Sécurité des Systèmes d'Information
La compartimentation système

Martin Peres

Doctorant de Francine Krief au LaBRI

October 9, 2013

Sommaire

Principe de base du contrôle d'accès

Un accès est caractérisé par:

- un sujet
- un objet
- une opération (lire, écrire, exécuter, etc...)

Les différents modèles

2 modèles très connus:

- MLS: MultiLevel Security
- RBAC: Role-Based Access Control

MLS: MultiLevel Security

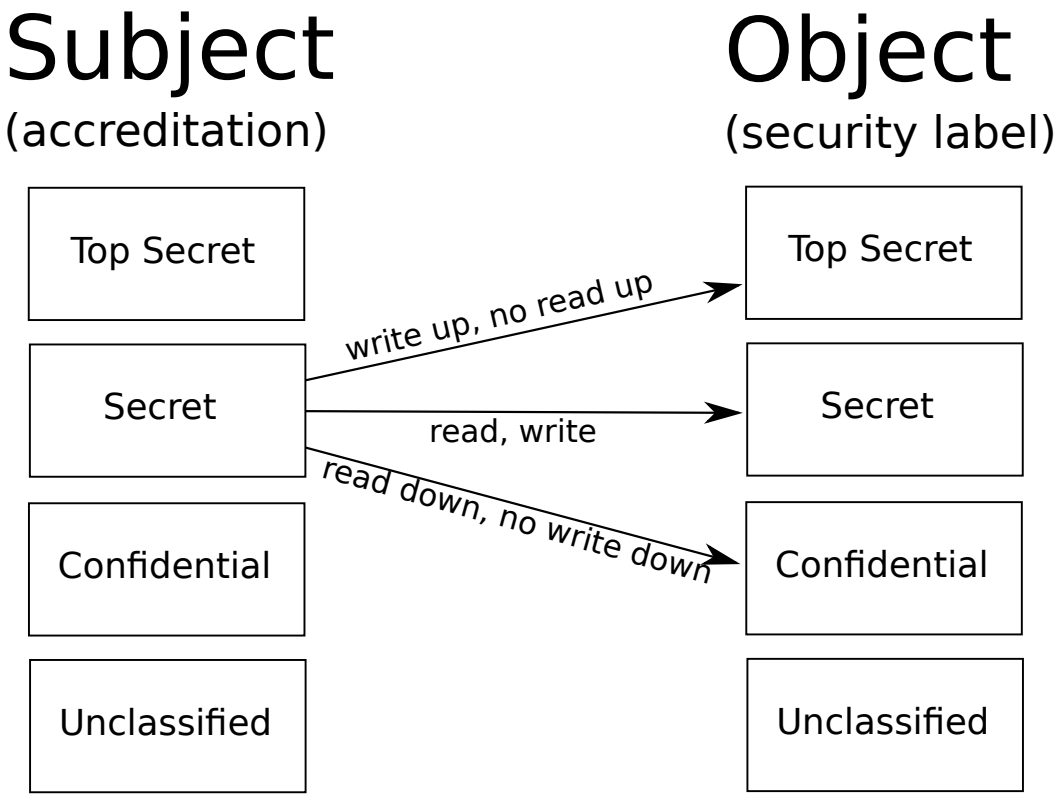
La plupart des systèmes "sécurisés" utilisent un modèle MLS.

- les objets sont taggés avec un label correspondant à un niveau d'intégrité
- les sujets ont un niveau d'accréditation permettant d'accéder aux objets
- les politiques Biba ou Bell Lapadula peuvent ensuite être appliquées

https://www.centos.org/docs/5/html/Deployment_Guide-en-

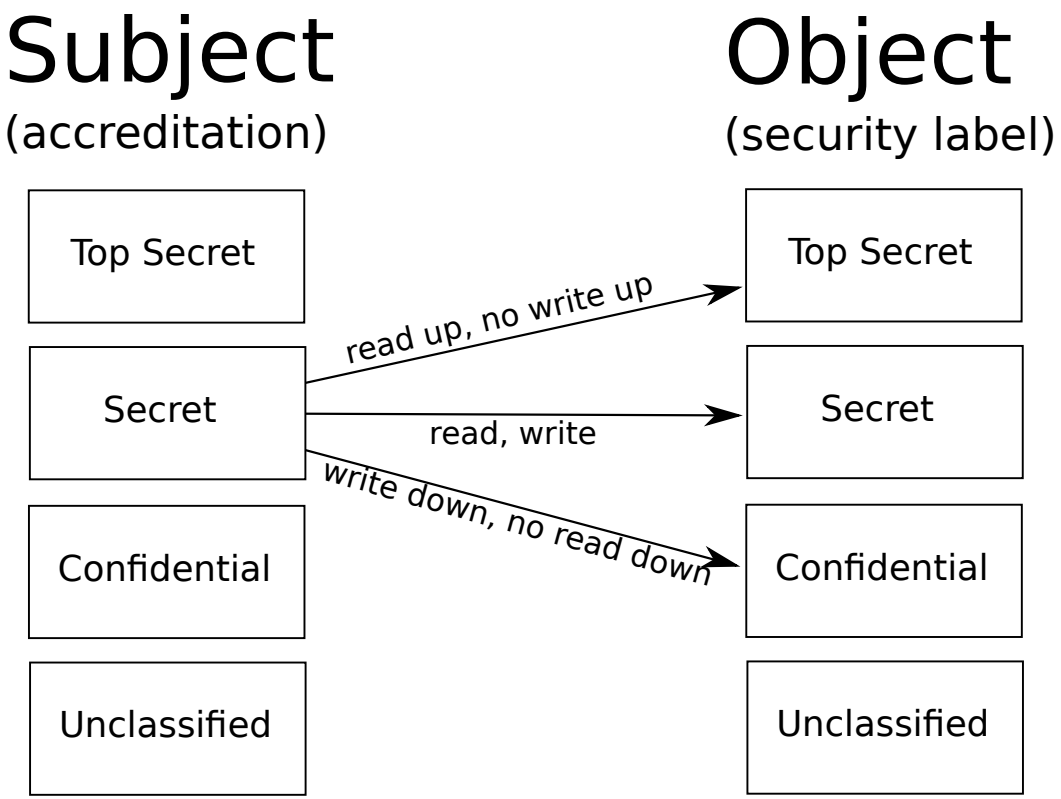
MLS: Modèle Bell Lapadula

- modèle le plus connu
- permet de garantir la confidentialité des données
- ne protège pas l'intégrité des données



MLS: Modèle Biba

- permet de garantir l'intégrité des données
- n'apporte aucune confidentialités aux données

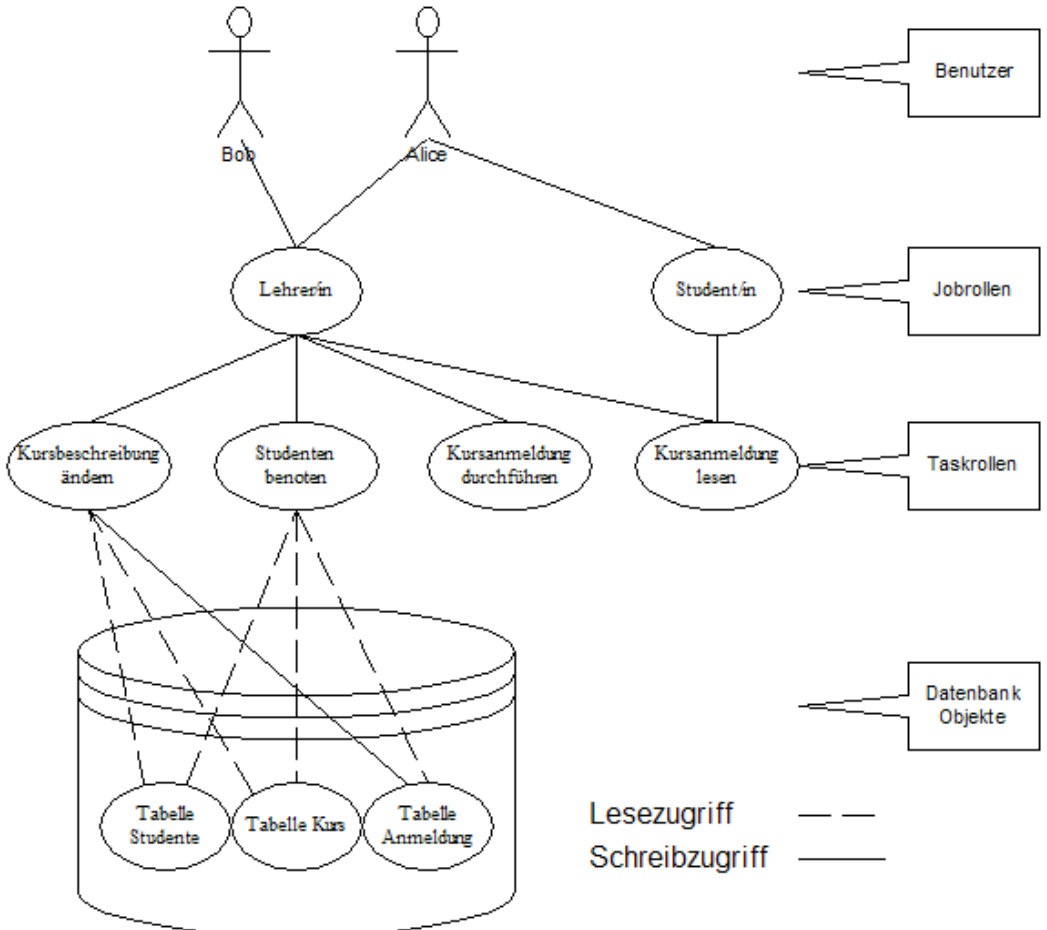


MLS: Inconvénients

- avec le temps, tous les objets ont tendance à être labellés avec l'IL le plus fort
- nécessite une procédure de dé-classification

RBAC: Role-Based Access Control

- Autorisations données à des groupes
- Simplification des politiques
- Utilisé dans la plupart des grandes entreprises



Sommaire

Philosophie UNIX

- tout est fichier
- système multi-utilisateur
- système multi-tache
- Un OS est dit UNIX si il respecte l'interface POSIX

Liste des OS POSIX

- (Free|Open|Dragonfly|Net) BSD
- MacOS
- Linux
- Windows (de façon très limitée)

Philosophie d'utilisation du compte administrateur

- root: compte administrateur
- tous les utilisateurs (admin compris) doivent utiliser un compte limité par défaut
- et devront récupérer les droits d'admin seulement quand c'est nécessaire et pour la durée de l'opération

DAC

Les droits Unix sont dit de type DAC:

- Discretionary Access Control
- À la charge de l'utilisateur de les spécifier

DAC: les droits UNIX

La commande "ls -l" permet de lister les informations importantes sur un fichier:

- type de fichier (d=directory, c=char, b=block, l=symlink, -=fichier régulier)
- droits d'accès (propriétaire: rwx, groupe: rwx, autres: rwx)
- nombre de sous dossier/fichiers dans le dossier. 1 si c'est un fichier, 2 mini si c'est un dossier (. et ..)
- user propriétaire / groupe propriétaire
- taille du fichier
- dernière modification
- nom du fichier

Représentation des droits UNIX

Les droits sont représentés par 3 chiffres en octal (3 bits). Chaque chiffre est un bitfield où le bit x veut dire:

- 4 (bit 2): read
- 2 (bit 1): write
- 1 (bit 0): execute

Signification du x

La signification de execute dépend du type de fichier:

- dossier: Les attributs des sous dossier peuvent être lu
- fichier régulier: Le fichier peut être exécuté (script, binaire).

Gestion des droits

Les personnes aptes à modifier les droits d'accès à un fichier sont:

- root: l'administrateur du système (uid 0)
- le propriétaire du fichier
- Ex: chmod 755 fichier

Changer de propriétaire

Root a le droit de changer le propriétaire d'un fichier en faisant:

- chown login:group fichier

Gestion automatique des droits UNIX: umask

Le umask:

- défini les droits que l'on veut enlever par défaut
- est visible et changeable par la commande umask
- est généralement égal à 022 (enlever les droits d'écriture au groupe + autres)
- un fichier régulier n'obtient pas par défaut le bit x pour des raisons de sécurité.

Exécution d'un programme

Un programme s'exécute:

- avec les droits de l'utilisateur qui l'a exécuté
- à moins que le programme change ses droits avec setuid() (root only)

SETUID: Exécuter avec les droits du propriétaire du fichier

Si le bit setuid est levé dans les droits d'un programme, le programme s'exécute avec les droits du propriétaire.

```
$ ls -l /bin/ping
-rwsr-xr-x 1 root root 35752  3 nov.  21:07 /bin/ping
```

ACL: Access Control List

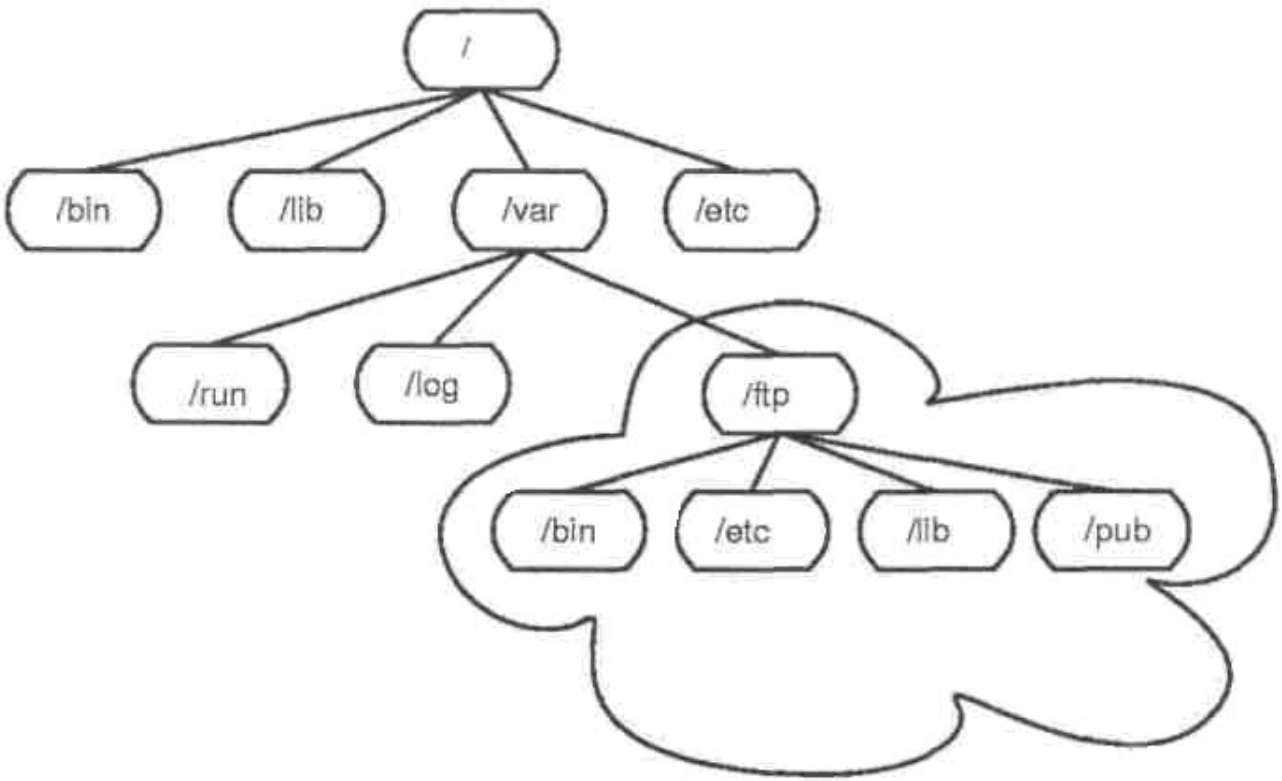
- augmente la flexibilité des droits unix
- permet de spécifier une liste blanche donner droit d'accès r, w ou x
- listés avec getfacl
- modifiés avec setfacl
- voir les pages man respectives

Exemples

- setfacl -m u:lisa:r file (donne le droit à lisa de lire le fichier file)
- setfacl -m g:élèves:rx file (donne le droit d'exécuter et lire le fichier file)

Chroot: compartimenter le système de fichier

- défini une nouvelle racine (CHange ROOT)
- puis exécute un programme
- le programme ne voit pas qu'il est dans un chroot
- chroot new_root [command] (/bin/sh par défaut)



S'échapper d'un chroot

- chroot n'a pas été conçu pour la sécurité
- il est possible de sortir d'un chroot
- GRSec, un patchset pour Linux, permet de rendre plus sécurisé l'utilisation d'un chroot
- Voir http://kerneltrap.org/Linux/Abusing_chroot
- Solution partielle: la virtualisation (vservers, Xen, kvm ou les jails d'OpenBSD)
- Vrai solution: MAC (SELinux, Tomoyo, etc...)

Capabilities: Limiter l'utilisation de root

Les capabilities:

- fragmentent les droits de root
- peuvent être hérités (et altérés) lors du spawn d'un nouveau process
- peuvent être droppés par une application sans pouvoir ensuite être regagnés
- peuvent être automatiquement ajustés lors de l'exécution d'un binaire (pour ce pid et ses futur fils)

Altérations sur les capabilities

Un binaire peut, lors de son exécution, modifier les capabilities héritées en utilisant setcap:

- ne pas hériter d'une capability: -i
- forcer l'acquisition d'une capability: +ep
- Voir "man cap_from_text"

Quelques capabilites

- CAP_CHOWN: Autorise le changement de propriétaire d'un fichier
- CAP_DAC_OVERRIDE: Contourne les droits DAC (utile quand on est admin)
- CAP_LINUX_IMMUTABLE: Autorise un fichier à être marqué comme immutable
- CAP_NET_BIND_SERVICE: Autorise un processus à binder un port privilégié (< 1024)
- CAP_NET_RAW: Autorise un processus à créer une socket RAW
- CAP_SYS_ADMIN: Autorise à utiliser les appels système quotactl, mount, swapon, sethostname, etc...
- Voir "man capabilities"

Exemple d'utilisation

Ping est marqué setuid car il a besoin de pouvoir forger des paquets (socket RAW)

- # chmod u-s /bin/ping (ping: icmp open socket: Operation not permitted)
- # setcap 'cap_net_raw=+ep' /bin/ping (ping works again!)
- getcap liste les altérations des capabilities (/bin/ping = cap_net_raw+ep)

Au delà des droits UNIX

Les systèmes de fichier proposent plus d'options que les simples droits UNIX. Ceux-ci sont:

- listés en utilisant lsattr
- modifiés en utilisant chattr (peut nécessiter CAP_LINUX_IMMUTABLE)

Exemple d'attributs

- a: append only
- c: compressed
- i: immutable
- s: secure deletion
- u: undeletable

Exemple d'utilisation

```
$ lsattr ./h2g2.jpg
-------------- ./h2g2.jpg
# chattr +i ./h2g2.jpg
$ rm ./h2g2.jpg
rm : supprimer fichier (protégé en écriture) ./h2g2.jpg ?
$ lsattr ./h2g2.jpg
----i---------- ./h2g2.jpg
$ chattr -i ./h2g2.jpg
chattr: Opération non permise lors de l'initialisation
des drapeaux sur ./h2g2.jpg
# chattr -i ./h2g2.jpg
$ lsattr ./h2g2.jpg
-------------- ./h2g2.jpg
```

Sommaire

MAC: Contrôle d'accès mandataire

Contrairement au DAC, le MAC est défini par l'administrateur pour garantir des propriétés de sécurité telles que:

- pas de flux d'informations entre 2 utilisateurs
- ne pas exécuter des fichiers hors des dossiers /bin et /usr/bin à moins d'être administrateur
- etc...

Domaines

Chaque application est confinée dans un domaine:

- identifié par un nom (SELinux) ou l'arbre d'appel (Tomoyo)
- a une liste d'opérations permises (liste blanche)
- facilite la gestion des droits (tout bloqué par défaut)

Windows: L'User Account Control (UAC)

- Constat: La plupart des utilisateurs windows sont administrateurs
- Cela ouvre la voie aux malwares qui peuvent infecter tout le système
- Problème: Les programmes nécessite la plupart du temps les droits administrateur
- Solution? Lancer les programmes en mode limité et demander à l'utilisateur de valider quand un programme passe en root
- Inconvénients: L'utilisateur ne comprend pas et valide la plupart du temps.

Windows et l'UAC: Vrai solution

- Installer les applications dans le répertoire utilisateur (plus besoin des droits admin pour l'install)
- Forcer les utilisateurs à avoir un compte limité par défaut (casse la retro-compatibilité par défaut)
- Avoir une demande pour passer en administrateur en moyenne par jour

Pub Apple tournant l'UAC en dérision

Windows: Mandatory Integrity Control (MIC)

- Problème: Comment faire varier les droits d'une application?
- Solution: Changer son niveau d'intégrité (MLS)
- 4 Niveaux: Low, Medium, High et System

Windows MIC: Intégrité des ressources

- Fichier: Niveau d'intégrité donné en fonction du chemin d'accès
- Registre: Niveau d'intégrité donné en fonction du chemin d'accès
- Processus: Se lance par défaut en Medium et peut transiter en Low

MIC: Utilisation

Peu de programmes baissent eux-même leurs privilèges, voici les plus connus:

- Internet Explorer 7+
- Adobe Reader 10+
- Google Chrome
- Firefox? Non, ce n'est pas normal

Remarques

- Tous les programmes ont le droit d'accéder à tous les fichiers utilisateur
- Utilise MLS au lieu d'une protection par domaine
- Au final, ça ressemble un peu aux capabilities (en moins souple) + MLS

Linux Security Module (LSM)

- Situé dans le noyau Linux
- "Hook" des appels de tous les appels systèmes et autres changement d'états interne
- Envoi de ces informations aux modules de sécurité qui doivent accepter ou non cet appel/changement
- Facilite l'écriture de modules de sécurité
- Unique sur les OS monolithiques modulaires

Exemple d'appels hookés

- .inode_create : création d'un inode
- .inode_getxattr : récupération des attributs étendus
- .dentry_open : ouvrir un dossier
- .socket_listen : listen sur une socket
- Plus de 100 appels hookés

Security Enhanced Linux (SELinux)

- Développé par la NSA, utilise LSM
- Travaille avec des domaines (Domain Type Enforcement), RBAC et MLS
- Basé sur des labels de sécurité plutôt que les chemins
- Extrêmement configurable et sécurité statique presque parfaite
- Dur à administrer (demande quelques mois d'expérience)

Utilisé par ?

- Fedora : Distribution Linux gratuite maintenue par Red Hat
- Red Hat Entreprise Linux: Distribution vendue avec du support utilisateur
- Gentoo Hardened: Distribution Linux communautaire basée sur Gentoo

Labelling du Système de Fichier

- Règles contenues dans les fichiers .fc
- Basé sur les expressions régulières

Exemple de fichier fc

- /usr/bin/honeyd system_u:object_r:honeyd_exec_t
- /usr/share/honeyd system_u:object_r:honeyd_conf_t

Autorisations

- Règles contenues dans les fichiers .te
- Défini les règles de transitions de domaine
- Défini les domaines utilisés par la politique de sécurité
- Défini les interactions autorisées

Exemple de fichier te

- daemon_domain(honeyd)
- domain_auto_trans(sysadm_t, honeyd_exec_t, honeyd_t)
- allow honeyd_t honeyd_conf_t:dir { search };
- allow honeyd_t honeyd_conf_t:file { getattr read };
- allow honeyd_t random_device_t:chr_file { read };

Tomoyo

- Utilise LSM
- Travaille avec des domaines
- Basé sur les chemins d'accès
- Propose un mode apprentissage activable par domaine
- Facile à administrer (GUI), permet d'avoir de bons résultats rapidement

Utilisé par ?

- ArchLinux: Distribution communautaire très facilement modifiable

Sommaire

MAC vs DAC

- Le DAC permet de spécifier des droits d'accès assez simplement
- Le DAC n'apporte que peu de garantie de sécurité
- Le DAC protège peu contre les erreurs d'inattention
- Le MAC apporte une plus grande expressivité
- Mais le MAC demande beaucoup de temps de configuration

Implémentations

Un MAC est disponible sur la plupart des OS:

- Windows: MIC, UAC, Anti-virus
- Linux: SELinux, Tomoyo, Smack, AppArmor, ...
- OpenBSD: Jails
- MAC OS: SEDarwin?