taille, couleurs des yeux, etc...)
- mot de passe

Authentification forte

Une authentification est dite forte si au moins 2 des moyens d'authentification suivants sont utilisés:

- je connais
- je possède
- je suis
- je sais faire
- lieu

Authentification: Je connais

L'entité a une connaissance que seule elle connaît.

- mot de passe
- clé privée
- code pin (de carte SIM ou carte bleu)

Authentification: Je possède

L'entité possède quelque chose d'unique et non recopiable.

- carte bleu
- token RSA (Sera expliqué plus loin dans le cours)

Authentification: Je suis

Utilisation d'une caractéristique unique d'une entité:

- taille + âge + couleurs des yeux et cheveux
- un nombre anormal de doigts ou orteils
- reconnaissance vocale
- empreinte digitale or de l'iris
- position des vaisseaux sanguins (doigt ou iris)

Authentification: Je sais faire

L'entité est seule capable de faire une action.

- Ulysse et son arc + faire passer la flèche dans 5 anneaux
- Jouer de la musique?

Authentification: Le lieu

L'entité se trouve dans un lieu bien défini. Exemple:

- donner l'accès à internet aux gens d'un bâtiment et pas ses voisins

Sommaire

1 I - Définitions

2 II - Sécurité par le “je possède”
- Comment vérifier qu’un utilisateur a bien un objet sur lui?
- Carte à puce
- Token RSA

3 III - Je connais: les mots de passe

Comment vérifier qu’un utilisateur a bien un objet sur lui?

- Accès physique: Carte à puce
- Accès distant: Token RSA

Carte à puce : Késako?

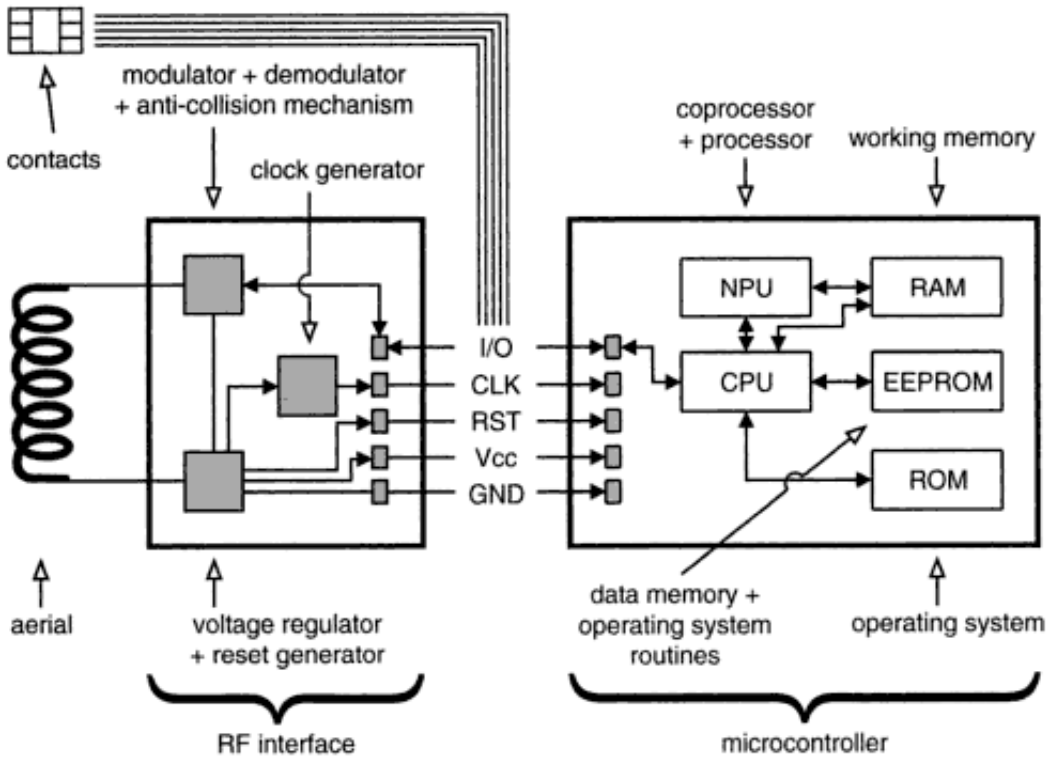
- micro ordinateur cryptographique
- stocke une clé privée protégée physiquement
- peut signer des données
- généralement seulement accessible après avoir tapé un code PIN

Utilisé dans les ...

- téléphones portables (authentification sur le réseau)
- cartes bancaires (authentification du porteur)
- cartes téléphoniques (décompte des unités inchangeable)
- carte vitale (stockage des informations santé du porteur)

Coût

- fixe: lecteur($>= 20€$), carte($>= 1€$)
- par mois: virtuellement nul



CC-By-SA. Author: Vlopes

Token RSA : Késako?

- micro ordinateur cryptographique
- stocke un secret partagé unique et protégé physiquement
- contient une horloge précise
- génère un token valide durant quelques secondes à chaque appuis sur un bouton

Utilisé dans les ...

- les grandes entreprises pour accéder à des données sensibles

Coût

- fixe: environ 90€
- par mois: environ 20€

Sommaire

1 I - Définitions

2 II - Sécurité par le “je possède”

3 III - Je connais: les mots de passe
- Qu’est ce qu’un mot de passe sûr?
- Comment stocker un mot de passe?
- L’authentification et les mots de passe dans Linux

Définition

Un mot de passe sûr est un mot de passe qu'il est difficile de deviner. Autrement dit, il doit avoir une certaine entropie.

Attaques sur les mots de passe

Un mot de passe peut être attaqué par les méthodes suivantes:

- connaissance de la personne et de son environnement
- attaque par transformations (a $->$ 4, e $->$ 3, o $->$ 0)
- attaque par dictionnaire/rainbow tables
- attaque par force brute

Source d'entropie

- longueur du mot de passe
- symboles utilisés (numériques, alphabétiques, majuscule/minuscules, caractères spéciaux, mots)

Le bon mot de passe?

- facile à retenir ou retrouver (pour ne pas devoir l'écrire quelque part)
- dur à deviner
- utilisé sur un et unique service

Idée de mots de passes

- formule mathématique/physique sans sens particulier
- ligne de code en C
- prendre la marseillaise et lire une lettre sur 3 (data connu, transformation inconnue)

Exemple de générateur de mot de passe sûr

pwd = sha1(master_pwd + '_' + service)
Génère un mot de passe:

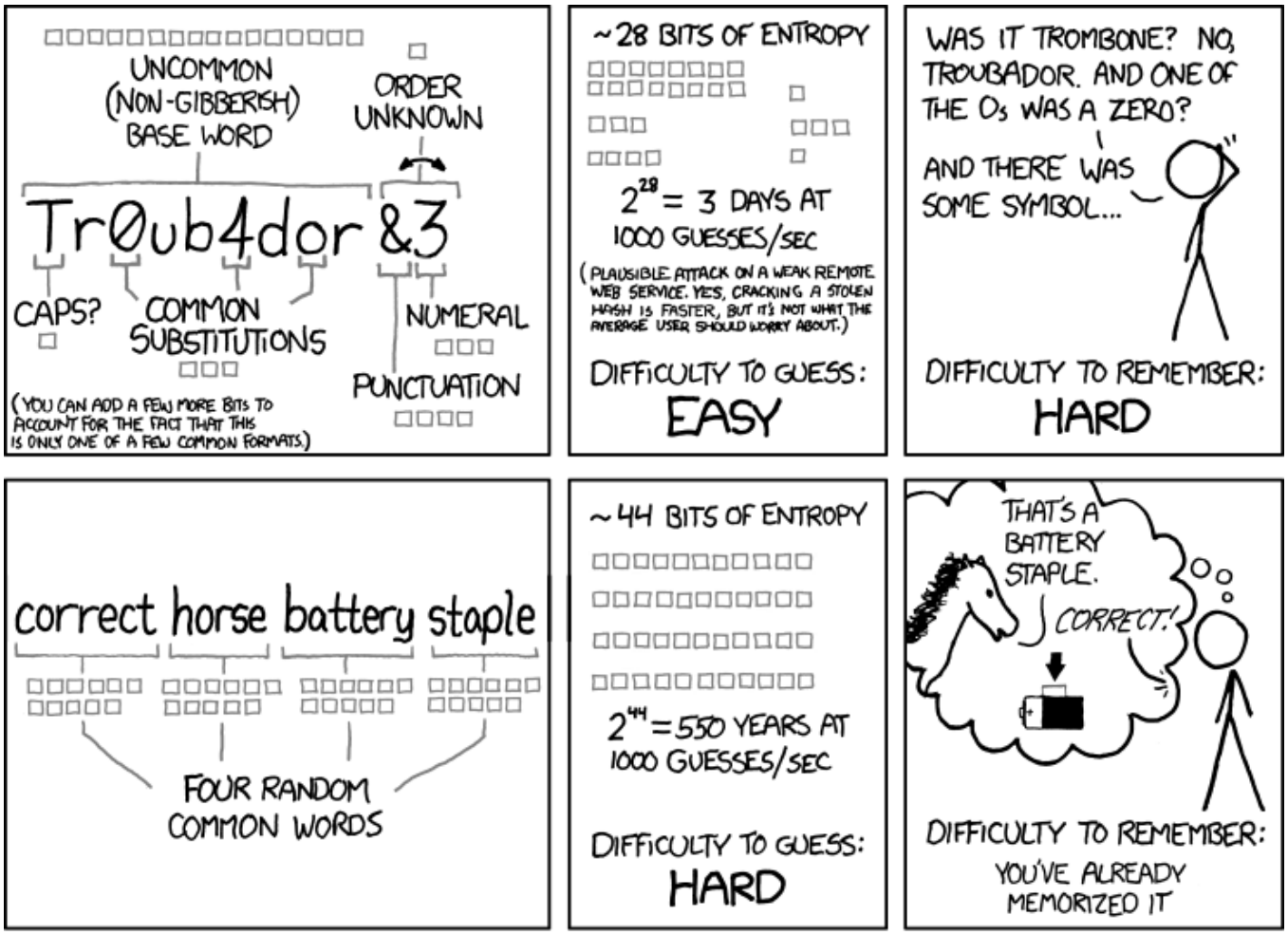
- long (20 chars)
- avec tous les caractères
- unique par service
- facile à re-générer (on doit juste se souvenir du master_pwd)

Exemples de mots de passes: identifiez les attaques probables

- toto
- azerty
- thomas
- rammstein
- l33t
- a£@23ùaz (8 chars, ANSI)
- bonjour monsieur soleil

À lire: Article très intéressant sur la sécurité des mots de passe

http://www.baekdal.com/insights/password-security-usability



Source: https://xkcd.com/936/

Comment stocker un mot de passe?

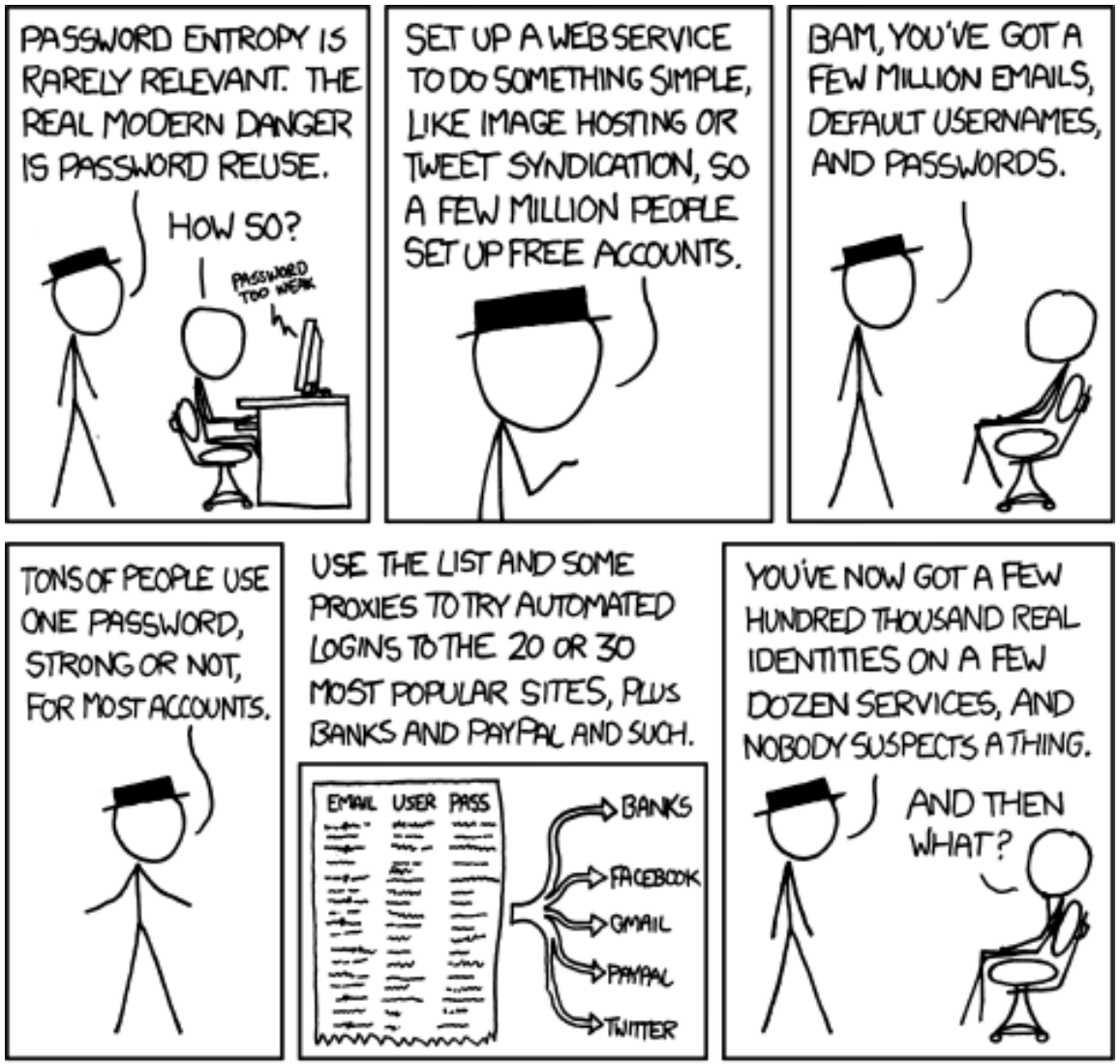
Un mot de passe doit être stocké:

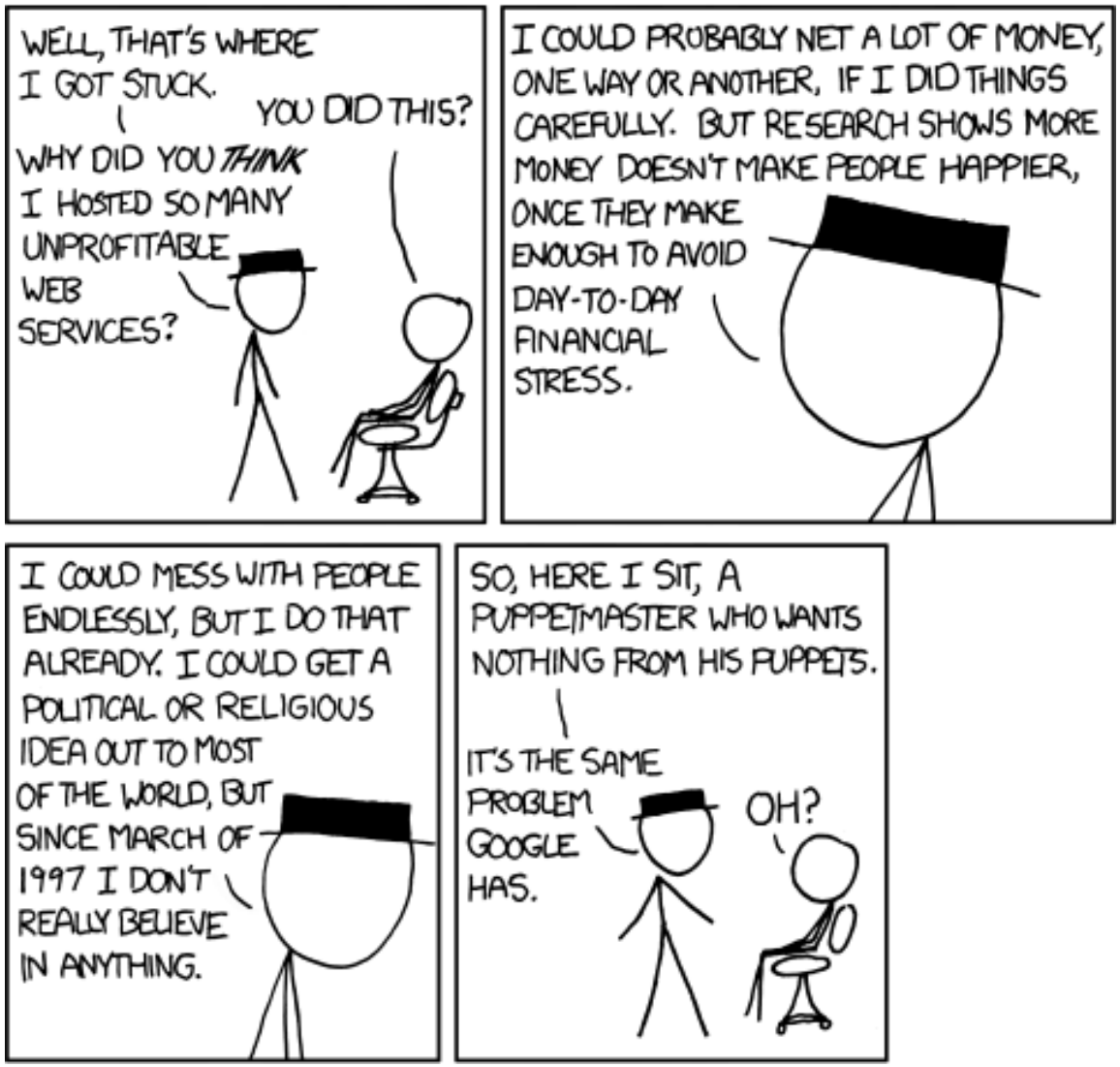
- haché et salé du coté du service authentifiant
- chiffré du coté du client

Sauvegarde coté serveur

Le mot de passe doit être stocké haché et salé car:

- un hacker pourrait dumper la bdd et retrouver les mots de passe
- le service n'a pas besoin du mot de passe lui même


PASSWORD ENTROPY IS RARELY RELEVANT. THE REAL MODERN DANGER IS PASSWORD REUSE.
HOW SO?
PASSWORD TOO WEAK
SET UP A WEB SERVICE TO DO SOMETHING SIMPLE, LIKE IMAGE HOSTING OR TWEET SYNDICATION, SO A FEW MILLION PEOPLE SET UP FREE ACCOUNTS.
BAM, YOU'VE GOT A FEW MILLION EMAILS, DEFAULT USERNAMES, AND PASSWORDS.
TONS OF PEOPLE USE ONE PASSWORD, STRONG OR NOT, FOR MOST ACCOUNTS.
USE THE LIST AND SOME PROXIES TO TRY AUTOMATED LOGINS TO THE 20 OR 30 MOST POPULAR SITES, PLUS BANKS AND PAYPAL AND SUCH.
EMAIL USER PASS
BANKS
FACEBOOK
GMAIL
PAYPAL
TWITTER
YOU'VE NOW GOT A FEW HUNDRED THOUSAND REAL IDENTITIES ON A FEW DOZEN SERVICES, AND NOBODY SUSPECTS A THING.
AND THEN WHAT?


WELL, THAT'S WHERE I GOT STUCK.
WHY DID YOU THINK I HOSTED SO MANY UNPROFITABLE WEB SERVICES?
YOU DID THIS?
I COULD PROBABLY NET A LOT OF MONEY, ONE WAY OR ANOTHER, IF I DID THINGS CAREFULLY. BUT RESEARCH SHOWS MORE MONEY DOESN'T MAKE PEOPLE HAPPIER, ONCE THEY MAKE ENOUGH TO AVOID DAY-TO-DAY FINANCIAL STRESS.
I COULD MESS WITH PEOPLE ENDLESSLY, BUT I DO THAT ALREADY. I COULD GET A POLITICAL OR RELIGIOUS IDEA OUT TO MOST OF THE WORLD, BUT SINCE MARCH OF 1997 I DON'T REALLY BELIEVE IN ANYTHING.
SO, HERE I SIT, A PUPPETMASTER WHO WANTS NOTHING FROM HIS PUPPETS.
IT'S THE SAME PROBLEM GOOGLE HAS.
OH?

Source: https://xkcd.com/792/

Attaque sur le mot de passe stocké haché: Les rainbow tables

Les rainbow tables sont des dictionnaires de mots de passes habituels déjà hashés pour diminuer le temps de l'attaque.

- Elles peuvent prendre plusieurs Go
- Elles sont spécifique à un algorithme de hash (md5, sha1, etc...)

Solution: Le salt

Pour augmenter la sécurité contre les attaques par rainbow tables, on peut concaténer un nombre connu mais unique.

- Augmente le nombre de possibilités donc la taille de la rainbow table
- $--> $ rend les rainbow tables inutiles
- pwd_stored = (salt, crypto_hash(salt + pwd))

Sauvegarde coté client

Le mot de passe doit être stocké chiffré:

- car on doit avoir le vrai mot de passe pour l'envoyer au service
- mais on ne veut pas qu'il soit lisible par n'importe qui

Solution classique de stockage de mots de passe

- en clair dans un fichier ou la base de registre (pidgin)
- en base64 pour le rendre illisible directement par un humain
- chiffré par un algorithme propriétaire sans clé (sécurité par l'obscurité)

Proposition

Stocker la liste de ses mots de passes dans un fichier chiffré par un mot de passe unique (master password):

- permet de n'avoir qu'un seul mot de passe à retenir
- permet d'avoir autant de mots de passes que de services

Key streching: Augmenter la sécurité de la master key

Comment obtenir une clé plus grande que la taille initiale ?

- Appliquer des transformations irréversibles et lentes
- Example: pwd = hash(hash(hash(hash(hash(...(passphrase)...)))))

Systèmes de stockages sécurisé de mots de passe

- Linux: KWallet, Gnome Keyring
- Windows: Windows Vault
- Mac OS X: Key chain

/etc/passwd

- contient la liste des utilisateurs “UNIX”
- format: login:x:uid:gid:Nom complet:home directory:shell
- lisible par tous (755)
- pour plus d'infos: man 5 passwd

Example: /etc/passwd

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/bin/false
http:x:33:33:http:/srv/http:/bin/false
kdm:x:135:135::/var/lib/kdm:/bin/false
mupuf:x:1000:1000:Martin Peres,68,,:/home/mupuf:/bin/bash
poulpe:x:1001:100::/home/poulpe:/bin/bash
```

/etc/shadow

- contient les mots de passe des utilisateurs "UNIX"
- Format: login:pwd_chiffré:last_changed:age_min:age_max: age_warn_period:age_expiriation:fin_validité
- lisible par root uniquement (700)
- pour plus d'infos: man 5 shadow, man passwd

Example: /etc/passwd

```
poulpe:d92.Y4wwOvWD2:15286:0:99999:7:::
```

Retrouver un mot de passe à partir du fichier shadow

Vous pouvez utiliser John the ripper. Il supporte:

- les dictionnaires
- l'écriture de transformations
- example: john crackme.txt

Example: core i7, mono thread

- user1:lol $--> 1$ min 25
- user2:plop $--> 45$ secondes
- user3:azerty $--> 3$ heures 21

PAM

Solution d'authentification sous Linux:

- fichiers situés dans /etc/pam.d/
- utilisations de modules
- défini une suite logique d'action pour authentifier (required, optional, sufficient, etc...)

Modules PAM

- auth: fourni l'authentification
- account: verifie les droits (compte valide, heure de la journée)
- password: utilisé pour définir un mot de passe
- session: utilisé après l'authentification d'un utilisateur (monter des partitions sécurisées, ...)

Indicateurs de contrôle PAM

- required: nécessaire pour la réussite
- sufficient: au moins un module indiqué suffisant doit réussir
- optional: pas obligatoire

/etc/pam.d/login example

```
#%PAM-1.0
auth       required   pam_securetty.so
auth       required   pam_unix.so shadow nullok
auth       required   pam_nologin.so
account    required   pam_unix.so
password   required   pam_cracklib.so difok=2 minlen=8 dcredi
password   required   pam_unix.so shadow nullok use_authtok
session    required   pam_unix.so
```

Le graal : Le Single Sign-On (SSO)

- S'authentifier une fois pour accéder à tous les services
- Un tier peut authentifier pour tout le monde (OpenID/Facebook/Google)

Avantages

- Réduit le nombre de mots de passes
- Pas besoin de créer un compte par service
- Mise à jour facilitée des données

Inconvénients

- L'authentification doit être solide pour ne pas donner les clés du chateau à un attaquant (king of the castle).