PAM et NSS

Olivier Thauvin

CNRS/LATMOS

ANF Les systèmes d'authentification dans la communauté ESR : étude, mise en œuvre et interfaçage dans un laboratoire de Mathématique
Angers, 22 - 26 septembre 2014

Progression

Moi...

Officiellement :

- Admin. Système et Réseau au LATMOS
  - Système Unix, réseau
  - Developeur WEB : PERL, PostgreSQL
- Admin. de la forge de l'IPSL

Sinon :

- Ex packager Mandriva puis Mageia
- Developeur de `http://sophie.zarb.org/`
- Quelques modules perl sur le CPAN `http://search.cpan.org/~nanardon/`

Merci Wikipédia TM

Identitification

Désigne l'action consistant à identifier (donner, attribuer un nom ou un code en propre à la chose ou la personne ainsi reconnue) un objet ou un individu.

Authentification

Est la procédure qui consiste, à vérifier l'identité d'une personne ou d'un ordinateur

En simple

L'identification permet donc de connaître l'identité d'une entité alors que l'authentification permet de vérifier cette identité.

- identification : login
- authentification : le mot de passe

Merci Wikipédia TM

Identitification

Désigne l'action consistant à identifier (donner, attribuer un nom ou un code en propre à la chose ou la personne ainsi reconnue) un objet ou un individu.

Authentification

Est la procédure qui consiste, à vérifier l'identité d'une personne ou d'un ordinateur

En simple

L'identification permet donc de connaître l'identité d'une entité alors que l'authentification permet de vérifier cette identité.

- identification : login
- authentification : le mot de passe

Merci Wikipédia TM

Identitification

Désigne l'action consistant à identifier (donner, attribuer un nom ou un code en propre à la chose ou la personne ainsi reconnue) un objet ou un individu.

Authentification

Est la procédure qui consiste, à vérifier l'identité d'une personne ou d'un ordinateur

En simple

L'identification permet donc de connaître l'identité d'une entité alors que l'authentification permet de vérifier cette identité.

- identification : login
- authentification : le mot de passe

Historique

Il était une fois...

Réinventons la roue

Chaque application gérait elle même l'identification et l'authentification.

- souvent codé en dur dans l'application
- parfois configurable via recompilation
- parfois sans recompilation (via fichier de configuration)

Les problèmes

- plusieurs implémentations :
  - une gestion des bogues et failles de sécurité difficile
  - des configurations différentes par application
- parfois configurable seulement via recompilation
- parfois sans recompilation (via fichier de configuration)

Il était une fois...

Réinventons la roue

Chaque application gérait elle même l'identification et l'authentification.

- souvent codé en dur dans l'application
- parfois configurable via recompilation
- parfois sans recompilation (via fichier de configuration)

Les problèmes

- plusieurs implémentations :
  - une gestion des bogues et failles de sécurité difficile
  - des configurations différentes par application
- parfois configurable seulement via recompilation
- parfois sans recompilation (via fichier de configuration)

L'authentification historique sous UNIX

Les fichiers

- `/etc/passwd` :
  `login:pass:uid:gid:gecos:home:shell`
- `/etc/group` :
  `nom:pass:gid:member...`

Chiffrement (Linux)

- mots de passe chiffrés avec `crypt()`
- salt de 2 caractères
- chiffrement DES

Exemple pour « azerty » (salt « xx ») : `xxtE64eMGwDYY`

L'authentification historique sous UNIX

Les fichiers

- `/etc/passwd` :
  `login:pass:uid:gid:gecos:home:shell`
- `/etc/group` :
  `nom:pass:gid:member...`

Chiffrement (Linux)

- mots de passe chiffrés avec `crypt()`
- salt de 2 caractères
- chiffrement DES

Exemple pour « azerty » (salt « xx ») : `xxtE64eMGwDYY`

Mettre les mots de passe à l'ombre

Problématiques des temps anciens

- mots de passe (chiffrés) lisible par tous
- pas d'expiration des comptes
- pas d'expiration des mots de passe

Le fichier shadow :

- lisible uniquement par `root` (droits : `-r--r-----`)
- contient :

  `login:passwd:lstchg:min:max:warn:inact:expire:flag`
- mots de passe dans le `/etc/passwd` :
  - `x` en temps normal
  - `!!` pour les comptes bloqués

Mettre les mots de passe à l'ombre

Problématiques des temps anciens

- mots de passe (chiffrés) lisible par tous
- pas d'expiration des comptes
- pas d'expiration des mots de passe

Le fichier shadow :

- lisible uniquement par `root` (droits : `-r--r-----`)
- contient :

  `login:passwd:lstchg:min:max:warn:inact:expire:flag`
- mots de passe dans le `/etc/passwd` :
  - `x` en temps normal
  - `!!` pour les comptes bloqués

De la bonne utilisation de crypt()

Généralité

- le salt doit être aléatoire
- on réutilise la version chiffrée pour authentifier
- on s'assure d'utiliser le crypt du système

Générer la version chiffrée

```
$salt = random_string();
$encrypted = crypt($clear, $salt);
```

Vérifier un mot de passe

```
if ($encrypted eq crypt($clear, $encrypted)) {
    return ok;
} else {
    return error;
}
```

De la bonne utilisation de crypt()

Généralité

- le salt doit être aléatoire
- on réutilise la version chiffrée pour authentifier
- on s'assure d'utiliser le crypt du système

Générer la version chiffrée

```
$salt = random_string();
$encrypted = crypt($clear, $salt);
```

Vérifier un mot de passe

```
if ($encrypted eq crypt($clear, $encrypted)) {
    return ok;
} else {
    return error;
}
```

De la bonne utilisation de crypt()

Généralité

- le salt doit être aléatoire
- on réutilise la version chiffrée pour authentifier
- on s'assure d'utiliser le crypt du système

Générer la version chiffrée

```
$salt = random_string();
$encrypted = crypt($clear, $salt);
```

Vérifier un mot de passe

```
if ($encrypted eq crypt($clear, $encrypted)) {
    return ok;
} else {
    return error;
}
```

De la bonne utilisation de crypt()

Généralité

- le salt doit être aléatoire
- on réutilise la version chiffrée pour authentifier
- on s'assure d'utiliser le crypt du système

Générer la version chiffrée

```
$salt = random_string();
$encrypted = crypt($clear, $salt);
```

Vérifier un mot de passe

```
if ($encrypted eq crypt($clear, $encrypted)) {
    return ok;
} else {
    return error;
}
```

De la bonne utilisation de crypt()

Généralité

- le salt doit être aléatoire
- on réutilise la version chiffrée pour authentifier
- on s'assure d'utiliser le crypt du système

Générer la version chiffrée

```
$salt = random_string();
$encrypted = crypt($clear, $salt);
```

Vérifier un mot de passe

```
if ($encrypted eq crypt($clear, $encrypted)) {
    return ok;
} else {
    return error;
}
```

Les algorythmes rencontrés

Les chiffrements et leur salt

Salt	Type	niveau de sécurité
`XX`	DES-based	très faible
`$1$...$`	MD5-based	faible
`$2$...$`	blowfish	nulle
`$2a$...$`		forte
`$2b$...$`		
`$2x$...$`		
`$2y$...$`		
`$5$...$`	SHA-256	forte
`$6$...$`	SHA-512	forte

Progression

Name Service Switch

Besoins :

Obetnir les informations concernant :

- les comptes et groupes
- les protocoles et services réseaux (ex : $http = 80$)
- les correspondances adresses ip/nom

Qu'est-ce que c'est :

- suite fonctions inclus dans la (g)libc :
  - getpwnam,uid,ent : passwd
  - getgrnam,uid,ent : group
  - ...
- un fichier de configuration : `/etc/nsswitch.conf`
- des modules : `/lib64/libnss_*.so.2`
- un outil : `getent`

Name Service Switch

Besoins :

Obetnir les informations concernant :

- les comptes et groupes
- les protocoles et services réseaux (ex : $http = 80$)
- les correspondances adresses ip/nom

Qu'est-ce que c'est :

- suite fonctions inclus dans la (g)libc :
  - getpwnam,uid,ent : passwd
  - getgrnam,uid,ent : group
  - ...
- un fichier de configuration : `/etc/nsswitch.conf`
- des modules : `/lib64/libnss_*.so.2`
- un outil : `getent`

Name Service Switch

Besoins :

Obetnir les informations concernant :

- les comptes et groupes
- les protocoles et services réseaux (ex : $http = 80$)
- les correspondances adresses ip/nom

Qu'est-ce que c'est :

- suite fonctions inclus dans la (g)libc :
  - getpwnam,uid,ent : passwd
  - getgrnam,uid,ent : group
  - ...
- un fichier de configuration : `/etc/nsswitch.conf`
- des modules : `/lib64/libnss_*.so.2`
- un outil : `getent`

Name Service Switch

Besoins :

Obetnir les informations concernant :

- les comptes et groupes
- les protocoles et services réseaux (ex : $http = 80$)
- les correspondances adresses ip/nom

Qu'est-ce que c'est :

- suite fonctions inclus dans la (g)libc :
  - getpwnam,uid,ent : passwd
  - getgrnam,uid,ent : group
  - ...
- un fichier de configuration : `/etc/nsswitch.conf`
- des modules : `/lib64/libnss_*.so.2`
- un outil : `getent`

Name Service Switch

Besoins :

Obetnir les informations concernant :

- les comptes et groupes
- les protocoles et services réseaux (ex : $http = 80$)
- les correspondances adresses ip/nom

Qu'est-ce que c'est :

- suite fonctions inclus dans la (g)libc :
  - getpwnam,uid,ent : passwd
  - getgrnam,uid,ent : group
  - ...
- un fichier de configuration : `/etc/nsswitch.conf`
- des modules : `/lib64/libnss_*.so.2`
- un outil : `getent`

NSS : Configuration

nsswitch.conf

```
# table: modules...
passwd:         files  ldap
shadow:         files  ldap
group:          files  ldap
hosts:          mdns4_minimal files nis dns mdns4
```

Actions :

```
[STATUS=ACTION]
[!STATUS=ACTION]
```

- STATUS : success, notfound, unavail, tryagain
- ACTION : return, continue

NSS : Configuration

nsswitch.conf

```
# table: modules...
passwd:         files  ldap
shadow:         files  ldap
group:          files  ldap
hosts:          mdns4_minimal files nis dns mdns4
```

Actions :

```
[STATUS=ACTION]
[!STATUS=ACTION]
```

- STATUS : success, notfound, unavail, tryagain
- ACTION : return, continue

Getent

Utilisation simple :

```
# getent passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/bin/sh
...
```

Recherche :

```
# getent passwd thauvin
thauvin:x:6033:5000:Olivier Thauvin:/net/nfs/home/thau
```

Getent

Utilisation simple :

```
# getent passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/bin/sh
...
```

Recherche :

```
# getent passwd thauvin
thauvin:x:6033:5000:Olivier Thauvin:/net/nfs/home/thau
```

Généralité

Besoins :

- valider l'authentification
- parfois éviter l'authentification
- permettre le changement du mot de passe

Dans la pratique PAM c'est :

- une librairie : la libpam
- des fichiers de configuration (un par service)
- des modules (`/lib64/security/pam_*.so`)

Généralité

Besoins :

- valider l'authentification
- parfois éviter l'authentification
- permettre le changement du mot de passe

Dans la pratique PAM c'est :

- une librairie : la libpam
- des fichiers de configuration (un par service)
- des modules (`/lib64/security/pam_*.so`)

Généralité

Besoins :

- valider l'authentification
- parfois éviter l'authentification
- permettre le changement du mot de passe

Dans la pratique PAM c'est :

- une librairie : la libpam
- des fichiers de configuration (un par service)
- des modules (`/lib64/security/pam_*.so`)

Généralité

Besoins :

- valider l'authentification
- parfois éviter l'authentification
- permettre le changement du mot de passe

Dans la pratique PAM c'est :

- une librairie : la libpam
- des fichiers de configuration (un par service)
- des modules (`/lib64/security/pam_*.so`)

Fonctionnement

4 types :

- **account** : valide les données comptes
- **auth**entication : vérifie le mot de passe
- **session** : execute des actions juste avant l'ouverture de session
- **password** : appelé lors d'un changement de mot de passe
- **@include** : inclus une configuration complète

Chaque type appelle un ou plusieurs modules successivement et applique une réponse.

Exemple (simple)

```
auth       sufficient pam_rootok.so
auth       required   pam_env.so
account    required   pam_access.so
session    required   pam_loginuid.so
```

Fonctionnement

4 types :

- **account** : valide les données comptes
- **auth**entication : vérifie le mot de passe
- **session** : execute des actions juste avant l'ouverture de session
- **password** : appelé lors d'un changement de mot de passe
- **@include** : inclus une configuration complète

Chaque type appelle un ou plusieurs modules successivement et applique une réponse.

Exemple (simple)

```
auth       sufficient pam_rootok.so
auth       required   pam_env.so
account    required   pam_access.so
session    required   pam_loginuid.so
```

Généralité

Fonctionnement

4 types :

- **account** : valide les données comptes
- **auth**entication : vérifie le mot de passe
- **session** : execute des actions juste avant l'ouverture de session
- **password** : appelé lors d'un changement de mot de passe
- **@include** : inclus une configuration complète

Chaque type appelle un ou plusieurs modules successivement et applique une réponse.

Exemple (simple)

```
auth        sufficient pam_rootok.so
auth        required   pam_env.so
account     required   pam_access.so
session     required   pam_loginuid.so
```

Généralité

Fonctionnement

4 types :

- **account** : valide les données comptes
- **auth**entication : vérifie le mot de passe
- **session** : execute des actions juste avant l'ouverture de session
- **password** : appelé lors d'un changement de mot de passe
- **@include** : inclus une configuration complète

Chaque type appelle un ou plusieurs modules successivement et applique une réponse.

Exemple (simple)

```
auth        sufficient pam_rootok.so
auth        required   pam_env.so
account     required   pam_access.so
session     required   pam_loginuid.so
```

Fonctionnement

4 types :

- **account** : valide les données comptes
- **auth**entication : vérifie le mot de passe
- **session** : execute des actions juste avant l'ouverture de session
- **password** : appelé lors d'un changement de mot de passe
- **@include** : inclus une configuration complète

Chaque type appelle un ou plusieurs modules successivement et applique une réponse.

Exemple (simple)

```
auth        sufficient pam_rootok.so
auth        required   pam_env.so
account     required   pam_access.so
session     required   pam_loginuid.so
```

Généralité

Fonctionnement

4 types :

- **account** : valide les données comptes
- **auth**entication : vérifie le mot de passe
- **session** : execute des actions juste avant l'ouverture de session
- **password** : appelé lors d'un changement de mot de passe
- **@include** : inclus une configuration complète

Chaque type appelle un ou plusieurs modules successivement et applique une réponse.

Exemple (simple)

```
auth        sufficient pam_rootok.so
auth        required   pam_env.so
account     required   pam_access.so
session     required   pam_loginuid.so
```

Fonctionnement

4 types :

- **account** : valide les données comptes
- **auth**entication : vérifie le mot de passe
- **session** : execute des actions juste avant l'ouverture de session
- **password** : appelé lors d'un changement de mot de passe
- **@include** : inclus une configuration complète

Chaque type appelle un ou plusieurs modules successivement et applique une réponse.

Exemple (simple)

```
auth        sufficient pam_rootok.so
auth        required   pam_env.so
account     required   pam_access.so
session     required   pam_loginuid.so
```

Configuration

Syntaxe

```
type    contrôle    module.so    options_du_modules...
```

Les contrôles, version simple

- required : retournera l'erreur mais le processus continue
- requisite : retourne l'erreur immédiatement
- sufficient : retourne le succès immédiatement
- optional : le module n'a d'importance que s'il est seul
- required : retournera l'erreur mais le processus continue
- include : inclus la configuration donné en argument
- substack : inclus la configuration mais continue le processus

Configuration

Syntaxe

```
type    contrôle    module.so    options_du_modules...
```

Les contrôles, version simple

- required : retournera l'erreur mais le processus continue
- requisite : retourne l'erreur immédiatement
- sufficient : retourne le succès immédiatement
- optional : le module n'a d'importance que s'il est seul
- required : retournera l'erreur mais le processus continue
- include : inclus la configuration donné en argument
- substack : inclus la configuration mais continue le processus

Configuration

Syntaxe

```
type    contrôle    module.so    options_du_modules...
```

Les contrôles, version simple

- required : retournera l'erreur mais le processus continue
- requisite : retourne l'erreur immédiatement
- sufficient : retourne le succès immédiatement
- optional : le module n'a d'importance que s'il est seul
- required : retournera l'erreur mais le processus continue
- include : inclus la configuration donné en argument
- substack : inclus la configuration mais continue le processus

Configuration

Syntaxe

```
type    contrôle    module.so    options_du_modules...
```

Les contrôles, version simple

- required : retournera l'erreur mais le processus continue
- requisite : retourne l'erreur immédiatement
- sufficient : retourne le succès immédiatement
- optional : le module n'a d'importance que s'il est seul
- required : retournera l'erreur mais le processus continue
- include : inclus la configuration donné en argument
- substack : inclus la configuration mais continue le processus

Configuration

Syntaxe

```
type    contrôle    module.so    options_du_modules...
```

Les contrôles, version simple

- required : retournera l'erreur mais le processus continue
- requisite : retourne l'erreur immédiatement
- sufficient : retourne le succès immédiatement
- optional : le module n'a d'importance que s'il est seul
- required : retournera l'erreur mais le processus continue
- include : inclus la configuration donné en argument
- substack : inclus la configuration mais continue le processus

Configuration

Syntaxe

```
type    contrôle    module.so    options_du_modules...
```

Les contrôles, version simple

- required : retournera l'erreur mais le processus continue
- requisite : retourne l'erreur immédiatement
- sufficient : retourne le succès immédiatement
- optional : le module n'a d'importance que s'il est seul
- required : retournera l'erreur mais le processus continue
- include : inclus la configuration donné en argument
- substack : inclus la configuration mais continue le processus

Configuration

Syntaxe

```
type    contrôle    module.so    options_du_modules...
```

Les contrôles, version simple

- required : retournera l'erreur mais le processus continue
- requisite : retourne l'erreur immédiatement
- sufficient : retourne le succès immédiatement
- optional : le module n'a d'importance que s'il est seul
- required : retournera l'erreur mais le processus continue
- include : inclus la configuration donné en argument
- substack : inclus la configuration mais continue le processus

Configuration

Syntaxe

```
type    contrôle    module.so    options_du_modules...
```

Les contrôles, version simple

- required : retournera l'erreur mais le processus continue
- requisite : retourne l'erreur immédiatement
- sufficient : retourne le succès immédiatement
- optional : le module n'a d'importance que s'il est seul
- required : retournera l'erreur mais le processus continue
- include : inclus la configuration donné en argument
- substack : inclus la configuration mais continue le processus

Nouvelle syntaxe

Syntaxe

Décisions plus fines en fonctions des réponses des modules
Contrôle entre [] composé d'une série
`code_de_retour=valeur` :

```
type    [default=ignore]  module.so   options
```

Quelques code de retour

- success : passage avec succès
- new_authtok_reqd : changement de mot de passe requis
- user_unknown : utilisateur inconnu
- default : valeur à adopter par défaut
- ...

Nouvelle syntaxe

Syntaxe

Décisions plus fines en fonctions des réponses des modules
Contrôle entre `[]` composé d'une série
`code_de_retour=valeur` :

```
type    [default=ignore]  module.so   options
```

Quelques code de retour

- success : passage avec succès
- new_authtok_reqd : changement de mot de passe requis
- user_unknown : utilisateur inconnu
- default : valeur à adopter par défaut
- ...

Nouvelle syntaxe

Réponses à apporter :

- ignore : ignorer cette réponse
- ok : à considérer comme un succès (success)
- done : succès, arrêt de la pile
- bad : à considérer comme une erreur
- die : erreur, arrêt de la pile

Exemple

```
auth [user_unknown=ignore success=ok ignore=ignore default=bad] \
    pam_securetty.so
```

(`pam_securetty` n'autorise root que sur certain tty).

Nouvelle syntaxe

Réponses à apporter :

- ignore : ignorer cette réponse
- ok : à considérer comme un succès (success)
- done : succès, arrêt de la pile
- bad : à considérer comme une erreur
- die : erreur, arrêt de la pile

Exemple

```
auth [user_unknown=ignore success=ok ignore=ignore default=bad] \
    pam_securetty.so
```

(`pam_securetty` n'autorise root que sur certain tty).

Nouvelle syntaxe

Réponses à apporter :

- ignore : ignorer cette réponse
- ok : à considérer comme un succès (success)
- done : succès, arrêt de la pile
- bad : à considérer comme une erreur
- die : erreur, arrêt de la pile

Exemple

```
auth [user_unknown=ignore success=ok ignore=ignore default=bad] \
    pam_securetty.so
```

(`pam_securetty` n'autorise root que sur certain tty).

Nouvelle syntaxe

Réponses à apporter :

- ignore : ignorer cette réponse
- ok : à considérer comme un succès (success)
- done : succès, arrêt de la pile
- bad : à considérer comme une erreur
- die : erreur, arrêt de la pile

Exemple

```
auth [user_unknown=ignore success=ok ignore=ignore default=bad] \
    pam_securetty.so
```

(`pam_securetty` n'autorise root que sur certain tty).

Nouvelle syntaxe

Réponses à apporter :

- ignore : ignorer cette réponse
- ok : à considérer comme un succès (success)
- done : succès, arrêt de la pile
- bad : à considérer comme une erreur
- die : erreur, arrêt de la pile

Exemple

```
auth [user_unknown=ignore success=ok ignore=ignore default=bad] \
    pam_securetty.so
```

(`pam_securetty` n'autorise root que sur certain tty).

La récupération du mot de passe

La demande de mot de passe :

Chaque module doit demander le mot si nécessaire, cependant :

- la demande se fait sur la console
- une module peut récupérer le mot de passe demander par son prédécésseur (cf l'option `use_first_pass`)
- l'authentification non interactive est possible (bien sûr)

Discussion technique de NSS et PAM

Des librairies ?

Conséquences sur la sécurité :

- Le programme intègre du code externe :
  - intégration des bogues des modules
  - bogue dans l'utilisation de nss/pam
- utilisation des privilèges de l'application

Conséquences sur la performance :

Pour chaque nouveau programme :

- duplication des codes 32/64 bits
- rechargement des configurations
- pas de cache des données
- déduplication des données

Des librairies ?

Conséquences sur la sécurité :

- Le programme intègre du code externe :
  - intégration des bogues des modules
  - bogue dans l'utilisation de nss/pam
- utilisation des privilèges de l'application

Conséquences sur la performance :

Pour chaque nouveau programme :

- duplication des codes 32/64 bits
- rechargement des configurations
- pas de cache des données
- déduplication des données

Des librairies ?

Conséquences sur la sécurité :

- Le programme intègre du code externe :
  - intégration des bogues des modules
  - bogue dans l'utilisation de nss/pam
- utilisation des privilèges de l'application

Conséquences sur la performance :

Pour chaque nouveau programme :

- duplication des codes 32/64 bits
- rechargement des configurations
- pas de cache des données
- déduplication des données

Des librairies ?

Conséquences sur la sécurité :

- Le programme intègre du code externe :
  - intégration des bogues des modules
  - bogue dans l'utilisation de nss/pam
- utilisation des privilèges de l'application

Conséquences sur la performance :

Pour chaque nouveau programme :

- duplication des codes 32/64 bits
- rechargement des configurations
- pas de cache des données
- déduplication des données

Des librairies ?

Conséquences sur la sécurité :

- Le programme intègre du code externe :
  - intégration des bogues des modules
  - bogue dans l'utilisation de nss/pam
- utilisation des privilèges de l'application

Conséquences sur la performance :

Pour chaque nouveau programme :

- duplication des codes 32/64 bits
- rechargement des configurations
- pas de cache des données
- déduplication des données

Des librairies ?

Conséquences sur la sécurité :

- Le programme intègre du code externe :
  - intégration des bogues des modules
  - bogue dans l'utilisation de nss/pam
- utilisation des privilèges de l'application

Conséquences sur la performance :

Pour chaque nouveau programme :

- duplication des codes 32/64 bits
- rechargement des configurations
- pas de cache des données
- déduplication des données

NSCD

Name Service Cache Daemon

Démon de mise en cache des données NSS (passwd, group, host).

NSCD en pratique

- service à lancer
- utilisé par tousles programmes automatiquement
- configuration dans `/etc/nscd.conf`
- peut causer des problèmes (cache)

NSCD

Name Service Cache Daemon

Démon de mise en cache des données NSS (passwd, group, host).

NSCD en pratique

- service à lancer
- utilisé par tousles programmes automatiquement
- configuration dans `/etc/nscd.conf`
- peut causer des problèmes (cache)

NSCD

Name Service Cache Daemon

Démon de mise en cache des données NSS (passwd, group, host).

NSCD en pratique

- service à lancer
- utilisé par tousles programmes automatiquement
- configuration dans `/etc/nscd.conf`
- peut causer des problèmes (cache)

NSCD

Name Service Cache Daemon

Démon de mise en cache des données NSS (passwd, group, host).

NSCD en pratique

- service à lancer
- utilisé par tousles programmes automatiquement
- configuration dans `/etc/nscd.conf`
- peut causer des problèmes (cache)

NSCD

Name Service Cache Daemon

Démon de mise en cache des données NSS (passwd, group, host).

NSCD en pratique

- service à lancer
- utilisé par tousles programmes automatiquement
- configuration dans `/etc/nscd.conf`
- peut causer des problèmes (cache)

Progression

Avant Propos

Provenance des modules

Provenance diverses et variées :

- modules natifs :
  - NSS (glibc) : `files`, `nis`
  - PAM : `pam_unix`, `pam_cracklib`
- projets tiers

Attention :

1. modules NSS et PAM pas forcément fournis ensemble
2. pérénité du projet à vérifier
3. sécurité des modules à vérifier (code + à l'usage)

Provenance des modules

Provenance diverses et variées :

- modules natifs :
  - NSS (glibc) : `files`, `nis`
  - PAM : `pam_unix`, `pam_cracklib`
- projets tiers

Attention :

1. modules NSS et PAM pas forcément fournis ensemble
2. pérénité du projet à vérifier
3. sécurité des modules à vérifier (code + à l'usage)

Provenance des modules

Provenance diverses et variées :

- modules natifs :
  - NSS (glibc) : `files`, `nis`
  - PAM : `pam_unix`, `pam_cracklib`
- projets tiers

Attention :

1. modules NSS et PAM pas forcément fournis ensemble
2. pérénité du projet à vérifier
3. sécurité des modules à vérifier (code + à l'usage)

Provenance des modules

Provenance diverses et variées :

- modules natifs :
  - NSS (glibc) : `files`, `nis`
  - PAM : `pam_unix`, `pam_cracklib`
- projets tiers

Attention :

1. modules NSS et PAM pas forcément fournis ensemble
2. pérénité du projet à vérifier
3. sécurité des modules à vérifier (code + à l'usage)

Provenance des modules

Provenance diverses et variées :

- modules natifs :
  - NSS (glibc) : `files`, `nis`
  - PAM : `pam_unix`, `pam_cracklib`
- projets tiers

Attention :

1. modules NSS et PAM pas forcément fournis ensemble
2. pérénité du projet à vérifier
3. sécurité des modules à vérifier (code + à l'usage)

pam_stack.so

Rôle

Importe une configuration d'un autre fichier

```
auth     required pam_stack.so service=foo
session  optional pam_stack.so service=foo
password optional pam_stack.so service=foo
account  optional pam_stack.so service=foo
```

N'existe plus

Remplacé par la directive `include`

pam_stack.so

Rôle

Importe une configuration d'un autre fichier

```
auth     required pam_stack.so service=foo
session  optional pam_stack.so service=foo
password optional pam_stack.so service=foo
account  optional pam_stack.so service=foo
```

N'existe plus

Remplacé par la directive `include`

pam_unix.so

Rôle

Valide l'authentification depuis les fichiers `passwd`, `group` et `shadow`

Rémplacé

Il faut utiliser `pam_tcb` :

- propose de séparer les mot de passe dans plusieurs fichiers
- `fork()` pendant l'authentification

Bases de comptes externes

NIS

Network Information Service

- module NSS de la glibc
- a besoin d'`ypbind`
- `pam_tcb` pour l'authentification

Peu d'avenir

- remplacé par `ldap`
- ce n'est qu'une version réseau des fichiers unix
- sécurité des mots de passe assuré seulement par leur chiffrement

NIS

Network Information Service

- module NSS de la glibc
- a besoin d'`ypbind`
- `pam_tcb` pour l'authentification

Peu d'avenir

- remplacé par `ldap`
- ce n'est qu'une version réseau des fichiers unix
- sécurité des mots de passe assuré seulement par leur chiffrement

LDAP*, Kerberos

LDAP (NSS et PAM)

- s'appuient sur la `libldap`
- configuration plus ou moins conséquente selon les schémas
- authentification par le serveur ldap (bind)
- mot de passe non récupérable

Kerberos (PAM)

- valide le couple login/mot de passe au près du serveur
- récupère un ticket au cours de l'authentification

LDAP*, Kerberos

LDAP (NSS et PAM)

- s'appuient sur la `libldap`
- configuration plus ou moins conséquente selon les schémas
- authentification par le serveur ldap (bind)
- mot de passe non récupérable

Kerberos (PAM)

- valide le couple login/mot de passe au près du serveur
- récupère un ticket au cours de l'authentification

MySQL, PgSQL, *SQL

Stockage en base SQL

- sources très diverses et très variées
- schéma SQL requis :
  - plus ou moins figé selon les modules
  - parfois incohérents (entre NSS et PAM)

Apport de fonctionnalités

pam_rootok.so

Description

- Vérifie que l'utilisateur est Root
- utile pour `sudo`, `su`, ...

pam_postgresok.so

Description

- module de RedHat
- vérifie que l'utilisateur est PostgreSQL
- que l'UID est 26 (postgresql sous RedHat)

pam_cracklib.so

Description

- passe le mot de passe à cracklib
  - vérification de la force du mot de passe
  - vérification sur dictionnaire
- très configurable

pam_mkhomedir.so

Description

- création de la home à la connection
- configurable :
  - choix des droits
  - choix du modèle
- attention au `root_quash` sur NFS
- ne fonctionne dans le cas de reception d'un mail

Progression

Programmation

Généralité

- NSS : de base dans les langages
- PAM via module additionnel

Dans les langages :

- PERL : module sur le CPAN
- Python : `http://atlee.ca/software/pam/` (encore maintenu ?)
- Ruby : plusieurs modules disponibles sur le net
- PHP : `http://pecl.php.net/package/PAM` (encore maintenu ?)

Pam est optionnel

- à la compilation
- par configuration (`UsePAM yes`)

Clef + PAM :

- le type `auth` n'est pas appelé
- les types `account` et `session` le sont

OpenSSH

Pam est optionnel

- à la compilation
- par configuration (`UsePAM yes`)

Clef + PAM :

- le type `auth` n'est pas appelé
- les types `account` et `session` le sont

Progression

Les documentations

NSS

`man`

- `nss`
- `nsswitch.conf`

PAM

`man`

- `pam`
- `pam.d`
- `pam_MODULE`

Les autres modules

Dépends des projets...

Les documentations

NSS

`man`

- `nss`
- `nsswitch.conf`

PAM

`man`

- `pam`
- `pam.d`
- `pam_MODULE`

Les autres modules

Dépends des projets...

Les documentations

NSS

`man`

- `nss`
- `nsswitch.conf`

PAM

`man`

- `pam`
- `pam.d`
- `pam_MODULE`

Les autres modules

Dépends des projets...

Ce qu'il faut retenir :

- NSS et PAM ne centralisent pas la gestion
  - de la validation des mots de passe
  - des autorisations et datent de validité
- testez vos configurations en situation hostile (sans réseau) !
- attention aux incompatibilités applications-modules pam

Merci

Question ?

(c) Wikipédia

Licence