

Polytech Nancy  
*laurent.vigneron@univ-lorraine.fr*

# Sécurité informatique

## Chapitre 2 : Règles d'hygiène informatique

2021/2022



Support de cours basé sur un document pédagogique mis à disposition par l'ANSSI sous licence Creative Commons Attribution 3.0 France.

# Plan du chapitre

- 1. Connaître le Système d'Information**
- 2. Maîtriser le réseau**
- 3. Sécuriser les terminaux**
- 4. Gérer les utilisateurs**
- 5. Sécuriser physiquement**
- 6. Contrôler la sécurité du S.I.**
- 7. Focus : contrôle d'accès**



## *Sécurité informatique*

# 1. Connaître le Système d'Information

- a) Identifier les composants du S.I.
- b) Inventorier les biens
- c) Types de réseaux
- d) Interconnexion

# 1. Connaître le Système d'Information

## *a. Identifier les composants du S.I.*

Au-delà de la connaissance des composants du S.I., l'inventaire permettra par la suite de mieux déterminer les menaces et les mesures de protection applicables.

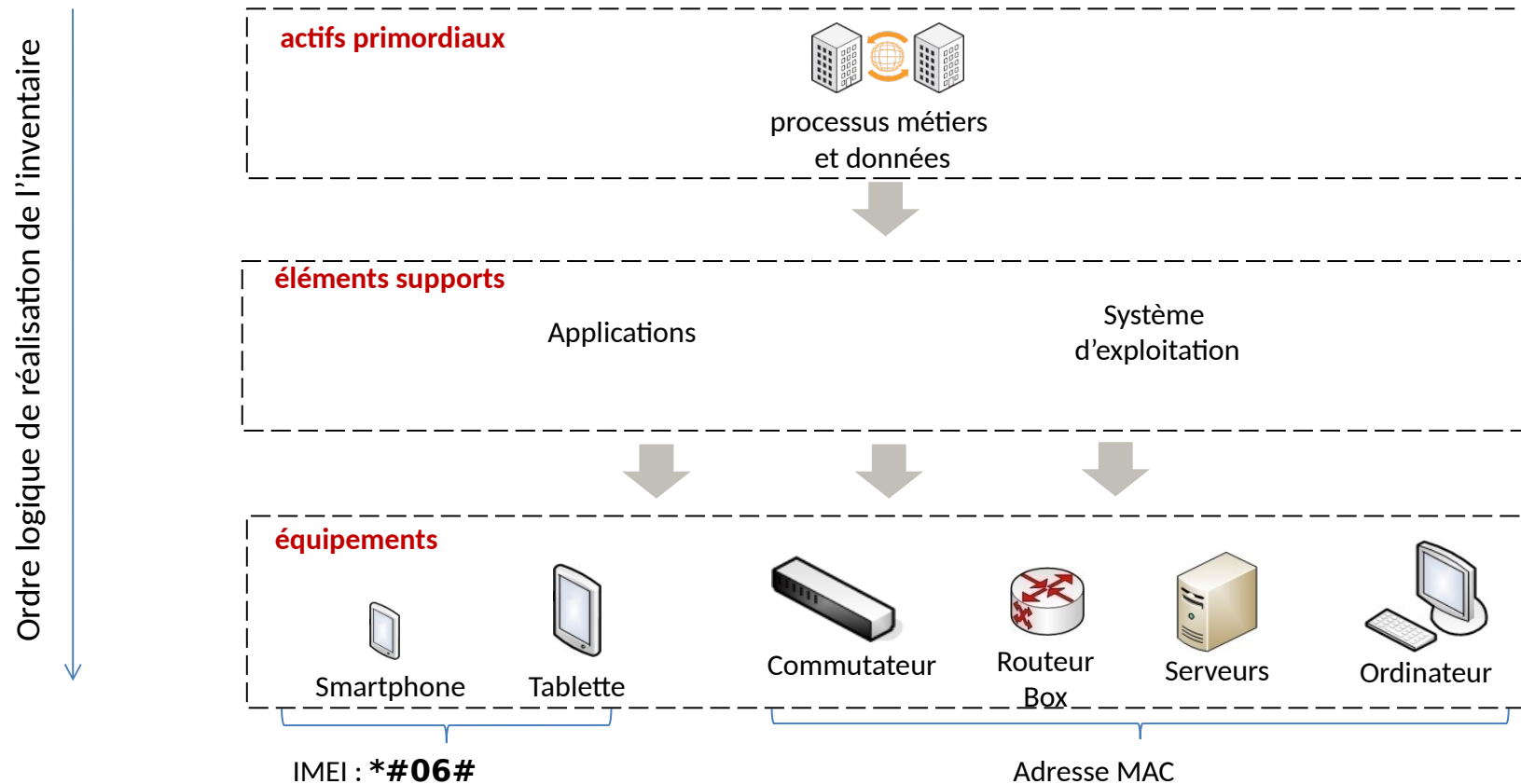
Tout projet sécurité doit donc forcément intégrer un inventaire des biens.

L'inventaire des biens doit suivre une **méthodologie logique** afin d'être exhaustif, en commençant par l'inventaire des métiers.

# 1. Connaître le Système d'Information

## a. Identifier les composants du S.I.

Différents éléments composent le SI



**Comprendre son S.I. passe par l'identification de ses composants.**

# 1. Connaître le Système d'Information

## *b. Inventorier les biens*

### Identifier

- Les **données sensibles** :
  - mots de passe, cartes de crédit, documents personnels, etc.
  - plan marketing, fichier client, brevets, contrats, etc.
- Les **applications** avec leur version : bureautique, navigateur web, etc.
- Les **systèmes d'exploitation** : Android, iOS, Windows, Linux, MacOS, etc.
- **Équipements** : ordinateur, tablette, téléphone, serveur, box, routeur, etc.

### Inventorier

- Outil d'identification des ordinateurs en réseau
  - Exemple : ServiceNow, MicroFocus
- Outil d'identification des logiciels installés sur un ordinateur/téléphone ainsi que des versions
  - Exemple : Everest.

# 1. Connaître le Système d'Information

## *c. Types de réseaux*

- BAN (Body Area Network) : réseau composé de télé transmetteur utilisé dans le domaine de la santé.
- PAN (Personal Area Network) : réseau centré autour d'une personne interconnectant ordinateur, téléphone, tablette, voiture... (moins de 10m).
  - WPAN (Wireless PAN) : réseau PAN sans fil utilisant des technologies telles que : IrDA, ZigBee, Bluetooth, Wireless USB.
- [W]LAN ([Wireless] Local Area Network) : Réseau local [sans fil] interconnectant plusieurs périphériques et permettant l'échange d'informations entre plusieurs individus.
- MAN (Metropolitan Area Network) : réseau plus large qu'un LAN et étendu par exemple sur une ville.
- CAN (Campus Area Network) : réseau s'étendant sur plusieurs LAN, et de la taille d'une université.
- WAN (Wide Area Network) : réseau d'une étendue nationale ou internationale. Exemple : Internet.

# 1. Connaître le Système d'Information

## *d. Interconnexion*

Connaître et maîtriser les points d'interconnexion

- Accès Internet via :
  - Box Internet (ADSL, Fibre, ...).
  - téléphone/carte 3G/4G, etc.
- Interconnexion avec d'autres réseaux (universités, partenaires, prestataires, etc.)
  - Liaison dédiée : E1/T1 carrier, fibre optique noire.
  - Réseau privé virtuel (VPN) sur un WAN appartenant à un opérateur ou sur Internet.
  - Liaison satellite.





## *Sécurité informatique*

# 2. Maîtriser le réseau

- a) Sécuriser le réseau interne
- b) BYOD (Bring Your Own Device)
- c) Contrôler les échanges internes
- d) Protéger le réseau interne d'Internet
- e) Accès distant
- f) Sécuriser l'administration
- g) Wifi

## 2. Maîtriser le réseau

### *a. Sécuriser le réseau interne*

Créer des zones dans le réseau interne

- Zones distinctes pour les serveurs, postes de travail, visiteurs.
- Assurer la confiance par l'authentification mutuelle des composants :
  - chaque composant s'authentifie avant le début de l'échange ;
  - permet d'éviter l'usurpation d'identité.
- Assurer le cloisonnement au moyen de : VLAN, VRF, sous-réseaux et ne pas oublier d'implanter un mécanisme de filtrage !

Restreindre les accès aux réseaux internes

- IEEE 802.1X permet de contrôler l'accès réseau et de s'assurer que l'autorisation n'est accordé qu'après authentification de l'utilisateur.
- Recourir à l'authentification avant d'autoriser l'accès au réseau :
  - l'authentification peut se faire par l'usage d'un certificat ou d'une carte à puce ;
  - l'authentification est centralisée sur un serveur qui donne les accès en fonction de l'identité de l'utilisateur (Exemple : serveur Radius).

## 2. Maîtriser le réseau

### *b. BYOD (Bring Your Own Device)*

- Le réseau permet de partager des informations, mais aussi de propager les infections de codes malveillants.
- Les terminaux personnels n'ont pas le même niveau de sécurité que les terminaux de l'entreprise / université :
  - Sur un terminal personnel, un utilisateur installe les logiciels de son choix, avec la configuration de son choix. L'antivirus n'est pas forcément à jour.
  - Sur un terminal professionnel, les logiciels sont installés de manière centralisée, et les sources vérifiées.
- Les terminaux personnels sont connus pour être une source de fuite de données sensibles pour l'entreprise (de façon volontaire ou par erreur).

**Le S.I. est un tout, un maillon faible affaiblit tout l'ensemble.**

## 2. Maîtriser le réseau

### *c. Contrôler les échanges internes*

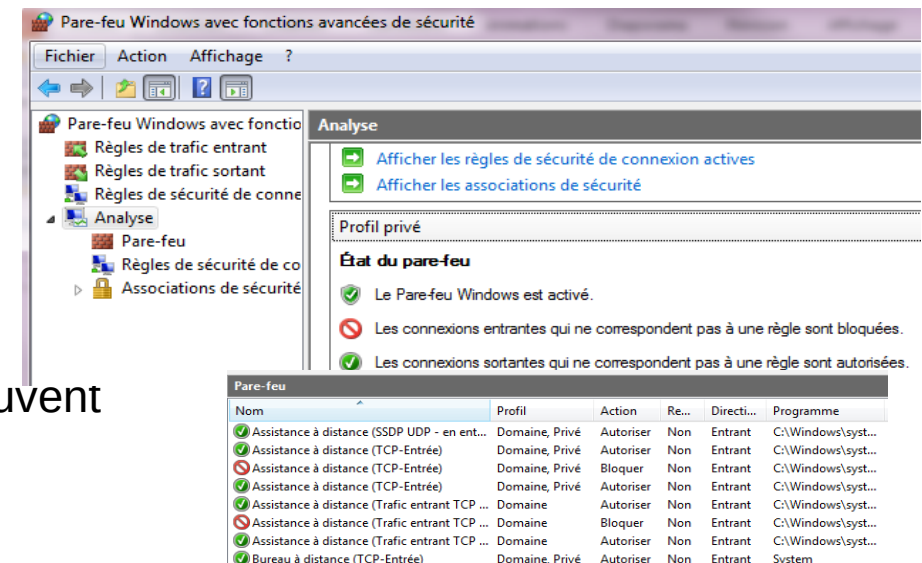
- Filtrer les flux pouvant être échangés entre les zones :
  - identifier les ports réseau utiles ;
  - identifier les protocoles réseau autorisés ;
  - disposer d'une matrice de flux indiquant les flux autorisés et interdits entre les zones.
- Autoriser explicitement des adresses IP (machines) d'une zone à échanger avec les adresses IP (machines) d'une autre zone
  - Utiliser une « liste blanche » d'adresse IP pour les échanges, et non pas une liste noire. Une liste noire ne peut en effet jamais être exhaustive, et est forcément d'un intérêt limité.

**Appliquer le principe « *Tout ce qui n'est explicitement autorisé est interdit* » lors de la gestion des flux.**

## 2. Maîtriser le réseau

### d. Protéger le réseau interne d'Internet

- Le réseau interne est à protéger et est considéré comme « **de confiance** ».
- Les équipements interagissant avec Internet peuvent être
  - placés dans une zone spéciale appelée « **Zone Démilitarisée (DMZ)** » ;
    - avec un niveau de filtrage et de contrôle plus accru que le réseau interne.
  - protégés d'Internet par des « pare-feux » filtrant les échanges de flux
    - Équipement dédié protégeant le réseau ou logiciel « pare-feu personnel »
- sous Windows, utiliser le pare-feu par défaut ou un pare-feu tiers (exemple Zone Alarm) ;
- toujours contrôler les connexions entrantes ;
- autoriser les applications au travers du pare-feu, au cas par cas.
- protégés derrière des IDS et des IPS qui peuvent
  - détecter les tentatives d'intrusion ;
  - prévenir les attaques.



## 2. Maîtriser le réseau

### ***e. Accès distant***

- Il est possible d'accéder à distance à un réseau pour faire :
  - du télétravail ;
  - de la téléassistance ;
  - de la télé-administration.
- Il est recommandé d'avoir des points d'entrée identifiés pour les accès distants :
  - Serveurs d'authentification : TACACS+, RADIUS.
  - Concentrateurs VPN.
  - Remote Access Server (RAS).

## 2. Maîtriser le réseau

### e. Accès distant

- Utiliser des moyens sécurisés pour les accès distants :
  - **SSH** au lieu de telnet : pour l'établissement de connexion à distance sur un équipement.
  - Secure remote desktop (**RDP**) : pour la prise en main à distance d'un bureau.
  - **SFTP** ou **SCP** : pour la copie distante.
  - **HTTPS** : pour l'accès à une interface Web (Exemple : Teamviewer).
  - Réseau Privé Virtuel (**VPN**) établi sur un réseau qu'on ne maîtrise pas, tel que Internet :
    - VPN IPSEC : permet l'authentification et le chiffrement ; il est utilisé pour protéger le trafic réseau.
    - VPN SSL : protège essentiellement le trafic Web, et est facile à déployer.

## 2. Maîtriser le réseau

### *f. Sécuriser l'administration*

- Restreindre/Interdire les interfaces d'administration depuis Internet
  - L'administration d'un composant ne doit pouvoir se faire que depuis le réseau interne (ouvrir un accès VPN en cas de nécessité d'accéder à distance).
- Restreindre les accès aux interfaces d'administration sur les sites Web :
  - Pour des sites web développés avec des CMS (Content Management System) comme Joomla! ou WordPress :
    - Le lien de la page d'administration peut être facilement trouvé (sauf à la modifier).
    - Des attaques en « brute force » peuvent être menées pour deviner le mot de passe administrateur.
    - Modifier le compte « admin » par défaut.
- Utiliser un réseau d'administration dédié :
  - Ce réseau doit être séparé du réseau de production de manière à ce que seuls les postes autorisés puissent s'y connecter.
  - Avoir une liste blanche des administrateurs autorisés à se connecter à ce réseau.
  - Authentifier mutuellement les postes des administrateurs et les équipements à administrer.



## 2. Maîtriser le réseau

### *g. Wifi*

- Pour sécuriser son réseau Wifi (fourni par sa box ou son téléphone), il faut :
  - Protéger la confidentialité des communications en effectuant un chiffrement à l'aide d'une clé :
    - La clé doit être composée de plusieurs caractères alphanumériques (min. 15).
  - Choisir la technologie WPA2 (**Wi-Fi Protected Access 2**).
  - Choisir l'algorithme de chiffrement CCMP (**C**ounter **C**ipher **M**ode **P**rotocol) lorsque possible.
  - Modifier le SSID (nom du réseau Wifi fourni).
  - Modifier les identifiants fournis par défaut pour accéder à l'interface d'administration :
    - en général, sur les box, saisir l'url « <http://192.168.1.1> » pour atteindre l'interface d'administration.
  - Ne pas divulguer (protéger) sa clé WIFI.



## 2. Maîtriser le réseau

### *g. Wifi : WPS*

- Ne pas utiliser le WPS (Wi-Fi Protected Setup), notamment fourni sur certaines box internet, car reconnu vulnérable à une attaque par force brute sur le code PIN. Sur les box Internet, il est donc préférable de configurer la connexion Wi-Fi manuellement et choisir son propre mot de passe (robuste) ;
- ou cocher l'option qui permet de désactiver automatiquement le WPS au-delà de 5 tentatives de clé.

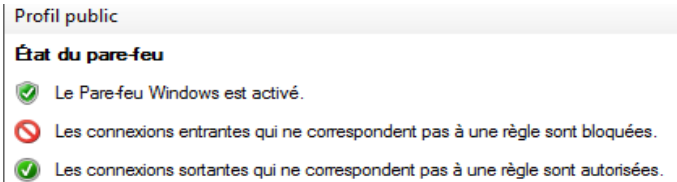
## 2. Maîtriser le réseau

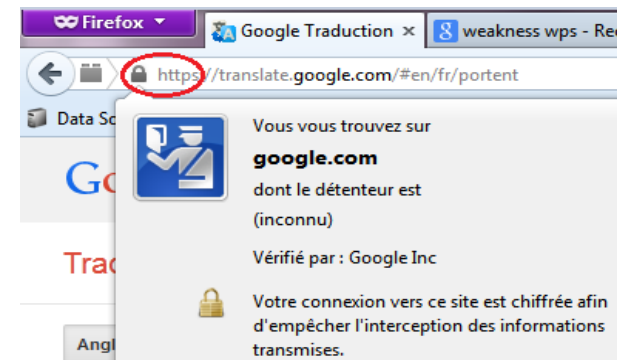
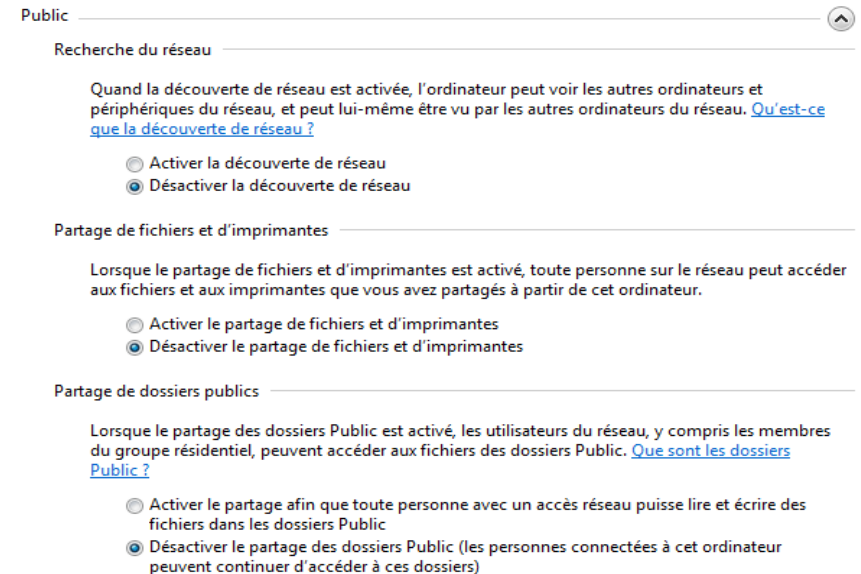
### *g. Wifi : Wifi privé vs Wifi public*

- Le Wifi privé peut être utilisé dans le réseau interne pour donner l'accès à des personnes de confiance. Dans un LAN, on peut utiliser le wifi comme moyen d'interconnexion. On parle alors de WLAN.
  - Pour ces wifi en entreprise, mettre en place si possible une authentification par certificats, cela évite que tous les utilisateurs partagent le même mot de passe.
- Le Wifi public (hotspots) : est fourni aux personnes « de non confiance » ou au grand public, et est généralement fourni pour un accès Internet uniquement :
  - Hotspot Wifi : Wifi dans les aéroports, McDo, etc.
  - Être conscient que tous les utilisateurs connectés sur le même hotspot peuvent écouter toutes les conversations (sauf si la page Web visitée est en HTTPS).

## 2. Maîtriser le réseau

### *g. Wifi : Bonnes pratiques en cas d'usage du Wifi Public*

- désactiver les options de partage :
    - arrêter la découverte réseau ;
    - arrêter le partage de fichier et d'imprimantes.
  - Activer le pare-feu du poste
    - Sous Windows, un pare-feu existe par défaut :
      - Contrôler les connexions entrantes et lorsque possible, les connexions sortantes.
- 
- Activer la journalisation.
  - Éviter de se connecter sur des sites peu sécurisés (utilisant du HTTP).
  - Vérifier que les communications vers les sites Internet se font en HTTPS.



**Si cela est possible, utiliser un VPN sur un Wifi public.**



## *Sécurité informatique*

### 3. Sécuriser les terminaux

- a) Choisir les applications
- b) Mises à jour logicielles et systèmes
- c) Antivirus / Antimalware / Antispyware
- d) Symptômes de présence des codes malicieux
- e) Protéger les données
- f) Durcissement de configuration des équipements

# 3. Sécuriser les terminaux

## a. Choisir les applications

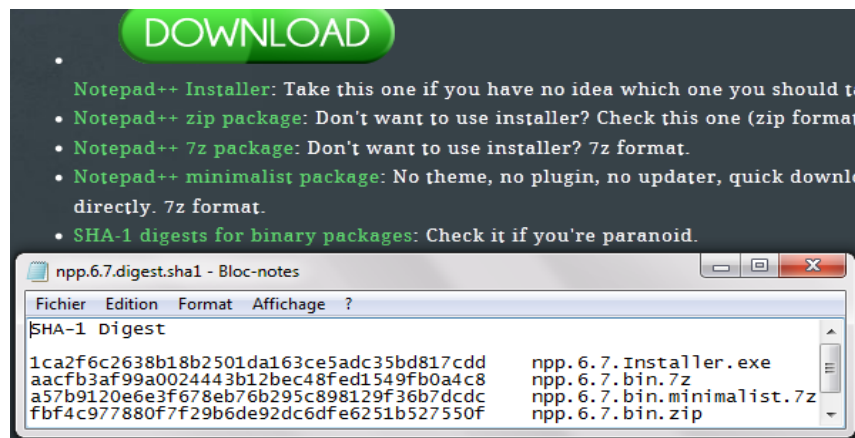
- Pourquoi faut-il être vigilant concernant les logiciels téléchargeables ?
  - On ne connaît pas forcément ni l'auteur, ni le site hébergeant ce logiciel.
  - Certains escrocs sont spécialisés dans la fourniture de chevaux de Troie : un malware est fourni avec le logiciel, dont l'objectif peut être de récupérer login, mot de passe, numéro de carte bancaire.
- Préférer des sources « sûres »
  - utiliser des sources « de confiance » pour télécharger les logiciels ;
    - Sous Android : interdire le téléchargement d'application depuis des sources inconnues.
  - utiliser les sites officiels (site de l'éditeur) pour les téléchargements.



# 3. Sécuriser les terminaux

## a. Choisir les applications

- Vérifier la signature d'un logiciel
  - recalculer la signature du fichier téléchargé avec la signature (checksum) indiquée sur le site, et comparer.



## 3. Sécuriser les terminaux

### *a. Choisir les applications*

- « Crack » logiciels
  - les sites proposant les « cracks » ou clés gratuites pour les logiciels payants sont souvent truffés de logiciels malveillants ;
  - les versions « crackées » de logiciels contiennent souvent des logiciels malveillants.
- Les logiciels gratuits
  - SnapDo est un pirate de navigateur qui infiltre les navigateurs Internet (Internet Explorer, Google Chrome, et Mozilla Firefox) via des téléchargements de logiciels gratuits.

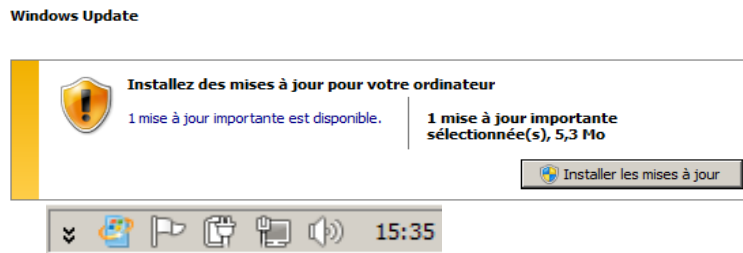




# 3. Sécuriser les terminaux

## *b. Mises à jour logicielles et systèmes*

- Rôle : apporter des modifications à un logiciel/application afin de corriger un dysfonctionnement ou une vulnérabilité.
- Les mises à jour s'appliquent :
  - aux applications, aux systèmes d'exploitation, etc.



- En entreprise, les mises à jour s'effectuent de manière centralisée
  - Téléchargement sur des serveurs dédiés, exemple serveur WSUS pour Windows.
  - Déploiement et observation sur des machines de test.
  - Sauvegarde des machines de production.
  - Déploiement sur les machines de production.

## 3. Sécuriser les terminaux

### *b. Mises à jour logicielles et systèmes*

- Les mises à jour ne concernent pas que le système d'exploitation : tous les logiciels peuvent présenter des failles et doivent être mis à jour régulièrement également.
  - Flash, Shockwave, Javascript, les lecteurs PDF sont connus pour nécessiter des mises à jour régulières.
  - La plupart des logiciels ont une option qui permet une « mise à jour automatique », il est recommandé de l'activer.
- Attention en entreprise, c'est à l'administrateur de planifier et valider les mises à jour (cela inclut notamment des tests préalables de non régression).

## 3. Sécuriser les terminaux

### *c. Antivirus / Antimalware / Antispyware*

- Ces logiciels peuvent être :
  - Gratuits :
    - installé par défaut lors de l'achat du terminal ou par l'éditeur du système d'exploitation (Microsoft Security Essential) ;
    - ou manuellement : Avast, Malwarebytes.
  - Payants : par exemple McAfee, Norton Antivirus.
- Ils nécessitent des mises à jour régulières du **moteur** et de la **base antivirale** pour détecter les nouveaux codes malveillants.
- Lors de l'apparition d'un nouveau code malveillant, des éditeurs de solutions antivirales effectuent des analyses afin de :
  - déterminer la « **signature** » de ce code malveillant pour l'identifier de manière unique ;
  - identifier les moyens de protection et des corrections ;
  - enrichir leur base antivirale avec ces informations.

**Éviter d'exécuter les scans gratuits depuis les pages Internet vous indiquant que votre ordinateur est infecté.**

# 3. Sécuriser les terminaux

## c. Antivirus / Antimalware / Antispyware

- Doivent être configurés de manière à :
  - Télécharger automatiquement les nouvelles signatures (base antivirale).
  - Être toujours actifs (faire attention si votre antivirus est désactivé).
  - Scanner tout l'ordinateur sans exception de répertoires / fichiers.
  - Effectuer des analyses complètes de manière périodique.
  - Analyser automatiquement de nouveaux périphériques tels que les clés USB.
  - Analyser les méls (entrants et sortants) et la messagerie instantanée.
- **Limites**
  - Il n'y a pas de base exhaustive pour les virus.
  - Un code malveillant peut sévir dans un système disposant d'un antivirus et y demeurer non détecté.
  - Les antivirus ne détectent que les virus dont les signatures sont « *connues* ».
  - De très nombreux codes malveillants sont créés chaque jour.

**L'antivirus n'est pas une « arme absolue ». La mise à jour des systèmes et des applications, ainsi qu'une bonne hygiène informatique sont indispensables.**

# 3. Sécuriser les terminaux

## *d. Symptômes de présence des codes malveillants*

- Ralentissement
  - du terminal : exemple pendant l'arrêt et le redémarrage ;
  - du débit : la bande passante semble partagée.
- Ouvertures régulières de fenêtres de pop-up et de publicités
- Modification de la configuration de votre navigateur web
  - Modification de votre page d'accueil ou de votre moteur de recherche ;
    - Exemple : Snapdo.
  - Présence de nouvelles extensions que vous n'avez pas installées.
- Surconsommation des ressources
  - Réduction de l'espace libre sur disque sans raison.
  - Surcharge du processeur.
- L'antivirus/antimalware ou pare-feu est désactivé sans votre intervention
- Les mises à jour système/antivirus/antimalware échouent systématiquement
- Messagerie
  - vos contacts (amis/famille) reçoivent des messages que vous n'avez pas envoyés ;
  - votre boîte d'envoi contient des messages que vous n'avez pas envoyés.

# 3. Sécuriser les terminaux

## e. Protéger les données

- Lors des échanges par mél
  - chiffrer les pièces jointes ou les données sensibles
    - exemple : AxCrypt, Zed Container ;
    - envoyer le mot de passe (Clé) par un autre moyen : SMS.
- Lors de l'usage du Cloud
  - utiliser des logiciels spécialisés pour protéger/chiffrer vos données dans le Cloud (DropBox, Box, SkyDrive...).



- En effectuant des sauvegardes
  - disque externe ;
  - Cloud.

**Chiffrer vos données sensibles avant de les stocker.**

# 3. Sécuriser les terminaux

## *f. Durcissement de configuration des équipements*

- Modifier les mots de passe des comptes par défaut.
  - exemple : administrateur.
- Désinstaller les logiciels/services inutiles (exemple : partage de fichiers).
- Désactiver les ports/lecteurs non utilisés.
  - port série / port USB ;
  - lecteur de disquette ;
  - désactiver le « débogage USB » sur les téléphones.
- Mettre un mot de passe BIOS lors de la phase de démarrage.
  - Lors du démarrage du poste, appuyer sur « F2 » pour rentrer dans le Setup ;
  - Aller dans l'onglet « Security » pour saisir les mots de passe.
- Désactiver le boot sur des périphériques externes (clé USB, CD Rom).
  - Dans le setup (touche « F2 » lors du démarrage), configurer l'ordre de démarrage pour avoir le disque dur en premier.
- Activer la journalisation.





## *Sécurité informatique*

### 4. Gérer les utilisateurs

- a) Attribution de privilèges
- b) Rôles utilisateur
- c) Mots de passe
- d) Autres méthodes d'authentification
- e) Sensibilisation des utilisateurs
- f) Spam
- g) Phishing / Spear phishing / Social engineering
- h) Réagir en tant que victime



## 4. Gérer les utilisateurs

### *a. Attribution de privilèges : grands principes*

- « Moindre privilège » : n'attribuer aux utilisateurs que les droits dont ils ont besoin pour effectuer leurs tâches
  - ne pas donner les privilèges importants à tous les utilisateurs, seulement à ceux qui en ont besoin ;
    - Exemple : le privilège « Administrateur »
  - pour un visiteur qui a juste besoin d'accéder à Internet : ne pas lui donner un accès aux disques ou aux applications sensibles.
- « Besoin de connaître » : donner les accès et les privilèges appropriés aux utilisateurs
  - donner accès seulement aux données nécessaires aux utilisateurs ;
  - restreindre l'accès aux répertoires contenant les données sensibles.

## 4. Gérer les utilisateurs

### *a. Attribution de privilèges : recommandations*

- Attribuer les comptes aux utilisateurs de manière nominative
  - Un utilisateur = un compte
  - Tracer les actions effectuées par chaque utilisateur.
  - Éviter les comptes partagés entre plusieurs utilisateurs.
- Faire signer une charte d'utilisation du SI, informant sur :
  - La conduite à tenir lors de l'usage du SI
    - Actions encouragées :
      - utiliser son poste pour des recherches, pour le travail qui est confié ;
      - protéger ses moyens d'accès : badge, identifiant, etc.
    - Actions interdites :
      - installer des logiciels malveillants / arrêter les outils de détection de codes malveillants ;
      - porter atteinte à un autre utilisateur du SI.
  - Les conditions et les règles d'utilisation des ressources du S.I.
  - Les responsabilités de l'utilisateur et ceux de l'entreprise/université.
  - Les sanctions internes, pénales, civiles encourues.
- Sous Windows, la commande « **GPEDIT.msc** » permet de configurer de manière fine les droits des utilisateurs.

## 4. Gérer les utilisateurs

### *a. Attribution de privilèges : procédures d'attribution / retrait de privilèges*

- Définir une procédure d'attribution/retrait de privilèges.
  - Tenir à jour une liste des droits attribués à chaque utilisateur.
  - Chaque nouveau compte utilisateur doit être créé en respectant les principes d'attribution de privilège.
  - Au besoin, chaque utilisateur doit avoir son répertoire personnel et sa boîte aux lettres.
  - Lorsque qu'un utilisateur n'a plus besoin d'accéder au système (démission, changement de poste...), la procédure de retrait de droit doit :
    - décrire la désactivation de son compte et la suppression de son compte ;
    - décrire la procédure de retrait des accès aux locaux (badge, clés).

## 4. Gérer les utilisateurs

### *b. Rôles utilisateur*

- Le rôle « **administrateur** » : ayant les privilèges les plus élevés sur le système. Il peut être de plusieurs types :
  - Administrateur système : en charge de l'administration des systèmes, de la gestion des disques.
  - Administrateur réseau : en charge des équipements réseaux, des règles de filtrage.
  - Administrateur sécurité : en charge de la journalisation, de la supervision.
- Le rôle « **utilisateur** » : ayant le droit d'utiliser le système et d'accéder à des répertoires sensibles.
- Le rôle « **invité** » : ayant peu de droits, et pas d'accès aux répertoires contenant les informations sensibles.

## 4. Gérer les utilisateurs

### *c. Mots de passe : politique de mots de passe*

- Définir une politique de mot de passe qui oblige à
  - Créer un mot de passe complexe :
    - différent d'un mot sorti du dictionnaire ;
    - différent d'une date de naissance (celle de votre conjoint, enfant...) ;
    - différent d'une partie du nom d'utilisateur, du nom, ou du prénom, etc.
  - Avoir un mot de passe d'au moins **8** caractères (**10** pour les admin).
  - Changer régulièrement les mots de passe (tous les 6 mois).
    - la fréquence des changements dépend de la sensibilité des systèmes accédés, par exemple le code pour accéder en ligne à son compte bancaire sera changé plus régulièrement.
  - Utiliser un mot de passe pour déverrouiller l'écran de veille.
- Consulter les recommandations élaborées par l'ANSSI.  
<http://www.ssi.gouv.fr/administration/bonnes-pratiques/>

## 4. Gérer les utilisateurs

### *c. Mots de passe : mémorisation*

- Ne pas choisir le même mot de passe pour différents comptes.
  - Même si ce principe devient difficile à respecter au vu du nombre de mots de passe que les utilisateurs doivent se rappeler.
  - A minima, **ne jamais réutiliser son mot de passe de messagerie**. Le compte mél devient en effet le pivot numérique de chacun.
    - En cas de perte de mot de passe, c'est souvent grâce à la boîte mél que l'on est en mesure d'en régénérer un nouveau.
    - Le mél sert aux sites marchands pour nous identifier lors de l'ouverture d'un compte.
    - Si le compte mél se fait pirater, c'est une partie significative de la vie numérique de l'utilisateur qui est affectée (usurpation d'identité, suppression malveillante de documents, changement forcé de mots de passe et impossibilité de les régénérer...).

## 4. Gérer les utilisateurs

### *c. Mots de passe : aide-mémoire*

- Aide-mémoire pour construire des mots de passe complexes :
  - Choisir une phrase comme mot de passe, on parle encore de « **passphrase** ».
    - Exemple : « Aujourd'hui je vais à l'aéroport. »
  - Ne garder que la première lettre de chaque mot puis un mot complet
    - **Ajv**à**l**a**é**ro**p**o**r**t
  - remplacer « s » par « **\$** », les « e » par « **3** » ;
  - Ajv@l@3r0port
- Le mot de passe est personnel et doit être mémorisé
  - Ne pas écrire les mots de passe sur les post-it.
  - Il existe des solutions pour stocker les mots de passe sous forme sécurisée (voir diapositive suivante).

**Les outils pour deviner les mots de passe, prennent parfois en compte le remplacement de « a » par « @ ».**

## 4. Gérer les utilisateurs

### *c. Mots de passe : stockage des mots de passe*

- Toujours stocker les mots de passe sous forme chiffrée
  - Mauvais exemple : Sony, répertoire nommé « Password » et contenant les mots de passe « en clair ».
- Utiliser des « porte-feuilles » de mots de passe :
  - Dashlane - KeyPass – 1Password ;
  - ou créer votre « porte-feuilles », chiffré et protégé par un mot de passe fort.
- ***Ne pas enregistrer les mots de passe sur les navigateurs Web.***



**Face aux limites des mots de passe et à leur difficulté d'utilisation, de nouveaux moyens d'authentification sont proposés.**



## 4. Gérer les utilisateurs

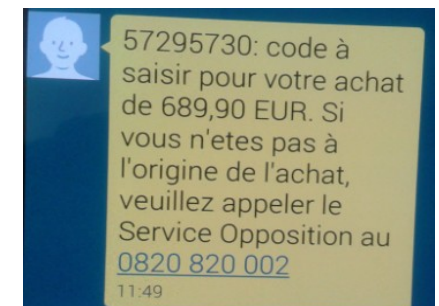
### *d. Autres méthodes d'authentification*

- Biométrie ;
  - permet l'authentification par la lecture des attributs physiques peu changeants d'une personne : empreinte digitale, voix, rétine, etc.
- Carte à puce + code pin
- SSO : Single Sign-On
  - L'utilisation du SSO permet d'éviter que les utilisateurs aient à ressaisir leurs mots de passe (utilisation d'un seul formulaire d'authentification pour accéder à différents services).
- OTP (One Time Password).
  - Généré à chaque demande et utilisable une seule fois. Durée de validité très courte :



Token Safenet

un OTP peut être un code de validation de paiement en ligne reçu par sms ;  
ou généré par un générateur matériel de jetons sécurisés.



## 4. Gérer les utilisateurs

### *e. Sensibilisation des utilisateurs*

- Se tenir informé de l'actualité liée à la sécurité :
  - vulnérabilités publiées ;
  - fuites d'information ;
  - « scam » arnaques sur Internet : arnaque à la nigériane, etc.
- Faire attention aux pièces jointes (même pour les expéditeurs connus).
  - télécharger d'abord et faire un scan avec l'antivirus, avant d'ouvrir la pièce jointe.

## 4. Gérer les utilisateurs

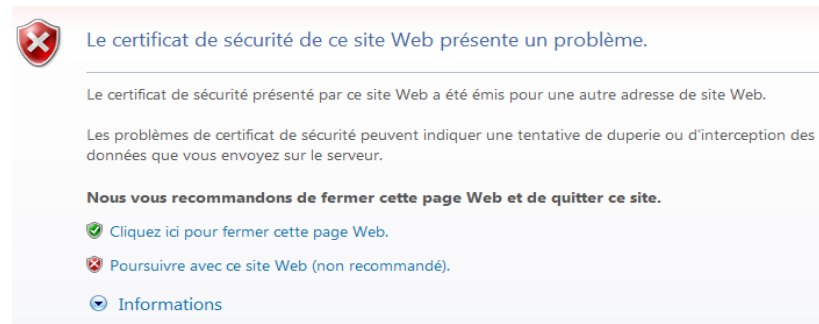
### *e. Sensibilisation des utilisateurs*

- Désactiver l'exécution des liens hypertextes et l'affichage des images dans les méls.
  - Il est préférable de copier et coller le lien hypertexte dans le navigateur. En effet, une technique dans le phishing consiste à faire afficher un lien qui paraît légitime à la lecture, mais qui pointe en fait vers une site malveillant. Cette technique ne fonctionne que si on clique sur le lien.
- Faire attention aux ralentissements/lenteurs de son poste.
- Applications web : penser à cliquer sur le bouton déconnecter lorsqu'on a fini de surfer sur le site afin de désactiver le cookie.
- Déconnecter son poste lorsqu'il n'est pas utilisé.

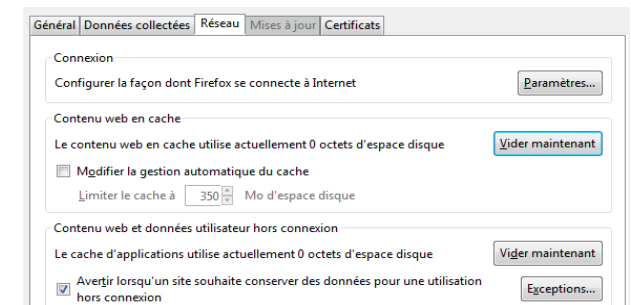
## 4. Gérer les utilisateurs

### e. Sensibilisation des utilisateurs

- Éviter les sites dont les certificats proposés ne sont pas reconnus.



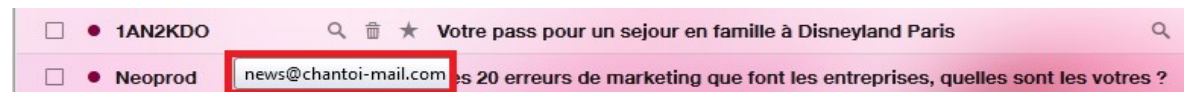
- Dans la mesure du possible, naviguer toujours en « https »
  - Cela est d'autant plus important sur les hotspots publics !
- Effacer régulièrement l'historique de navigation, les fichiers temporaires, les cookies de votre navigateur Web.



## 4. Gérer les utilisateurs

### f. Spam

- Traiter le spam
  - protéger son adresse mél ;
  - au besoin, créer une adresse poubelle : [xxx@yopmail.com](mailto:xxx@yopmail.com) ;
  - marquer les méls indésirables comme tel afin d'affiner la politique de détection des spams ;
  - ne pas ouvrir les spams, et ne pas cliquer sur les liens contenus.
- Faire attention aux méls envoyés dont l'émetteur est inconnu.



- Faire attention aux contenus des méls.

P. St, You are receiving this message because you opted-in your email address  
@yahoo.com to receive emails from diploe.antsy.bgxxbxx.com.

If you would like to be removed from our mailing list, please [click here](#).

To ensure ongoing optimal receipt of these communications, please add  
[customerservice@tabard.bgxxbxx.com](mailto:customerservice@tabard.bgxxbxx.com) to your address book.

If, for any reason, this promotion is not capable of running as planned, sponsor reserves the right to  
cancel, terminate, modify or suspend the promotion. This includes, but is not limited to, infection by  
computer virus, bugs, tampering, unauthorized intervention, fraud, technical failures or any other causes  
beyond the control of the sponsor. Why did the Onion Price Go Up So Suddenly?.. BecauseRajnikanth  
Ordered An Onion Dosa. ! :)

## 4. Gérer les utilisateurs

### *g. Phishing / Spear phishing / Social engineering*

- Ne pas donner suite au mél, coup de fil vous demandant de :
  - Rappeler rapidement votre conseiller bancaire alors que vous ne l'avez pas contacté.
  - Donner des informations personnelles parce que vous avez gagné un voyage, un prix, etc.
  - D'envoyer votre mot de passe/code bancaire/code pin par mél sous le prétexte urgent :
    - d'éviter la fermeture de votre adresse mél (car vérification en cours) ;
    - de valider l'existence de votre carte bancaire désactivée, etc.
  - De faire un transfert d'argent à un de vos contacts dans le besoin à l'étranger.

**En cas de doute, renseignez-vous mais ne répondez pas au mél.**

## 4. Gérer les utilisateurs

### *g. Phishing / Spear phishing / Social engineering*

- Limiter les informations que vous partagez par les réseaux sociaux ou méls
  - Date de départ en voyage
    - <http://pleaserobme.com> : sur la base de tweet (position) indique les maisons vides.
  - Informations personnelles
  - Données (photos/vidéos) potentiellement compromettantes
    - Chantage menant à des suicides d'adolescents « **chantage à la webcam** »
      - « Le jeune homme, prénommé Gauthier, a mis fin à ses jours le 10 octobre après avoir été victime d'un chantage sur Facebook de la part d'une jeune fille avec qui il venait de faire virtuellement connaissance. »
      - En janvier, Cédric, 17 ans, s'est pendu dans sa chambre à Marseille, 3 mois après avoir été piégé au cours d'un "plan webcam".
- Quelques liens utiles :
  - <https://www.arnaque-chantage-webcam.com>
  - <http://www.arnacoeurs.com>
  - <https://www.cnil.fr/fr/reagir-en-cas-de-chantage-la-webcam>

## 4. Gérer les utilisateurs

### *h. Réagir en tant que victime*

- Ne jamais payer de rançons
- En cas de chantage / usurpation d'identité / atteinte à la réputation :
  - Ne communiquez plus avec l'escroc
    - bloquer ses messages / son contact.
  - Signalez et recevez de l'aide
    - faire bloquer ce contact sur le site du chat, ou sur Facebook ou Skype ;
    - exercer le droit à l'oubli sur Google : [https://support.google.com/legal/contact/lr\\_eudpa?product=websearch&hl=fr](https://support.google.com/legal/contact/lr_eudpa?product=websearch&hl=fr);
    - signaler : <https://www.internet-signalement.gouv.fr/>
    - pour les mineurs : <https://www.netecoute.fr/>
- En cas de ransomware (rançongiciel) :
  - En entreprise ou à l'université : signalez aux responsables informatiques
  - A la maison : rechercher de l'aide sur des sites et forums spécialisés :
    - <https://stopransomware.fr/nettoyer-son-ordinateur/>
    - Les sites d'éditeurs de solutions antivirus : Symantec, etc.
- Porter plainte à la police ou à la gendarmerie.





## *Sécurité informatique*

# 5. Sécuriser physiquement

- a) Protection physique des locaux
- b) Imprimantes / Photocopieuses
- c) Sécuriser les équipements

## 5. Sécuriser physiquement

### *a. Protection physique des locaux*

- Protéger physiquement les locaux contenant les biens sensibles :
  - Contrôler l'accès aux locaux : usage de badges par exemple.
  - Utiliser des alarmes pour identifier les intrusions.
  - Protéger les clés ou badges dans des coffres par exemple.
- Les prises d'accès réseau doivent être protégées de manière à être inaccessibles aux visiteurs/personnes mal intentionnées.
  - Si les prises d'accès réseau doivent être exposées, ne pas les connecter au réseau. Mais plutôt le faire au besoin et désactiver ensuite.
- Protéger contre les incidents environnementaux :
  - Incendie : extincteur, détecteur de fumée, etc.
  - Inondation : s'installer en zone non inondable, surélever les éléments, etc.
  - Panne électrique : utiliser des onduleurs, etc.

## 5. Sécuriser physiquement

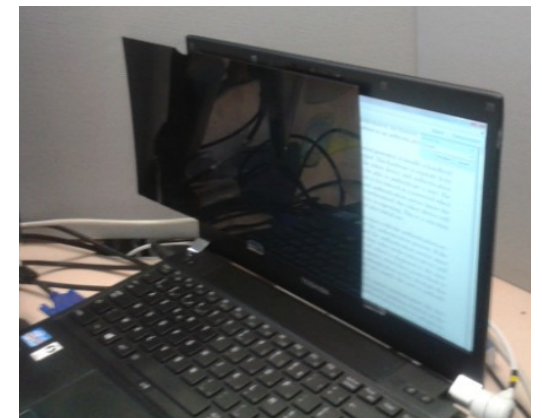
### ***b. Imprimantes / Photocopieuses***

- Faire attention lors des photocopies à ne pas oublier les originaux.
- Aller rapidement retirer les documents imprimés pour éviter que des informations sensibles soient révélées.
- Ne pas oublier que les imprimantes disposent :
  - de disques durs ;
  - d'historique des impressions : dont les titres de documents pourraient être révélateurs ;
  - de configuration IP pouvant être usurpée.
- Les documents papiers sensibles doivent être détruits à la déchiqueteuse.
- Les imprimantes ne doivent être accessibles depuis Internet.

# 5. Sécuriser physiquement

## *c. Sécuriser les équipements*

- Attacher avec un câble de sécurité les équipements le permettant.
- Protéger l'accès aux équipements :
  - Avoir un code/mot de passe pour restreindre l'accès à son équipement :
    - lecteur d'empreinte ou signe sur téléphone ;
    - code PIN ou mot de passe ;
  - Demander un code/mot de passe pour sortir de la veille.
- Verrouiller son écran en cas d'inactivité de quelques minutes.
- Faire attention aux médias USB :
  - Des clés USB piégées sont parfois offertes ou abandonnées.
  - Toujours scanner (anti-virus) une clé USB avant de l'utiliser.
- Utiliser les filtres de confidentialité d'écran.
  - Écran d'ordinateur (fixe, portable).
  - Écran de téléphone.





## *Sécurité informatique*

# 6. Contrôler la sécurité du S.I.

- a) Contrat/Maintenance/Professional Services
- b) Surveiller/Superviser
- c) Incidents de sécurité
- d) Plans de secours
- e) Audit

# 6. Contrôler la sécurité du S.I.

## *a. Contrat/Maintenance/Professional Services*

- Lors de l'achat de :
  - matériel : souscrire à des contrats de maintenance ou d'assurance pour vous garantir une assistance en cas de difficulté ;
  - application : souscrire à des contrats de support et d'assistance.
    - niveau 1 : description et enregistrement du problème rencontré ; conseil/information basique ;
    - niveau 2 : intervention de technicien ;
    - niveau 3 : intervention d'expert.
- SLA (Service Level Agreement) : indique le niveau de service garanti par le prestataire pour une prestation de service donnée.
  - Exemple : couverture 3G/4G/5G.

## 6. Contrôler la sécurité du S.I.

### *a. Contrat/Maintenance/Professional Services*

- Cyber-assurance : une assurance visant à indemniser et assister les victimes de cyber-attaque (fuite de données, attaque à la e-réputation...) :
  - Exemple : AXA propose pour les particuliers « Protection Familiale Intégr@ale »
  - Noter que la souscription d'une assurance est considérée comme une mesure de sécurité permettant de réduire les risques portant sur l'entreprise (au même titre qu'une assurance habitation n'empêchera pas un incendie, mais compensera/limitera les pertes financières de la victime).

**Souscrire à des services d'assurance/support/maintenance pour les composants sensibles.**

# 6. Contrôler la sécurité du S.I.

## ***b. Surveiller / Superviser***

- Activer la journalisation d'évènements.
  - Enregistrer les tentatives d'accès réussies ou pas.
  - Enregistrer les tentatives de modifications d'informations sensibles.
  - Etc.
- Consulter les journaux d'évènements.
- Définir une politique de supervision :
  - Définir les seuils : au-delà de tel taux d'occupation du disque, recevoir une alerte.
  - Définir le type d'alerte souhaité : SMS, mél, etc.



# 6. Contrôler la sécurité du S.I.

## *c. Incidents de sécurité : catégories d'incidents*

- Divulcation d'information personnelle.
  - Carte de crédit, vol d'identité, numéro de sécurité sociale, etc.
- Dénî de service.
  - Entrant ou sortant.
- Activité causée par un code malveillant.
  - Vers, virus, keylogger, rootkit.
- Enquête et activité criminelle.
  - Vol de terminal, fraude, pornographie infantile.
- Non respect de la politique de sécurité.
  - Partage de mot de passe.
- Défacement Web.
  - Redirection de site, défacement d'un site internet.
- Vulnérabilité non corrigée.
  - Système/application vulnérable, non application d'un correctif important.

# 6. Contrôler la sécurité du S.I.

## *c. Incidents de sécurité : gestion des incidents de sécurité*

- Un processus de gestion des incidents de sécurité permet de :
  - Réagir rapidement et de réduire l'impact en cas d'incident.
  - Améliorer la prévention et la sensibilisation.
  - Détecter et identifier les incidents.
  - Améliorer le niveau de sécurité.
- Exemple de réaction en cas d'une infection virale :
  - Déconnecter le poste du réseau ou d'Internet, sans l'éteindre.
  - S'assurer que l'antivirus/antimalware est à jour avec les dernières signatures.
  - Exécuter le scan complet (en « mode sans échec » par exemple) avec un antivirus.
  - Contacter un spécialiste au besoin.
  - Chercher à identifier la cause.

**La norme ISO 27035 décrit le processus de gestion des incidents.**

# 6. Contrôler la sécurité du S.I.

## ***d. Plan de secours***

- Avoir un plan de secours en cas de dysfonctionnement important (électrique, télécom...) :
  - Double alimentation :
    - pour un téléphone : batterie de secours ;
    - ordinateur/serveur : onduleur, batterie de secours, groupe électrogène.
  - Accès Internet :
    - utiliser son téléphone comme modem en cas de dysfonctionnement de sa Box ;
    - en entreprise, souscription à une offre Internet comme ligne de secours fournie par un opérateur différent.
  - Avoir une sauvegarde de ses données en cas de panne de son disque dur.
- En entreprise, il y a des :
  - PRA : Plan de Reprise d'Activité qui permet de « reprendre » après une interruption inattendue comme la perte d'un site de travail.
    - Exemple : utilisateur d'un site de secours « B » et déplacement du personnel en cas d'incendie dans le site principal « A »
  - PCA : Plan de Continuité d'Activité qui permet de s'assurer que l'activité ne s'arrêtera pas.
    - Exemple : usage d'une architecture réseau redondée en haute disponibilité.
      - Routeur en actif/actif.
  - Les PCA et les PRA doivent être testés et mis à jour régulièrement.

# 6. Contrôler la sécurité du S.I.

## ***e. Audit : informations générales***

- Un audit peut porter sur tout ou partie du S.I., une application, etc.
- Le but de l'audit est généralement :
  - d'évaluer le niveau de sécurité par rapport à un référentiel (interne ou à une norme) ;
  - obtenir un agrément ou une certification :
    - ASIP Santé, PCI DSS, 27001, etc.
  - trouver des faiblesses et les corriger :
    - site Web ;
    - application développée « in-house »
- L'audit peut être réalisé par :
  - des experts appelés « auditeur sécurité », « pen-testeur »
  - des sociétés spécialisées.
- Un cadre légal et contractuel est requis pour les audits :
  - Pour les audits de site Web, il faut l'accord du propriétaire du site (par exemple l'association étudiante), de l'hébergeur du site (OVH ou l'université) et parfois celui de l'opérateur.
  - L'auditeur doit indiquer à partir de quelles adresses IP publiques son audit sera effectué.
  - L'auditeur doit s'engager à ne pas provoquer d'incident de sécurité (déni de service par exemple) au cours de son audit.

# 6. Contrôler la sécurité du S.I.

## *e. Audit : types d'audit*

- Audit de conformité pour déterminer les écarts par rapport à un référentiel :
  - Politique de sécurité interne ou exigences de sécurité d'un cahier de charge.
  - Norme internationale : exemple 27001, PCI DSS, ASIP Santé.
- Audit en vue de l'obtention d'une certification/agrément :
  - Audit physique des datacenters pour obtention d'un agrément SAS 70.
  - Audit 27001 en vue de démontrer la bonne application des principes de la norme.
- Audit technique.
  - « Boite noire » ou « Pentest » : sans aucun accès, on évalue le système (site web par exemple) du point de vue d'un attaquant quelconque.
  - « Boite grise » ou « test du stagiaire » : on dispose de quelques informations et on essaye d'élever ses privilèges.
  - « Boite blanche » pour faire des « audits de configuration » par exemple : on dispose d'accès, y compris administrateur et on évalue le système par rapport à un référentiel.
  - « Forensic » ou « Post-mortem » : à effectuer sur un système après une attaque.



## *Sécurité informatique*

# 7. Focus : Contrôle d'accès

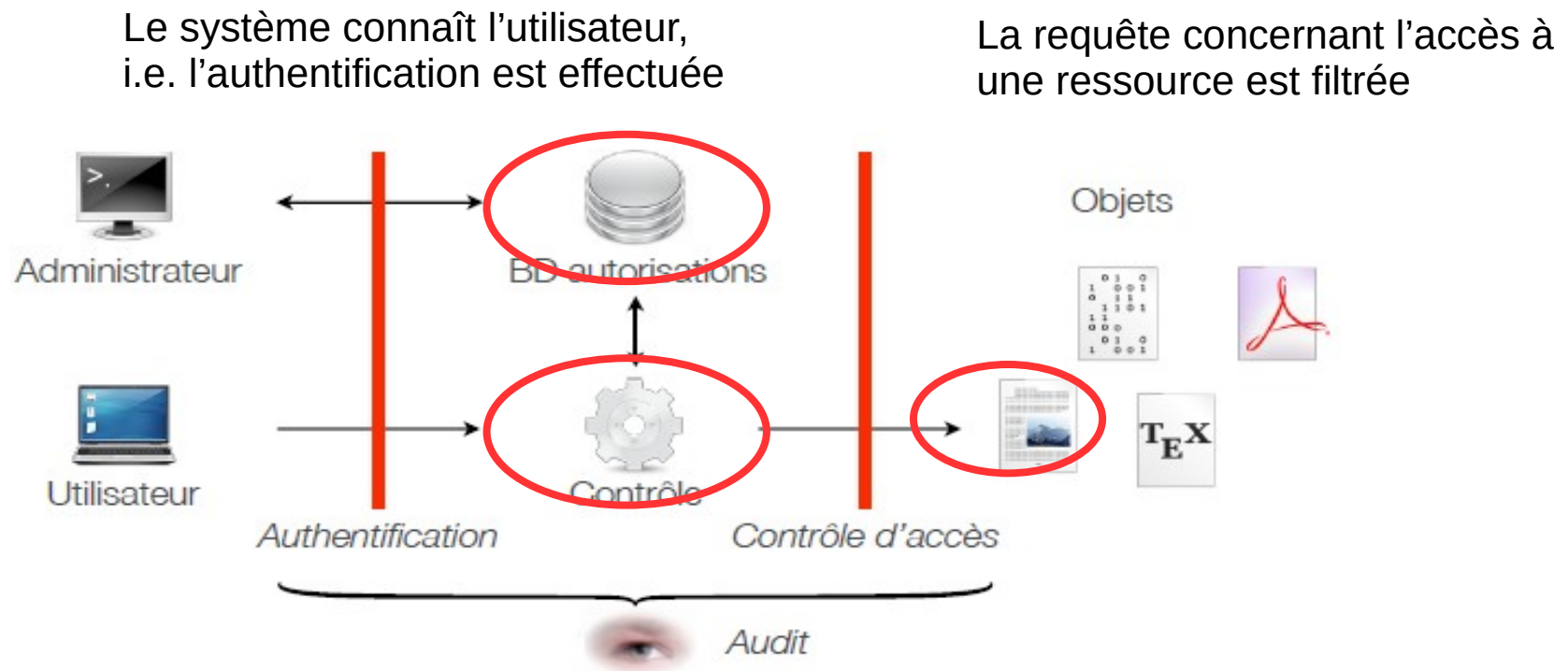
- a) Éléments de base
- b) Modèles DAC (ex : HRU)
- c) Modèles MAC (ex : BLP, Biba)
- d) Modèle RBAC

## a. Éléments de base

### ***Contrôle d'accès***

*Protéger les objets...*

# Composants du contrôle d'accès



Il existe plusieurs modèles de contrôle d'accès

Le mécanisme de contrôle d'accès vérifie les opérations qui peuvent être exécutées sur les données ou les ressources protégées



# Modèles de contrôle d'accès

## Les 3 A

- ***authentification***

- procédure qui consiste à vérifier l'identité d'un système

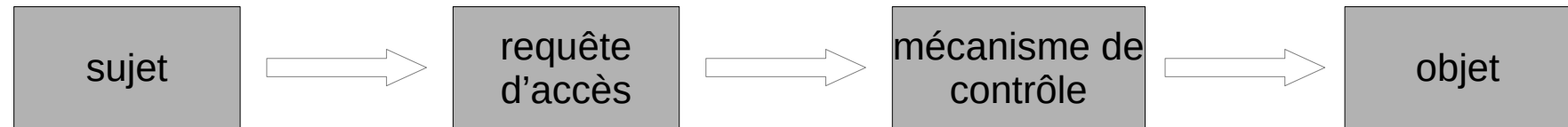
- ***autorisation***

- c'est un droit (ou une permission) accordé à un système pour l'accès à une ressource

- ***accès contrôlé***

- protection des ressources d'un système contre les accès non autorisés
- procédé par lequel l'utilisation des ressources est régulée en accord avec les politiques de sécurité et n'est autorisée qu'aux entités déclarées dans ces politiques

# Contrôle d'accès



B. Lampson. Protection. ACM Operating System Reviews, 8, 1974.

Sujets : entités **actives** du système

- personnes, processus, threads

Objets: entités **passives** du système

- données, programmes, fichiers
- attention : les sujets peuvent être des objets (processus et threads par ex.)

Les sujets demandent un accès (**requête**) : lire un fichier, écrire un fichier,...

Un **mécanisme de contrôle d'accès** détermine si l'accès demandé est autorisé et décide d'accorder ou de refuser cet accès

# Mécanisme de contrôle d'accès

Partie du système qui décide si l'accès demandé est autorisé ou non

La **décision** est basée sur

- l'identité du sujet qui fait la requête
- l'identité de l'objet qui doit être accédé
- le mode d'accès
- les autorisations actuelles
  - définies par une relation *Sujets* × *Objets* × *Permissions*
  - peuvent évoluer dans le temps

**Hypothèses** qui doivent être **vérifiées**

- le mécanisme de contrôle ne peut **pas être détourné**
- le mécanisme de contrôle ne peut **pas être altéré**

# Mécanisme de contrôle d'accès

Le **mécanisme** de contrôle d'accès définit comment le contrôle d'accès **agit**, alors que le **modèle** de contrôle d'accès définit comment il **devrait agir**

- comment décide-t-on si une requête est autorisée ou non ?
- comment les autorisations sont-elles modifiées ?

La décision est basée sur ... (voir slide précédent)

Les hypothèses concernant le mécanisme sont identiques (voir slide précédent)

La différence ?

- une spécification est plus abstraite qu'un programme
  - le comportement est souvent spécifié par une machine à état qui opère à partir d'un état du système de protection
  - un mécanisme de contrôle d'accès (une implantation) peut être vérifié par rapport à un modèle de contrôle d'accès

# Objet

N'importe quel artefact qui possède des données

- fichier, relation, répertoire, message, paquet réseau, périphérique d'E/S, disque, mémoire,...

## **Deux catégories d'objets :**

- ceux qui doivent être protégés
  - pris en compte par le mécanisme de contrôle d'accès
- ceux qui ne sont pas nécessairement protégés
  - non considérés par le mécanisme de contrôle d'accès

# Sujet

Entité active du système qui peut exécuter une opération

Les sujets peuvent être classés :

- **utilisateurs**
  - entités physiques qui se connectent au système
- **groupes**
  - ensembles d'utilisateurs
- **rôles**
  - collections de permissions/entités fonctionnelles dans une organisation
- **processus**
  - programmes qui s'exécutent sous la responsabilité des utilisateurs

Il peut exister des **relations** entre les sujets

# Modes d'accès

**Opérations** qu'un sujet peut exécuter sur les objets protégés du système

Chaque classe d'opération correspond à un mode d'accès

Modes d'accès les plus simples :

- read
- write

Dans la réalité, **nombreux modes d'accès** :

- dépendent des ressources à protéger: fichiers, tables, n-uplets, périphériques,...
- read, write, execute, select, insert, update, delete,...

# Modes d'accès (suite)

## Exemple : Linux

Modes d'accès définis pour les fichiers :

- read : lire le fichier
- write : écrire dans le fichier
- execute : exécuter un fichier

Modes d'accès pour les répertoires :

- read : lister le répertoire
- write : créer ou renommer un fichier du répertoire
- execute : rechercher dans un répertoire



# Permissions d'accès

Également appelées *autorisations* ou *droits*

Définies en termes de sujet, objets, et modes d'accès

Conceptuellement, une autorisation est un triplet  $\langle s, o, a \rangle$  tel que

- $s$  est un sujet
- $o$  est un objet
- $a$  est un mode d'accès

Cela signifie que : « *le sujet  $s$  possède la permission d'exécuter l'opération  $a$  sur l'objet  $o$*  »

## Exemple

$\langle \text{Bob}, f1, \text{read} \rangle$  signifie que Bob possède la permission de lire le fichier  $f1$

# Hiérarchies

Les sujets, objets et modes d'accès peuvent être organisés hiérarchiquement

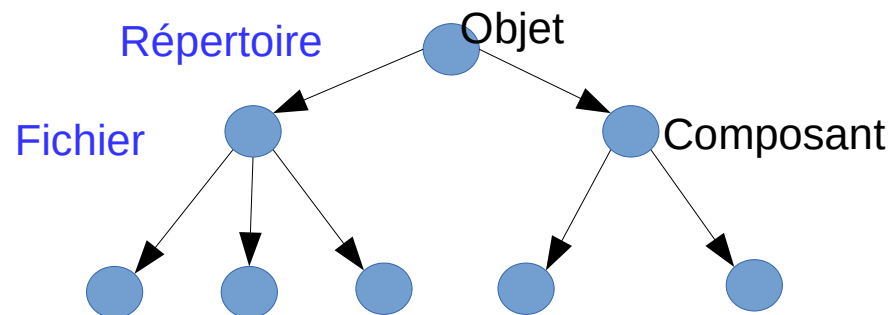
La sémantique de la hiérarchie dépend du domaine considéré

## Avantages

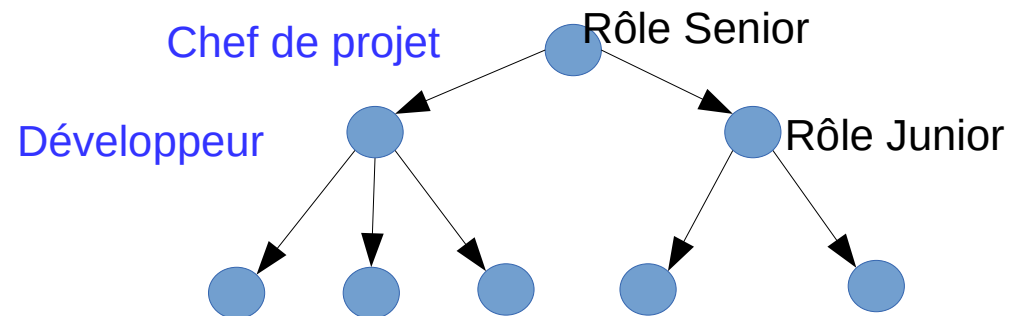
- réduction du nombre de permissions, d'où réduction du coût d'administration
- association possible avec des autorisations négatives, permettant ainsi de définir des exceptions

# Hiérarchies (suite)

## Hiérarchie d'objets (part-of)

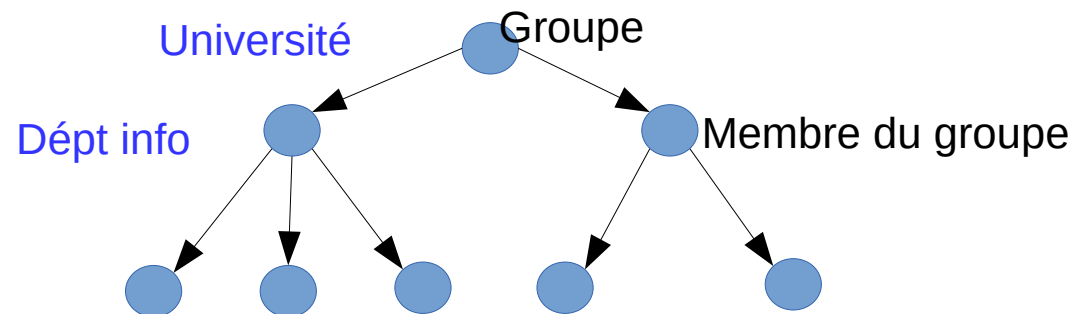


## Hiérarchie de rôles



# Hiérarchies (suite)

## Hiérarchie de groupes



On suppose que le groupe Dépt info regroupe 200 personnes, et que le groupe Université regroupe 5000 personnes

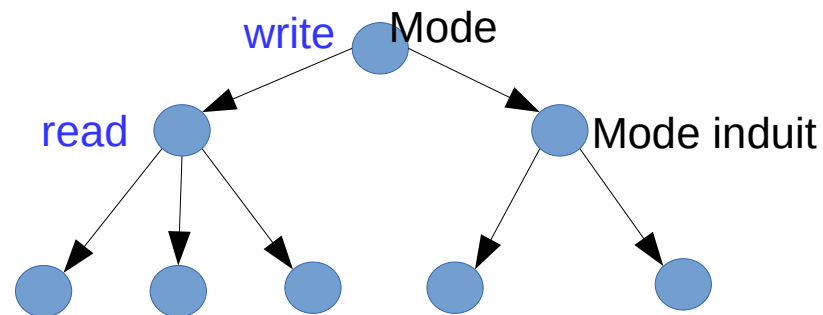
On suppose que l'on a une politique qui stipule que l'agenda du département peut être consulté par tous les membres de l'Université, mais peut uniquement être modifié par les membres du département info

Deux permissions sont nécessaires :

- <Université, agenda, read>
- <Dépt info, agenda, write>

# Hiérarchies (suite)

Hiérarchie de modes d'accès : subsumption



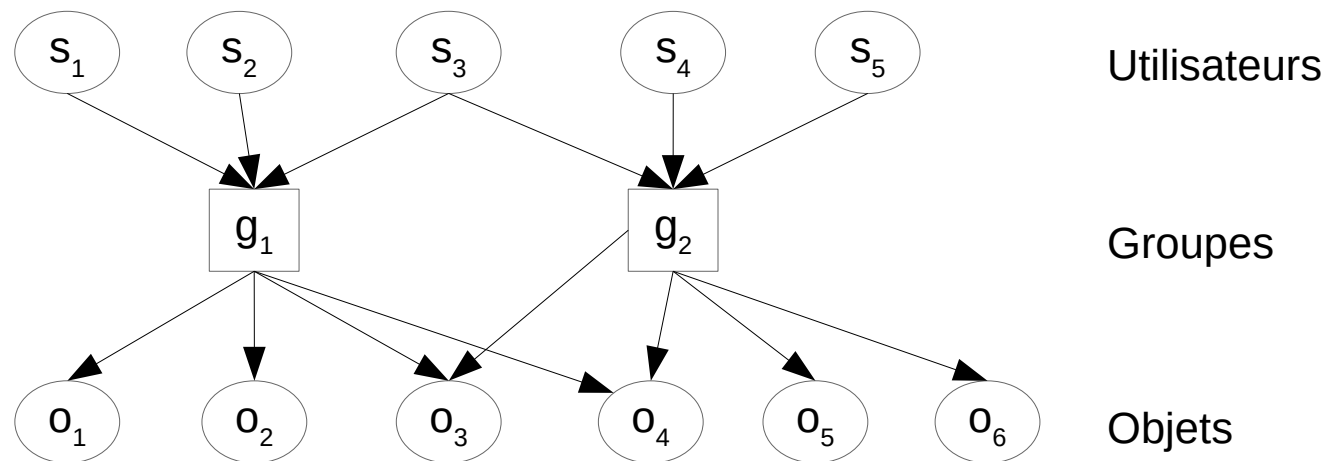
# Groupes

Un groupe peut être vu comme un niveau intermédiaire entre les utilisateurs et les objets

Un groupe est un ensemble de sujets

En définissant des droits d'accès en fonction du groupe plutôt que pour chacun de ses membres, on simplifie la définition de ses droits d'accès

Exemple :



# Permissions négatives

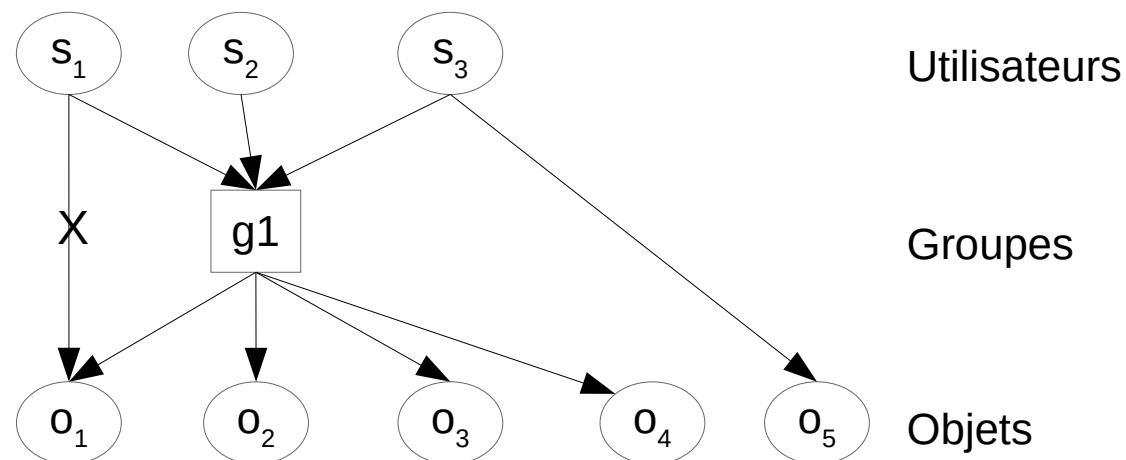
Quelques fois, il est plus pratique

- de donner directement une permission à un utilisateur sur un objet
- ou bien de lui retirer une permission qui est normalement induite par son appartenance à un groupe

Une permission négative spécifie les opérations qu'un sujet ne peut pas accomplir sur un objet

La représentation des permissions négatives devient alors un n-uplet  $\langle s, o, a, \text{signe} \rangle$  où  $\text{signe} \in \{+, -\}$

Exemple :



# Administration

Question de l'administration des autorisations ?

Qui définit quels sont les sujets qui peuvent accéder à quels objets pour quelles opérations ?

Qui possède l'autorisation de modifier les autorisations ? d'en définir de nouvelles ? d'en supprimer ?

Suivant le type d'approche :

- **discretionnaire**

- le propriétaire de l'objet définit qui peut accéder
- Question : qui est le propriétaire de l'objet ?

- **obligatoire**

- la politique au niveau du système définit qui peut accéder



# Structures de contrôle d'accès

## Deux types

- **DAC : Discretionary Access Control model**

- « si un utilisateur individuel peut définir un mécanisme de contrôle d'accès pour autoriser ou refuser l'accès à un objet, le mécanisme est un mécanisme discrétionnaire (DAC) (appelé également IBAC – *identity-based access control*) »

- **MAC : Mandatory Access Control model**

- « lorsque le mécanisme système contrôle l'accès à un objet et qu'un utilisateur individuel ne peut modifier cet accès, le contrôle est dit obligatoire (MAC) (appelé également *rule-based access control*) »

## b. DAC: contrôle d'accès discrétionnaire

**Idée :** les utilisateurs possèdent les objets et contrôlent les accès

Le propriétaire de l'information ou de la ressource est susceptible de modifier les permissions à sa discrétion

Le propriétaire peut également transférer la possession aux autres utilisateurs

### **Avantages**

- flexible

### **Inconvénients**

- risques d'erreurs, de négligence, d'abus
- les utilisateurs doivent comprendre le mécanisme et respecter les politiques de sécurité
- abus possible avec des chevaux de Troie : application qui transfère les droits des utilisateurs

# Exemple : Linux

Linux propose un mécanisme implantant une classe restreinte de DAC

- contrôle d'accès sur les objets qui utilise un schéma de permissions basé sur *propriétaire/groupe/autre*
- les permissions sont attribuées aux objets par leur propriétaire
- toutes les politiques ne sont pas modélisables avec ce mécanisme
  - par exemple, comment exprimer qu'un patient peut accéder à toutes les informations le concernant à l'hôpital ? qui possède ces informations ?

Support limité de la délégation (*setuid* ou *setgid*)

- l'exécutant prend l'identité du propriétaire (utilisateur ou groupe) à l'exécution
- exemple : un utilisateur normal prend les droits root pour changer son mot de passe
- abus possibles, trous de sécurité

# Définition

Un **état protégé** (relativement à un ensemble de privilèges) est un triplet (S,O,P)

- S un ensemble de sujets
- O un ensemble d'objets
- P un ensemble de privilèges

Une matrice  $M$  sur  $S$ ,  $O$  et  $P$  est définie comme

$$M = (M_{so})_{s \in S, o \in O} \text{ avec } M_{so} \subset P$$

$M_{so}$  définit les opérations que le sujet  $s$  peut effectuer sur l'objet  $o$

Les **transitions entre les états protégés** sont définies grâce à des commandes du type

- **entrer** privilège/droit  $p$  dans  $M_{so}$
- **créer** sujet  $s$
- **détruire** objet  $o$

# Matrice de contrôle d'accès

		<i>Objets</i>		
<i>Sujets</i>		<b>bill.doc</b>	<b>edit.exe</b>	<b>fun.dir</b>
	<b>Alice</b>	-	{execute}	{execute, read}
	<b>Bill</b>	{read, write}	-	{execute, read, write}

← *Privilèges*

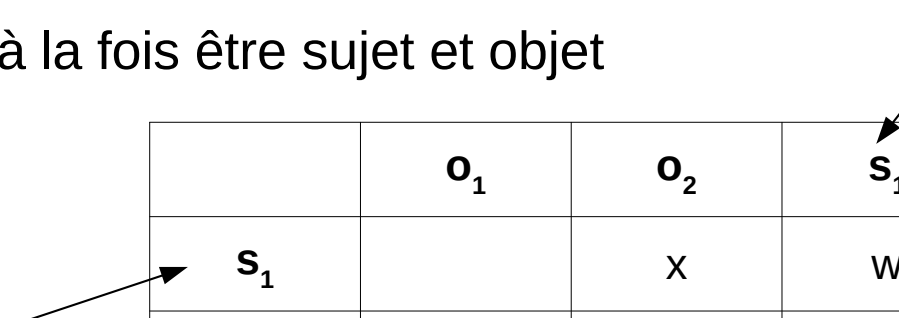
# Matrice de contrôle d'accès (suite)

Une entité peut à la fois être sujet et objet

Exemple

- un processus

*processus  $s_1$   
vu comme un sujet*



	$o_1$	$o_2$	$s_1$	$s_2$	$s_3$
$s_1$		x	w		
$s_2$	r	r			
$s_3$	r,w				

- le processus  $s_1$  ne pourra jamais lire ou écrire  $o_1$  ou  $o_2$ , mais il pourra exécuter  $o_2$  (“x” dans la cellule  $M[s_1, o_2]$ ) et il pourra se modifier lui-même (“w” pour *write* dans  $M[s_1, s_1]$ )
- le processus  $s_2$  pourra lire (“r”)  $o_1$  et  $o_2$ , mais ne pourra pas les écrire ou les exécuter
- le processus  $s_3$  pourra lire et écrire  $o_1$ , mais n’a pas d’autres autorisations

# Exemple

Considérons la politique suivante

- « aucun sujet ne peut obtenir les droits d'accès en lecture à un fichier sauf si le droit *read* a été explicitement accordé par le propriétaire »

Formalisation

- soit  $s = (S, O, M)$  un état autorisé, avec  $own \in M(p, f)$  pour le sujet  $p$  et le fichier  $f$
- $read \notin M(q, f)$  pour un sujet  $q$
- soit  $s' = (S', O', M')$  un état tel que  $s \vdash_c s'$  et  $read \in M'(q, f)$
- $s'$  est autorisé ssi  $c = confer.read(p, q, f)$  où  
    command confer.read( $s_1, s_2, o$ )  
        if  $own \in M(s_1, o)$   
        then enter read into  $M(s_2, o)$   
    end

# Structures de données

La matrice est la structure abstraite qui définit les droits d'accès

Comment implanter cette matrice ?

	Fichier 1	Fichier 2	Fichier 3	Fichier 4
Alice	Own R W		W X	
Bob	R	Own R W	W	R
Charlie	R W	R		Own R W



# Matrice

Représentation la plus simple

Assez inefficace

- la plupart du temps, matrices creuses

# ACL: Access-control list

Utilisation de listes pour une définition **basée sur les objets**

- les objets sont associés à des listes de paires <sujet, attributs d'accès>
- les ACL correspondent aux colonnes de la matrice
- exemple
  - *Fichier 1* : [(Alice,own),(Alice,r),(Alice,w),(Bob,r),(Charlie,r),(Charlie,w)]
- les ACLs sont utilisées dans Linux par exemple

# Listes de capacités

Utilisation de listes pour une définition **basée sur les sujets**

- les sujets sont associés à des listes de paires <objets, attributs d'accès>
- les listes de capacités correspondent aux lignes de la matrice
- exemple : *Charlie* :  $[(Fichier\ 1, r), (Fichier\ 1, w), (Fichier\ 2, r), \dots]$

Moins utilisé que ACL

- pas très compatible avec la vision orientée objet
- difficile d'évaluer qui possède les permissions sur un objet
- difficile de révoquer une permission

**Application dans le contexte distribué**

- les utilisateurs sont dotés de certificats qu'ils présentent aux objets distribués

# Comparaison

Y-a-t-il une différence en fonction de la représentation choisie ?

- **oui**, si on parle implantation du mécanisme de contrôle d'accès
  - la représentation avec des listes nécessite moins d'espace mémoire
  - les autorisations peuvent être directement associées aux sujets ou aux objets
- **non**, il n'y a pas de différence quant à l'expressivité des politiques de sécurité
  - les mêmes politiques sont exprimables
  - on peut transformer une représentation dans une autre

# Comparaison (suite)

Étant donné un sujet, quels sont les objets auxquels il peut accéder, et comment ?

Étant donné un objet, quels sont les sujets qui peuvent y accéder, et comment ?

ACL et capacités peuvent répondre aux 2 questions

# Comparaison (suite)

Étant donné un sujet, quels sont les objets auxquels il peut accéder, et comment ?

Étant donné un objet, quels sont les sujets qui peuvent y accéder, et comment ?

ACL et capacités peuvent répondre aux 2 questions

## **Première question**

- le plus simple : capacités

## **Deuxième question**

- le plus simple : ACL

Quelle est la question à laquelle on devra répondre le plus souvent ?

# Exemple : Le modèle HRU

Proposé par Harrison, Ruzzo et Ullman en 1976

Article : *M.Harrison, W. Ruzzo, J. Ullman. Protection in Operating Systems. Comm. of ACM 19(8), August 1976.*

Deux concepts importants

- notion de système d'autorisations
- notion de sûreté (safety)

# Le modèle HRU (suite)

Dans HRU, on considère :

- $O$  : ensemble d'objets
- $S \subseteq O$  : ensemble de sujets
- $P$  : un ensemble de droits, par ex.  $P = \{own, read, write, execute\}$

Un état du système de protection est représenté par une matrice  $M$  tel que :

- chaque objet de  $O$  est associé à une colonne de  $M$
- chaque sujet de  $S$  est associé à une ligne de  $M$
- à chaque objet  $o$  et sujet  $s$  est associé une cellule  $M(s,o)$  qui contient un ensemble de droits  $p \in P$

Il existe 6 primitives permettant de modifier l'état du système :

Enter $p$ into $M(s,o)$	Delete $p$ into $M(s,o)$
Create subject $s$	Create object $o$
Destroy subject $s$	Destroy object $o$



# Le modèle HRU (suite)

Les **transitions possibles** du système de protection sont définies par un ensemble fini  $C$  de **commandes**

```
command  $c(x_1, \dots, x_k)$   
  if  $r_1 \in M(x_{s1}, x_{o1})$  and ... and  $r_m \in M(x_{sm}, x_{om})$   
  then  $op_1, \dots, op_n$   
end
```

avec au moins une opération :  $n \geq 1$

Un **système de protection** est un quintuplet  $(P, O, S, M_0, C)$  où

- $P$  est un ensemble fini de droits,
- $O$  est un ensemble d'objets,
- $S \subseteq O$  est un ensemble de sujets,
- $M_0$  est une matrice de droits d'accès, représentant l'état initial du système de protection,
- $C$  est un ensemble fini de commandes

# Le modèle HRU (suite)

La partie **condition** de la commande

- vérifie si les permissions sont présentes dans la matrice
- peut être vide

Si toutes les conditions sont vérifiées

- la séquence d'opérations est exécutée

Une **commande contient au moins une opération**

Les commandes qui ne contiennent qu'une et une seule opération sont dites mono-opérationnelles

Une matrice  $M$  de droits d'accès est accessible à partir de  $M_0$  s'il existe une suite finie de commandes de  $C$  qui permettent d'aboutir à  $M$

# Le modèle HRU (suite)

## Exemples de commandes

command *create\_file(s,f)*

create *f*

enter *o* into  $M_{s,f}$

enter *r* into  $M_{s,f}$

enter *w* into  $M_{s,f}$

end

command *grant\_read(s,p,f)*

if *o* in  $M_{s,f}$

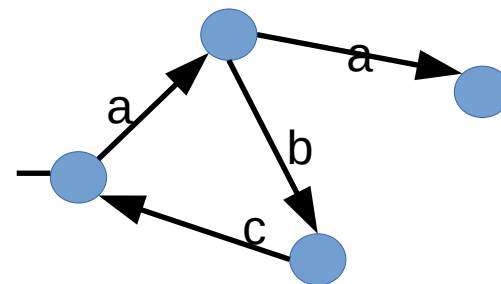
then enter *r* into  $M_{p,f}$

end

# Le modèle HRU (suite)

On écrit  $(S, O, M) \vdash_c (S', O', M')$  pour indiquer une transition associée à la commande  $c$

L'état de départ  $s_0 : (S_0, O_0, M_0)$  et l'ensemble de commandes  $C$  définissent un système état-transition



Un modèle définit donc un ensemble de traces autorisées, de la forme

- $st_0, st_1, st_2, st_3, \dots$

où  $st_0 = s_0$ , et  $st_i \vdash_{c_j} st_{i+1}$  pour  $c_j \in C$

# Sémantique des primitives

$$(S, O, M) \xrightarrow{\text{Enter } r \text{ into } (s^*, o^*)} (S', O', M')$$

$$(s^* \in S \wedge o^* \in O) \Rightarrow$$
$$(S' = S \wedge O' = O \wedge M'[s^*, o^*] = M[s^*, o^*] \cup \{r\}$$
$$\wedge \forall s \in S \setminus \{s^*\}, o \in O : M'[s, o] = M[s, o]$$
$$\wedge \forall s \in S, o \in O \setminus \{o^*\} : M'[s, o] = M[s, o])$$

$$(s^* \notin S \vee o^* \notin O) \Rightarrow (S' = S \wedge O' = O \wedge M' = M) \quad \text{et la requête échoue}$$

$$(S, O, M) \xrightarrow{\text{Delete } r \text{ into } (s^*, o^*)} (S', O', M')$$

$$(s^* \in S \wedge o^* \in O) \Rightarrow$$
$$(S' = S \wedge O' = O \wedge M'[s^*, o^*] = M[s^*, o^*] \setminus \{r\}$$
$$\wedge \forall s \in S \setminus \{s^*\}, o \in O : M'[s, o] = M[s, o]$$
$$\wedge \forall s \in S, o \in O \setminus \{o^*\} : M'[s, o] = M[s, o])$$

$$(s^* \notin S \vee o^* \notin O) \Rightarrow (S' = S \wedge O' = O \wedge M' = M) \quad \text{et la requête échoue}$$

**Exercice :** écrire create/destroy subject/object  $s^*/o^*$

# HRU : état

Quand dit-on qu'un état est sûr ?

- Définition 1 : l'accès aux ressources sans la participation du propriétaire est impossible [HRU76]
- Définition 2 : l'utilisateur doit être en mesure de dire si ce qu'il va faire (donner un droit) peut entraîner une fuite de ce droit à des sujets non autorisés

HRU introduit la notion de *sûreté*

# Sûreté

Le problème à la base de la motivation de la sûreté peut être défini comme suit :

« Supposons un sujet  $s$  qui prévoit de donner au sujet  $s'$  le privilège  $p$  sur l'objet  $o$ . La question est alors de savoir si la matrice des droits d'accès, avec  $p$  étant une entrée dans  $(s', o)$  est telle qu'il devient possible que le privilège  $p$  puisse alors apparaître dans une nouvelle entrée de la matrice ? »

On dit alors qu'il existe une **fuite du privilège  $p$**  à partir d'un état, i.e. une matrice d'accès  $M$ , si il existe une commande  $c$  qui ajoute le privilège  $p$  dans une cellule de la matrice qui ne contenait pas auparavant  $p$

**Théorème** : étant donnés une matrice  $M$  et un privilège  $p$ , vérifier la sûreté de  $M$  par rapport à  $p$  est indécidable

# Sûreté (suite)

Un système de protection est sûr si un privilège  $p$  ne peut pas être ajouté dans une cellule de la matrice où il n'était pas au départ

## Définition 1

Soit  $(P, O, S, M_0, C)$  un système de protection. Une matrice  $M$  laisse fuir le privilège  $p \in P$  s'il existe une commande  $c \in C$  telle que la matrice  $M'$  après exécution de la commande  $c$  contienne une cellule  $M'_{s,o}$  telle que  $p \in M'_{s,o}$

## Définition 2

Un système de protection  $(P, O, S, M_0, C)$  est sûr pour le privilège  $p \in P$  s'il n'existe pas de matrice de droit d'accès  $M$  accessible à partir de  $M_0$ , telle que  $M$  laisse fuir  $p$

## Exemple

Si  $o$  est un fichier qui contient des données privées de l'utilisateur  $s$ , et que  $s$  ne veut pas partager ces données, alors tous les états avec " $p \in M[s',o]$ " tels que  $s' \neq s$  ne sont pas sûrs



# Sûreté (suite)

Supposons que Bob ait développé une application A1

Il veut que son application puisse être exécutée par d'autres, mais pas modifiée

Le système de protection suivant n'est pas sûr par rapport à cette politique

Pourquoi ?

Considérons la séquence de commandes suivante :

- Bob : *grant\_execute*(Bob, Tom, A1)
- Tom : *modify\_own\_right*(Tom, A1)

Le résultat est :  $M_{Tom, A1}$  contient le droit  $w$

```
command grant_execute(s,p,f)
  if o in  $M_{s,f}$ 
    then enter x into  $M_{p,f}$ 
  end

command modify_own_rights(s,f)
  if x in  $M_{s,f}$ 
    then enter w into  $M_{s,f}$ 
  end
```

# Décidabilité

La **sûreté** est un problème

- **décidable** pour les systèmes de protection mono-opérationnels
- **indécidable** pour le cas général

Création d'un fichier sous Linux :

- 2 commandes: création, positionnement des privilèges

Les résultats de décidabilité montrent que la sécurité est un problème difficile

- si on conçoit un système complexe qui ne peut être décrit que par des modèles complexes, il est difficile de trouver les preuves
- dans le pire des cas (indécidabilité), il n'existe pas d'algorithme capable de vérifier la sécurité pour toutes les instances du problème

# Chevaux de Troie

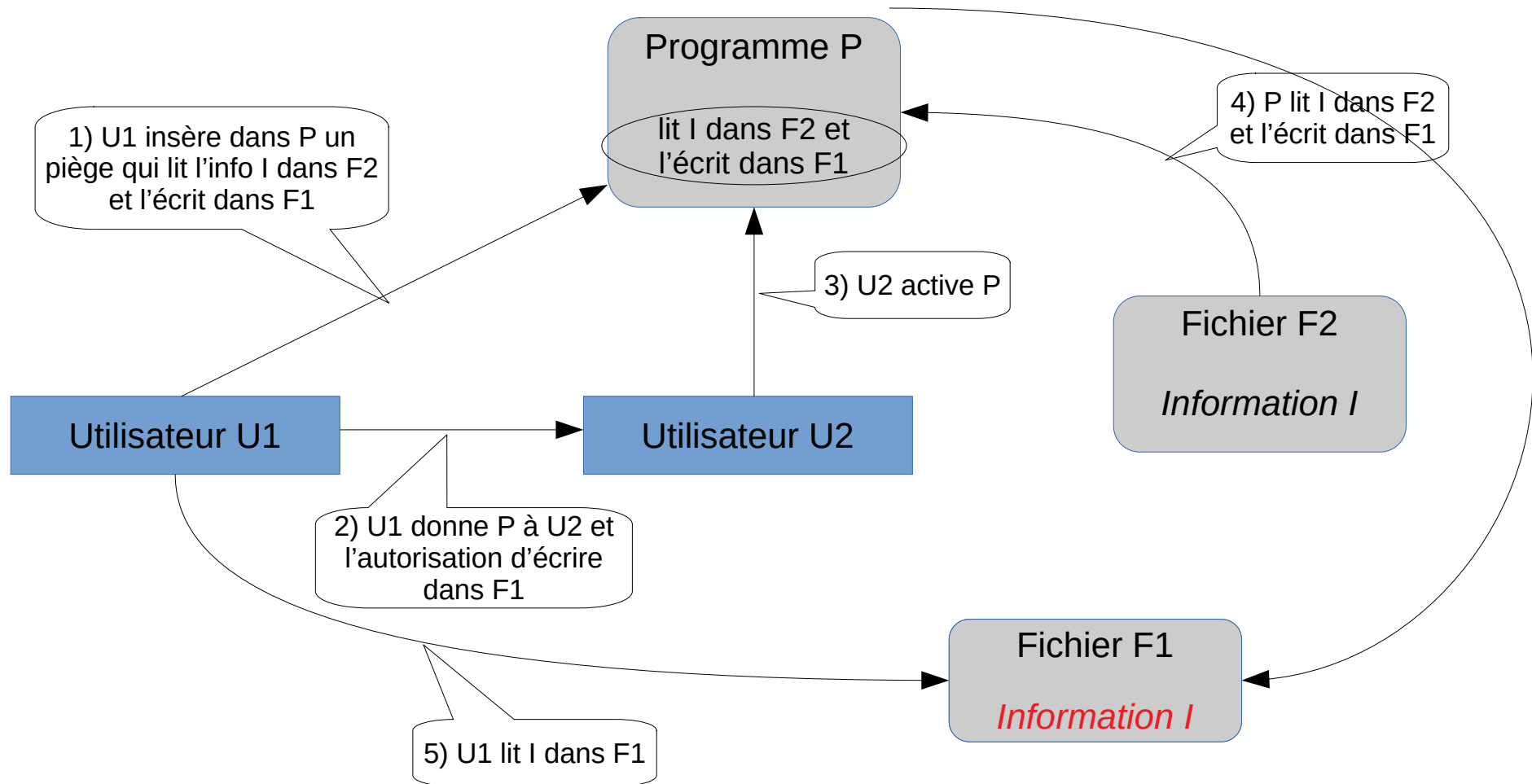
Les modèles DAC sont vulnérables aux chevaux de Troie

Soit :

- le programme P
- deux fichiers F1 et F2
- l'utilisateur U1
  - propriétaire de P et de F1
  - sans privilège sur F2
- l'utilisateur U2
  - propriétaire de F2
  - sans privilèges sur F1

Comment U1 peut-il obtenir une information I contenue dans F2 alors qu'il n'a pas le droit de le lire ?

# Chevaux de Troie (suite)



# Synthèse

Les modèles discrétionnaires sont

- simples à mettre en œuvre
- les plus implantés : Linux, SQL,...
- vulnérables aux chevaux de Troie
- difficiles (voire impossibles) à prouver

## c. MAC: contrôle d'accès obligatoire

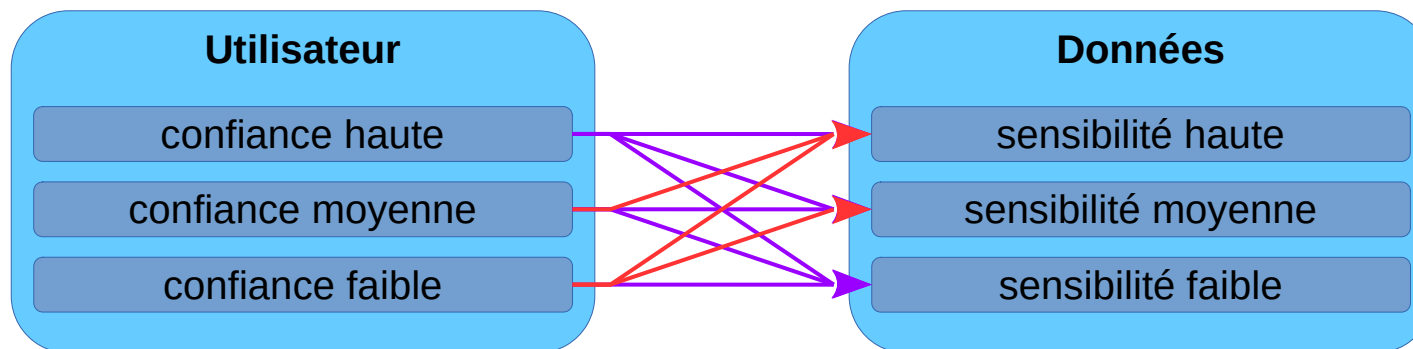
Également appelée sécurité multi-niveaux

Comment ça fonctionne ?

- les utilisateurs reçoivent un niveau d'*habilitation*
- les objets reçoivent un niveau de *classification*
- comparaison niveau de classification/d'habilitation

Niveaux dans un ensemble partiellement ordonné

- Public < Confidentiel < Secret < Très secret



# MAC: contrôle d'accès obligatoire (suite)

Pourquoi « obligatoire » ?

les niveaux de sécurité ne peuvent pas être modifiés

En résumé :

- issu du monde militaire
- les utilisateurs et les objets ont un niveau
- un utilisateur ne peut lire (resp. écrire) des objets de niveau égal ou inférieur (resp. supérieur)
- plus rigide que DAC
- mais plus sûr

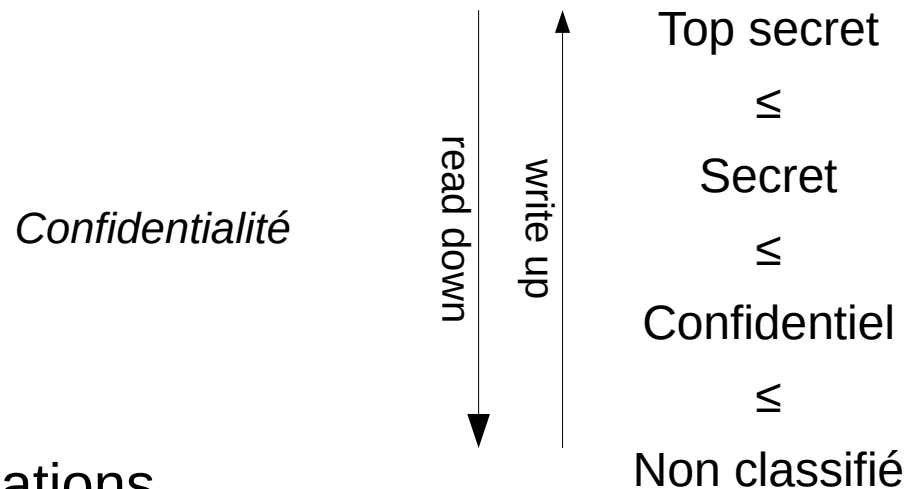
Exemples : Bell-La Padula, Biba

# Sécurité multi-niveaux

Une politique de sécurité multi-niveaux est une paire  $(L, \leq)$  telle que

- $L$  est un ensemble de niveaux de sécurité
- $\leq \subseteq L \times L$  est un ordre partiel sur  $L$

Exemple :



Habilitations et classifications

- habilitation : le niveau de sécurité d'un sujet
- classification : le niveau de sécurité d'un objet



# Exemple : Modèle Bell-La Padula

Élaboré en 1975 pour le Département de la Défense américaine

Permet de modéliser la **confidentialité** dans les systèmes multi-utilisateurs

Probablement le modèle le plus célèbre et le plus influent

- développé pour le gouvernement US de manière à prévenir les attaques dans les systèmes multi-utilisateurs
- base de plusieurs standards dont le Department of Defense TCSEC (*Trusted Computer System Evaluation Criteria*) ou « Orange book »
- sujet à controverse (définition d'un modèle de sécurité)

BLP modélise la confidentialité en combinant certains aspects de DAC et de MAC

- permissions d'accès sont définies en utilisant une matrice d'accès et des niveaux de sécurité
- sécurité multi-niveaux (*MLS - Multi-level security*): des politiques obligatoires interdisent que des informations puissent fuir vers un niveau inférieur

**Important** : modèle statique : les niveaux de sécurité ne peuvent pas changer

# Diagrammes des niveaux



— : frontières entre les niveaux (avec un ordre partiel  $\leq$ )

 : Sujet       : Objet

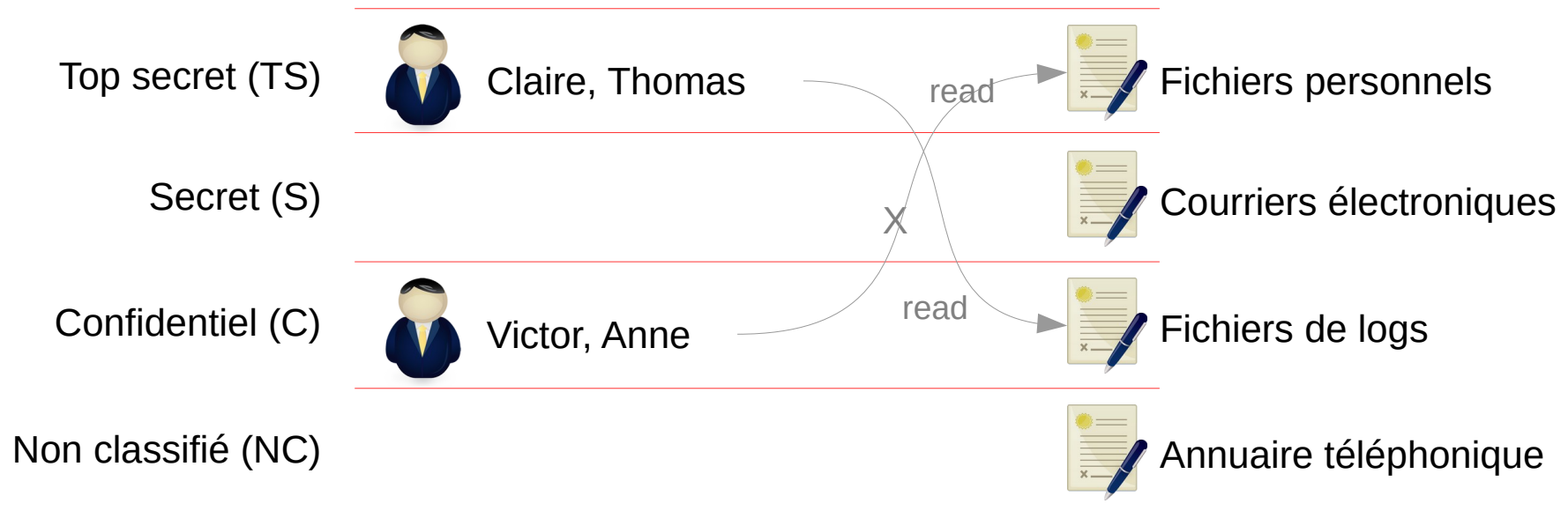
→ : opérations (*read*, *write*, *execute*, *append*)

Les diagrammes ne permettent pas de représenter fidèlement le treillis

- par exemple, lorsqu'un sujet  $s$  essaie d'accéder à un objet  $o$  sans niveau de sécurité (ni  $niveau(s) \leq niveau(o)$  et  $niveau(o) \leq niveau(s)$  sont vraies)

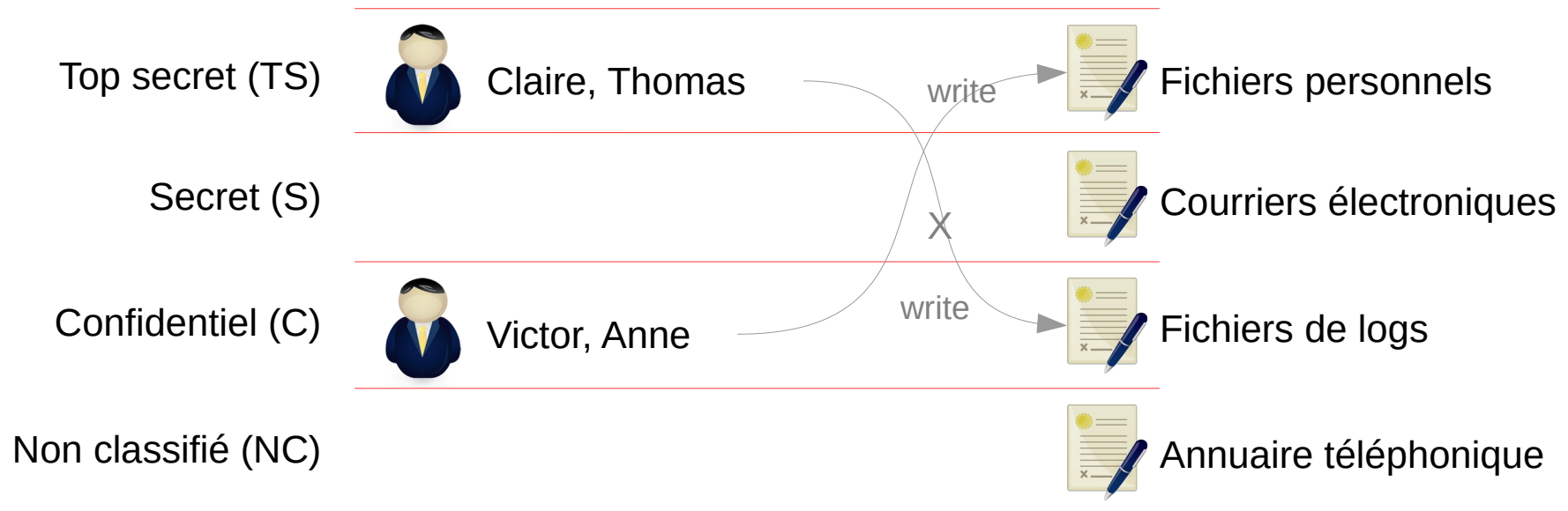
# BLP : version basique

**Première règle :**  $s$  ne peut lire  $o$  que si  $niveau_o \leq niveau_s$ , et  $s$  possède le droit *read* sur  $o$



# BLP : version basique (suite)

**Deuxième règle :**  $s$  peut écrire  $o$  si  $niveau_s \leq niveau_o$ , et  $s$  possède le droit *write* sur  $o$



# BLP : système sûr

Soit  $\Sigma$  un système possédant un état initial  $\sigma_0$  sûr, et soit  $T$  l'ensemble des transitions

- si chaque élément de  $T$  préserve la première règle et la deuxième règle
- alors chaque état  $\sigma_i$  ( $i \geq 0$ ) est sûr

On va ajouter des catégories aux niveaux de classification

- les catégories dépendent du contexte de l'application à sécuriser
- par exemple, dans un domaine militaire, on peut prendre les catégories suivantes

- Armée de terre
- Marine
- Armée de l'air
- Nucléaire

Thomas peut avoir le  
niveau Top secret pour  
la catégorie Marine :  
(Top secret, {Marine})

# BLP : treillis et niveaux de sécurité

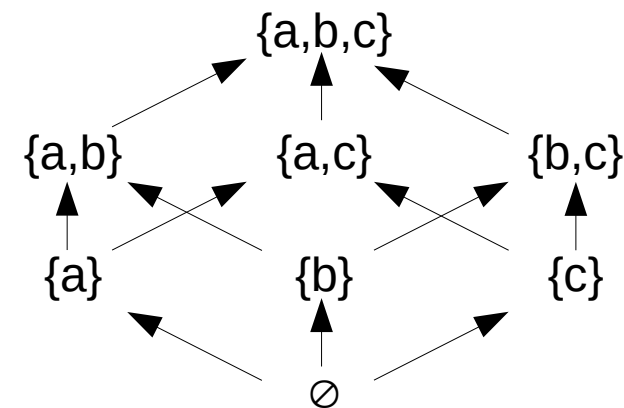
Un treillis  $(L, \leq)$  est constitué d'un ensemble  $L$  et d'un ordre partiel  $\leq$ , tels que pour chaque paire d'éléments  $a, b \in L$ , il existe une borne supérieure minimale  $u \in L$  et une borne inférieure maximale  $v \in L$

On dit que " $a$  est dominé par  $b$ " ou " $b$  domine  $a$ " si  $a \leq b$

Le niveau de sécurité dominé par tous les autres est appelé *Bas du Système*, alors que le niveau qui domine tous les autres s'appelle *Haut du Système*

Exemple :

- $L$  est l'ensemble des parties de  $\{a,b,c\}$
- la relation d'ordre partiel est l'inclusion
- le haut du système est  $\{a,b,c\}$  et le bas du système est  $\emptyset$



# BLP : niveaux de sécurité

Le niveau de sécurité  $(L, C)$  **domine** le niveau de sécurité  $(L', C')$  ssi

$$L' \leq L \text{ et } C' \subseteq C$$

## Exemple :

- Bob est habilité (Secret, {Armée de Terre, Marine})
- le document A est classifié (Confidentiel, {Armée de Terre})
- le document B est classifié (Secret, {Marine, Armée de l'air})
- le document C est classifié (Secret, {Marine})
- Bob ? Doc A
- Bob ? Doc B
- Bob ? Doc C

# BLP : niveaux de sécurité

Le niveau de sécurité  $(L, C)$  **domine** le niveau de sécurité  $(L', C')$  ssi

$$L' \leq L \text{ et } C' \subseteq C$$

## Exemple :

- Bob est habilité (Secret, {Armée de Terre, Marine})
- le document A est classifié (Confidentiel, {Armée de Terre})
- le document B est classifié (Secret, {Marine, Armée de l'air})
- le document C est classifié (Secret, {Marine})
- Bob domine Doc A: confidentiel  $\leq$  secret et {Armée de Terre}  $\subseteq$  {Armée de Terre, Marine}
- Bob ? Doc B
- Bob ? Doc C



# BLP : niveaux de sécurité

Le niveau de sécurité  $(L, C)$  **domine** le niveau de sécurité  $(L', C')$  ssi

$$L' \leq L \text{ et } C' \subseteq C$$

## Exemple :

- Bob est habilité (Secret, {Armée de Terre, Marine})
- le document A est classifié (Confidentiel, {Armée de Terre})
- le document B est classifié (Secret, {Marine, Armée de l'air})
- le document C est classifié (Secret, {Marine})
- Bob domine Doc A: confidentiel  $\leq$  secret et {Armée de Terre}  $\subseteq$  {Armée de Terre, Marine}
- Bob ne domine pas Doc B: {Armée de l'air, Marine}  $\not\subseteq$  {Armée de Terre, Marine}
- Bob ? Doc C

# BLP : niveaux de sécurité

Le niveau de sécurité  $(L, C)$  **domine** le niveau de sécurité  $(L', C')$  ssi

$$L' \leq L \text{ et } C' \subseteq C$$

## Exemple :

- Bob est habilité (Secret, {Armée de Terre, Marine})
- le document A est classifié (Confidentiel, {Armée de Terre})
- le document B est classifié (Secret, {Marine, Armée de l'air})
- le document C est classifié (Secret, {Marine})
- Bob domine Doc A: confidentiel  $\leq$  secret et {Armée de Terre}  $\subseteq$  {Armée de Terre, Marine}
- Bob ne domine pas Doc B: {Armée de l'air, Marine}  $\not\subseteq$  {Armée de Terre, Marine}
- Bob domine Doc C: secret  $\leq$  secret et {Marine}  $\subseteq$  {Armée de Terre, Marine}

# BLP : niveaux de sécurité (suite)

## Exercice :

Soient les classes d'accès suivantes

- $c1 = (TS, \{Armée, Nucléaire\})$
- $c2 = (TS, \{Nucléaire\})$
- $c3 = (C, \{Armée\})$

Donner les relations entre

- $c1$  et  $c2$  ?
- $c2$  et  $c3$  ?
- $c1$  et  $c3$  ?

# BLP : axiomes

Soient  $S$  un ensemble de sujets,  $O$  un ensemble d'objets

Soit  $A = \{ execute, read, append, write \}$  les opérations d'accès

Soit  $L$  un ensemble de niveaux de sécurité muni d'un ordre partiel  $\leq$

Un état est un triplet  $(b, M, f) \in B \times M \times F$ , où

- $B = P(S \times O \times A)$  est l'ensemble des opérations d'accès en cours (matrice)
- $M$  est l'ensemble des matrices de permissions d'accès  $M = (M_{so})_{s \in S, o \in O}$
- $F \subset L^S \times L^S \times L^O$  est l'ensemble des attributions de niveaux de sécurité. Un élément  $f$  de  $F$  est un triplet  $(f_s, f_c, f_o)$  tel que
  - $f_s : S \rightarrow L$  donne le niveau maximal qu'un sujet peut atteindre
  - $f_c : S \rightarrow L$  donne le niveau courant de chaque sujet
  - $f_o : O \rightarrow L$  donne la classification de chaque objet

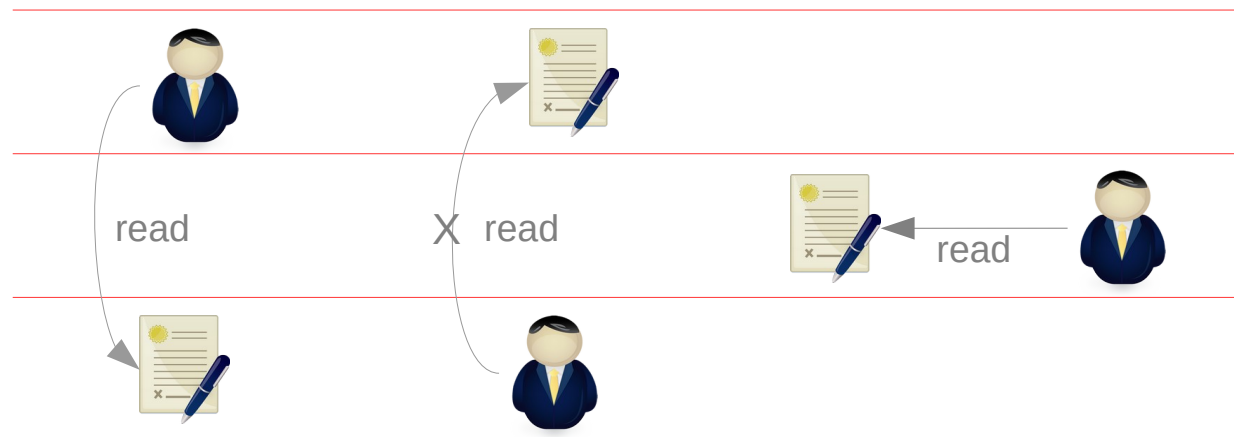
Remarque : le niveau maximal d'un sujet domine toujours le niveau courant

# BLP : propriétés de simple sécurité (ss)

Un état  $(b, M, f)$  satisfait la **propriété ss** si

- pour chaque élément  $(s, o, a) \in b$  pour lequel  $a$  est *read* ou *write*
- le niveau maximal de sécurité du sujet  $s$  domine la classification de l'objet  $o$ , c'est-à-dire  $f_o(o) \leq f_s(s)$
- c'est-à-dire:  $\forall s \in S, o \in O: (read \in M[s, o] \vee write \in M[s, o]) \Rightarrow f_o(o) \leq f_s(s)$
- un sujet peut lire les objets seulement si leur niveau de sécurité est dominé par celui du sujet

NRU : no read up



# BLP : propriétés de sécurité (suite)

## Exemple

- Bob est habilité (Secret, {Armée de Terre, Marine})
- le document A est classifié (Confidentiel, {Armée de Terre})
- le document B est classifié (Secret, {Marine, Armée de l'air})
- le document C est classifié (Secret, {Marine})
- Bob peut lire Doc A et Doc C, mais pas Doc B

# BLP : propriétés de sécurité (suite)

## Exemple

- Bob est habilité (Secret, {Armée de Terre, Marine})
- le document A est classifié (Confidentiel, {Armée de Terre})
- le document B est classifié (Secret, {Marine, Armée de l'air})
- le document C est classifié (Secret, {Marine})
- Bob peut lire Doc A et Doc C, mais pas Doc B

## Exercice :

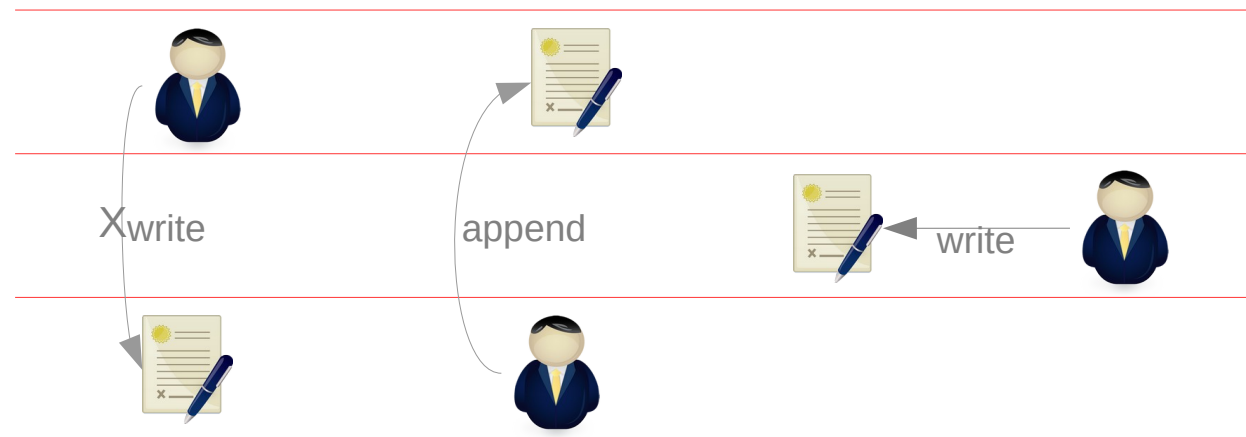
- Bob est habilité (Confidentiel, {Université})
- peut-il lire le document suivant ?
  - doc A est classifié (Confidentiel, {Étudiant})

# BLP : propriété « étoile » (\*-property)

Un état  $(b, M, f)$  satisfait la **propriété-\*** si

- pour chaque élément  $(s, o, a) \in b$  pour lequel  $a$  est *append* ou *write*
- le niveau courant du sujet  $s$  est dominé par la classification de l'objet  $o$ , c'est-à-dire  $f_c(s) \leq f_o(o)$  pour *append*,  $f_c(s) = f_o(o)$  pour *write*
- un sujet  $s$  ne peut écrire (écrire est utilisé pour signifier écriture seule ou ajout) dans un objet  $o$  que si la classification de l'objet domine l'autorisation du sujet

NWD : no write down





# BLP : propriétés de sécurité (suite)

Supposons

- un colonel possède le niveau (Secret, {Nucléaire, Armée})
- un major possède le niveau (Secret, {Armée})
- le colonel doit envoyer un message au major
  - impossible à cause de la propriété-\*
  - (Secret, {Nucléaire, Armée}) domine (Secret, {Armée})
- comment faire ?

# BLP : propriétés de sécurité (suite)

Introduction de 2 niveaux

- niveau **maximum** de sécurité
- niveau **courant** de sécurité

Rappelez-vous la définition d'un état de sécurité :

- $F \subset L^S \times L^S \times L^O$  est l'ensemble des attributions de niveaux de sécurité.  
Un élément  $f$  de  $F$  est un triplet  $(f_S, f_C, f_O)$  tel que
  - $f_S : S \rightarrow L$  donne le niveau maximal qu'un sujet peut atteindre
  - $f_C : S \rightarrow L$  donne le niveau courant de chaque sujet
  - $f_O : O \rightarrow L$  donne la classification de chaque objet

Un sujet peut changer son niveau courant, mais le niveau maximal d'un sujet domine toujours le niveau courant

Exemple : le colonel peut modifier son niveau courant en (Secret, {Armée})

# BLP : propriété de sécurité discrétionnaire (sd)

Un état  $(b, M, f)$  satisfait la propriété **sd** si, pour chaque élément  $(s, o, a) \in b$ , nous avons  $a \in M_{so}$

Discrétionnaire veut dire que le contrôle d'accès peut être manipulé directement par le sujet

Par opposition, les **propriétés ss** et **\*** implantent un contrôle d'accès obligatoire, imposé par les niveaux de sécurité assignés

# BLP : sécurité d'un système

## Sûreté d'un état

- un état est sûr s'il satisfait aux trois propriétés de sécurité décrites plus haut (ss, \* et sd)

## Sûreté d'un système

- un système est sûr si tous les états atteignables sont sûrs
- soit  $\Sigma$  un système possédant un état initial  $\sigma_0$  sûr, et soit  $T$  l'ensemble des transitions
- si chaque élément de  $T$  préserve la propriété ss et la propriété-\*, alors chaque état  $\sigma_i$  ( $i \geq 0$ ) est sûr

# Exemple

$S = \{ \text{Alice, Bob, Carl} \}$

$O = \{ \text{alice.doc, winword.exe, time.log, jokes.txt} \}$

$L = \{1, 2, 3, 5, 7, 15, 21, 35, 70, 105, 210\}$  et l'ordre partiel est la divisibilité (1 divise tout le monde (haut du système) et 210 ne divise personne (bas du système))

Voici un exemple d'état  $(b_1, M_1, f_1)$

- $b_1 = \{(\text{Alice, alice.doc, read}), (\text{Bob, time.log, read}), (\text{Carl, winword.exe, execute})\}$
- $M_1$  et  $f_1$  sont représentés dans le tableau ci-dessous :

"s(m/c)" : pour chaque sujet s, le niveau maximal (m) et courant (c) d'habilitation

	Alice (3/21)	Bob (7/105)	Carl (5/70)
alice.doc (21)	r, w	r	
winword.exe (70)			x
time.log (7)		r	a
jokes.txt (105)	r	w	r

"o(c)": pour chaque objet o, la classification (c) de cet objet

# Exemple (suite)

Voici un autre exemple d'état ( $b_2$ ,  $M_2$ ,  $f_2$ )

- $b_2 = \{(Alice, windord.exe, read), (Bob, jokes.txt, write), (Carl, time.log, append)\}$
- $M_2$  et  $f_2$  sont représentés dans le tableau ci-dessous :

	Alice (3/21)	Bob (7/35)	Carl (5/70)
alice.doc (21)	r, w	r	
winword.exe (2)	r		x
time.log (35)		r	
jokes.txt (105)	r	w	r

# Exemple (suite)

Est-ce que nos exemples d'état sont sécuritaires ?

- $(b_1, M_1, f_1)$  est-il sécuritaire ?
- $b_1 = \{(Alice, alice.doc, read), (Bob, time.log, read), (Carl, winword.exe, execute)\}$

	Alice (3/21)	Bob (7/105)	Carl (5/70)
alice.doc (21)	r, w	r	
winword.exe (2)			x
time.log (35)		r	a
jokes.txt (105)	r	w	r

# Exemple (suite)

- et  $(b_2, M_2, f_2)$  ?
- $b_2 = \{(Alice, \text{windord.exe}, \text{read}), (Bob, \text{jokes.txt}, \text{write}), (Carl, \text{time.log}, \text{append})\}$

	Alice (3/21)	Bob (7/35)	Carl (5/70)
alice.doc (21)	r, w	r	
winword.exe (2)	r		x
time.log (35)		r	
jokes.txt (105)	r	w	r

- l'accès d'Alice au fichier winword.exe contrevient à la propriété ss
- l'accès de Bob au fichier jokes.txt contrevient à la propriété \*
- l'accès de Carl au fichier time.log contrevient à la propriété sd



# Comparaison HRU/BLP

## Le modèle HRU

- se concentre sur la spécification du système (les propriétés de sécurité ne font pas partie du modèle)
- permet la modification des ensembles de sujets et d'objets
- fournit un langage simple (avec une sémantique opérationnelle) pour spécifier les requêtes

## Le modèle BLP

- définit trois propriétés de sécurité qui font partie intégrante du modèle
- ne permet pas la modification des ensembles de sujets et d'objets
- ne fournit pas de langage pour définir les requêtes ou spécifier les règles
- contient des canaux cachés

# Synthèse

McLean souligne le problème de la situation suivante :

- on assigne à tous les sujets le niveau minimal (bas du système)
- on assigne à tous les objets le niveau minimal
- on entre à chaque cellule de la matrice tous les droits d'accès possibles
- on peut prouver que la transition est sûre

***Mais peut-on appeler cet état sûr ?***

Débat sur l'utilité du modèle BLP :

- pour : si les besoins de l'utilisateur requièrent un tel état, alors ça doit faire partie de sa politique de sécurité
- contre : intuitivement, un système qui peut être amené à ce niveau n'est pas sûr. BLP n'est pas suffisant et doit être amélioré

Malgré ce débat, BLP reste un modèle de sécurité tout à fait significatif

# Synthèse

Lorsqu'un nouveau modèle de sécurité est proposé, on ne peut s'empêcher de le comparer à BLP

Il est cohérent et simple, alors que la réalité est complexe et pas toujours cohérente

BLP est critiqué pour les raisons suivantes :

- ne s'occupe que de confidentialité, mais pas d'intégrité
- c'est normal pour un modèle de ne pas tout couvrir...
- n'adresse pas la gestion du contrôle d'accès
- conçu pour les systèmes à droits d'accès statiques (tranquilles)
  - le niveau de confidentialité d'un objet actif ne peut pas être changé
- permet l'utilisation d'un canal caché
  - canal permettant à un sujet d'envoyer un bit à un sujet de moindre niveau, ce qui contrevient à la propriété \*

# Exemple : Biba

Biba (K.J Biba, 1977) : traite de la protection de l'intégrité

Appelé aussi BLP à l'envers (*BLP upside down*)

Biba a constaté que confidentialité et intégrité sont des concepts duaux

- confidentialité : contrainte sur qui a le droit de lire la donnée
- intégrité : contrainte sur qui a le droit de l'écrire ou de la modifier

Dans BLP :

- l'information ne peut pas circuler vers les niveaux inférieurs pour éviter la fuite de données sensibles

Dans Biba :

- Inverse : les informations ne peuvent pas migrer vers des niveaux hauts d'intégrité
- ainsi, une donnée piégée par un cheval de Troie ne peut pas contaminer des données saines d'un niveau d'intégrité supérieur

# Biba (suite)

Biba n'empêche pas un cheval de Troie, mais confine son effet

- un cheval de Troie ne pourra pas attaquer des données ayant un niveau d'intégrité supérieur où le cheval de Troie a été installé (*no-write-up*)
- exemple proche : sandboxing de Java
- l'attaquant ne peut pas non plus piloter le cheval de Troie à cause du *no-read-down*

Biba propose une classe de modèles d'intégrité avec les règles suivantes :

- modèle d'intégrité obligatoire : *no-read-down* et *no-write-up*
- *no-read-down* dynamique (« *subject low watermark property* »)
  - un sujet peut lire des objets du niveau inférieur, mais il doit d'abord abaisser son niveau d'intégrité au niveau de celui de l'objet
- *no-write-up* dynamique (« *object low watermark property* »)
  - abaisse le niveau de l'objet à celui du sujet qui veut réaliser l'écriture

On peut combiner BLP et Biba pour modéliser à la fois confidentialité et intégrité

# Biba : modèle d'intégrité

Le modèle Biba associe un niveau d'intégrité aux sujets et aux objets.

Les niveaux sont la base pour exprimer des politiques d'intégrité qui empêchent des objets propres d'être contaminés par une information souillée.

**Attention** : les niveaux d'intégrité **ne sont pas** des niveaux de sécurité.

L'intégrité est modélisée avec modèle états/transitions (comme BLP)

- un treillis  $(L, \leq)$  de niveaux d'intégrité
- $f_s : S \rightarrow L$  et  $f_o : O \rightarrow L$  allouent des niveaux d'intégrité aux sujets et aux objets
- l'information ne peut que descendre dans le treillis des niveaux d'intégrité

Contrairement à BLP (politique unique)

- plusieurs politiques possibles (éventuellement mutuellement incompatibles)
  - niveaux d'intégrité statiques
  - niveaux d'intégrité dynamiques
  - alternatives de politiques pour l'invocation

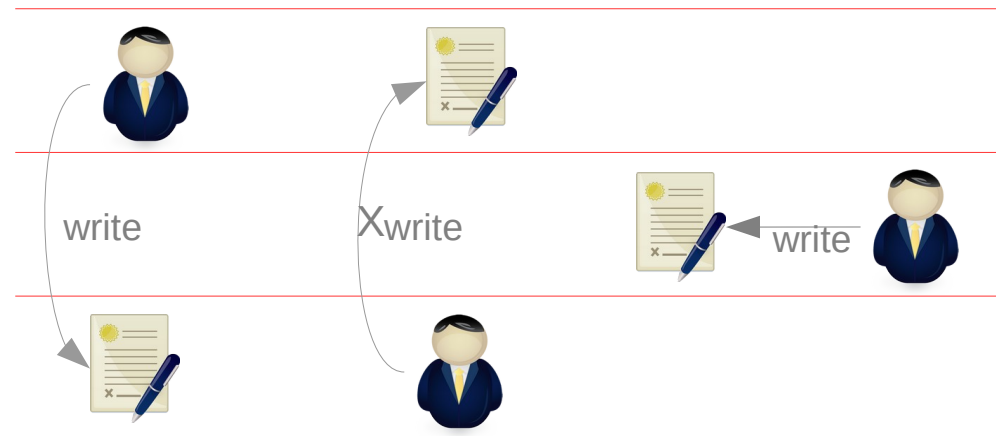
# Biba : niveaux d'intégrité statiques

Politiques dont les niveaux d'intégrité ne peuvent être modifiés (équivalent à la règle de tranquillité de BLP)

Deux propriétés (duales de BLP) :

## Propriété de simple intégrité

- $s$  peut modifier  $o$  ssi  $f_s(s) \geq f_o(o)$
- *no-write-up*



## Propriété \* d'intégrité

- si  $s$  peut lire  $o_1$ , alors  $s$  peut écrire un objet  $o_2$  ssi  $f_o(o_2) \leq f_o(o_1)$

Ces 2 politiques empêchent des objets propres d'être contaminés par une information *souillée*

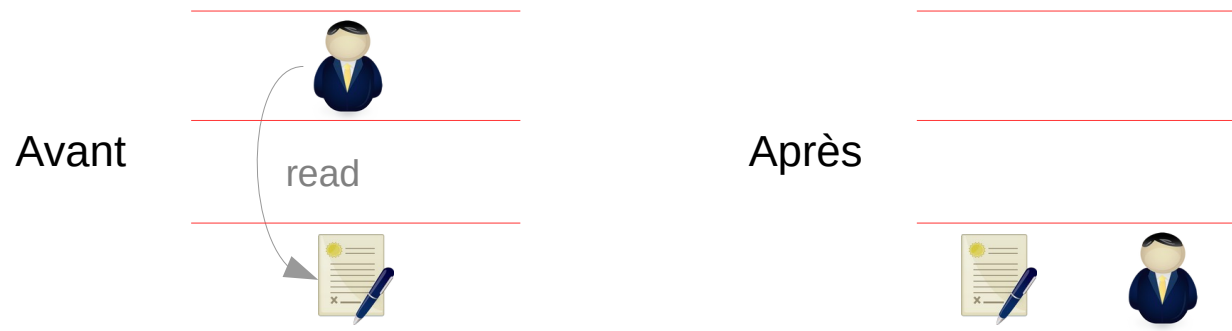
# Biba : niveaux d'intégrité dynamiques

Utilisation d'une politique de nivellement par le bas (*low Water Mark*)

- ajustement du niveau d'intégrité d'une entité en contact avec une donnée bas niveau

Nivellement par le bas du sujet

- on relâche le *no-read-down*
- on permet à un sujet de lire à un niveau inférieur, mais auparavant, on abaisse son niveau d'intégrité au niveau de l'objet qu'il veut lire
- un sujet  $s$  peut lire un objet  $o$  à n'importe quel niveau d'intégrité. Le nouveau niveau d'intégrité de  $s$  devient  $\inf(f_s(s), f_o(o))$  où  $f_s(s)$  et  $f_o(o)$  représentent les niveaux d'intégrité avant l'opération (où  $\inf(x,y)$  est la borne inférieure maximale entre  $x$  et  $y$ )

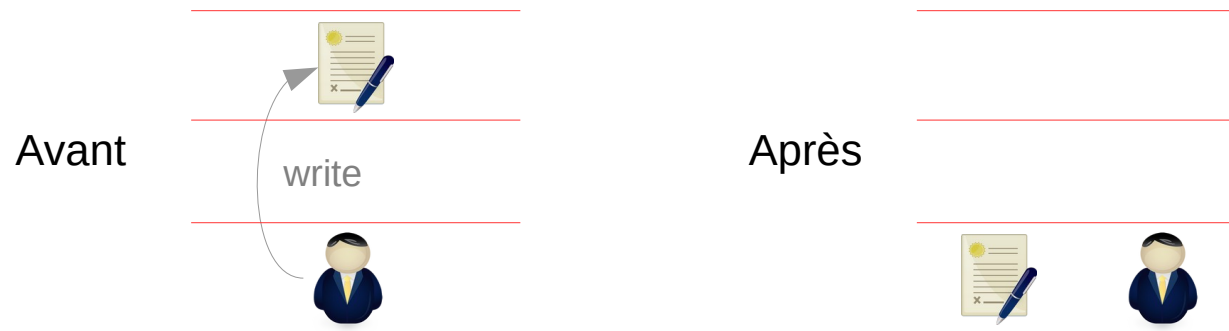




# Biba : niveaux d'intégrité dynamiques (suite)

Nivellement par le bas de l'objet

- on relâche le *no-write-up*
- on abaisse le niveau de l'objet à celui du sujet qui va exécuter la modification
- un sujet  $s$  peut modifier un objet  $o$  à n'importe quel niveau d'intégrité. Le nouveau niveau d'intégrité de  $o$  devient  $\inf(f_s(s), f_o(o))$  où  $f_s(s)$  et  $f_o(o)$  représentent les niveaux d'intégrité avant l'opération



Plus le système évolue dans le temps, plus les sujets et les objets se souillent

# Biba : alternatives de politiques d'invocation

Biba peut être étendu avec une opération *invoke* afin qu'un sujet puisse invoquer un autre sujet (par ex. un programme) pour avoir accès à un objet

## Propriété d'invocation

- on veut s'assurer qu'une invocation ne détourne pas les politiques d'intégrité obligatoires
- $s_1$  peut invoquer  $s_2$  ssi  $f_s(s_2) \leq f_s(s_1)$
- cela veut dire qu'un outil  $s_2$  ne peut échapper aux règles d'accès globales

## Propriété d'anneau (*ring property*)

- un sujet  $s_1$  peut lire les objets à tous les niveaux d'intégrité. Pour modifier un objet  $o$ , il faut que  $f_o(o) \leq f_s(s_1)$  ou qu'il invoque un sujet  $s_2$  tel que  $f_s(s_1) \leq f_s(s_2)$  pour le faire
- cela correspond à l'utilisation d'un outil propre pour permettre à un sujet moins propre d'accéder à un objet propre

Les deux alternatives ne sont pas consistantes : il faut choisir la plus appropriée en fonction de l'application visée

# Biba : conclusion

La dualité avec BLP fait que Biba souffre du problème soulevé par McLean

- concerne le principe de tranquillité (niveaux statiques)

Pour éviter ce problème, une solution adoptée dans les systèmes assurant des propriétés d'intégrité sur la base du modèle de Biba, est l'utilisation d'une politique de type Low Water Mark dans laquelle le label d'intégrité d'un objet est affecté au niveau le plus bas en lecture par le processus qui l'a créé.

# Role-Based Access Control

# Rappels

Quoi ?	Objectifs
	Modèle de politique
	Modèle de contrôle
	Modèle d'implantation
Comment ?	Implantation

# Rappels (suite)

## DAC

Quoi ?	Autorisations à la discrétion du propriétaire
Comment ?	ACL, capacités

# Rappels (suite)

## MAC

Quoi ?	Pas de fuite d'information
	Treillis (BLP)
	Noyau de sécurité
Comment ?	Étiquettes (labels) de sécurité

# RBAC : motivations

Problème important du modèle IBAC : complexité de l'expression et de l'administration de la sécurité

- complexité pour énumérer les autorisations pour chaque sujet, action et objet pour un grand nombre de sujets ou d'objets
- complexité de la mise à jour de la politique d'autorisation pour définir les nouvelles permissions associées à un sujet ou objet nouvellement créé
- de plus, si la population est dynamique, le nombre d'opérations à exécuter pour accorder ou révoquer les autorisations peut être difficile à administrer

Les utilisateurs finaux sont rarement propriétaires des données auxquelles ils accèdent

- le contrôle est souvent basé sur les fonctions d'un utilisateur plutôt que sur la possession des données

RBAC a été proposé comme une approche alternative à DAC et MAC pour supporter l'administration et le support du contrôle d'accès en se basant sur les fonctions



# RBAC : exemples de fonctions

## Banques

- responsable
- chargés de clientèle
- responsables des prêts

## Hôpitaux

- médecins
- infirmières
- sage-femmes
- administratifs

## Université

- enseignants
- non enseignants
- étudiants

# RBAC : principes

Moindre privilège

Séparation des devoirs

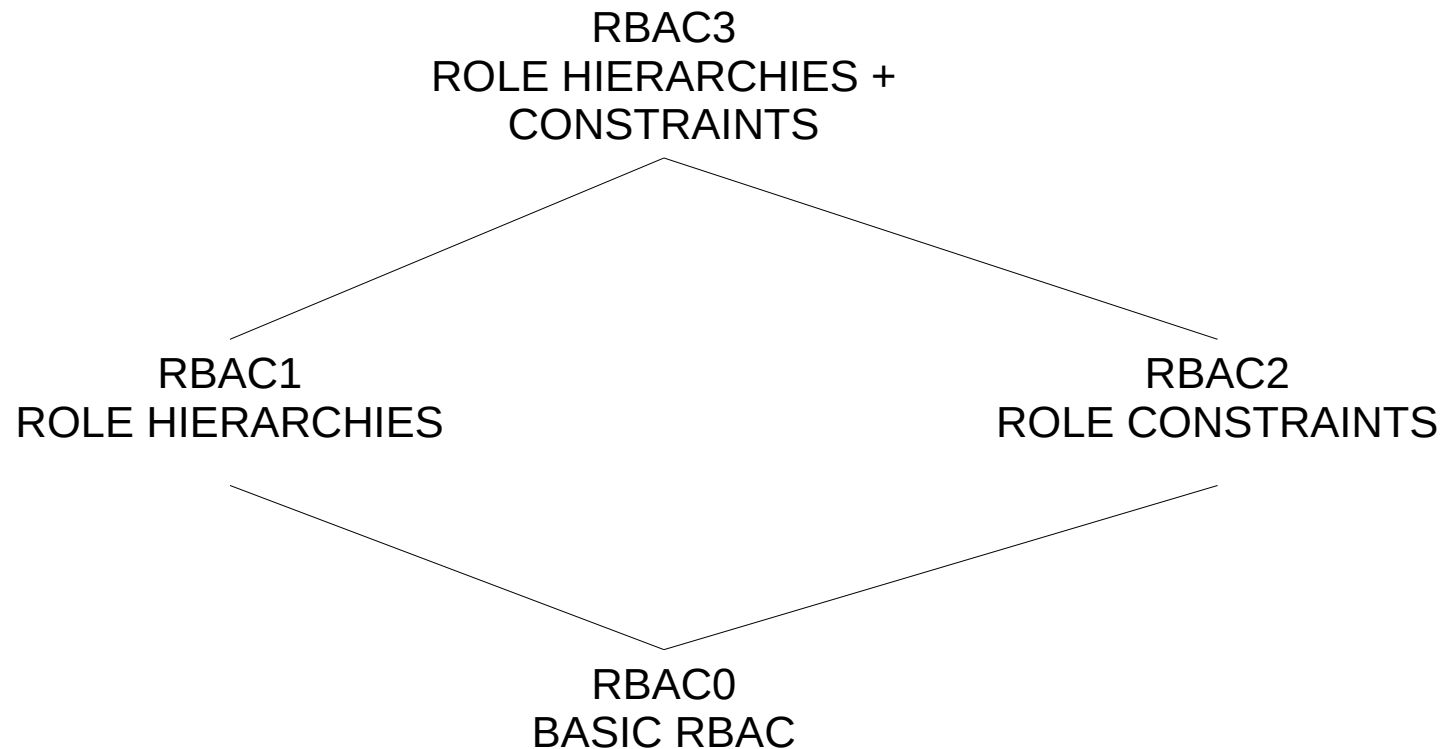
Séparation de l'administration et de l'accès

Opérations abstraites

Neutre vis-à-vis des politiques

- peut-être configuré pour faire du MAC
- peut-être configuré pour faire du DAC

# RBAC : un ensemble de modèles



# RBAC : concepts

RBAC propose de structurer l'expression de la politique d'autorisation autour du concept de rôle

Un **rôle** est un concept organisationnel qui représente les fonctions qu'un utilisateur peut jouer dans une organisation

- les autorisations sont directement associées aux rôles

Des rôles sont affectés aux utilisateurs

- ceux-ci sont alors autorisés à jouer un certain nombre de rôles pour acquérir les autorisations associées au rôle

Bénéfices

- comme un rôle représente une fonction organisationnelle, RBAC supporte plus facilement les politiques d'une organisation
- accorder ou révoquer des autorisations est plus simple
- les modèles RBAC sont neutres vis-à-vis des politiques

# RBAC : concepts (suite)

Un autre concept introduit par le modèle RBAC est celui de ***session***

Pour pouvoir réaliser une action sur un objet

- un utilisateur doit d'abord créer une session
- dans cette session, il doit ensuite activer un *rôle* qui a reçu l'autorisation de réaliser cette action sur cet objet

Si le rôle existe et si cet utilisateur a été affecté à ce rôle

- alors il aura la permission de réaliser cette action sur cet objet une fois ce rôle activé

# RBAC

Les vendeurs de SGBD ont été les premiers à voir l'intérêt de RBAC

Il existe un certain consensus sur un modèle RBAC standard

Le modèle proposé par le NIST [Sandhu,Ferraiolo,Kuhn00] est la première étape vers la définition d'un standard

Une proposition plus récente émane de l'ANSI

- *American national standard for information technology – role based access control. ANSI INCITS 359-2004, February 2004*

Le modèle NIST

- 3 niveaux croissants de fonctionnalités
  - *Core RBAC*
  - *Hierarchical RBAC*
  - *Constrained RBAC*

# RBAC : définitions

**Utilisateur** : entité active du système (être humain, machine, processus,...)

**Rôle** : fonction dans le contexte de l'organisation qui possède une sémantique quant à l'autorité et la responsabilité

**Permission** :

- mode d'accès qui peut être appliqué sur les objets du système
- les objets et les modes d'accès dépendent du domaine (tables, n-uplets et insert/update/delete pour un SGBD)

**Session** :

- lie un utilisateur à un ensemble de rôles activés
- à un instant donné, un utilisateur peut avoir plusieurs sessions actives

# RBAC : permissions

## Permissions de **base**

- read, write, append, execute

## Permissions **abstraites**

- crédite, débite,...

## Permissions **système**

- audit

## Permissions sur les **objets**

- read, write, append, execute, crédite, débite,...



# RBAC : permissions (suite)

Les permissions sont positives

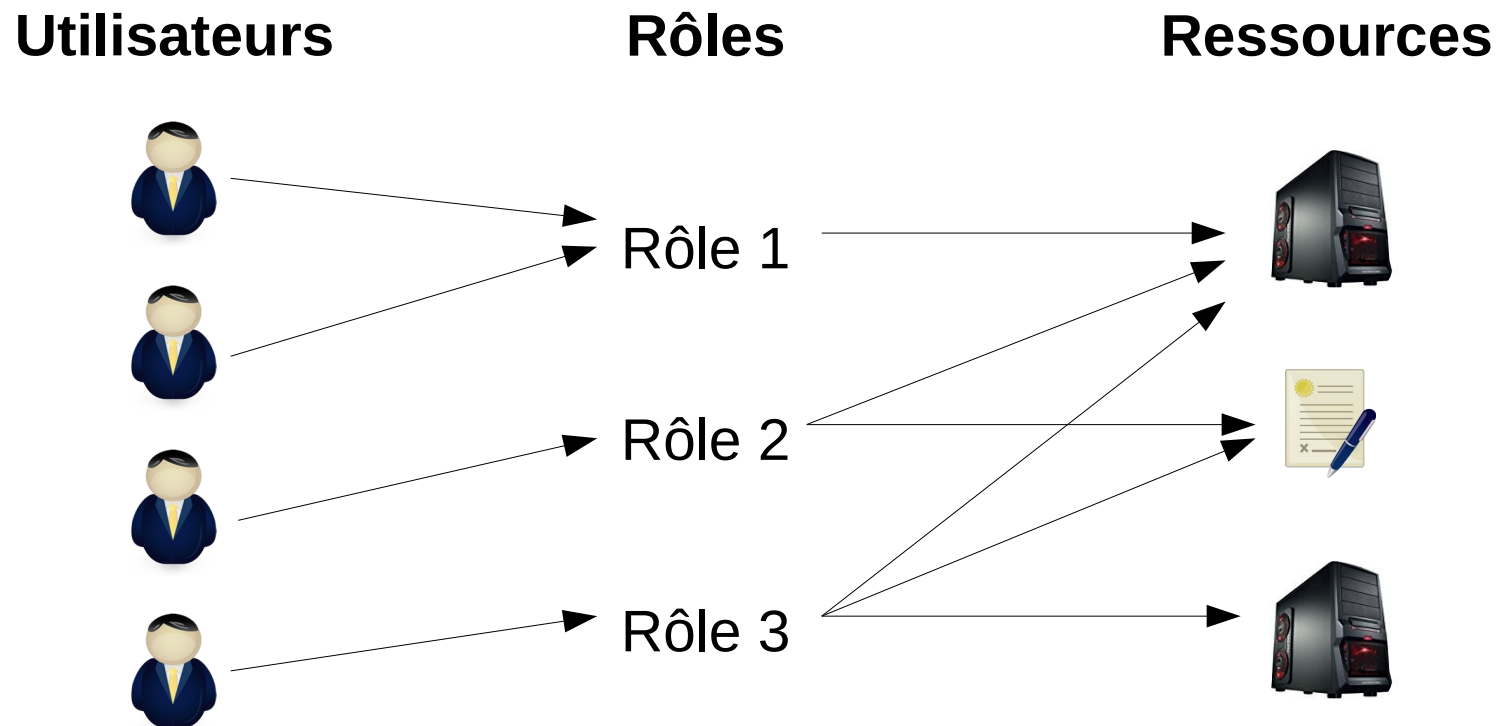
Pas de permissions négatives

- on peut gérer ce type de permissions avec les contraintes

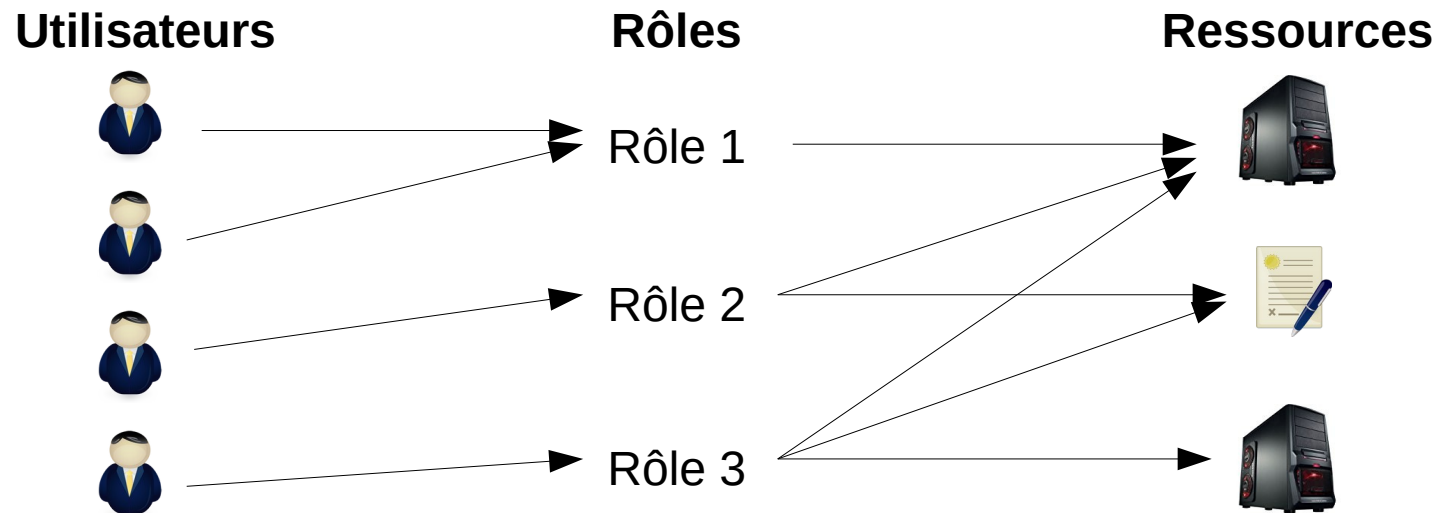
Pas de devoirs ou d'obligations

- en dehors du cadre du contrôle d'accès

# RBAC : basé sur les rôles



# RBAC : basé sur les rôles (suite)



## Abstraction

- nombreux sujets (ou objets) qui ont des attributs identiques et les politiques sont basées sur ces attributs

## Hiérarchie

- hiérarchies fonctionnelles/organisationnelles déterminent des droits

# RBAC : basé sur les rôles (suite)

Un rôle regroupe

- une collection d'utilisateurs
- une collection de permissions

Les collections peuvent varier au cours du temps

- mais un rôle possède une sémantique qui va au-delà des utilisateurs et des permissions à un instant donné

Rôle vs. groupes

- un groupe est une collection d'utilisateurs
- un rôle est une collection d'utilisateurs **et** une collection de permissions

# Core RBAC

## **Rôle actif** (*Active Role*)

- $AR(s : \text{sujet}) = \text{le rôle actif pour le sujet } s$

## Rôles autorisés (*Authorized roles*)

- $RA(s : \text{sujet}) = \{\text{rôles autorisés pour le sujet } s\}$

## **Transactions autorisées** (*Authorized transactions*)

- $TA(r : \text{rôle}) = \{\text{transactions autorisées pour le rôle } r\}$

## **Prédicat exec** (*predicate exec*)

- $\text{exec}(s : \text{sujet}, t : \text{transaction}) = \text{true}$  ssi  $s$  peut exécuter  $t$

## **Session**

- lie un utilisateur à un ensemble de rôles activés

# Core RBAC

Modèle RBAC généralisé (Ferraiolo, Kuhn 1992)

## **Attribution des rôles (statique)**

- un sujet peut exécuter des transactions seulement si le sujet possède le rôle

## **Autorisation d'un rôle (dynamique)**

- le rôle actif d'un sujet doit être autorisé pour le sujet

## **Autorisation de transaction**

- un sujet peut exécuter une transaction seulement si la transaction est autorisée pour le rôle actif du sujet

# Core RBAC: ensembles, fonctions, relations

## USERS, ROLES, OPS et OBS

$UA \subseteq \text{USERS} \times \text{ROLES}$  : relation permettant d'assigner un rôle à un utilisateur

*assigned\_users*:  $(r: \text{ROLES}) \rightarrow 2^{\text{USERS}}$  : mapping d'un rôle vers un ensemble d'utilisateurs

- formellement,  $\text{assigned\_users}(r) = \{u \in \text{USERS} \mid (u, r) \in UA\}$

$\text{PRMS} = 2^{(\text{OPS} \times \text{OBS})}$  : l'ensemble des permissions

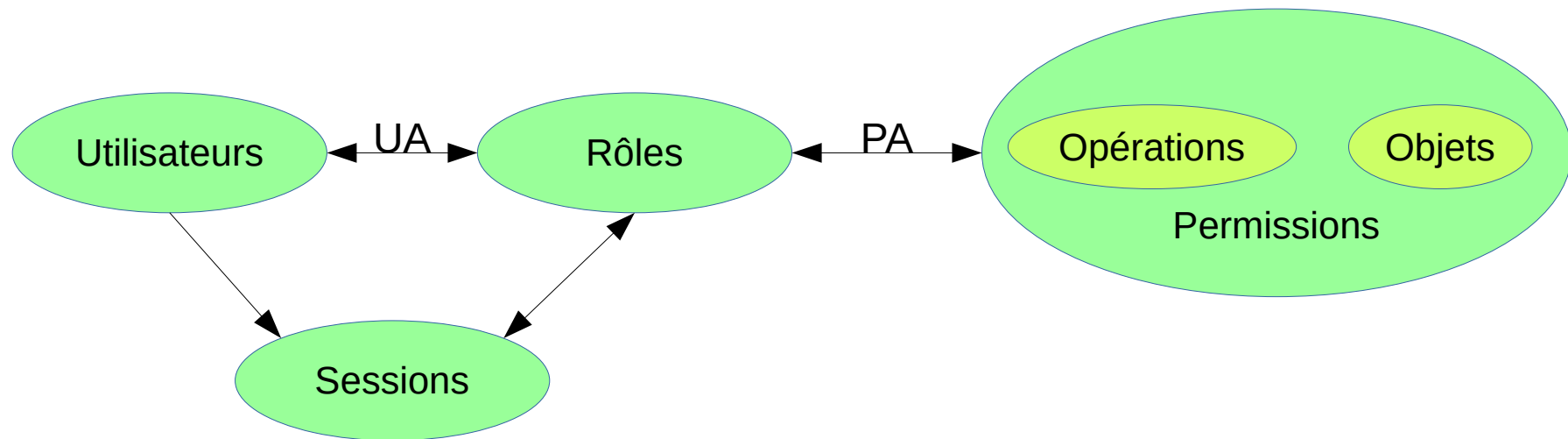
$PA \subseteq \text{PRMS} \times \text{ROLES}$  : relation permettant d'assigner les permissions aux rôles

*assigned\_permissions*:  $(r: \text{ROLES}) \rightarrow 2^{\text{PRMS}}$  : mapping d'un rôle vers un ensemble de permissions

- formellement,  $\text{assigned\_permissions}(r) = \{p \in \text{PRMS} \mid (p, r) \in PA\}$

# Core RBAC: ensembles, fonctions, relations

## Core RBAC (RBAC 0)





# Core RBAC: ensembles, fonctions, relations (suite)

$Op(p: PRMS) \rightarrow \{op \subseteq OPS\}$  : ensemble des opérations associées à la permission  $p$

$Ob(p: PRMS) \rightarrow \{op \subseteq OBS\}$  : ensemble des objets associés à la permission  $p$

SESSIONS = ensemble des sessions

$session\_users(s: SESSIONS) \rightarrow USERS$  : mapping d'une session  $s$  sur l'utilisateur correspondant

$session\_roles(s: SESSIONS) \rightarrow 2^{ROLES}$  : mapping d'une session  $s$  sur un ensemble de rôles

- formellement,  $session\_roles(s) = \{r \in ROLES \mid (session\_users(s), r) \in UA\}$

$avail\_session\_perms(s: SESSIONS) \rightarrow 2^{PRMS}$  : permissions disponibles pour un utilisateur dans une session  $s$

- formellement,  $avail\_session\_perms(s) = \bigcup_{r \in session\_roles(s)} assigned\_permissions(r)$

# Core RBAC : session

## Définition

- projection d'un utilisateur sur un sous-ensemble des rôles affectés à l'utilisateur

## Quelques variantes

- *Single-role activation* (SRA)
  - un seul rôle à la fois
- *Multi-role activation* (MRA)
  - plusieurs rôles peuvent être activés dans une session
  - séparation des devoirs pour restreindre l'activation de certains rôles

# RBAC hiérarchique

## Motivations

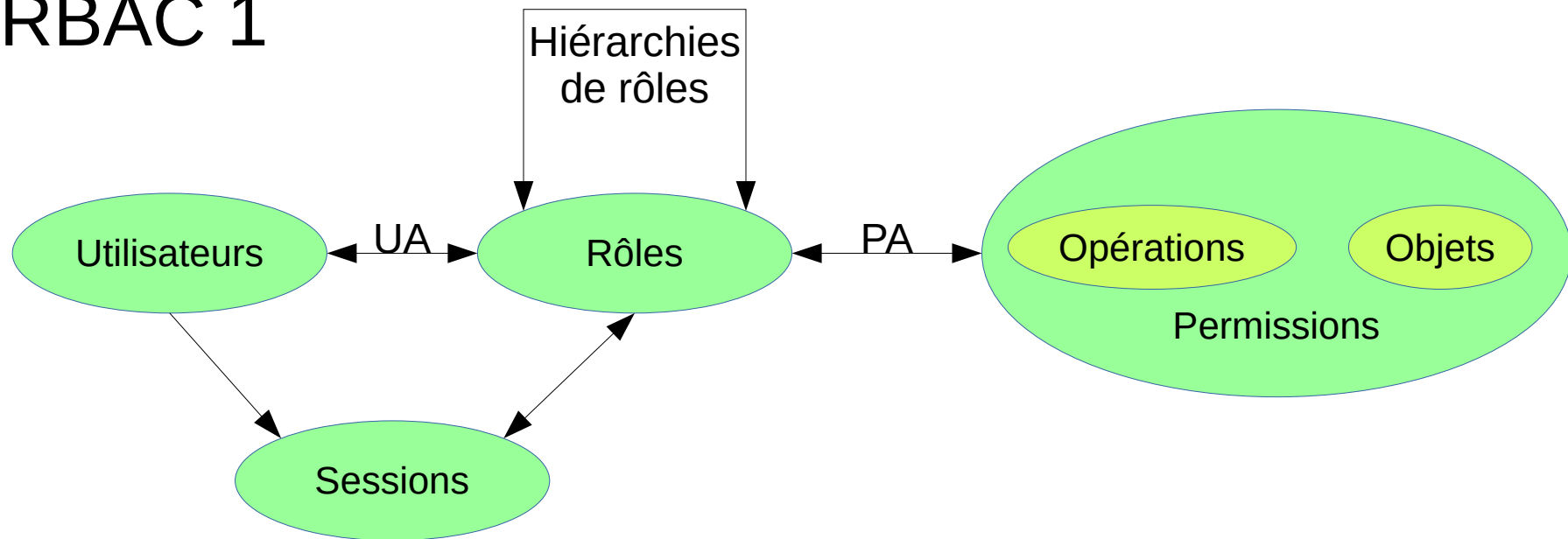
- les hiérarchies de rôles représentent un moyen intéressant pour structurer les rôles entre eux

## Modèle

- $RH \subseteq \text{ROLES} \times \text{ROLES}$  : une hiérarchie est définie sur un ensemble de rôles
- non réflexive et acyclique
  - relation de domination : si  $(r_i, r_j) \in RH$  on dit que  $r_i$  domine  $r_j$
- ordre partiel  $\geq$  qui est la fermeture transitive et réflexive de  $RH$ 
  - calculé ou stocké

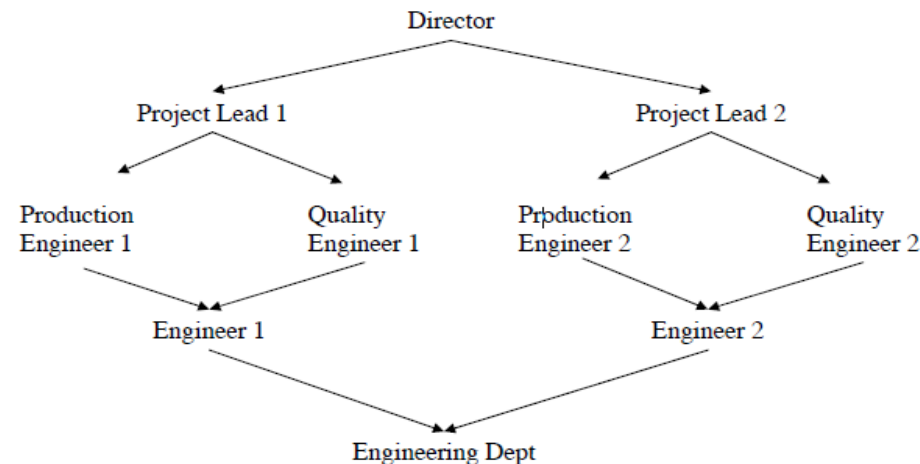
# RBAC hiérarchique

## RBAC 1



# RBAC hiérarchique (suite)

Exemple :



*User Inheritance (UI)* : tous les utilisateurs autorisés pour le rôle  $r$  sont également autorisés pour tout rôle  $r'$  tel que  $r \geq r'$

*Permission Inheritance (PI)* : un rôle  $r$  est autorisé pour toutes les permissions pour lesquelles tout rôle  $r'$ , tel que  $r \geq r'$ , est autorisé

*Activation Inheritance (AI)* : si on active un rôle  $r$ , tous les rôles  $r'$  tels que  $r \geq r'$  sont activés (uniquement dans le cas où l'on a MRA)

# RBAC contraint

RBAC contraint permet de gérer la séparation des devoirs (SoD)

Deux types :

- SoD statique
- SoD dynamique

## **SoD**

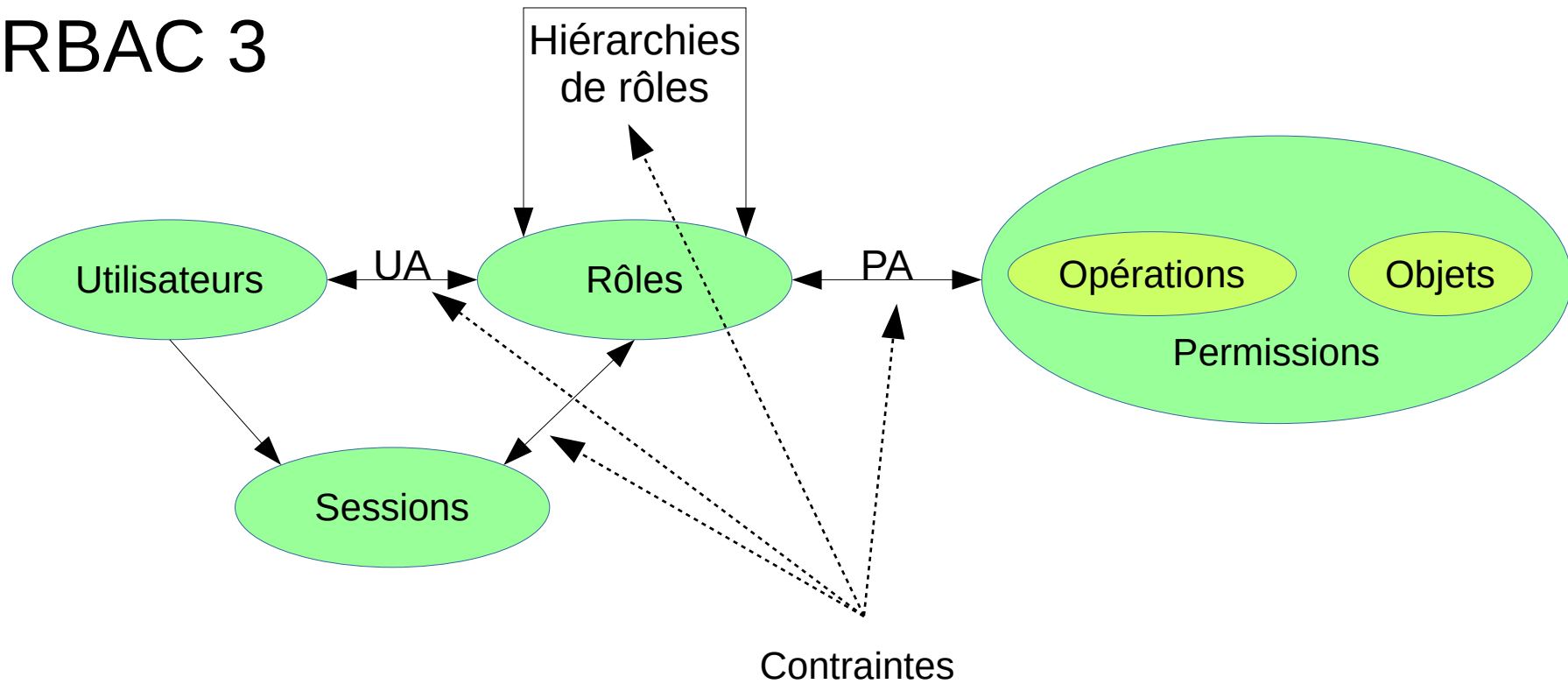
- permet de prendre en compte les conflits d'intérêts (voir Chinese Wall)
- permet d'éviter la collusion entre plusieurs rôles et les problèmes qui pourraient en découler

## **Définition (ANSI)**

- “division de la responsabilité pour les informations sensibles de manière à ce que les personnes agissant seules ne puissent pas compromettre la sécurité d'un système”

# RBAC constraint

## RBAC 3



# RBAC : SoD

## Statiques

- expression des restrictions sur l'ensemble des rôles et en particulier sur les relations RA (rôles autorisés)
- possibilité d'avoir au plus un certain nombre de rôles
- impossible qu'un utilisateur puisse utiliser trop de rôles
- contrôle des contraintes en surveillant l'assignation de rôles aux utilisateurs

## Dynamiques

- possibilité de limiter le nombre de rôles actifs dans une session
- exemples de contraintes :
  - un utilisateur ne peut avoir plus de  $x$  rôles dans une session
  - si l'utilisateur possède le rôle  $r1$  dans une session, il ne peut pas acquérir le rôle  $r2$  dans la même session
- contrôle des contraintes en surveillant l'historique des relations utilisateur-rôles dans une session



# RBAC contraint

## Contraintes

- rôles mutuellement exclusifs
  - exclusion statique/dynamique (cf. slide précédent)
- permissions mutuellement exclusives
  - statique : le rôle ne peut avoir deux permissions données
  - dynamique : le rôle ne peut avoir deux permissions données dans un contexte identique
- cardinalité sur l'affectation utilisateur-rôle (UA)
  - au plus  $k$  utilisateurs pour un rôle
  - au moins  $k$  utilisateurs pour un rôle
  - exactement  $k$  utilisateurs pour un rôle
- cardinalité sur l'affectation rôle-permissions
  - au plus (moins, exactement)  $k$  rôles doivent avoir la permission

# RBAC : administration

## **Opérations de gestion**

- Create, Delete, Maintain

## **Surveillance**

- Query

## **Fonctions de niveau système**

- création de sessions
- activation/désactivation de rôle
- respect des contraintes
- calcul de décisions sur la possibilité d'accéder ou non à une ressource