DE LA RECHERCHE À L'INDUSTRIE

Sécurité Windows

Système

Commissariat à l'Énergie Atomique
et aux Énergies Alternatives

13 janvier 2020

www.cea.fr

Sommaire

1 Introduction

2 Bases

3 Contrôle d'accès

4 Authentification

5 Mécanismes de protection

Sommaire

1 Introduction

2 Bases

3 Contrôle d'accès

4 Authentification

5 Mécanismes de protection

Microsoft Windows

Il s'agit d'un **système d'exploitation** qui :

- est développé par Microsoft ;
- existe depuis **1985** ;
- est massivement utilisé ;
  - environ 90% des postes et 30% des serveurs dans le monde ;
  - il est donc important de l'étudier.
- est un **produit commercialisé** et non *open source*.

Critiques

Critiques

Beaucoup de critiques peuvent être faites à Windows, notamment :

- son code est **non *open source***;
- il y a relativement **peu de documentation** de qualité;
- il a essuyé beaucoup d'**échecs en terme sécurité**;
- il s'utilise via un ***cliquodrome***.

⇒ certains des points négatifs ci-dessus sont en voie d'amélioration :

- apparition de plus de documentations de qualité;
- prise de conscience de l'importance de la sécurité et véritables améliorations;
- développement d'alternatives au *cliquodrome*.

Versions Windows

- La famille DOS shells :

Dénomination	Parution	Fin de support
Windows 1.0	1985	2001
Windows 2.0	1987	2001
Windows 2.1x	1988	2001
Windows 3.0	1990	2001
Windows 3.1x	1992	2001

- La famille Windows 9x :

Dénomination	Parution	Fin de support
Windows 95	1995	2001
Windows 98	1998	2006
Windows 98 SE	1999	2006
Windows Me	2000	2006

Versions Windows

- La famille **Windows NT** (*New-Technology*), partie 1/2 :

Version	Dénomination	Parution	Fin de support
3.1	Windows NT 3.1	1993	2000
3.5	Windows NT 3.5	1994	2001
3.51	Windows NT 3.51	1995	2001
4.0	Windows NT 4.0	1996	2004
5.0	Windows 2000	1999	2010
5.1	Windows XP	2001	2014
5.2	Windows Server 2003	2003	2015
5.2	Windows Server 2003 R2	2005	2015
6.0	Windows Vista	2006	2017

Versions Windows

- La famille **Windows NT** (*New-Technology*), partie 2/2 :

Version	Dénomination	Nom de code	Parution	Fin de support
6.0	Windows Server 2008	Longhorn Server	2008	2020
6.1	Windows 7	7, Blackcomb, Vienna	2009	2020
6.1	Windows Server 2008 R2	7 Server	2009	2020
6.2	Windows 8	8	2012	2016
6.2	Windows Server 2012	8 Server	2012	2023
6.3	Windows 8.1	Blue	2013	2023
6.3	Windows Server 2012 R2	Blue Server	2013	2023
10.0	Windows 10	Threshold, Redstone	2015	2025-2027
10.0	Windows Server 2016	Redstone Server	2016	2025-2027
10.0	Windows Server 2019	Redstone Server 5	2019	-

Versions Windows

- Détails des versions de Windows 10 :

Version	Nom de code	Parution	Fin de support
1507	Threshold 1	2015	2017
1511	Threshold 2	2015	2017
1607	Redstone 1	2016	2019
1703	Redstone 2	2017	2019
1709	Redstone 3	2017	2020
1803	Redstone 4	2018	2020
1809	Redstone 5	2018	2021

Composants de sécurité Windows

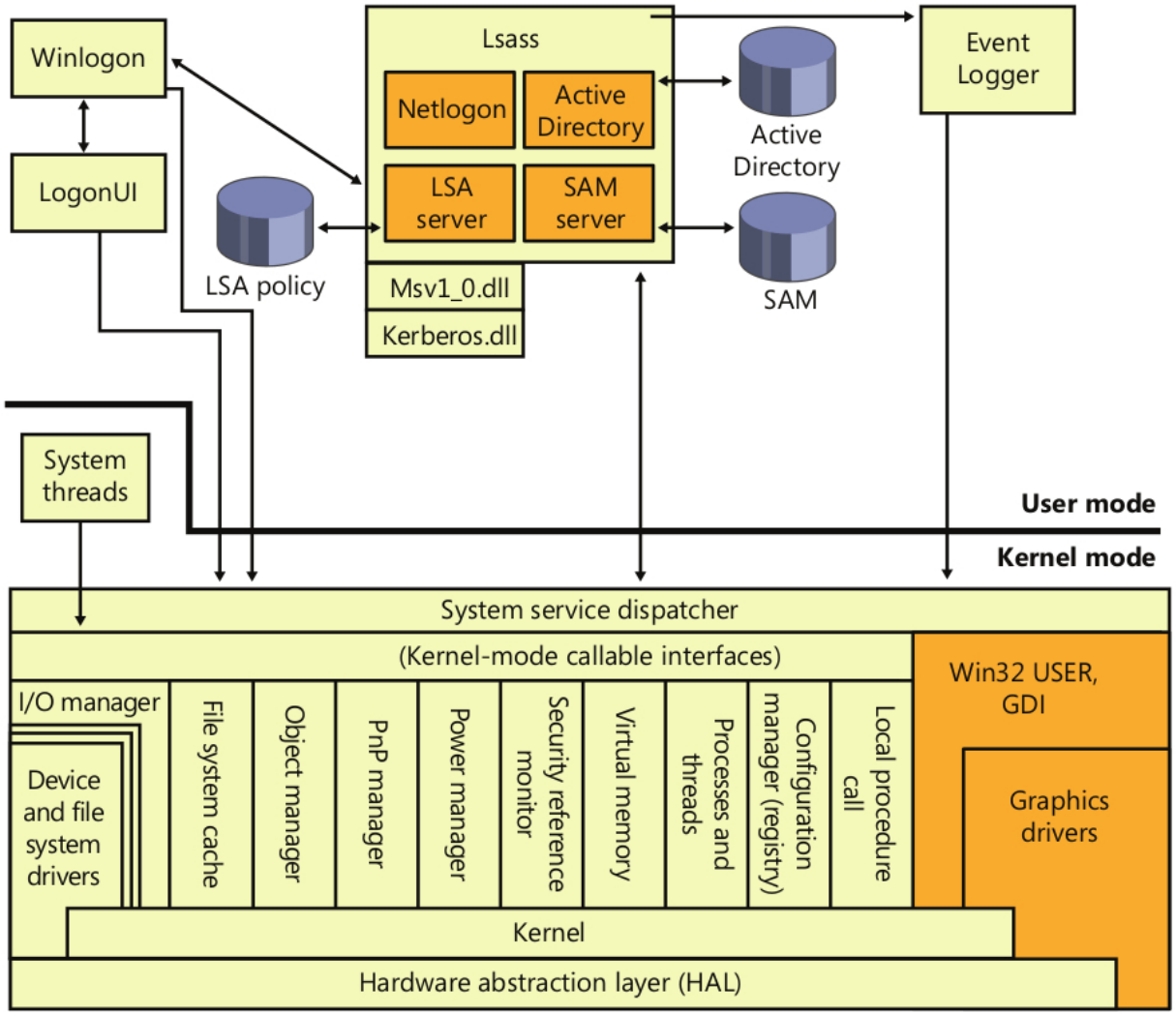


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Composants de sécurité Windows

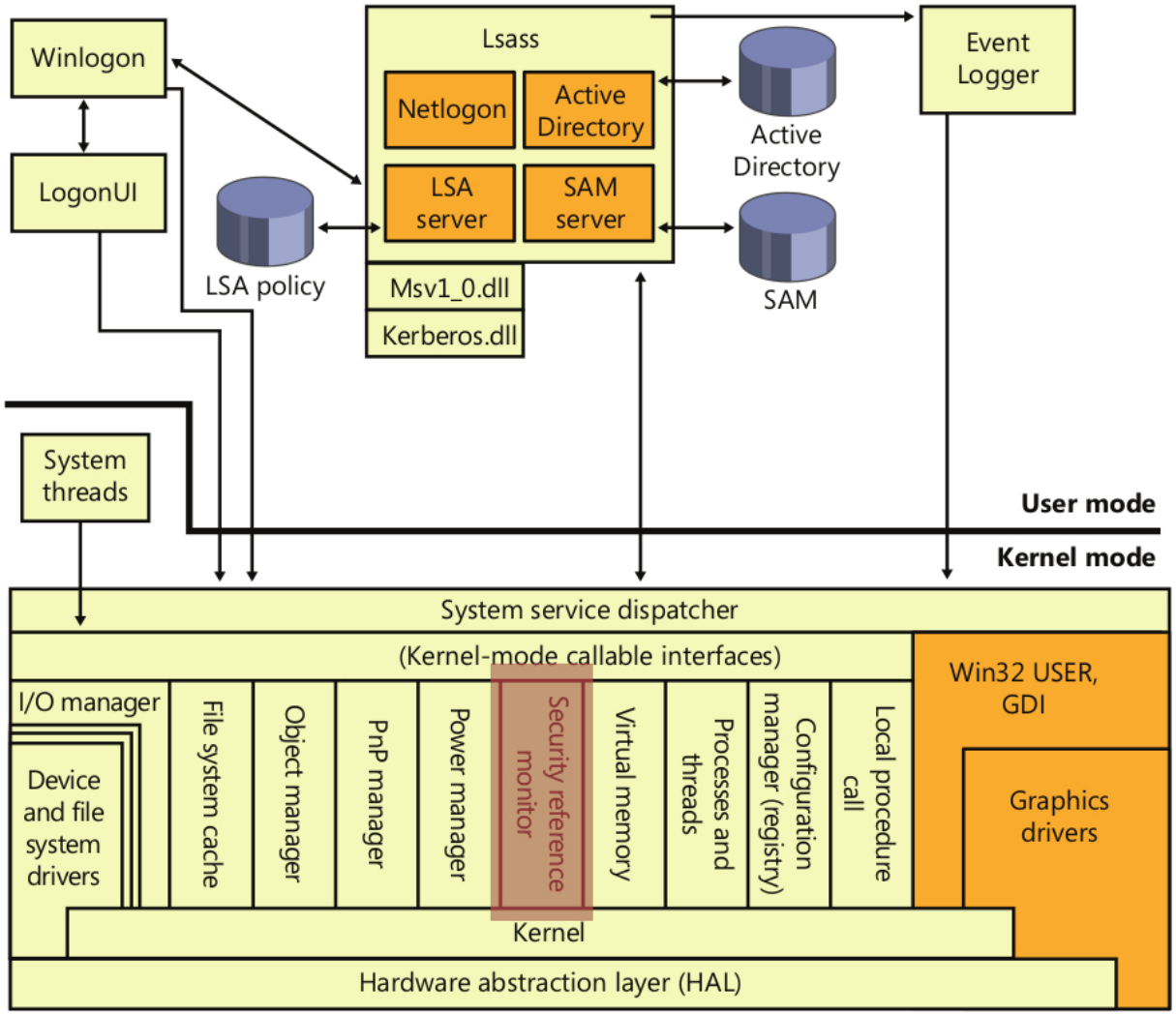


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Security Reference Monitor

SRM est un des composants de l'**exécutif Windows** (`ntoskrnl.exe`, qui contient aussi le noyau Windows).

SRM a pour rôles de :

- définir la structure de données des jetons d'accès ;
- effectuer les **contrôles d'accès** sur les objets ;
- manipuler les **privilèges** et droits utilisateur ;
- générer les **messages d'audit de sécurité** résultants.

Composants de sécurité Windows

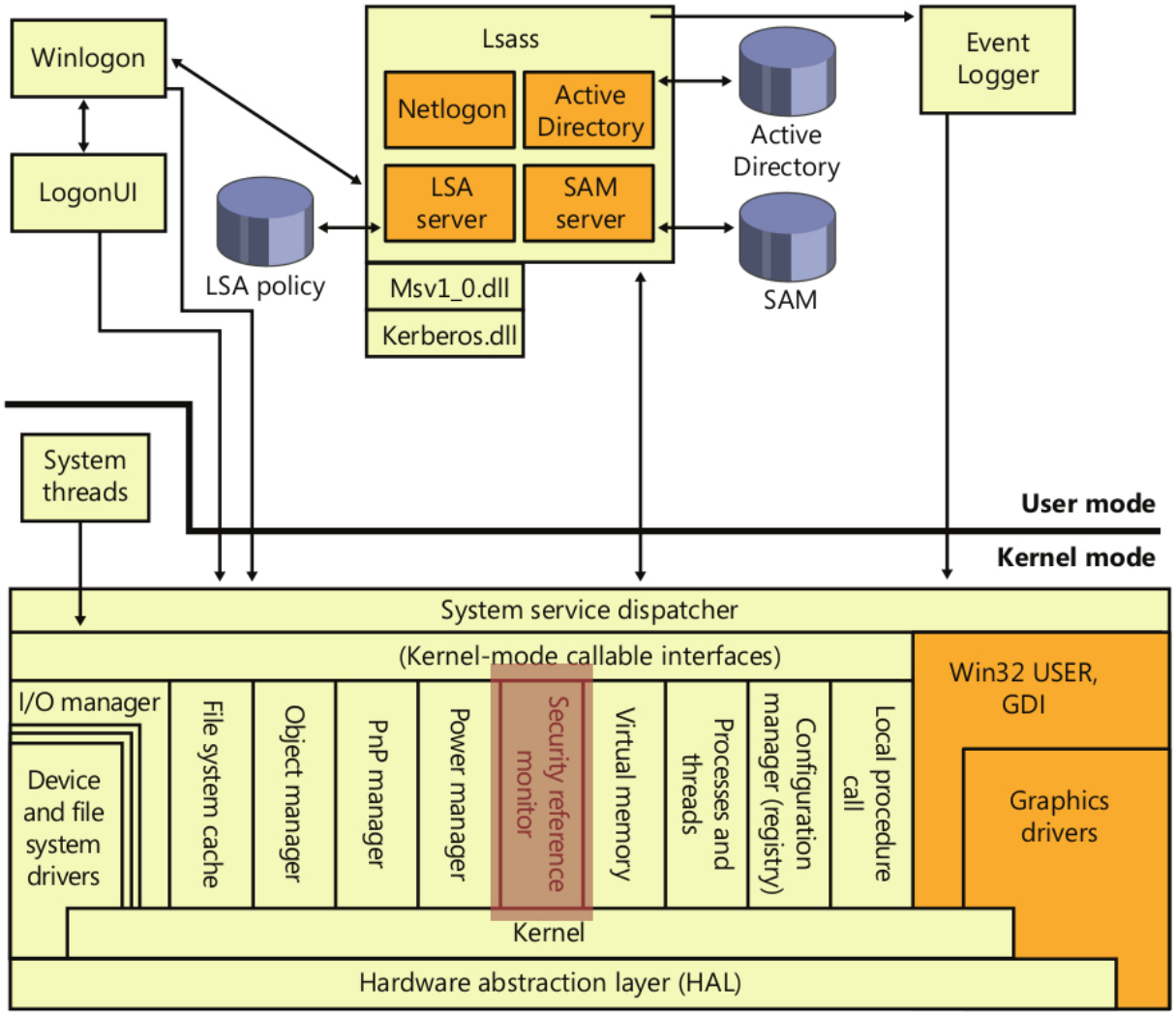


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Composants de sécurité Windows

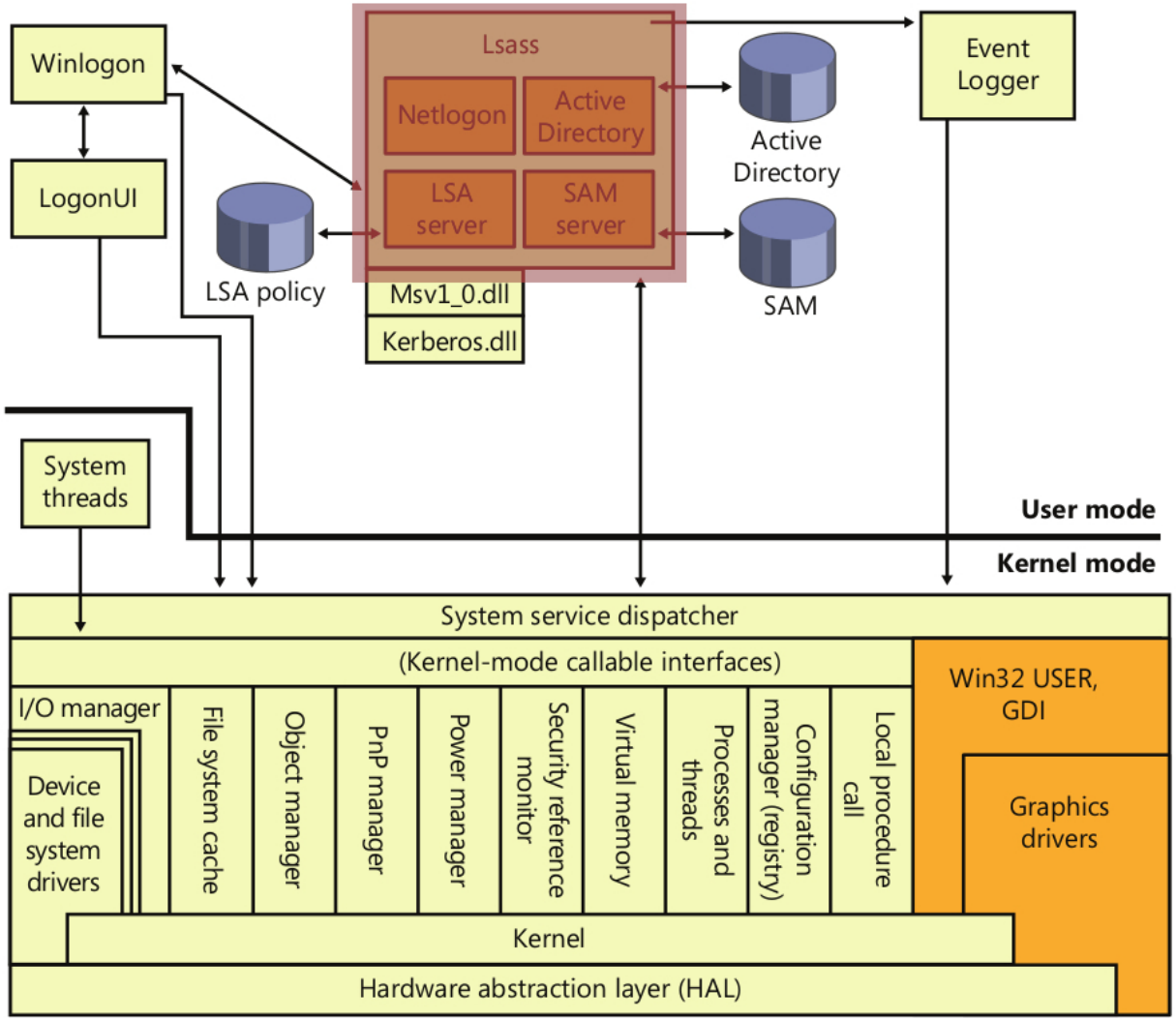


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Local Security Authority SubSystem

LSASS est un processus en mode utilisateur qui exécute l'image `lsass.exe`.

LSASS a pour rôles de gérer :

- la **politique de sécurité locale**, qui définit entre autres :
  - les utilisateurs autorisés à se connecter sur la machine ;
  - la politique de mots de passe ;
  - les privilèges donnés aux utilisateurs et aux groupes ;
  - la configuration du système d'audit de sécurité.
- **l'authentification** des utilisateurs ;
- **l'envoi des messages d'audit** de sécurité au journal d'événement.

Composants de sécurité Windows

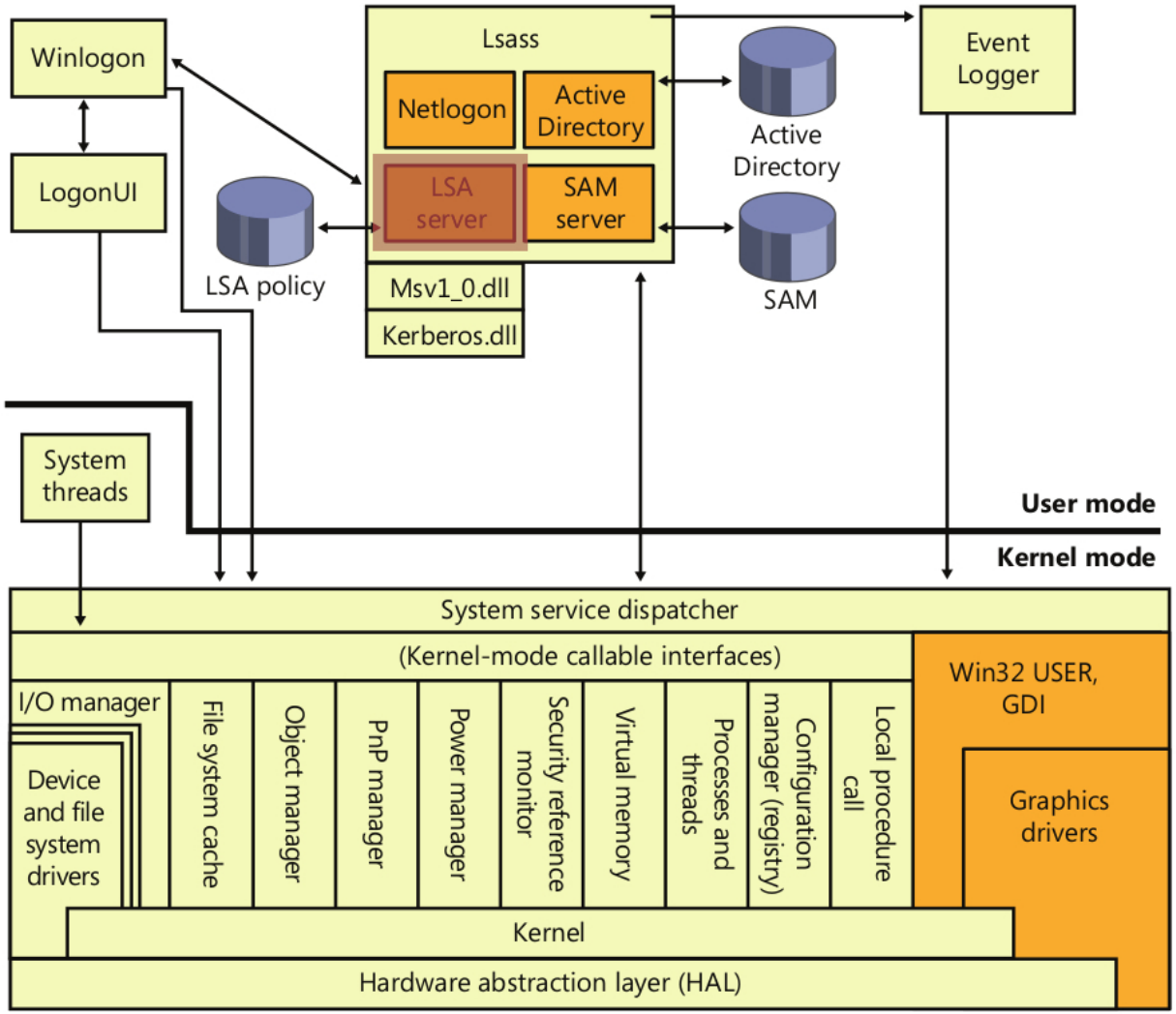


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Local Security Authority server

LSA *server* est une *dynamic-link libraries* (`lsasrv.dll`) chargée par LSASS qui implémente la plus grande partie des fonctionnalités de LSASS.

Composants de sécurité Windows

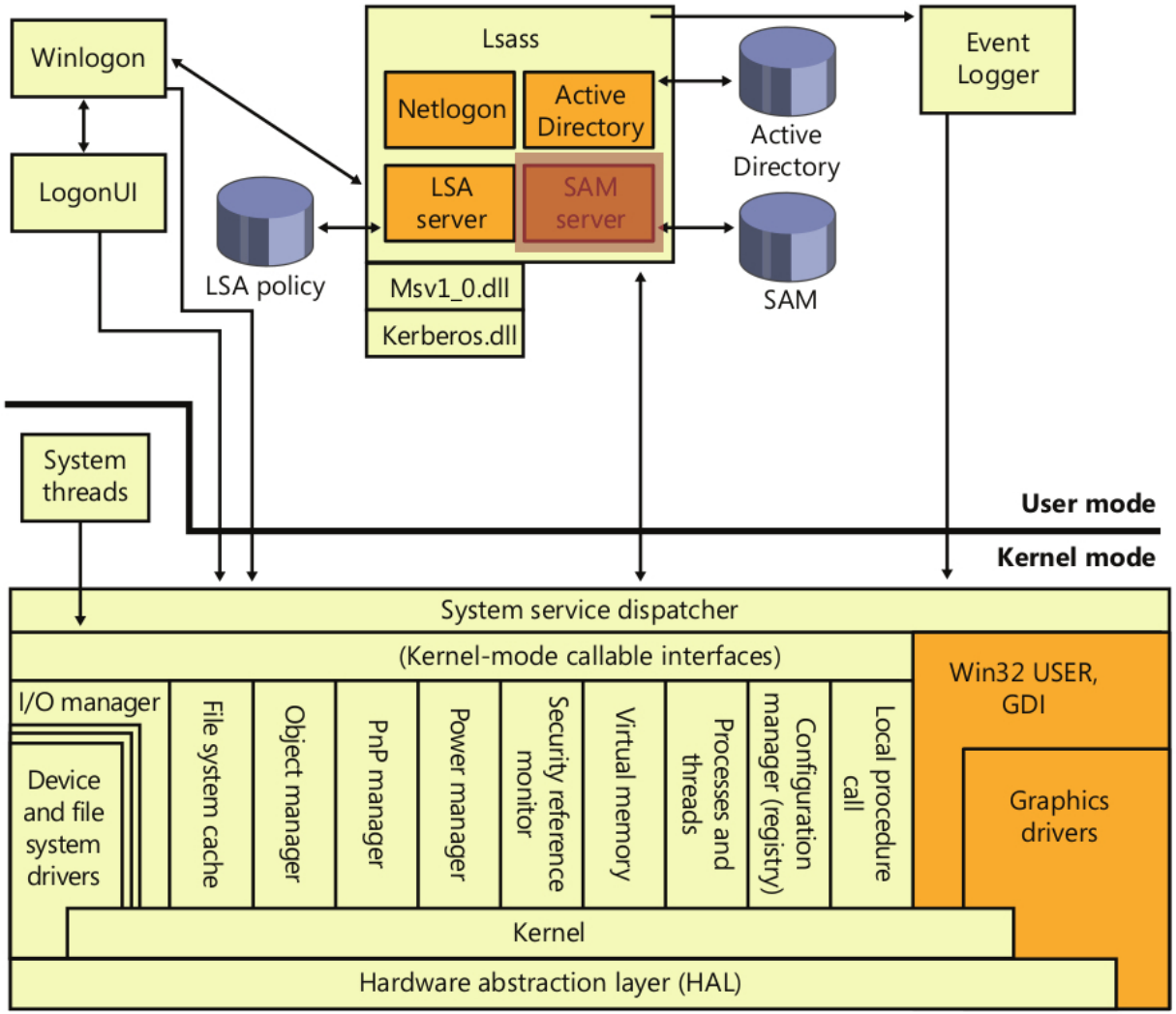


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Security Accounts Manager server

SAM *server* est une DLL (`samsrv.dll`) chargée par LSASS.

SAM *server* gère la **base de données SAM**.

Base SAM

La base de données SAM :

- est stockée dans la base de registre sous **`HKLM\SAM`** ;
- est stockée au niveau du système de fichiers dans `%SystemRoot%\System32\Config\Sam` ;
- contient :
  - les noms des utilisateurs et groupes locaux ;
  - les **empreintes de leur mot de passe** ;
  - quelques informations supplémentaires (comme le nom complet ou le chemin de leur profil).

Composants de sécurité Windows

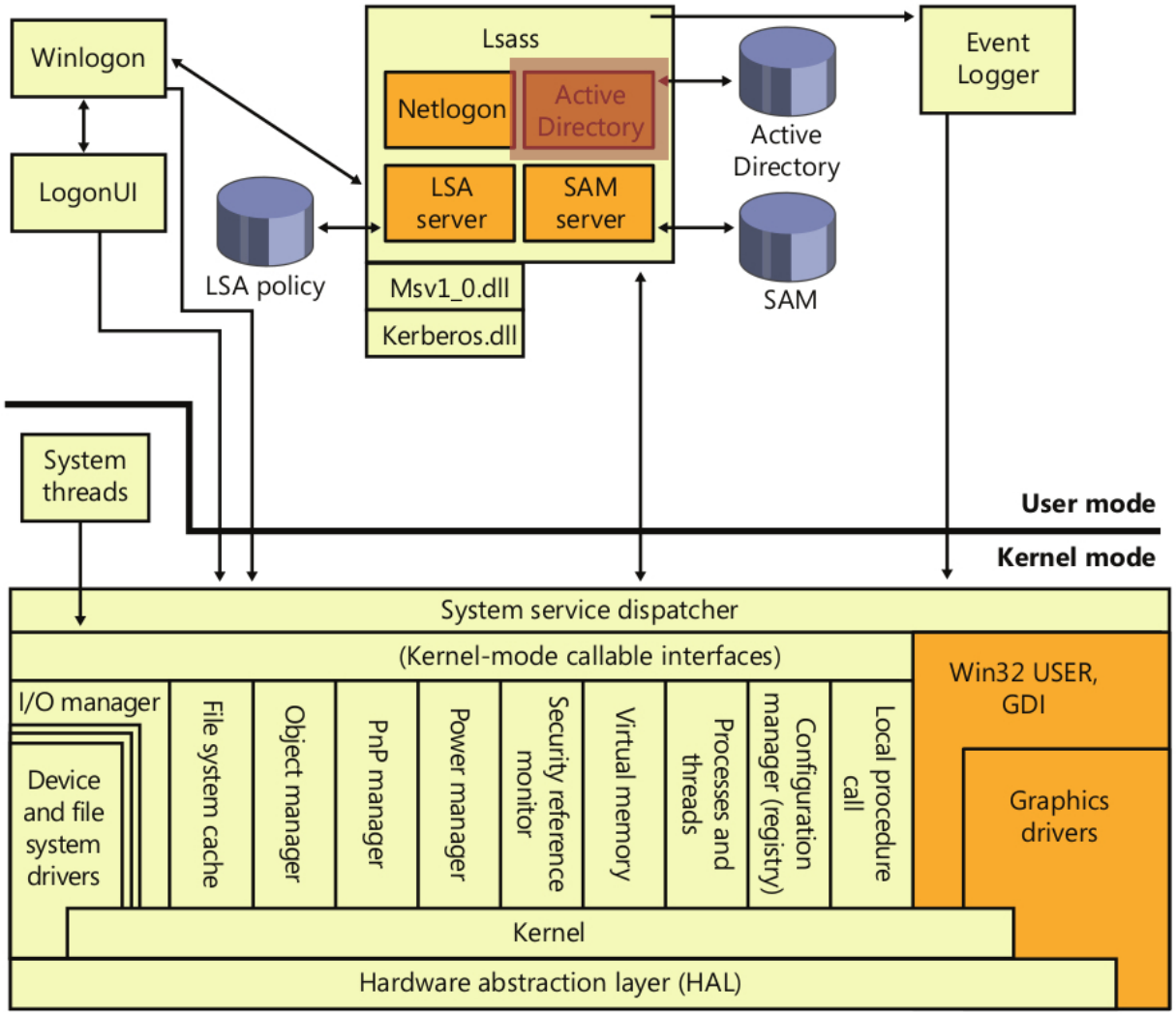


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Active Directory server

Active Directory server est une DLL (`ntdsa.dll`) chargée par LSASS.

Cette DLL est utilisée sur **les contrôleurs de domaine**. Elle implémente le *Directory System Agent* (DSA) qui contient une grande partie des fonctionnalités d'*Active Directory*.

Active Directory

Il s'agit d'un **service d'annuaire** qui stocke les informations au sujet des objets dans un **domaine** (collection de machines gérées comme une entité) comme :

- la liste des utilisateurs, groupes et machines ;
- les mots de passe ;
- les privilèges des utilisateurs et groupes.

Les informations d'un domaine sont répliquées sur les machines appelées DC (*Domain Controllers* ou **contrôleurs de domaine**).

Composants de sécurité Windows

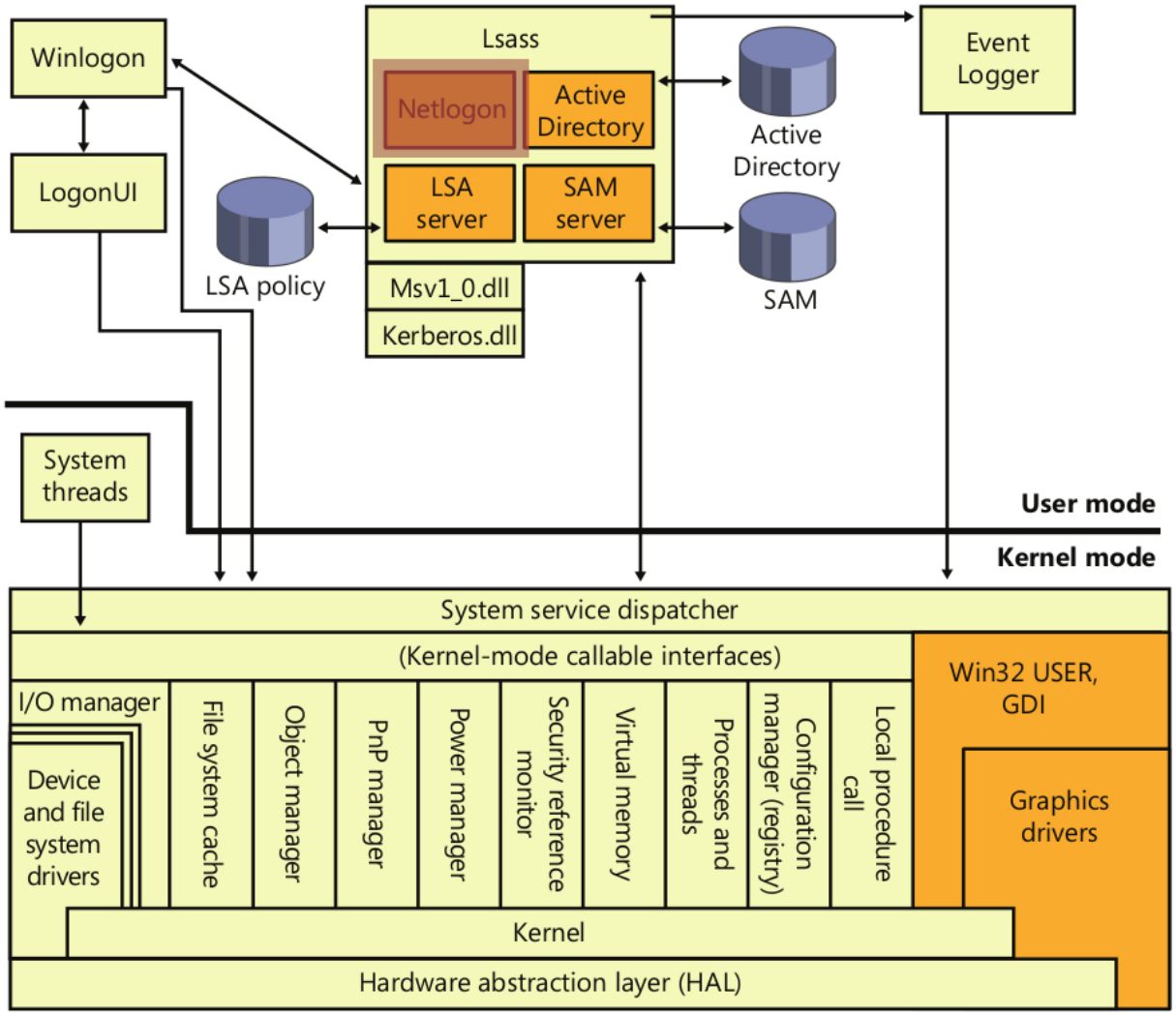


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Service *Network Logon*

Netlogon est un service Windows dont les fonctionnalités sont implémentées dans la DLL `netlogon.dll`.

Ce service permet :

- de **localiser les contrôleurs de domaine** ;
- d'établir un **canal sécurisé** avec un contrôleur de domaine ;
- d'**effectuer les requêtes** de sécurité (comme les requêtes d'authentification NTLM) via ce canal ;
- de **recevoir les requêtes** de sécurité du côté du contrôleur de domaine.

Composants de sécurité Windows

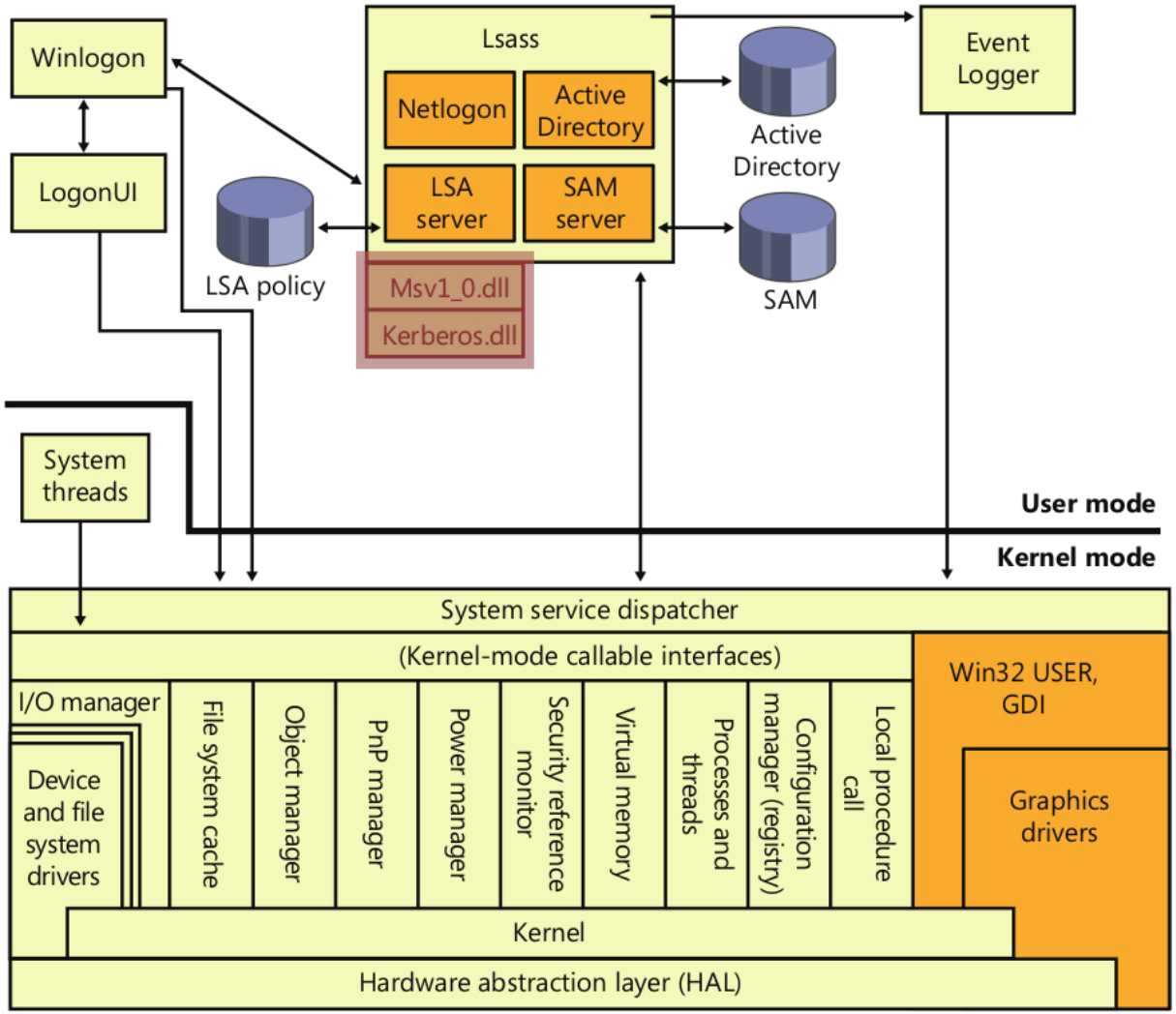


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Composants de sécurité Windows

Des **packages d'authentification**, des DLL chargés par LSASS, implémentent la politique d'authentification Windows.

Ces DLL permettent :

- d'**authentifier les utilisateurs** (en vérifiant que l'identifiant et le mot de passe correspondent) ;
- de communiquer à LSASS les informations détaillant l'**identité de l'utilisateur** (afin que LSASS puisse générer un jeton d'accès).

Parmi les packages d'authentification on trouve :

- `msv1_0.dll` qui s'occupe entre autres de l'authentification NTLM ;
- `kerberos.dll` pour l'authentification Kerberos ;
- `wdigest.dll` pour l'authentification *digest*.

Composants de sécurité Windows

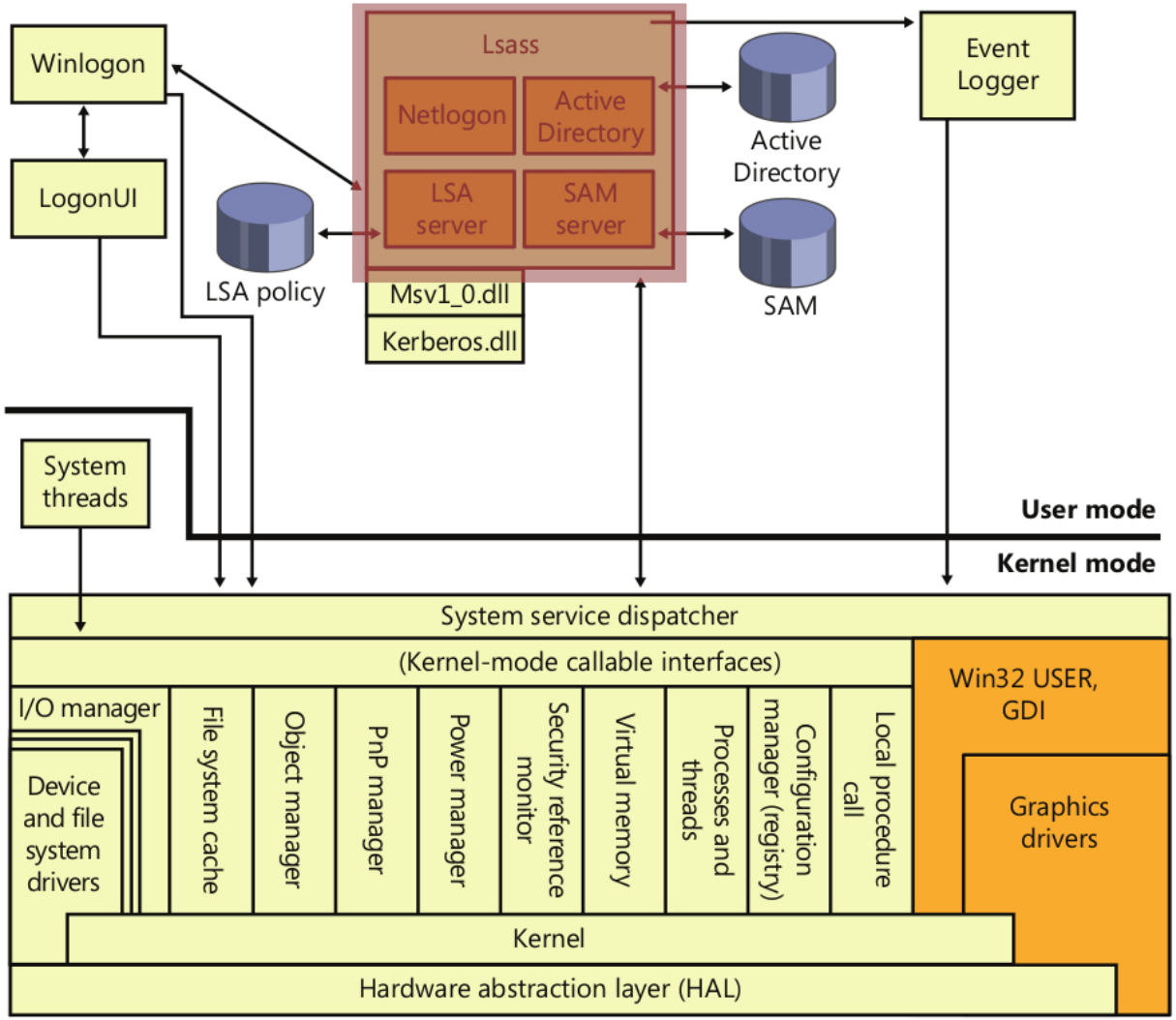


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Composants de sécurité Windows

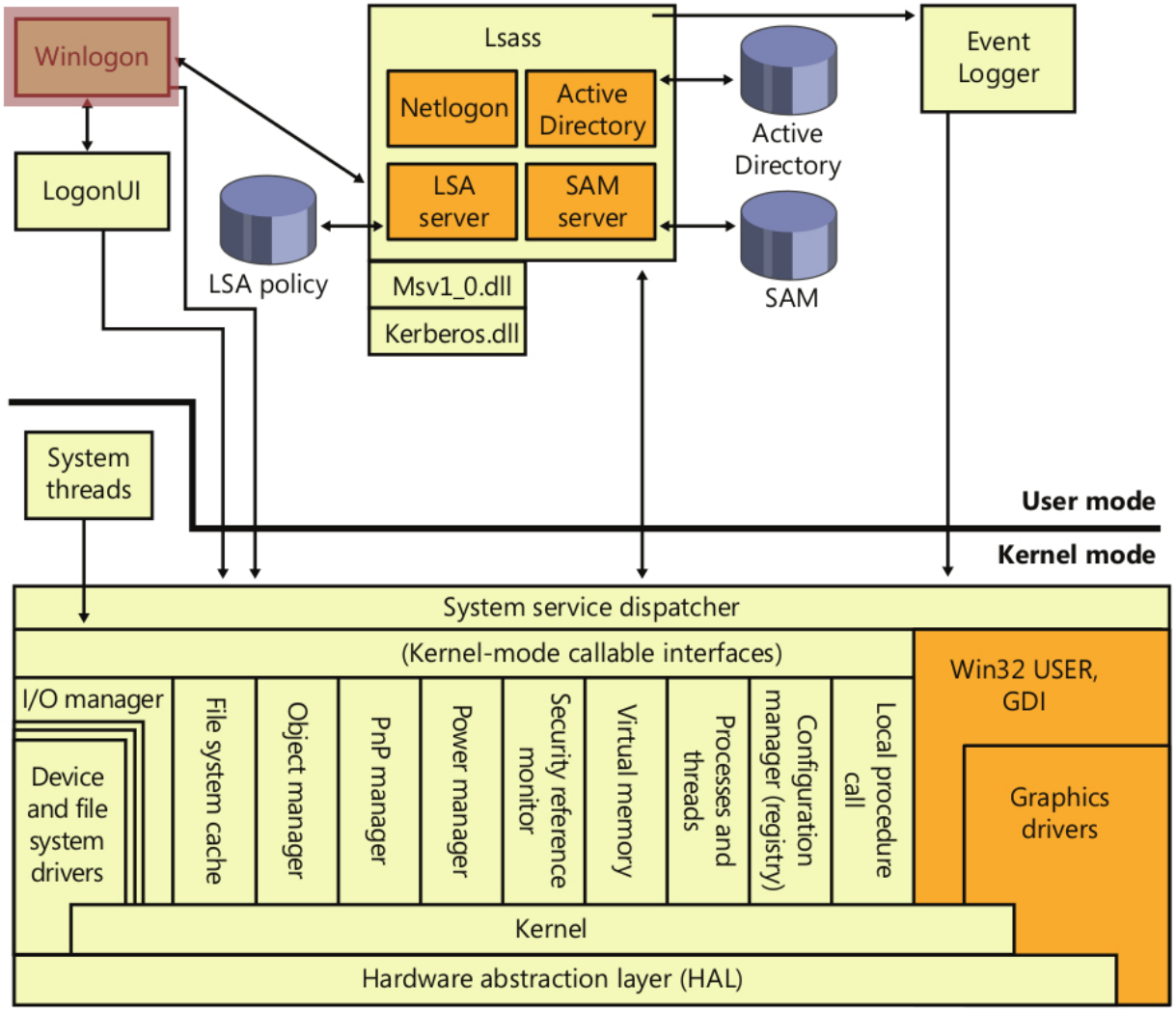


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Winlogon

Interactive logon manager (ou *Winlogon*) est un processus en mode utilisateur qui exécute l'image `winlogon.exe`.

Winlogon a pour rôles de :

- répondre au SAS (*Secure Attention Sequence*), pression de `Ctrl+Alt+Delete`;
- **gérer les sessions interactives** :
  - par exemple, crée le premier processus de l'utilisateur quand il se connecte.

Composants de sécurité Windows

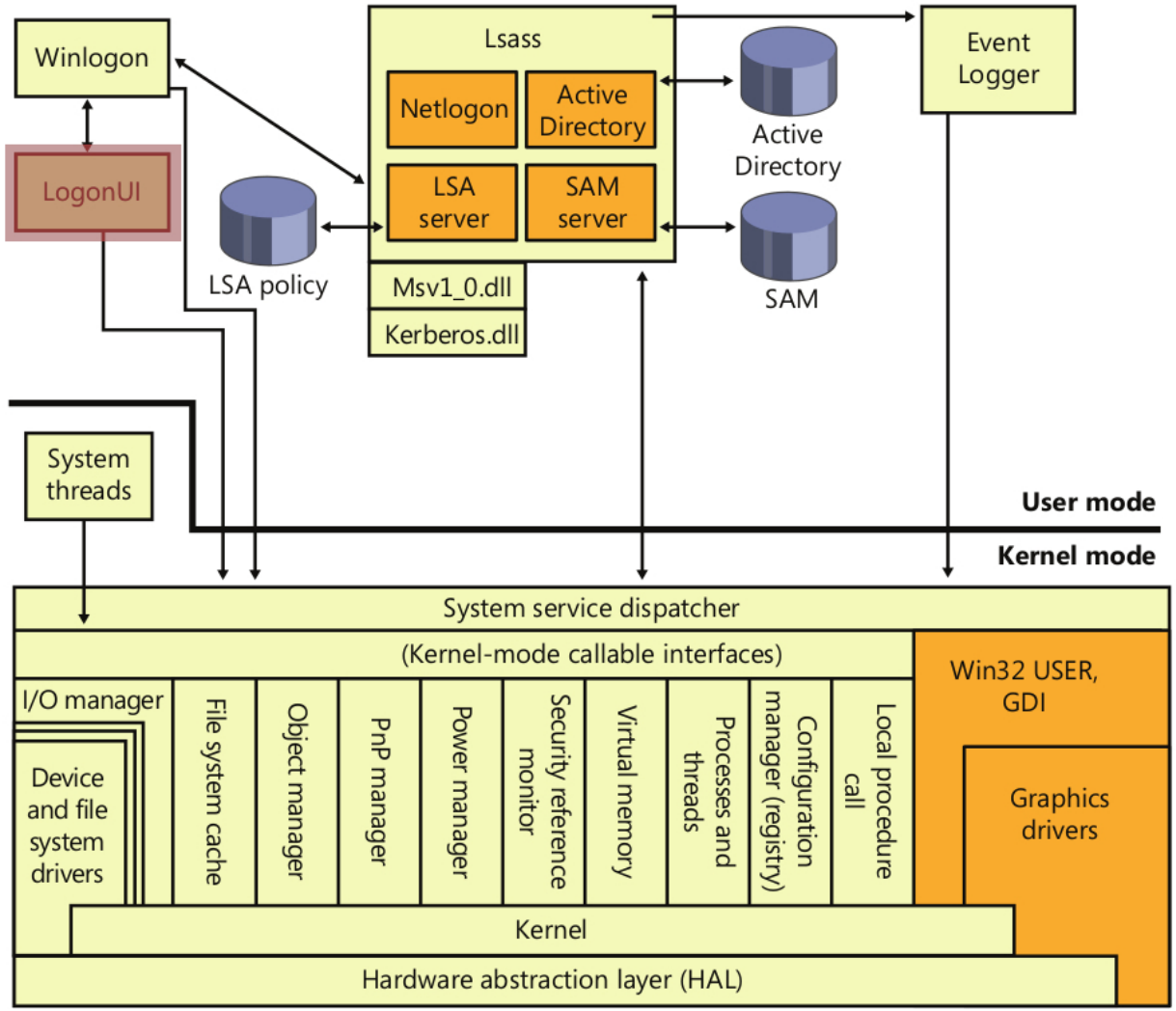


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Logon user interface

LogonUI est un processus en mode utilisateur qui exécute l'image `logonUI.exe`.

LogonUI présente une interface aux utilisateurs pour qu'ils s'authentifient sur le système. *LogonUI* utilise des ***credential providers*** pour récupérer les authentifiants des utilisateurs de différentes manières.

Credential Providers

Les CP sont des DLL qui sont exécutés dans le processus *LogonUI* qui implémentent différentes façons de récupérer les authentifiants des utilisateurs.

Les CP standards sont par exemple utilisés pour obtenir :

- l'identifiant et le mot de passe de l'utilisateur (via le CP `authui.dll`) ;
- le PIN d'une carte à puce (via le CP `SmartcardCredentialProvider.dll`).

Il est possible de créer ses propres CP.

Composants de sécurité Windows

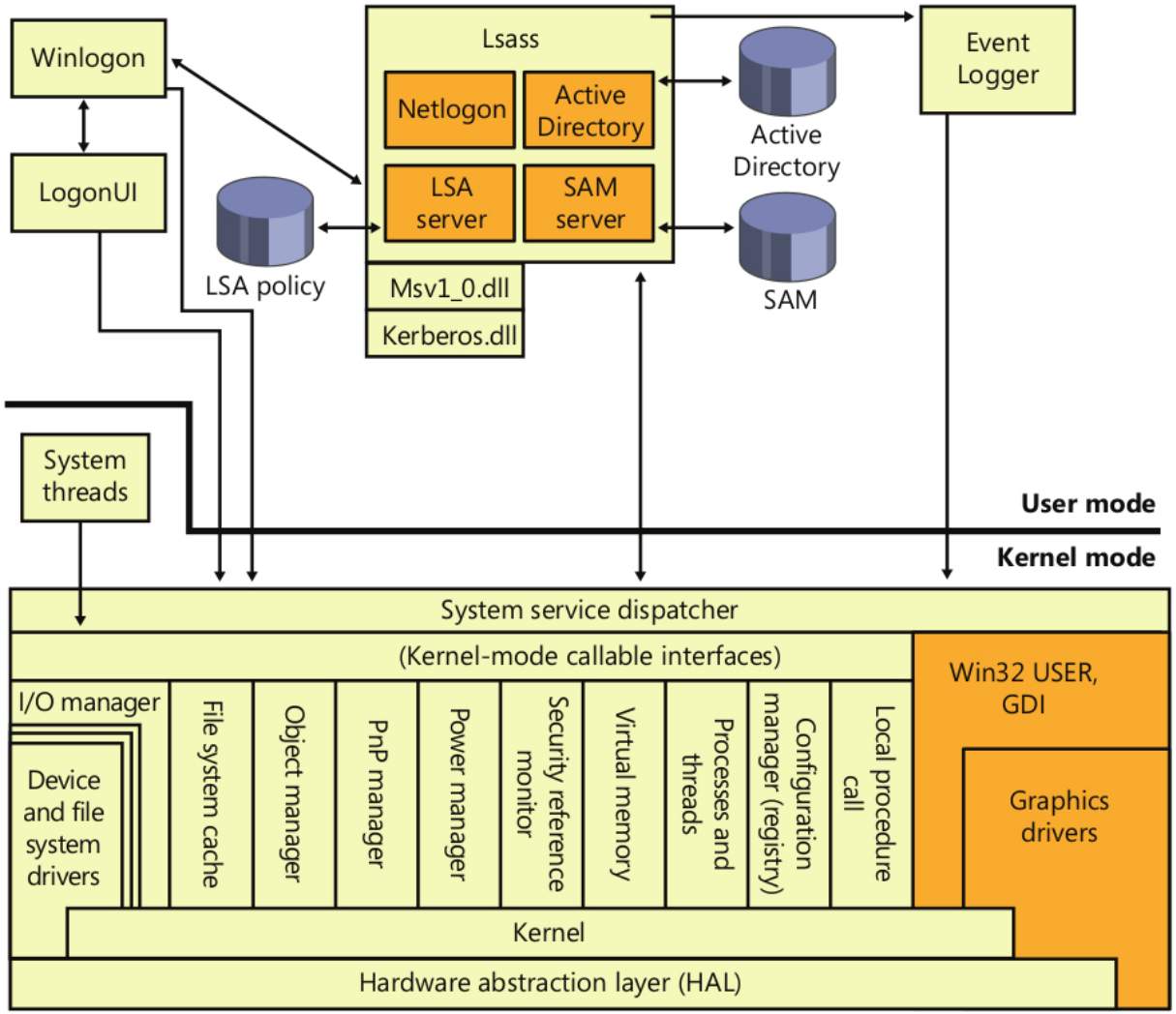


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Sommaire

1 Introduction

2 Bases

3 Contrôle d'accès

4 Authentification

5 Mécanismes de protection

Format PE

- *Portable Executable* (PE) est un **format binaire** employé dans Windows pour l'enregistrement et l'organisation de nombreuses formes de **contenu actif** ;
- Le format PE est à la base de nombreux composants exécutables de Windows, comme par exemple :
  - les programmes exécutables (**.exe**) ;
  - les bibliothèques logicielle (**.dll**) ;
  - les pilotes de périphériques (.sys) ;
  - les enfichables du panneau de configuration (.cpl).
- Pour des informations détaillées sur le format se référer au document "*Microsoft Portable Executable and Common Object File Format Specification*" trouvable sur `docs.microsoft.com` ;
- Dans cette présentation, quand rien n'est précisé, les PE dont nous parlerons se trouveront dans :
  - `c:\Windows\System32\` ;
  - ou plus généralement `%SystemRoot%\System32\`.

Fichiers fondamentaux du système

Fichier	Rôles
Hal.dll	Module noyau implémentant la couche d'**abstraction matérielle**.
Ntoskrnl.exe [1]	Exécutif et **noyau**.
Win32k.sys	Pilote de périphérique en mode noyau qui héberge le **gestionnaire de fenêtres**, les affichages écran et les saisies clavier-souris (et autres périphériques).
Ntdll.dll	DLL exportant des **fonctions de point d'entrée vers les services systèmes** ainsi que des fonctions de support internes (comme le chargeur d'image, fonctions commençant par *Ldr*).
Kernel32.dll	DLL exportant des **fonctions fondamentales** comme les fonctions de gestion de fichiers, de création de processus ou encore de gestion du tas.

1. il existe des variantes comme *Ntkrnlpa.exe* pour le support de PAE

Modes d'exécution

- Pour empêcher les applications utilisateurs d'avoir accès ou de modifier des données critiques du système d'exploitation, Windows utilise deux modes du processeur :
  - *kernel mode* ou **mode noyau** ;
  - *user mode* ou **mode utilisateur**.
- Les processeurs proposes différents modes d'exécution avec différents niveaux de privilège ;
  - ces niveaux sont appelés anneaux de protection ou ***rings*** ;
  - Windows n'utilise que deux modes :
    - *ring 0* pour le mode noyau ;
    - *ring* 3 pour le mode utilisateur.
  - permettent de restreindre :
    - les instructions disponibles (comme l'instruction `HLT` pour les processeurs x64) ;
    - les zones mémoires qui peuvent être lues ou écrites.

Anneaux de protections

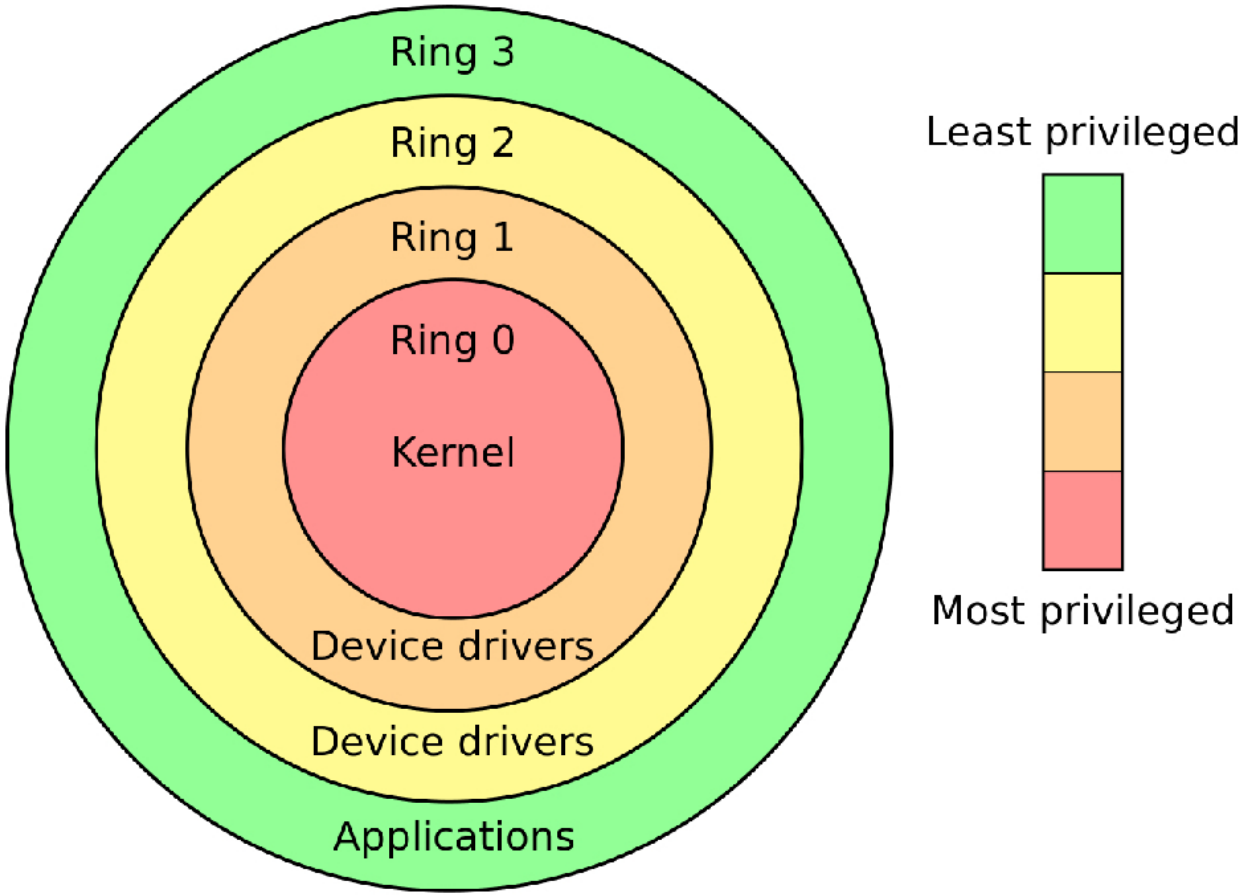


Figure – Représentation des anneaux (*ring*) de protection d'un CPU

Modes d'exécution

Kernel mode

Dans ce mode d'exécution :

- le processeur donne accès à l'**ensemble de la mémoire** du système ;
  - les pilotes s'exécutant en mode noyau peuvent prendre le contrôle du système ;
  - les pages en *read-only* ne peuvent être altérées depuis aucun mode.
- le processeur donne accès à l'**ensemble des instructions** CPU ;
- les *crashs* dans ce mode arrêtent l'exécution de l'ensemble du système (**BSoD**, *Bleue Screen of Death*) ;
- tous le monde partage un même espace d'adressage virtuel.

Modes d'exécution

User mode

Dans ce mode d'exécution :

- le processeur ne donne accès **qu'à la mémoire du processus** ;
- le processeur bloque l'accès aux **instructions privilégiées** du CPU ;
- les *crashs* dans ce mode donne lieu à une exception qui peut être gérée ;
- l'accès au matériel et aux structures du système passe par l'API Windows ;
  - exemple de l'appel de `CreateFile` :
    - appel de `CreateFile` (de `kernel32.dll`) ;
    - qui appelle `NtCreateFile` (de `ntdll.dll`) ;
    - qui via l'instruction `SYSCALL` passe en mode noyau
    - et appelle `NtCreateFile` de `ntoskrnl.exe`.
  - lors de ces appels systèmes, les transitions entre le mode utilisateur et le mode noyau passe par les instructions :
    - `SYSCALL` et `SYSRET` en x64 (et 32 bits AMD) ;
    - `SYSENTER` et `SYSEXIT` en x86.

SYSCALL

```
; Exported entry 316. NtReadFile
; Exported entry 1149. ZwReadFile


public ZwReadFile
ZwReadFile proc near
mov     r10, rcx        ; NtReadFile
mov     eax, 3
syscall
retn
ZwReadFile endp
```

Figure – Fonction *NtReadFile* de *ntdll.dll*

System Service Dispatcher

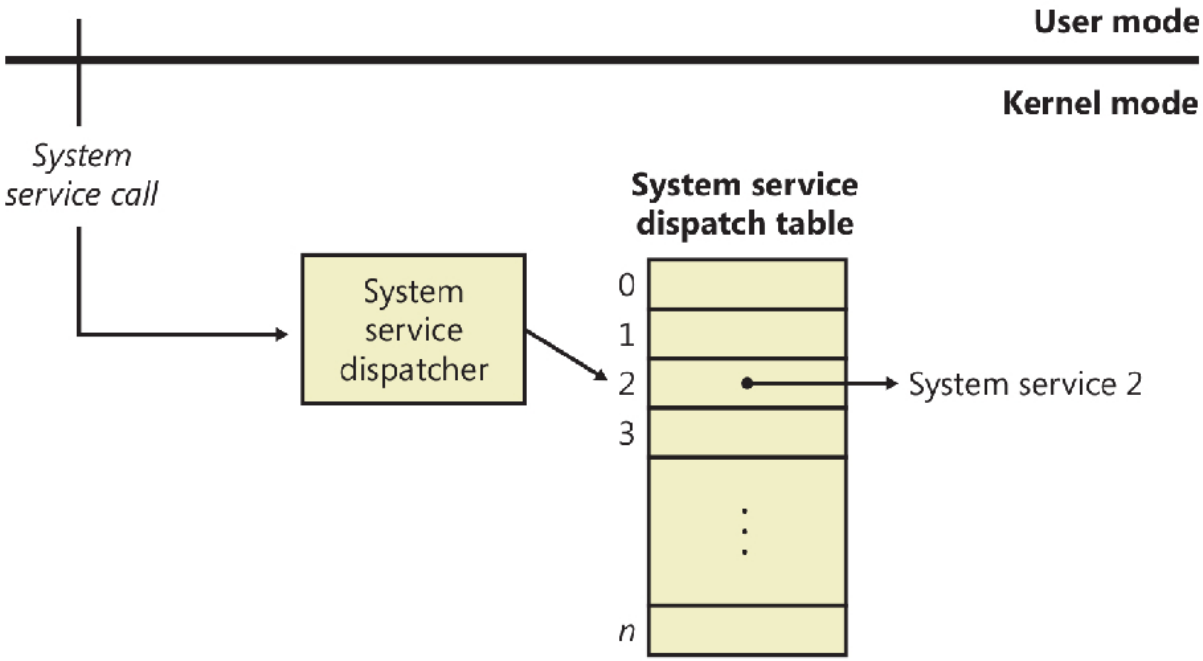
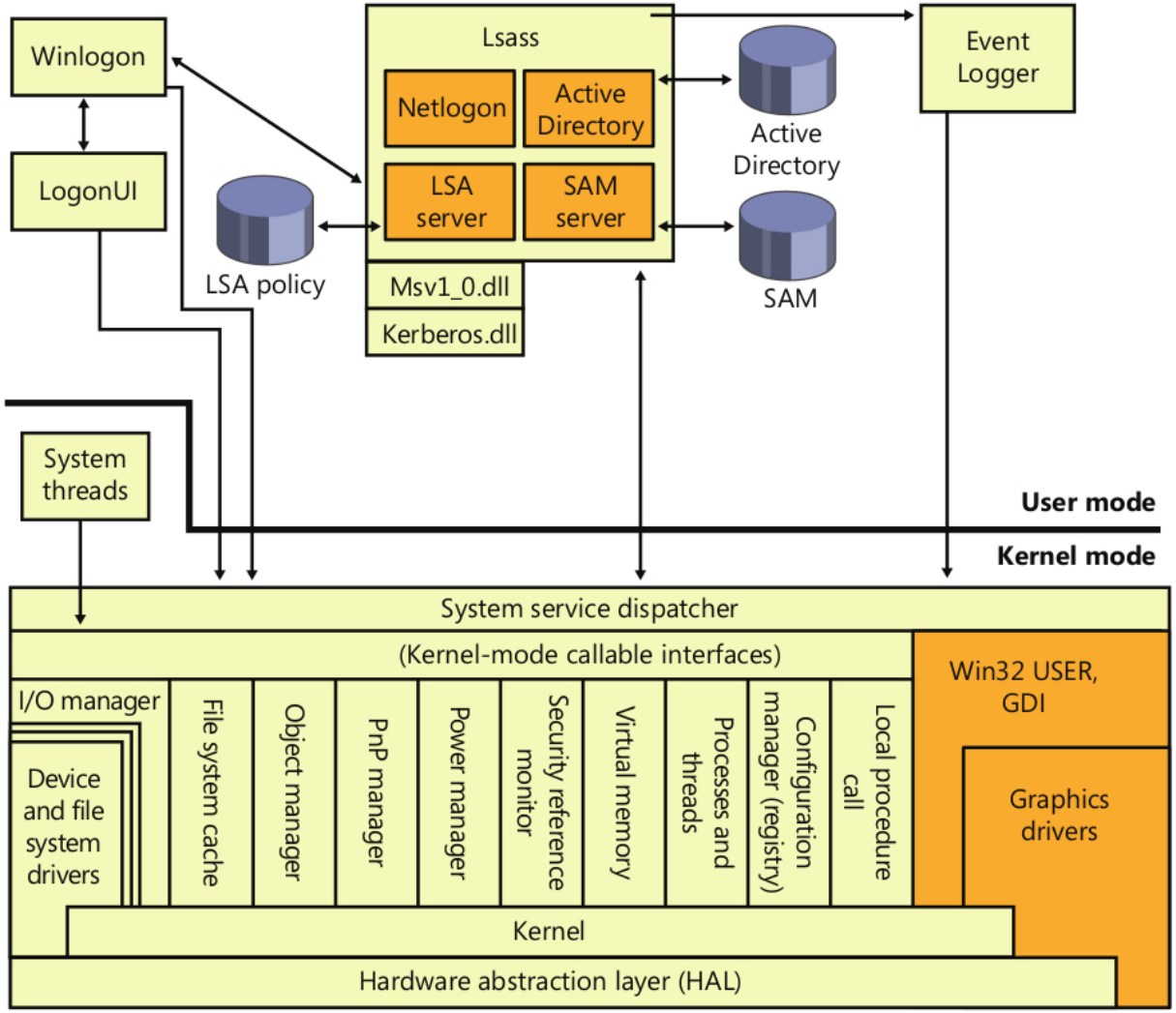


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Pilotes et modes d'exécution

- Les pilotes s'exécutant en mode noyau peuvent ;
  - entrainer l'arrêt complet du système en cas de *crash* ;
  - lire et écrire la mémoire de l'ensemble du système ;
- La compromission d'un pilote entraine potentiellement la compromission du système ;
- Un **système de signature de pilote** a donc été créé.

Composants de sécurité



Base de registre

- Il s'agit d'une **base de données** ;
- Elle contient :
  - la configuration du système ;
  - la configuration des applications ;
  - la configuration des profils utilisateur ;
  - des informations plus dynamiques (comme la liste des services actifs).
- Avant l'utilisation de la base de registre (avant Windows NT), le système et les applications utilisaient des fichiers INI (fichiers de configuration textuels).

Base de registre

- La structure de la base de registre est analogue à celle d'un système de fichier :
  - les clés de premier niveau ou clés racine correspondant aux racines ;
  - les clés et sous-clés correspondant aux dossiers :
    - une clé peut contenir des valeurs ou des sous-clés.
  - les valeurs correspondant aux fichiers :
    - contiennent des données ;
    - peuvent être de différents types (DWORD, STRING, BINARY, LINK...).

Base de registre

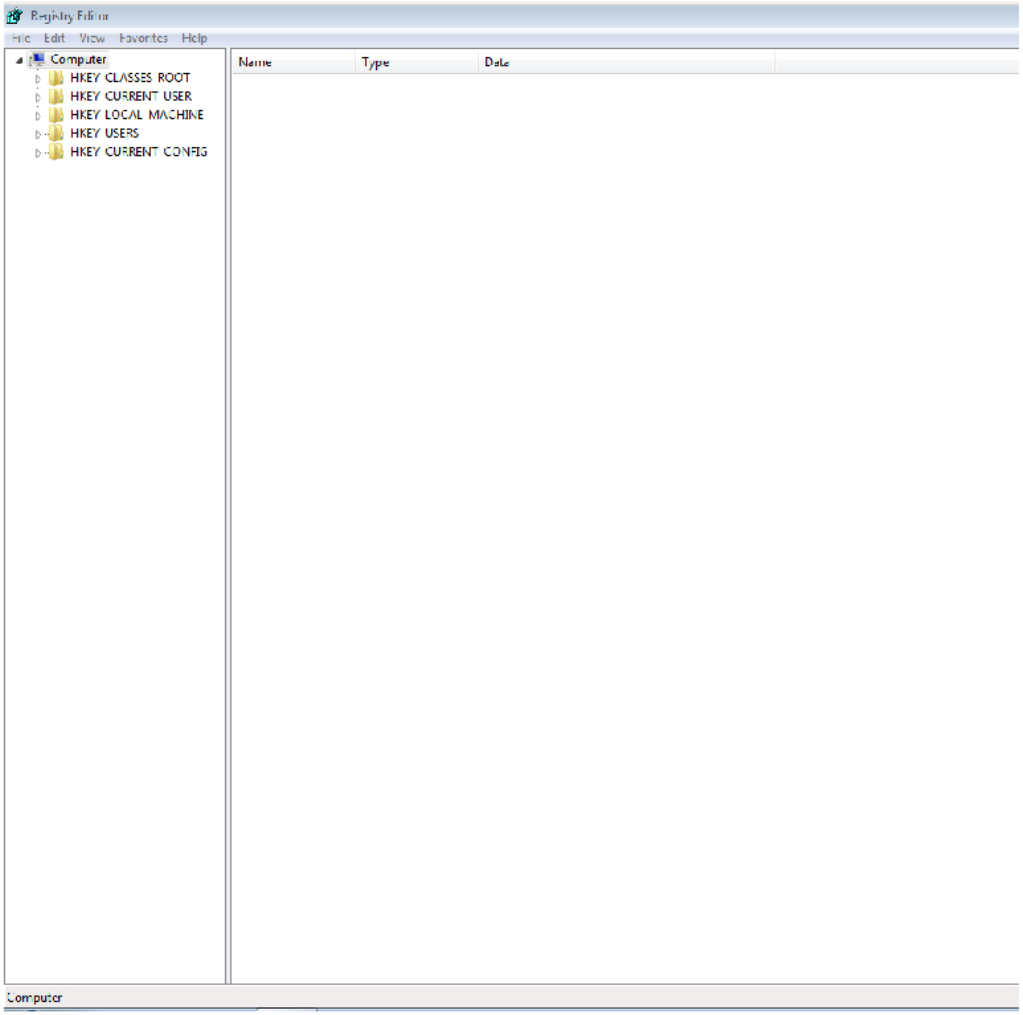


Figure – Visualisation des clés racine de la base de registre dans l'outil *regedit.exe*

Base de registre

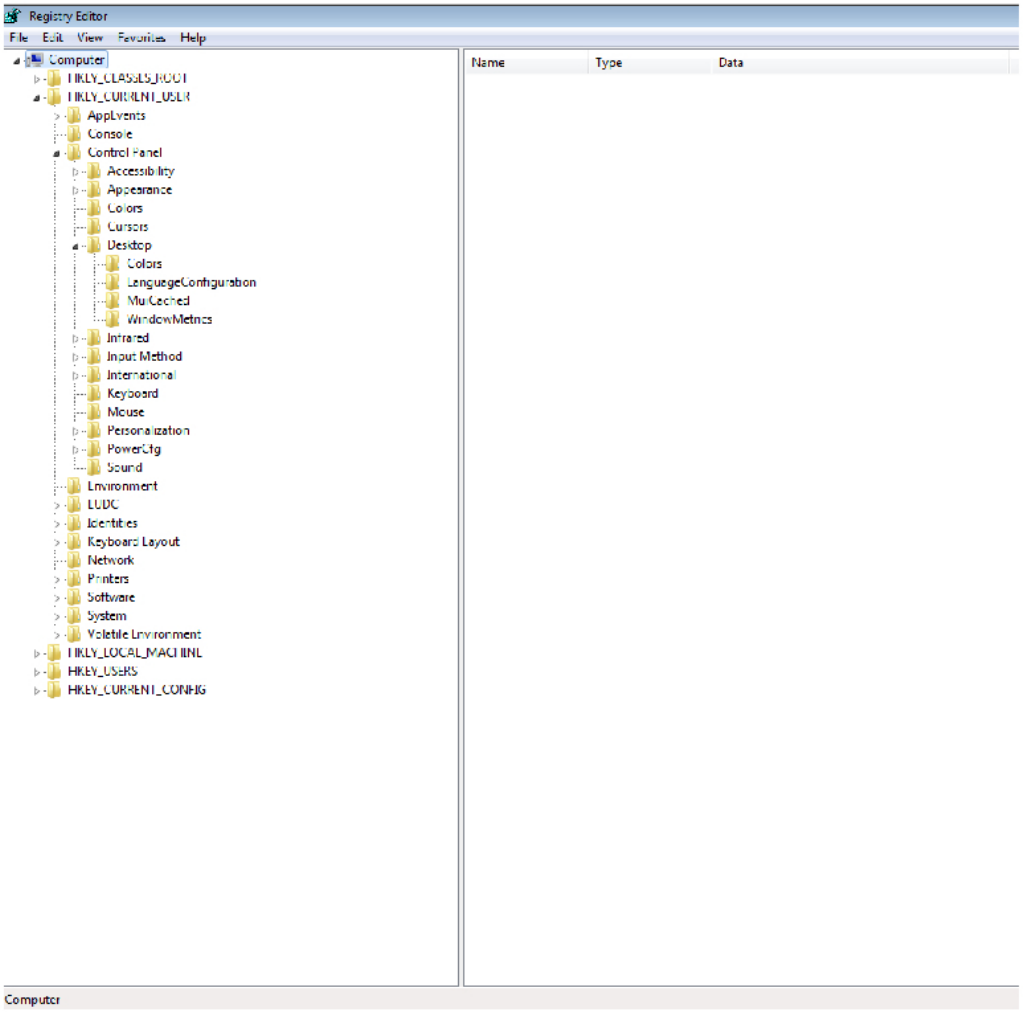


Figure – Visualisation de clés et sous-clés de la base de registre dans l'outil *regedit.exe*

Base de registre

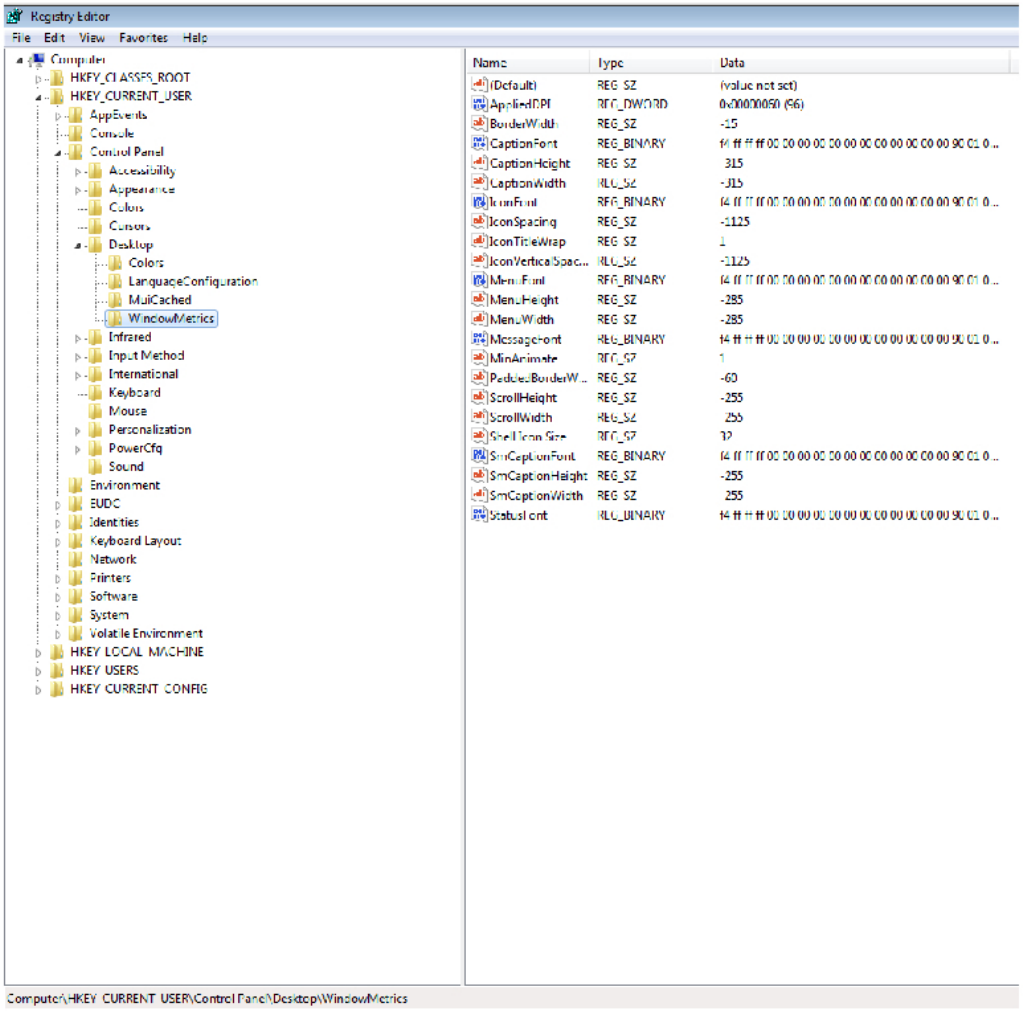


Figure – Visualisation de valeurs de la base de registre dans l'outil *regedit.exe*

Clés spécifiques

Clés volatiles

- Clés et valeurs de la base de registre qui peuvent disparaitre à tout moment ;
- Elles ne perdurent pas au-delà de l'extinction de la machine ;
- Elles sont créées et gérées en mémoire uniquement ;
- Exemple : `HKLM\HARDWARE`, qui stocke des informations sur les périphériques matériels et sur les ressources qui leur sont assignées.

Clés spécifiques

Clés racine

- Ces clés sont les racines de l'arborescence de la base de registre ;
- Il est impossible d'en ajouter ou d'en supprimer ;
- Il y a 6 clés racine :

Clé	Description
HKEY_CLASSES_ROOT	données d'associations de fichier et de déclarations d'objet COM
HKEY_CURRENT_CONFIG	données relatives au profil matériel courant
HKEY_CURRENT_USER	données associées à l'utilisateur courant
HKEY_LOCAL_MACHINE	informations de configuration spécifiques à l'ordinateur
HKEY_PERFORMANCE_DATA	données concernant les performances
HKEY_USERS	données concernant les profils utilisateur

- Les noms des clés racine ont une forme plus brève, par exemple HKCU pour HKEY_CURRENT_USER.

Base de registre

- Le stockage sur disque de la base de registre se fait à deux emplacements :
  - `%SystemRoot%\System32\Config` pour le système ;
  - `%UserProfile%\ntuser.dat` pour les utilisateurs.
- Chaque fichier correspond à une **ruche** ;
  - Les chemins de toutes les ruches sont configurées dans `HKLM\SYSTEM\CurrentControlSet\Control\hivelist` ;
- On ne peut pas ouvrir les fichiers de la base de registre pendant que le système s'exécute.
  - il faut passer par l'API spécifique pour modifier la base de registre ;
  - un mécanisme de synchronisation reportera les modifications de la mémoire aux fichiers.

Base de registre

- La base de registre est modifiée :
  - par le système : par exemple lors d'une modification de la configuration depuis le panneau de configuration ;
  - par l'utilisateur à l'aide d'outils ou directement à l'aide de l'API.
- Parmi les outils permettant de manipuler la base de registre on retrouve :
  - *Regedit* : un outil graphique ;
  - *Reg* : un outil en ligne de commande ;
  - *Powershell* : permet de manipuler la base de registre avec le confort du *scripting* ;
  - Les fichiers d'inscription : fichiers *.reg* à base de texte permettant la lecture et l'écriture de clés et valeurs ;
  - L'API Windows : tous les outils précédents utilisent l'API Windows, il est possible de l'utiliser directement. Il faut utiliser les fonctions commençant par *Reg* de *advapi32.dll* comme *RegOpenKey*.

Service

- Les services Windows sont des processus qui fonctionnent en **arrière-plan** et qui effectuent des tâches non attachées explicitement à l'utilisateur interactif ;
- Ils sont analogues aux ***daemons*** sous UNIX ;
- Exemple de service : le gestionnaire de journal d'événements ;
- Les services Windows sont constitués de trois composants :
  - une **application service** : ensemble logiciel dont au moins un composant fonctionne en tant que service Windows ;
  - un programme de contrôle de service (**SCP**, *Service Control Program*) ;
  - le gestionnaire de contrôle de service (**SCM**, *Service Control Manager*).

Service

SCM

Le gestionnaire de contrôle de service (*Service Control Manager*) est un processus système spécial responsable du démarrage, de l'arrêt et de l'interaction avec les processus de service.

- Il est lancé au *boot* du système ;
- Il exécute *Services.exe*.

SCP

Pour intéragir avec un service, tout utilisateur passe par un programme de contrôle de service (*Service Control Program*) ;

- Windows fournit des SCP avec des fonctionnalités génériques de démarrage, d'arrêt, de suspension et de reprise ;
- certaines applications service incluent leurs propres SCP.

Service

- Les services doivent être déclarer auprès du système ;
- La déclaration se fait avec la fonction *CreateService* de *advapi32.dll* ;
  - fonction qui a pour rôle de partager au SCM les 13 paramètres de configuration du service.
- Lors de la création du service, SCM crée une clé de registre pour le service sous `HKLM\SYSTEM\CurrentControlSet\Services` où sont stockés les paramètres ;
- Parmi les paramètres on trouve :
  - *ImagePath* : le chemin de fichier exécutable ;
  - *Desciption* : pour la description du service ;
  - *Type* : pour le type de service (interactif, à processus dédié, pilote...).

Service

- Les services peuvent être dans différents états (*running*, *stopped*, *paused*...) ;
- Il existe 3 modes de démarrage des services :
  - manuel : démarré seulement sur demande (comme le service de support Bluetooth) ;
  - automatique : démarré automatiquement par SCM au démarrage du système (comme le service Windows Firewall) :
  - différé : démarré automatiquement avec un délai (comme le service Windows Defender) :

Service

- Dans la configuration par défaut, trois comptes sont disponibles pour les service :
  - Système local ;
  - Service réseau ;
  - Service local.
- Il est aussi possible de configurer un service pour qu'il soit soit exécuté sous un **compte de remplacement** (en spécifiant un compte et un mot de passe).

Tâches planifiées

Elles sont analogues aux ***crons*** sous UNIX.

- Elles nécessitent qu'un utilisateur ait ouvert une session ;
- Elles sont déclenchées en fonction d'un calendrier ou par des conditions système (démarrage, connexion d'un utilisateur...) ;
- Elles sont gérées par le **planificateur de tâches** (qui est lui-même un service) ;
- exemple de tâche : le scan quotidien de Windows Defender.

Outils

- Les services peuvent être administrés via :
  - l'outil d'administration Services (*Services.msc*) de la console MMC ;
  - l'onglet Services du Gestionnaire des tâches ;
  - la commande *sc.exe*.
- Les tâches planifiées peuvent être administrées via :
  - le planificateur de tâches (outil graphique) ;
  - la commande *schtasks* (remplaçant de la commande *at*).

Notion d'objet

- Les ressources qu'il faut partager, protéger, nommer ou rendre visibles aux programmes en mode utilisateur sont représentées comme des **objets** au niveau du noyau ;
- Un objet contient les propriétés type d'une même catégorie d'entités ;
  - par exemple l'objet de type processus contient toutes les informations de contrôle requises pour sa gestion (comme sa liste de *threads*).
- Entre autres, sont des objets :
  - les processus ;
  - les fichiers ;
  - les périphériques ;
  - les pilotes ;
  - les clés de registre.
- Ces objets sont manipulés indirectement à l'aide d'***handles*** par les processus en mode utilisateur.

Notion d'objet

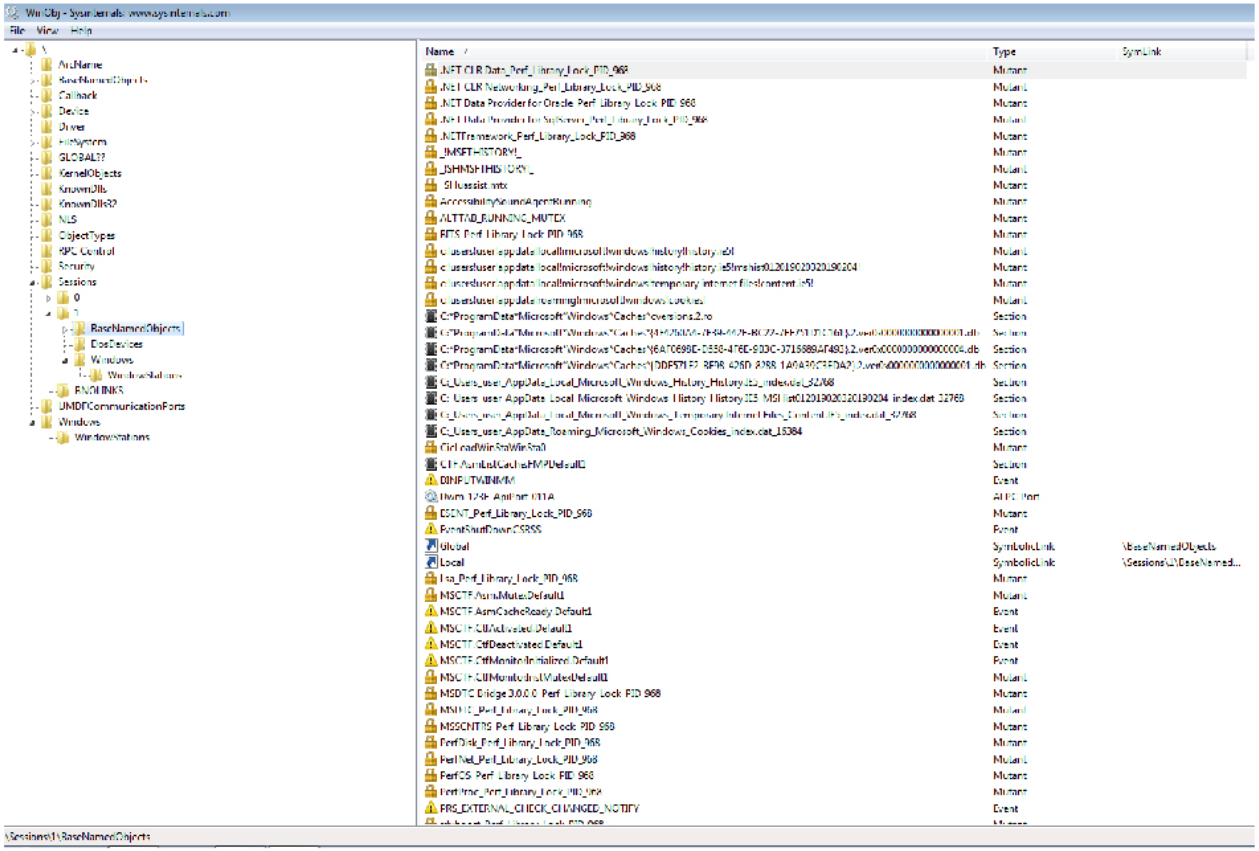


Figure – Utilisation de *WinObj* de la suite *Sysinternals*

Notion d'objet

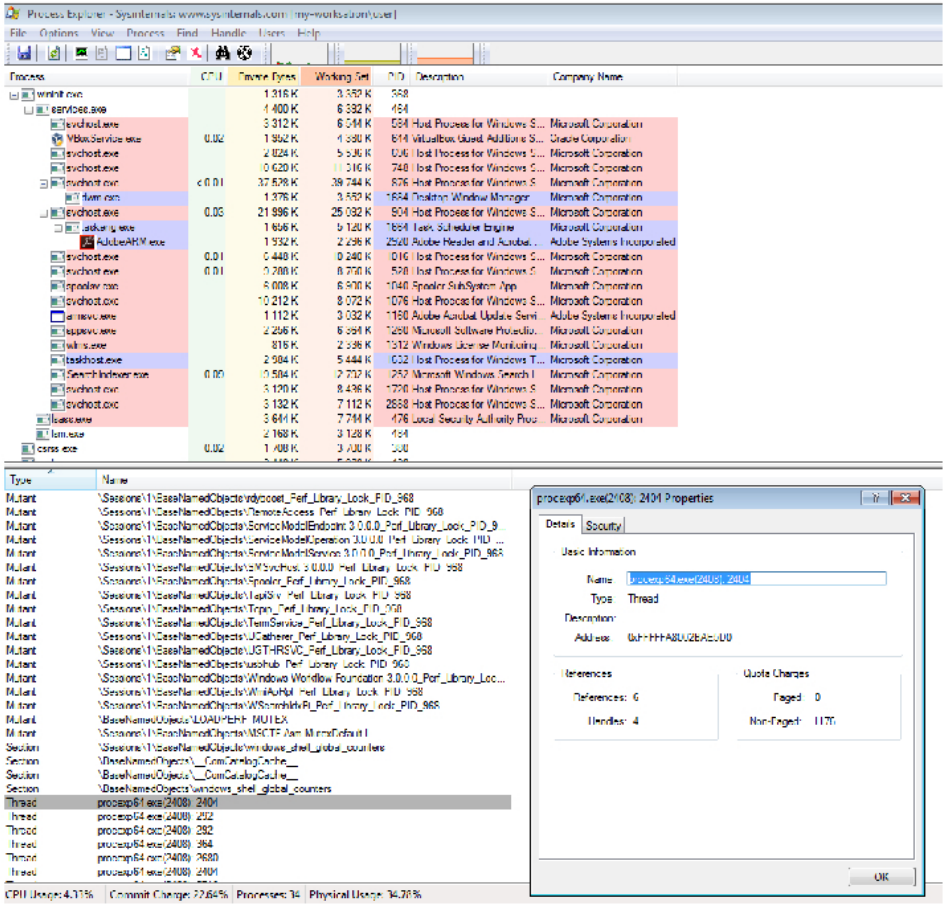


Figure – Utilisation de *procexp* de la suite *Sysinternals* pour visualiser les *handles*

Sommaire

1 Introduction

2 Bases

3 Contrôle d'accès

4 Authentification

5 Mécanismes de protection

Introduction

- Windows met en place un système de contrôle d'accès aux objets ;
- Chaque objet bénéficiant d'un contrôle d'accès possède un **descripteur de sécurité** ;
  - spécifie quelles opérations peuvent être faites par quels utilisateurs et quelle information sera enregistrée dans le journal d'audit de la sécurité ;
  - exemples d'objets possédant un descripteur de sécurité : les fichiers, les dossiers, les processus.
- Les utilisateurs (en réalité leurs processus) possèdent un **jeton d'accès** ;
  - représente le contexte de sécurité (privilèges, groupes...).
- Quand un processus veut obtenir un *handle* pour un objet, le SRM (*Security Reference Monitor*) :
  - vérifie le jeton de sécurité du processus et le descripteur de sécurité de l'objet ;
  - détermine si le processus a des droits d'accès suffisants pour mener à bien l'opération.

SID

- Un SID est un identifiants de sécurité (*Security Identifiers*) ;
- Tout ce qui peut être authentifié par le système d'exploitation a un SID qui lui est associé ;
  - c'est-à-dire les utilisateurs et leurs processus, les groupes locaux, les groupes de domaine, les ordinateurs locaux...
- Permet d'identifier de manière **unique** les entités d'un système ;
- Windows se base dessus pour le contrôles d'accès.

SID

- Un SID est une valeur numérique de longueur variable ;
- Structure d'un SID :

```
S-1-A-Sub1-...-SubN-RID
```

  - S : indique que c'est un SID (fixe) ;
  - 1 : numéro de version ;
  - A : autorité ayant généré le SID, généralement un système local ou un domaine Windows ;
  - Sub1-...-SubN : sous-autorité du SID ;
  - RID : identifiant relatif.

SID

- Exemple : S-1-5-21-3535373721-3146749226-1307819366-1000
  - S-1 : version 1 du SID ;
  - 5 : correspond à l'autorité `SECURITY_NT_AUTHORITY`, indique qu'il s'agit d'un SID Windows ;
  - 21 : première sous-autorité correspondant à `SECURITY_NT_NON_UNIQUE`, indiquant que les sous-autorités suivantes représentent un domain ;
  - 3535373721-3146749226-1307819366 : sous-autorités correspondant à un domaine local ;
  - 1000 : RID du compte (premier utilisateur créé).

SID

- Exemple : **S-1**-5-21-3535373721-3146749226-1307819366-1000
  - **S-1 : version 1 du SID ;**
  - 5 : correspond à l'autorité `SECURITY_NT_AUTHORITY`, indique qu'il s'agit d'un SID Windows ;
  - 21 : première sous-autorité correspondant à `SECURITY_NT_NON_UNIQUE`, indiquant que les sous-autorités suivantes représentent un domain ;
  - 3535373721-3146749226-1307819366 : sous-autorités correspondant à un domaine local ;
  - 1000 : RID du compte (premier utilisateur créé).

SID

- Exemple : S-1-**5**-21-3535373721-3146749226-1307819366-1000
  - S-1 : version 1 du SID ;
  - **5 : correspond à l'autorité `SECURITY_NT_AUTHORITY`, indique qu'il s'agit d'un SID Windows ;**
  - 21 : première sous-autorité correspondant à `SECURITY_NT_NON_UNIQUE`, indiquant que les sous-autorités suivantes représentent un domain ;
  - 3535373721-3146749226-1307819366 : sous-autorités correspondant à un domaine local ;
  - 1000 : RID du compte (premier utilisateur créé).

SID

- Exemple : S-1-5-**21**-3535373721-3146749226-1307819366-1000
  - S-1 : version 1 du SID ;
  - 5 : correspond à l'autorité `SECURITY_NT_AUTHORITY`, indique qu'il s'agit d'un SID Windows ;
  - **21 : première sous-autorité correspondant à `SECURITY_NT_NON_UNIQUE`, indiquant que les sous-autorités suivantes représentent un domain ;**
  - 3535373721-3146749226-1307819366 : sous-autorités correspondant à un domaine local ;
  - 1000 : RID du compte (premier utilisateur créé).

SID

- Exemple : S-1-5-21-**3535373721-3146749226-1307819366**-1000
  - S-1 : version 1 du SID ;
  - 5 : correspond à l'autorité `SECURITY_NT_AUTHORITY`, indique qu'il s'agit d'un SID Windows ;
  - 21 : première sous-autorité correspondant à `SECURITY_NT_NON_UNIQUE`, indiquant que les sous-autorités suivantes représentent un domain ;
  - **3535373721-3146749226-1307819366 : sous-autorités correspondant à un domaine local ;**
  - 1000 : RID du compte (premier utilisateur créé).

SID

- Exemple : S-1-5-21-3535373721-3146749226-1307819366-**1000**
  - S-1 : version 1 du SID ;
  - 5 : correspond à l'autorité `SECURITY_NT_AUTHORITY`, indique qu'il s'agit d'un SID Windows ;
  - 21 : première sous-autorité correspondant à `SECURITY_NT_NON_UNIQUE`, indiquant que les sous-autorités suivantes représentent un domain ;
  - 3535373721-3146749226-1307819366 : sous-autorités correspondant à un domaine local ;
  - **1000 : RID du compte (premier utilisateur créé).**

SID

Well-known SID :

SID	Description
S-1-1-0	Tout le monde
S-1-5-7	Anonyme
S-1-5-11	Utilisateurs authentifiés
S-1-5-18	Système local
S-1-5-domain-500	Administrateur local
S-1-5-domain-501	Guest
S-1-5-32-544	Groupe Administrateurs
S-1-5-32-545	Groupe Utilisateurs

C:\Windows\system32\cmd.exe

```
C:\Users\user\Desktop\SysinternalsSuite>psgetsid system

PsGetSid v1.45 - Translates SIDs to names and vice versa
Copyright (C) 1999-2016 Mark Russinovich
Sysinternals - www.sysinternals.com

SID for NT AUTHORITY\system:
S-1-5-18


C:\Users\user\Desktop\SysinternalsSuite>psgetsid Administrators

PsGetSid v1.45 - Translates SIDs to names and vice versa
Copyright (C) 1999-2016 Mark Russinovich
Sysinternals - www.sysinternals.com

SID for BUILTIN\Administrators:
S-1-5-32-544


C:\Users\user\Desktop\SysinternalsSuite>psgetsid Guest

PsGetSid v1.45 - Translates SIDs to names and vice versa
Copyright (C) 1999-2016 Mark Russinovich
Sysinternals - www.sysinternals.com

SID for my-worksation\Guest:
S-1-5-21-3857084477-1910035935-2444439189-501
```

Figure – Utilisation de *psgetsid* de la suite *Sysinternals*

Jeton d'accès

- Le jeton d'accès (*access token*) identifie le contexte de sécurité d'un processus (d'un utilisateur) ;
  - on y retrouve notamment les privilèges et les groupes.

- Ce jeton est généré au cours de l'authentification par *Winlogon* qui l'attache au premier processus de l'utilisateur ;
- Tous les programmes qu'exécute l'utilisateur héritent d'une copie du jeton initial ;

- Il est possible de générer un jeton avec la fonction *LogonUser* et de lancer un processus avec ce jeton avec la fonction *CreateProcessAsUser* ;
- La fonction *CreateProcessWithLogon* permet de faire les deux actions en une fonction.
  - notamment utilisé par *runas*.

Composition d'un jeton d'accès

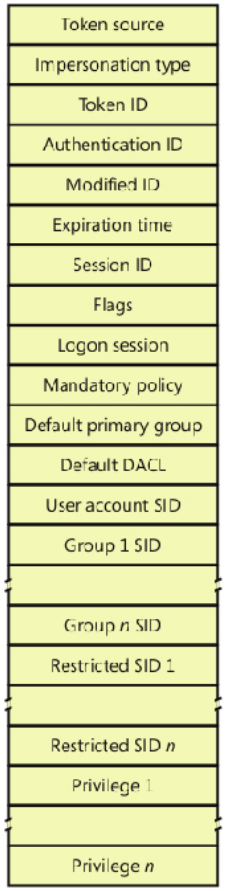


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Jeton d'accès

- Quelques attributs notables du jeton d'accès :
  - *Default Primary Group* et *Default Discretionary Access Control* (DACL) définissent les permissions par défaut des objets qui seront créés par le processus ;
    - équivalent au *mask* sous UNIX.
  - *Expiration time* n'est pas supporté pour l'instant.

Privilèges

- Un privilège est le droit d'un compte de faire des actions systèmes sur la machine ;
  - concerne les actions qui ne sont pas relative à un objet en particulier.
- Exemple de privilège :
  - *SeLoadDriverPrivilege* : autorise le chargement et déchargement des pilotes de périphériques ;
  - *SeSystemtimePrivilege* : autorise le changement de l'heure du système.

Privilèges

- Ces privilèges peuvent être *enabled* et *disabled* ;
- Certains privilèges permettent de prendre le contrôle d'une partie ou de tout le système ;
  - Comme par exemple *SeTakeOwnershipPrivilege* ou encore *SeDebugPrivilege*.

Privilèges

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\John>whoami /priv

PRIVILEGES INFORMATION
----------------------

Privilege Name                Description                           State
============================= ===================================== ========
SeShutdownPrivilege           Shut down the system                  Disabled
SeChangeNotifyPrivilege       Bypass traverse checking              Enabled
SeUndockPrivilege             Remove computer from docking station  Disabled
SeIncreaseWorkingSetPrivilege Increase a process working set        Disabled
SeTimeZonePrivilege           Change the time zone                  Disabled

C:\Users\John>_
```

Figure – Visualisation de privilèges avec *whoami*

Droits utilisateur

- Les droits utilisateur (*user rights* ou *logon rights*) peuvent être considérés comme des privilèges qui concerne les types d'ouvertures de session autorisés ;
- Ils sont vérifiés par LSA ;
- Contrairement aux privilèges :
  - ils ne peuvent pas être *enabled* et *disabled* ;
  - étant utilisés seulement pour l'ouverture de session, ils ne sont pas présent dans le jeton d'accès.
- Exemple de droits utilisateur :
  - *SeInteractiveLogonRight* : autorise la connexion locale interactive ;
  - *SeDenyServiceLogonRight* : interdit la connexion en tant que service.

Droits utilisateur

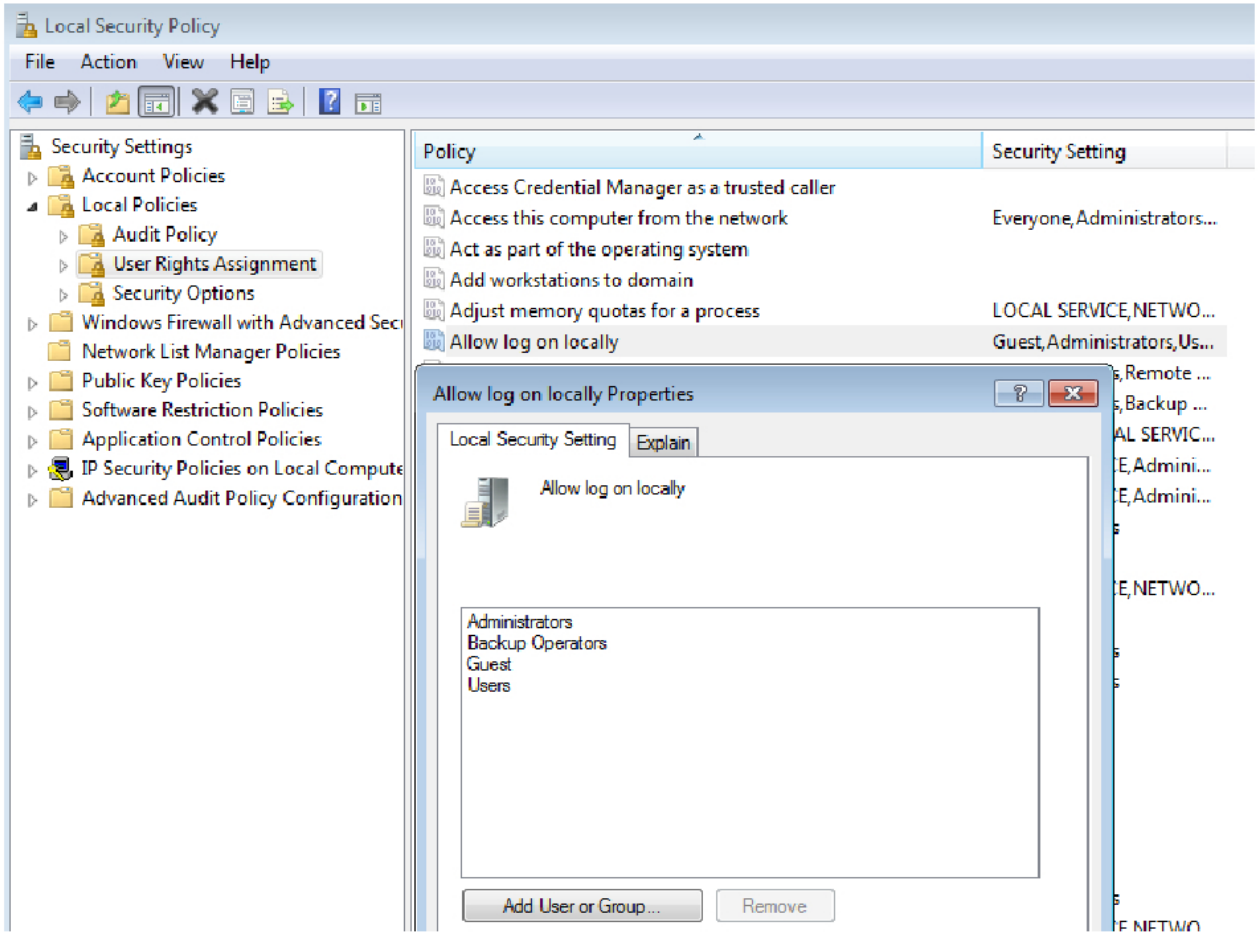


Figure – Edition des droits utilisateur avec *secpol.msc*

Jeton d'accès

- Il existe deux types de jetons :
  - *Primary token* : jeton qui décrit le contexte de sécurité de l'utilisateur associé au processus ;
  - *Impersonation token* : jeton utilisé pour prendre temporairement une autre identité.
- L'impersonation se fait au niveau du *thread* ;
- Les *threads* effectuant une impersonation ont à la fois :
  - un *primary token* (celui du processus) ;
  - un *impersonation token*.
- Souvent utilisé dans le cadre client/serveur où le serveur accède à des objets au nom du client.

Jeton restreint

- Un jeton restreint est un *primary token* ou *impersonation token* qui a été modifié par la fonction *CreateRestrictedToken* de *advapi32.dll* ;
- Un jeton restreint permet de limiter l'accès aux objets et les privilèges ;
- La fonction *CreateRestrictedToken* permet :
  - d'enlever des privilèges ;
  - de marquer des SID avec l'attribut DENY-ONLY ;
    - ne considère ces SID que pour les règles empêchant l'accès.
  - d'ajouter des SID à la liste de *restricted* SID.
    - permet de restreindre la liste de SID du jeton (sans les supprimer) ;
    - pour accéder à un objet il faut obtenir le droit d'accès pour la liste de SID <u>et</u> pour la liste des *restricted* SID.

Jeton d'accès

- Pour visualiser un jeton d'accès il est possible d'utiliser :
  - la commande *whoami /all* ;
  - *procexp* de *Sysinternals* ;
  - la commande *!token* dans le *kernel debugger*.

Jeton d'accès

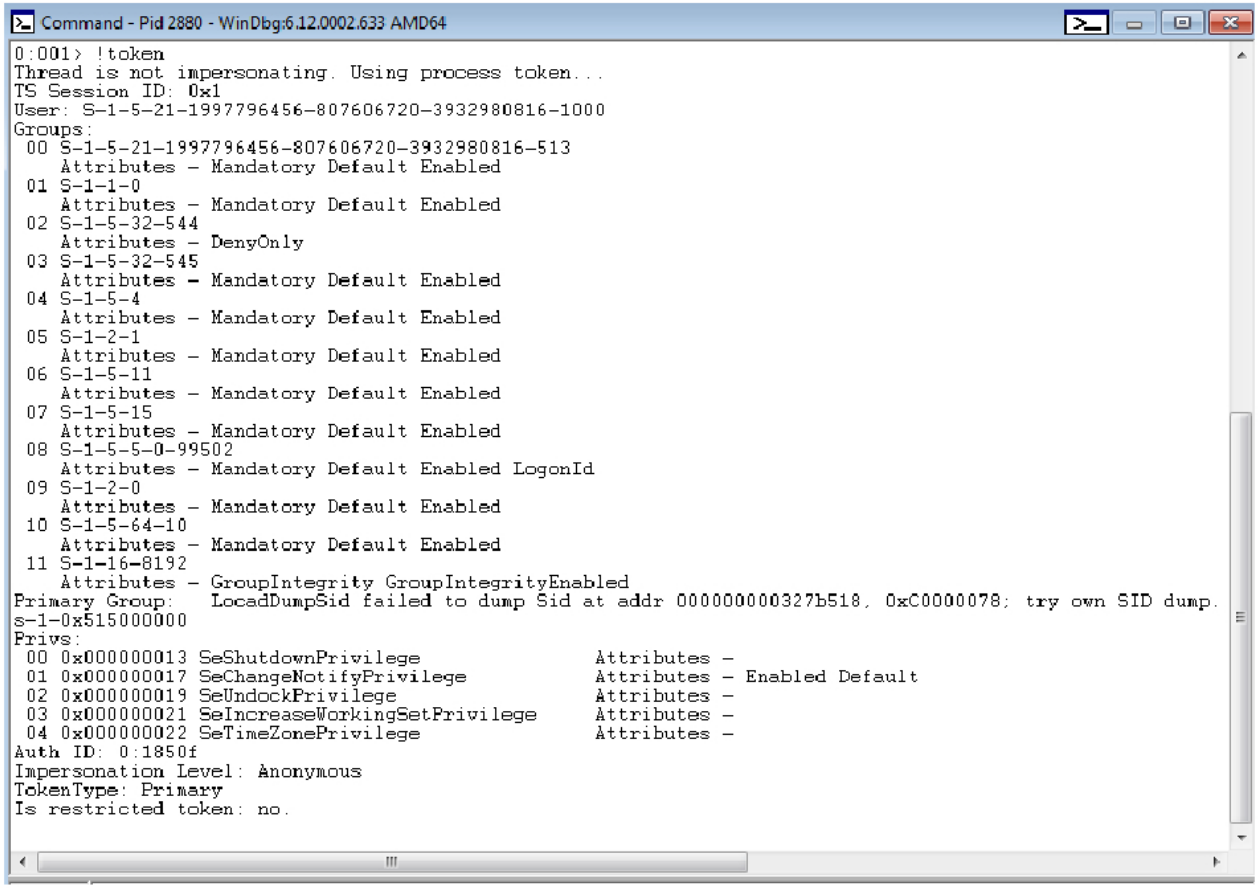


Figure – Visualisation d'un jeton avec *WinDBG*

Descripteur de sécurité

- On a vu que chaque processus possède un jeton d'accès ;
- Pour mener à bien le contrôle d'accès, SRM confronte ce jeton aux descripteurs de sécurité des objets ;

- Chaque objet qui peut être protégé par le mécanisme de contrôle d'accès possède un descripteur de sécurité ;
- Ces descripteurs de sécurité définissent **qui peut faire quoi** sur l'objet auquel ils sont attachés ;
  - ils définissent aussi quelles informations seront enregistrées dans le journal d'audit de la sécurité.
- Exemples d'objets avec des descripteurs de sécurité : les fichiers, les dossiers, les processus ou encore les jetons d'accès.

Descripteur de sécurité

- Un descripteur de sécurité est stocké dans la structure `SECURITY_DESCRIPTOR` qui contient les attributs :
  - *Revision number* : la version du modèle de sécurité du SRM utilisé pour créer le descripteur ;
  - *Flags* : un champ de bits qui définit le comportement du descripteur ;
  - *Owner SID* : le SID du propriétaire ;
  - *Group SID* : le SID du groupe primaire (utilisé uniquement par POSIX) ;
  - DACL : *Discretionary access control list*, spécifie qui a quel accès à l'objet ;
  - SACL : *System access control list*, spécifie quelles opérations seront enregistrées dans le journal d'événements de sécurité.

Descripteur de sécurité

- L'attribut *Flags* est un champ de bits qui définit le comportement du descripteur ;
- Par exemple, le bit `SE_DACL_PRESENT` (0x0004) indique si le descripteur de sécurité à une DACL :
  - si ce bit est à 0, le descripteur de sécurité **donne l'accès complet à tout le monde**.

Descripteur de sécurité

- L'attribut *Owner SID* contient le SID du propriétaire du descripteur de sécurité ;
- Le priopriétaire **peut toujours modifier les permissions** (c'est-à-dire les DACL) d'un objet ;
  - et ce même si les permissions de l'objet bloquent tout accès au prioriétaire.
- Peuvent modifier l'attribut *Owner SID* :
  - de tous les objets :
    - les utilisateurs ayant le privilège *SeTakeOwnershipPrivilege* (cas de l'administrateur) ;
    - les utilisateurs ayant le privilège *SeRestorePrivilege* (cas de l'administrateur).
  - les utilisateurs ayant la permission *Take ownership* sur l'objet ;
  - le propriétaire de l'objet.

ACL

ACL

Une liste de contrôle d'accès (ACL, *Access Control List*) est une collection d'ACE.

ACE

Les entrées de contrôle d'accès (ACE, *Access Control Entry*) associent un compte (un utilisateur ou un groupe) à une règle définissant les autorisations ou interdictions sur une ressource.

DACL

Une liste de contrôle d'accès discrétionnaire (DACL, *Discretionary Access Control List*) gère les permissions sur un objet, spécifiant qui est autorisé et à qui il est interdit de faire certains actions sur un objet.

SACL

Une liste de contrôle d'accès système (SACL, *System Access Control List*) contrôle la génération de messages d'audit lors des tentatives d'accès à un objet sécurité.

ACL

- Une ACE est composée de :
  - un SID qui identifie l'ayant droit (utilisateur ou groupe) auquel l'ACE s'applique ;
  - un masque d'accès qui détermine les différents types d'accès possibles (comme lecture, écriture) ;
    - dépend du type de l'objet.
  - le type de l'ACE (autorise ou interdit) ;
  - différents flags définissant les règles d'héritage de l'ACE.

ACL

- Certains types d'objet ont des objets conteneur et des objets contenu ;
  - comme les fichiers et les dossiers.
- Dans ce cas, en manipulant les flags d'héritage des ACE d'un objet conteneur, il est possible de faire hériter un objet contenu de certaines ACE de l'objet conteneur.

ACL

- La représentation textuelle des ACL est faite avec SDDL (*security descriptor definition language*) ;
- Syntaxe pour une ACL :

  `acl_header(ace_1)(ace_2)...`
- Syntaxe pour une ACE :

  `ace_type;ace_flags;rights;object_guid;inherit_object_guid;account_sid;`

ACL

- La partie `acl_header` indique le type d'ACL :
  - ”D :” pour les DACL ;
  - ”S :” pour les SACL.
- Ainsi que le comportement face à l'héritage :
  - *PAI* pour bloquer l'héritage (`SDDL_PROTECTED`) ;
  - *AI* pour authoriser l'héritage (`SDDL_AUTO_INHERITED`).

ACL

- Quelques valeurs possibles de `ace_type` :

Chaîne de caratère	type d'ACE
"A"	`ACCESS_ALLOWED_ACE_TYPE`
"D"	`ACCESS_DENIED_ACE_TYPE`
"AL"	`SYSTEM_ALARM_ACE_TYPE`
"OU"	`SYSTEM_AUDIT_OBJECT_ACE_TYPE`
"OL"	`SYSTEM_ALARM_OBJECT_ACE_TYPE`

- Quelques valeurs possibles de `ace_flags` :

Chaîne de caratère	description
”CI”	les objets enfants qui sont des conteneurs héritent de l’ACE
”OI”	les objets enfants qui ne sont pas des conteneurs héritent de l’ACE
”ID”	l’ACE est héritée

ACL

- Quelques valeurs possibles de `rights`

Chaîne de caratère	description
”GA”	`GenericALL`
”GR”	`GenericREAD`
”GW”	`GenericWRITE`
”GX”	`GenericEXECUTE`
”WO”	`WRITE_OWNER`
”FA”	`FILE_ALL_ACCESS`
”FR”	`FILE_GENERIC_READ`

ACL

- Le champ `account_sid` peut être un SID ou un *SID string* :

SID string	description
”BA”	Groupe Administrateurs (*built-in*)
”SY”	Système local
”WD”	Tout le monde

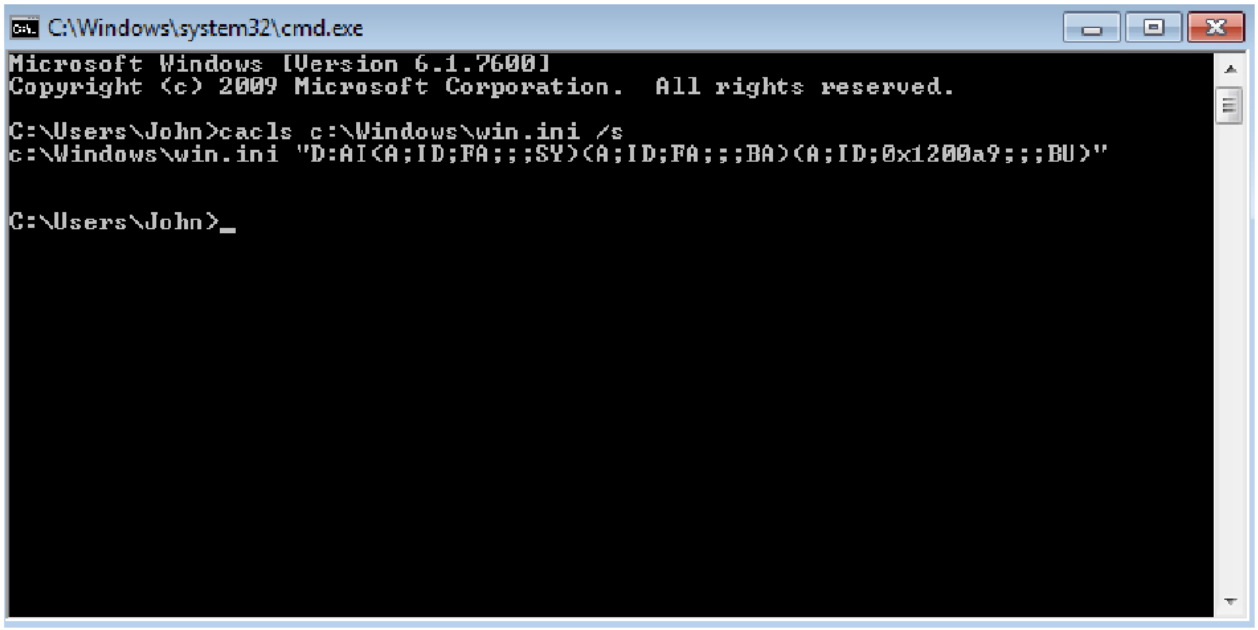


Figure – Visualisation des DACL d'un fichier avec *cacls* sous forme SDDL

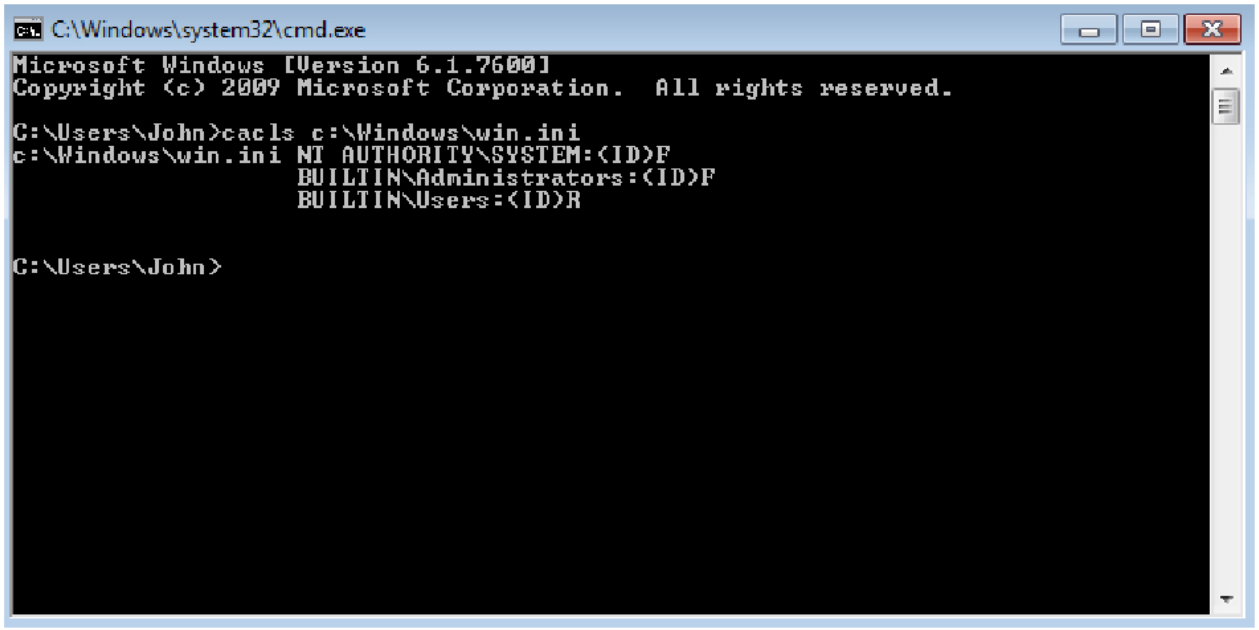


Figure – Visualisation des DACL d'un fichier avec *cacls*

Descipteur de sécurité

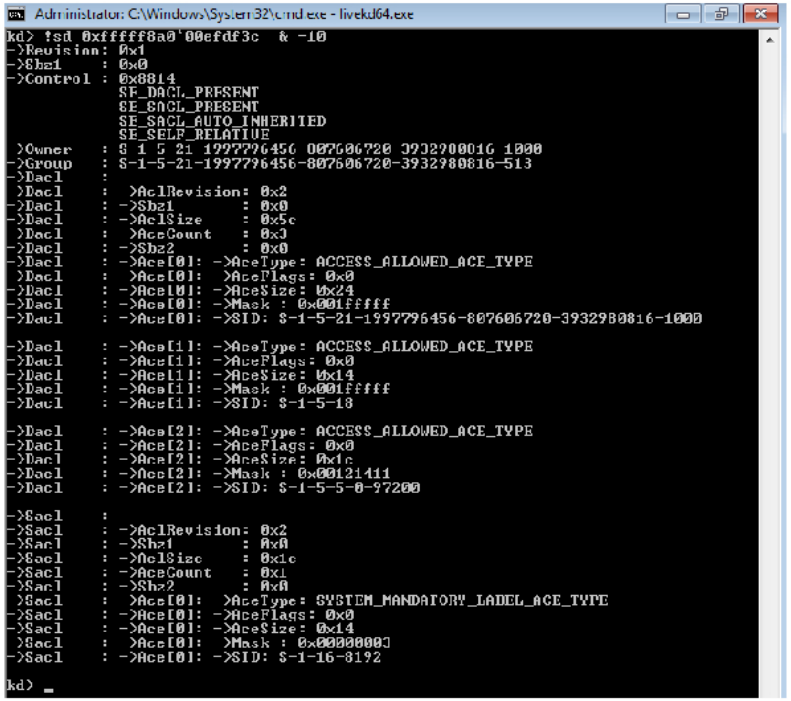


Figure – Visualisation d'un descripteur de sécurité avec la commande *!sd* du *kernel debugger*

Descipteur de sécurité

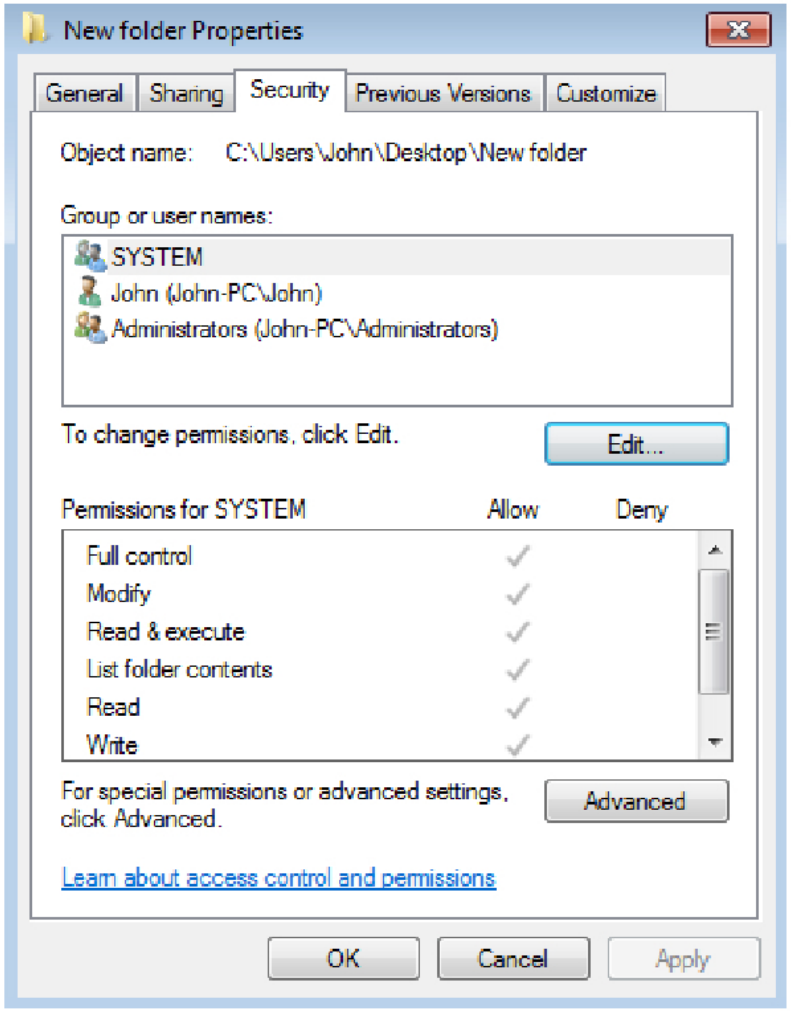


Figure – Il est possible de visualiser et modifier le descripteur de sécurité graphiquement

Sommaire

1 Introduction

2 Bases

3 Contrôle d'accès

4 Authentification

5 Mécanismes de protection

Introduction

- Nous allons voir :
  - les étapes d'initialisation de *Winlogon* ;
  - les étapes de l'authentification d'un utilisateur ;
  - Les différentes méthodes d'authentification ;
  - Le stockage des secrets d'authentification.

Composants impliqués dans l'authentification

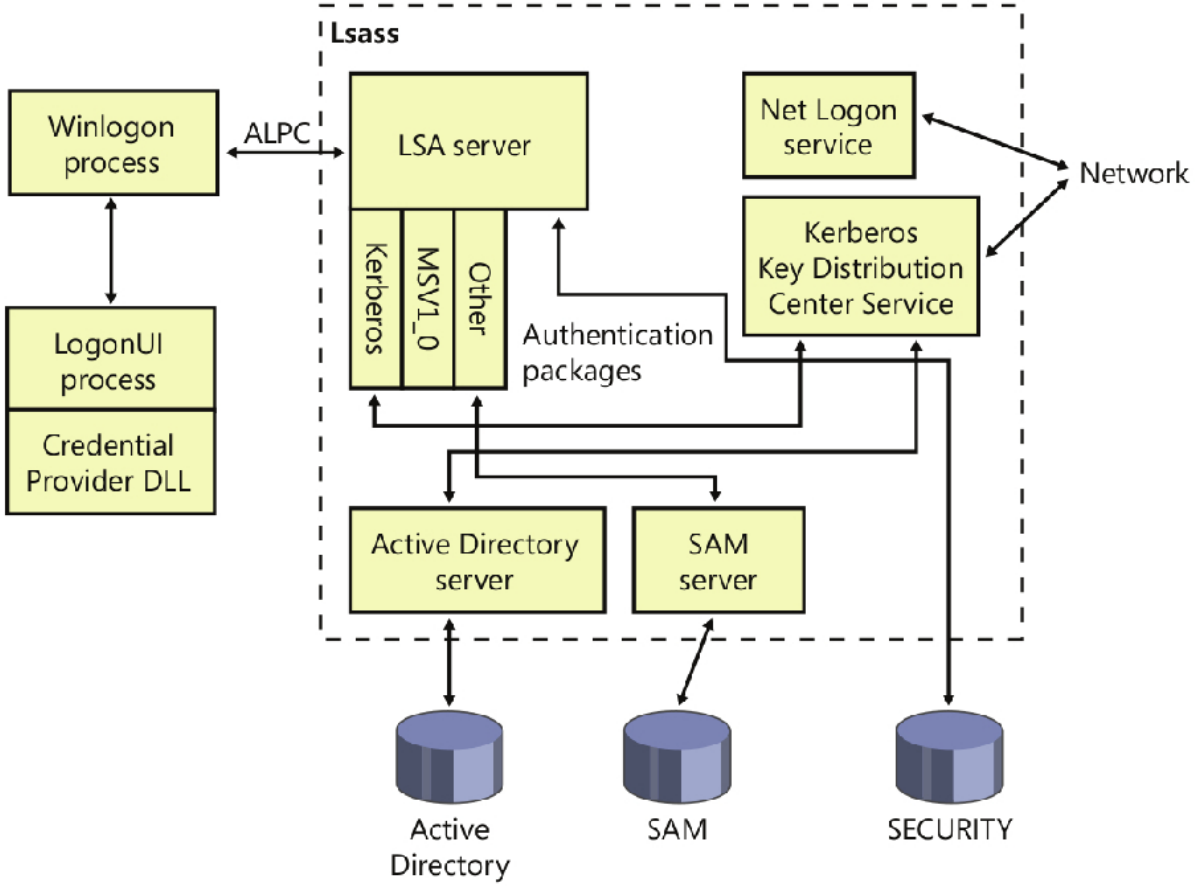


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Initialisation de *Winlogon*

- Le mécanisme d'authentification est orchestré par *Winlogon* ;
- L'initialisation de *Winlogon* se fait en plusieurs étapes :
  0. Le gestionnaire de sessions (*Smss*) démarre *Winlogon* ;
     - à lieu pendant le démarrage de Windows.
  1. *Winlogon* créé un objet *Window Stations* interactif ;
     - cet objet représente l'écran, le clavier et la souris ;
     - `Sessions\1\Windows\WindowStations\WinSta0`.
  2. *Winlogon* créé deux objets *Desktop* ;
     - un *desktop* d'application (`\Sessions\1\Windows\WinSta0\Default`) ;
     - un *desktop* sécurisé (`\Sessions\1\Windows\WinSta0\Winlogon`), seulement accessible par *Winlogon* ;
     - avant la première connexion, le *desktop Winlogon* est visible ;
     - la combinaison *Ctrl+Alt+Suppr* fait basculer du *desktop* d'application vers le *desktop Winlogon*.

Initialisation de *Winlogon*

- Suite :
  3. *Winlogon* se connecte à LSASS avec la fonction *LsaRegisterLogonProcess* ;
     - il s'agit d'une connexion via ALPC ;
     - cette connexion sert à échanger des informations pendant le *logon*, le *logoff* et pour les opérations concernant les mots de passe.
  4. *Winlogon* enregistre son serveur RPC qui écoute, entre autres, les notifications de SAS provenant de *Win32k*.

SAS

- SAS (*Secure Attention Sequence*) est une combinaison de touches spéciale (*Ctrl+Alt+Suppr*) ;
- Auncune application ne peut intercepter la combinaison ou empêcher *Winlogon* de la recevoir :
  - *Winlogon* utilise la fonction *RegisterHotKey* pour réserver *Ctrl+Alt+Suppr* ;
  - Seul le *thread* qui a réservé une combinaison peut la relâcher avec *UnregisterHotKey* ;
  - *SetWindowsHook* permet d'intercepter toutes les séquences de touches sauf *Ctrl+Alt+Suppr* ;

> Une application malveillante ne peut pas prendre le contrôle de l'écran quand il y a un SAS.
  - pourrait par exemple être utilisé pour récupérer les mots de passe en clair.

Composants impliqués dans l'authentification

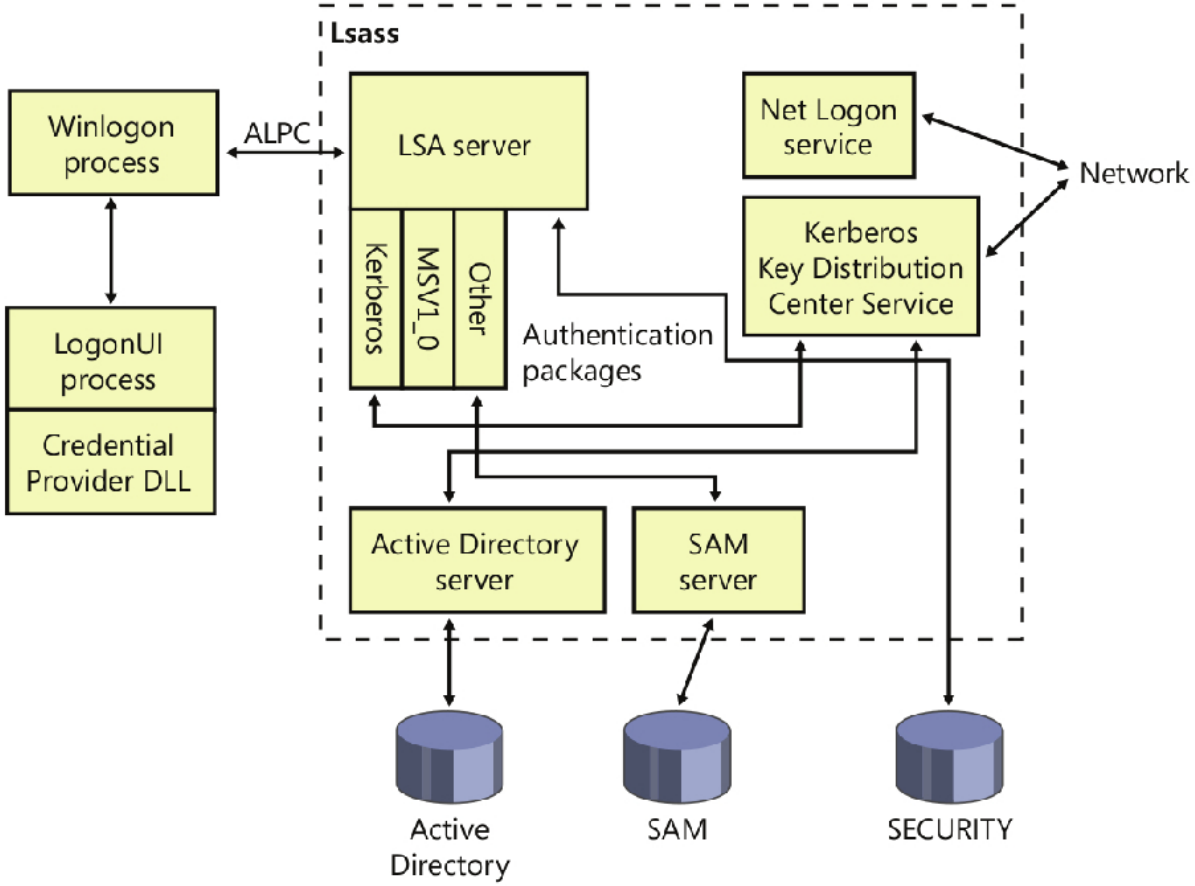


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Connexion d'un utilisateur

- La connexion interactive d'un utilisateur se déroule en plusieurs étapes :
  1. L'utilisateur tappe Ctrl+Alt+Del (SAS)
  2. *Winlogon* démarre *LogonUI* ;
     - *LogonUI* fait appel aux *credential providers* pour obtenir l'identifiant et l'authentifiant (le mot de passe) de l'utilisateur.
  3. *Winlogon* récupère des *handles* sur les packages d'authentification ;
     - via la fonction *LsaLookupAuthenticationPackage* (de *secur32.dll*) ;
     - les packages d'authentification sont listés sous `HKLM\SYSTEM\CurrentControlSet\Control\Lsa`.

Connexion d'un utilisateur

- Suite :
  4. *Winlogon* transmet l'identifiant et l'authentifiant à chaque package d'authentification ;
     - via la fonction *LsaLogonUser* (de *secur32.dll*) ;
     - le fait pour chaque package jusqu'au succès ;
     - si aucun ne retourne un succès, l'authentification est échouée.
  5. *LSASS* vérifie que l'utilisateur a le droit de se connecter ;
     - récupération des droits utilisateur pour l'utilisateur et ses groupes dans la base locale de sécurité ;
     - comparaison avec le type de tentative de connexion (Interactive, Network, Service...) ;
     - s'il n'a pas le droit, la tentative de connexion échoue.
  6. *LSASS* complète les informations sur l'utilisateur ;
     - ajout des SID nécessaires (*Interactive*, *Authenticated Users*...) ;
     - récupèration les privilèges de tous les SID de l'utilisateur.

Connexion d'un utilisateur

- Suite :
  7. *LSASS* crée le jeton d'accès et retourne un *handle* sur le jeton à *Winlogon* ;
     - la création du jeton se fait par un appel à l'exécutif Windows (*ntoskrnl.exe*).
  8. *Winlogon* exécute Userinit ;
     - paramétré dans `HKLM\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\Winlogon\Userinit` ;
     - Par défaut, il s'agit de *Userinit.exe* ;
     - *Winlogon* y attache le jeton d'accès de l'utilisateur.
  9. *Userinit.exe* charge le profil et exécute le *shell* de l'utilisateur.
     - le *shell* est paramétré dans `HKCU\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\Winlogon\Shell` ;
     - par défaut le *shell* est *explorer.exe* ;
     - *Userinit.exe* s'arrête après avoir démarré le *shell*.

Composants impliqués dans l'authentification

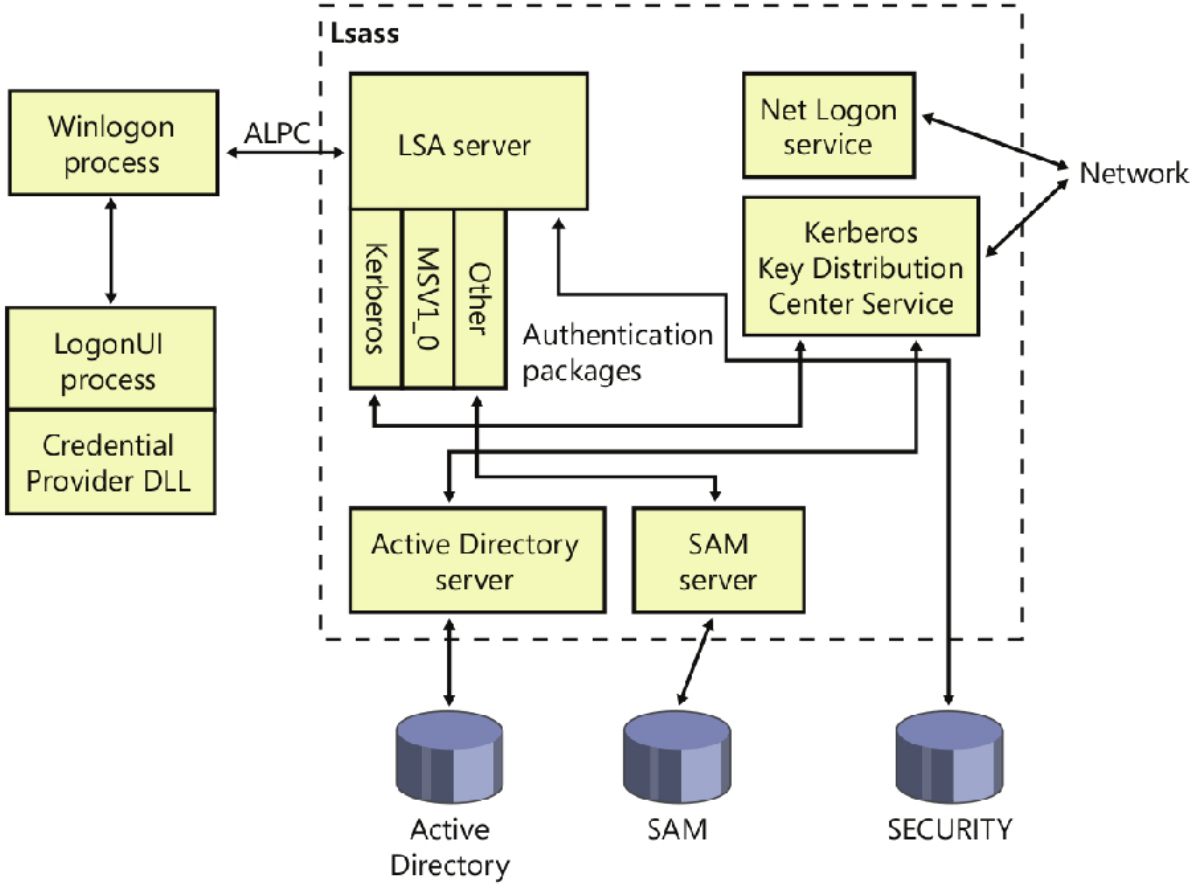


Figure – Extrait de *Windows Internals, 6th edition, part 1*

Packages d'authentification

- Les packages d'authentification implémentent les différentes façon de s'authentifier ;
- Exemple de packages d'authentification :
  - *msv1_0.dll* ;
  - *kerberos.dll* ;
  - *schannel.dll* ;
  - *wdigest.dll*.
- Ces DLL sont utilisées par LSASS ;
  - elles implémentent la fonction *LsaApLogonUserEx2* qui est appelé par LSASS ;
  - cette fonction prend notamment pour paramètres :
    - *LogonType* : le type d'authentification (Interactive, Network...) ;
    - *ProtocolSubmitBuffer* : contient les informations d'authentification ;
    - *AccountName* : le nom du compte.

Packages d'authentification

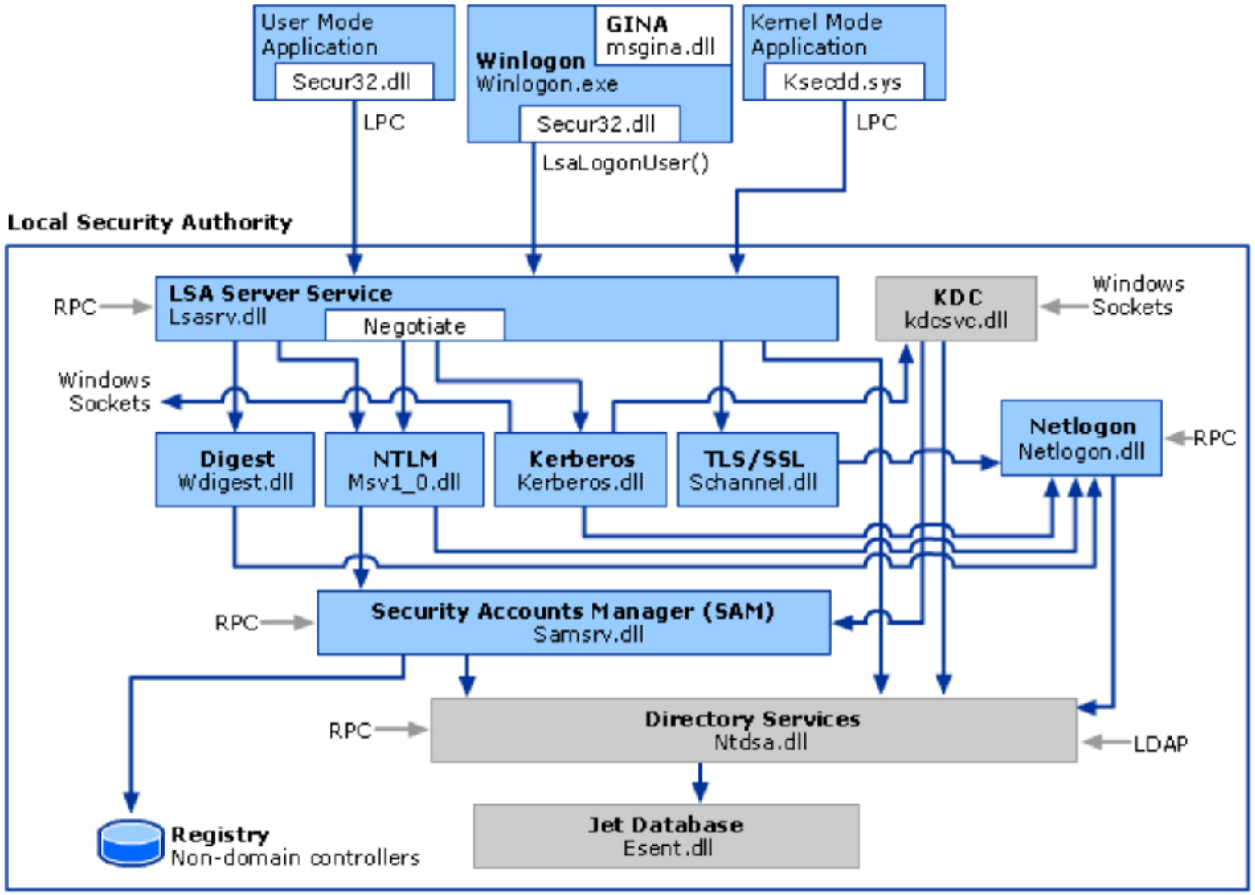


Figure – Extrait de docs.microsoft.com

Packages d'authentification

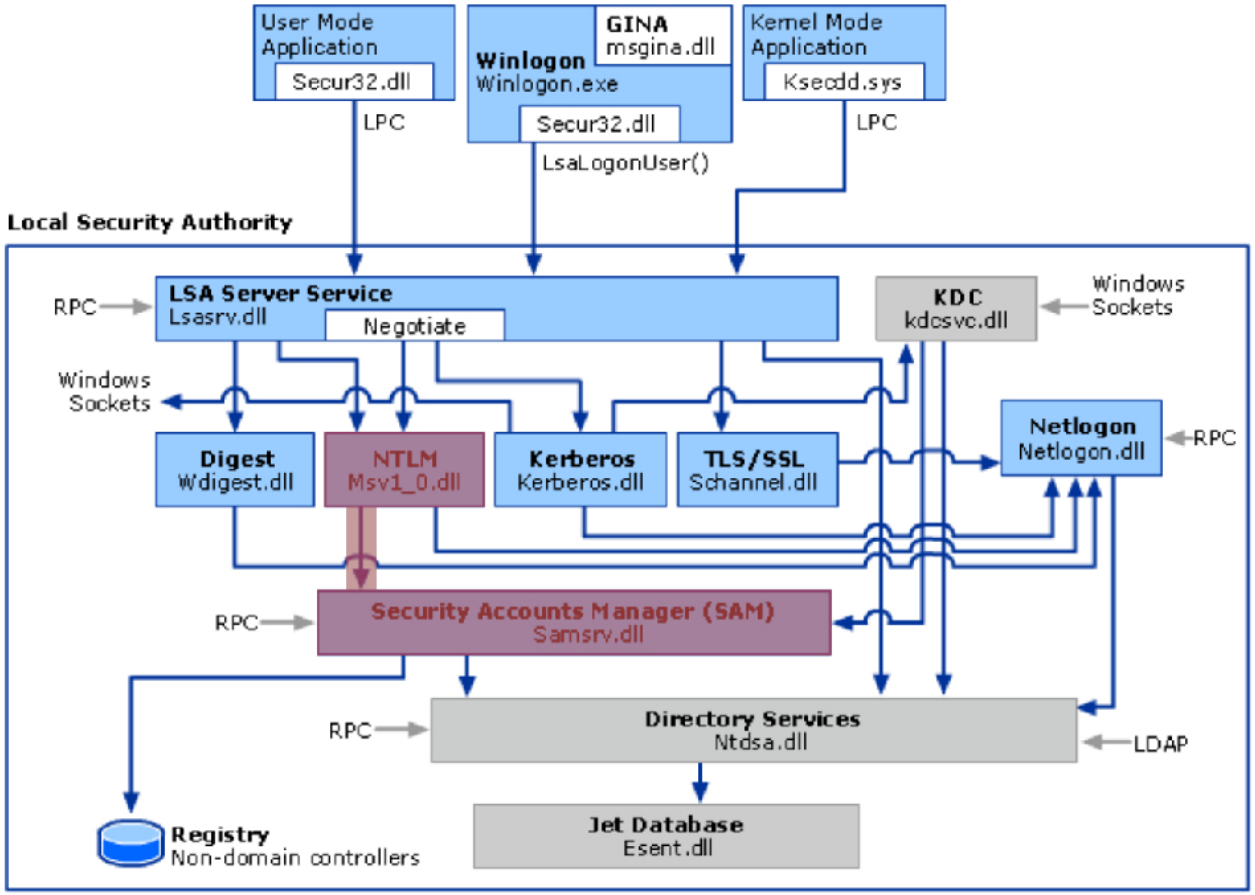


Figure – Extrait de docs.microsoft.com

Packages d'authentification

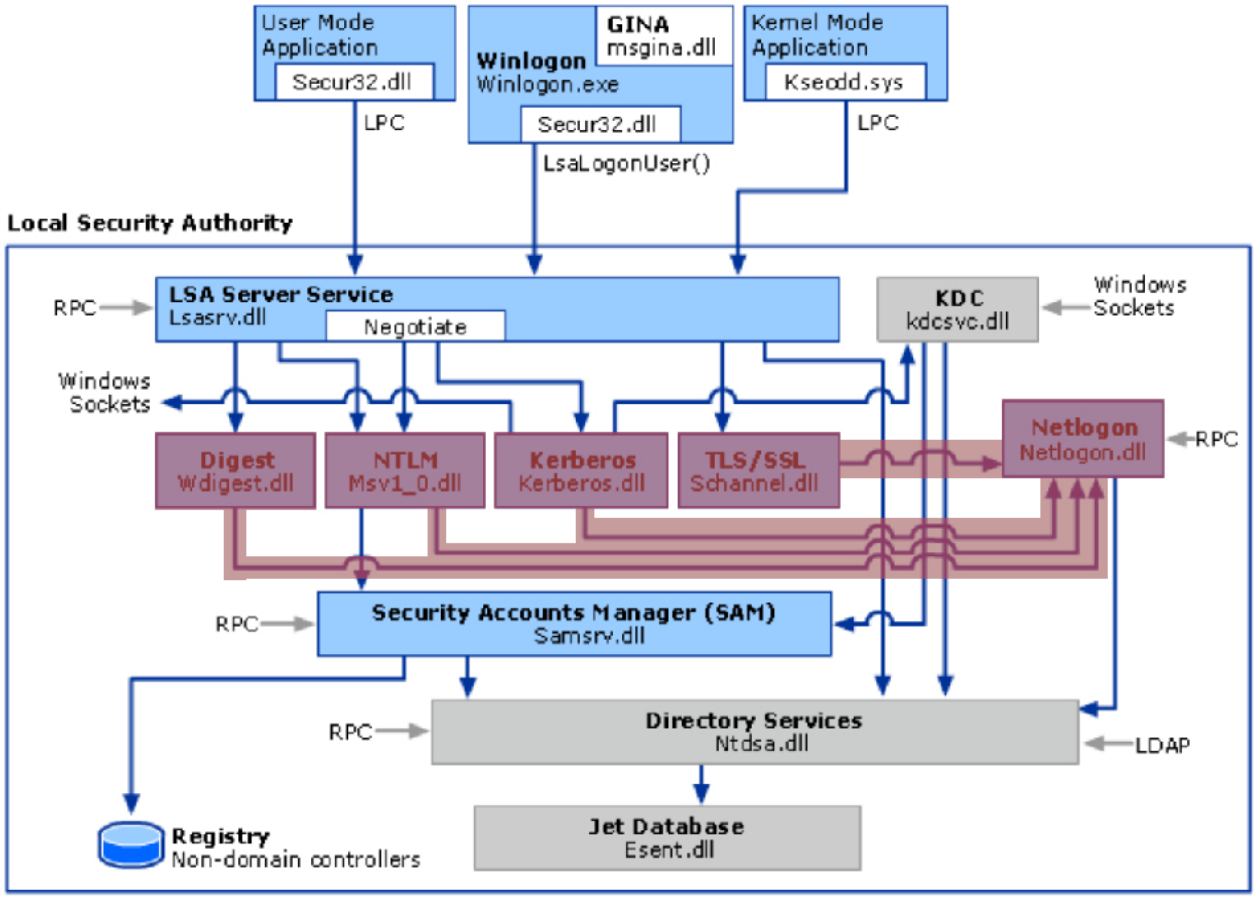


Figure – Extrait de docs.microsoft.com

Authentification locale

- Il existe différent types d'authentification locale :
  - *Interactive* :
    - il s'agit du *LogonType* 2 ;
    - effectuée avec le clavier de l'ordinateur.
  - *Batch* :
    - il s'agit du *LogonType* 4 ;
    - pour les tâches planifiées.
  - *Service* :
    - il s'agit du *LogonType* 5 ;
    - pour les services.
- Cette authentification est gérée par le *msv1_0.dll* ;

Authentification locale

- L'authentification locale se déroule ainsi :
  1. LSASS appel *LsaApLogonUserEx2* de *msv1_0.dll* ;
     - prend notamment en paramètre le nom du compte et son mot de passe ;
  2. *msv1_0.dll* récupère dans la base SAM le *hash* du mot de passe ;
  3. *msv1_0.dll* compare les *hashs* ;
     - *msv1_0.dll* doit donc calculer le *hash* du mot de passe ;
     - s'ils sont identiques l'authentification est un succès.
  4. *msv1_0.dll* retourne à LSASS le succès.

Authentification locale

- Il est possible de s'authentifier directement avec un *hash* ;
- Cette *attaque* est connue sous le nom de ***pass-the-hash***.

Packages d'authentification

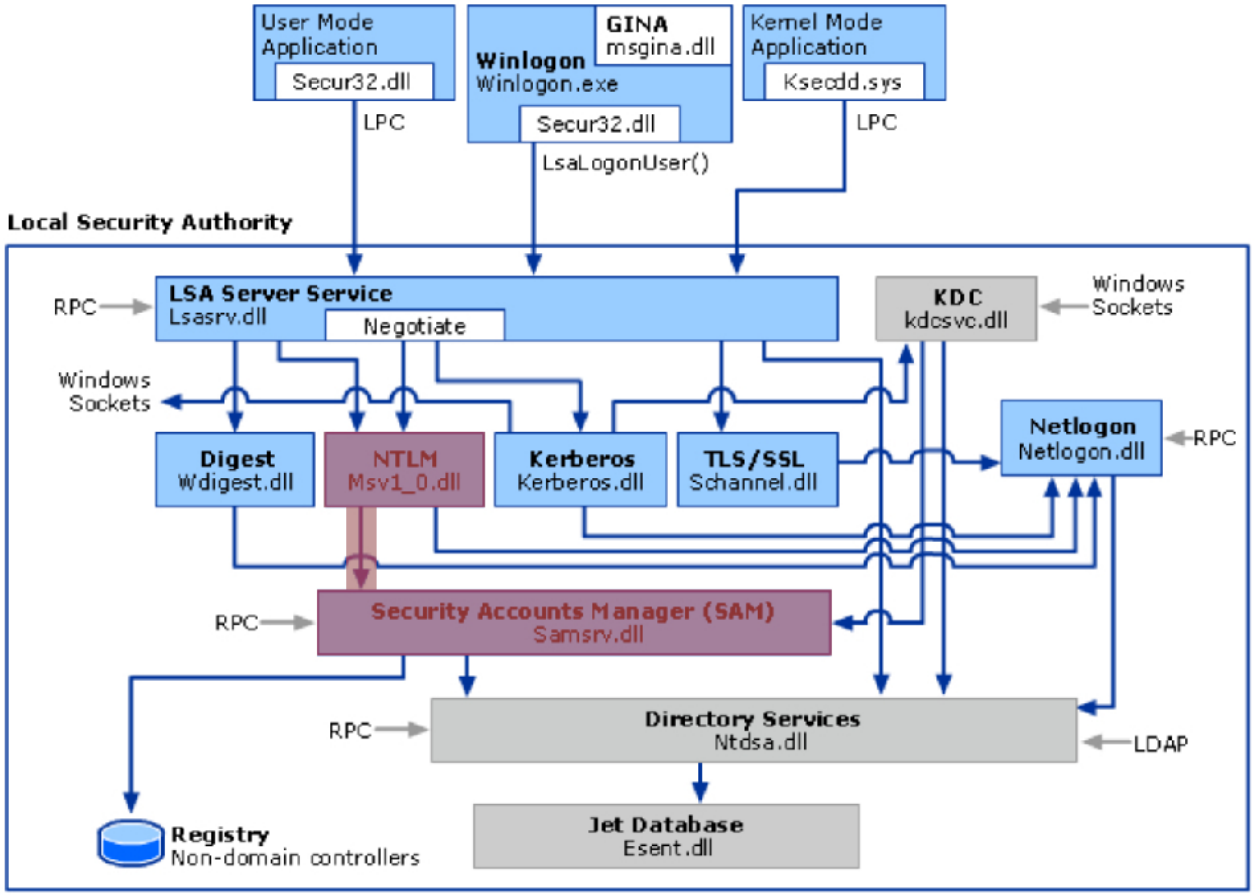


Figure – Extrait de docs.microsoft.com

Packages d'authentification

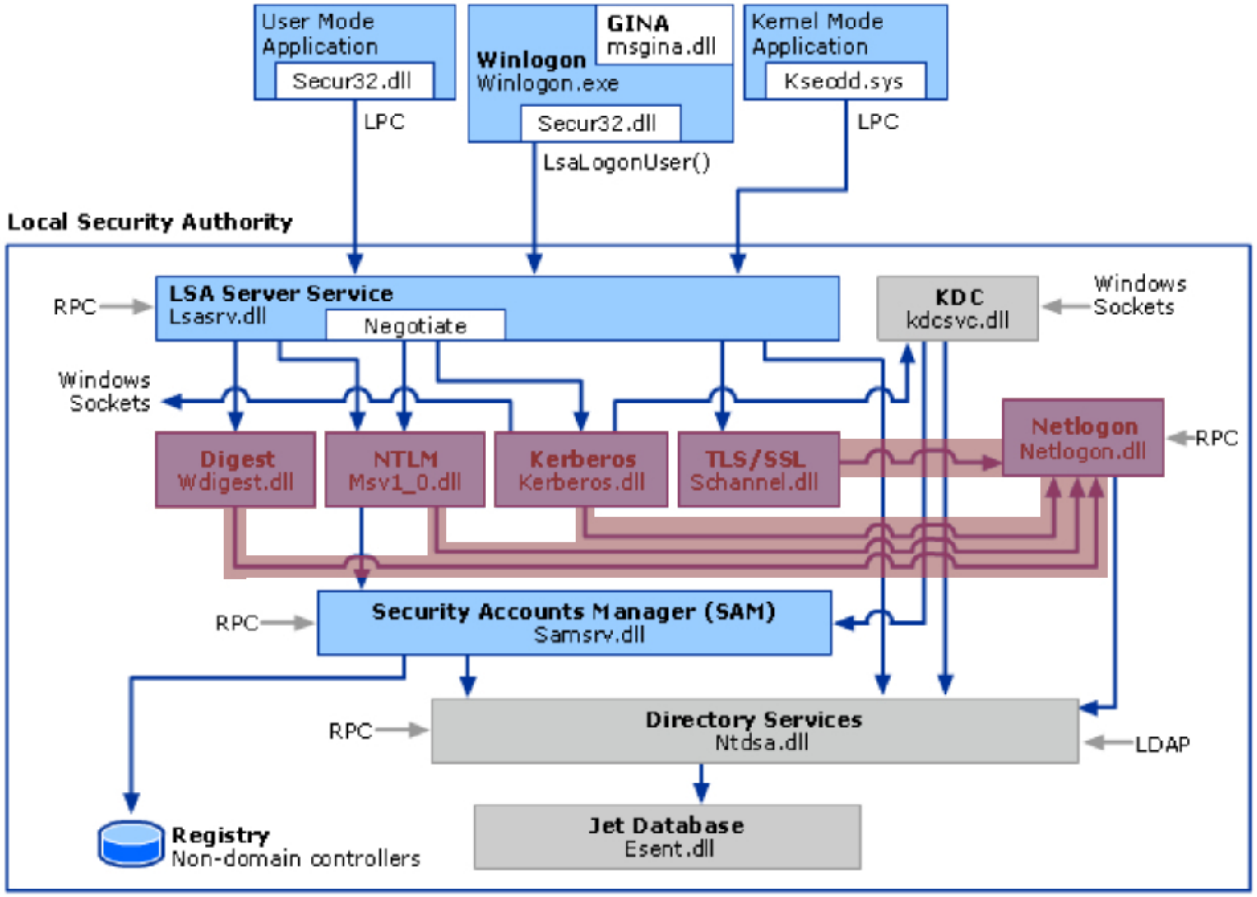


Figure – Extrait de docs.microsoft.com

Authentification distante

- Il existe plusieurs méthodes d'authentification distante ;
- *Netlogon* s'occupe de la communication entre le client et le serveur ;
- Différents packages d'authentification sont utilisés comme :
  - *msv1_0.dll* pour l'authentification avec les protocoles LM et NTLM ;
  - *kerberos.dll* pour l'authentification avec le protocole Kerberos ;
  - *wdigest.dll* pour l'authentification avec le protocole *digest*.

NTLM

- Il existe plusieurs versions de l'authentification NTLM (NT *Lan Manager*) :
  - NTLMv1 ;
  - NTLMv2 ;
- Implémenté dans *msv1_0.dll* ;
- Authentification par défaut pour les machines hors domaine ;
- Également utilisé pour l'authentification aux domaines pré-Windows 2000.

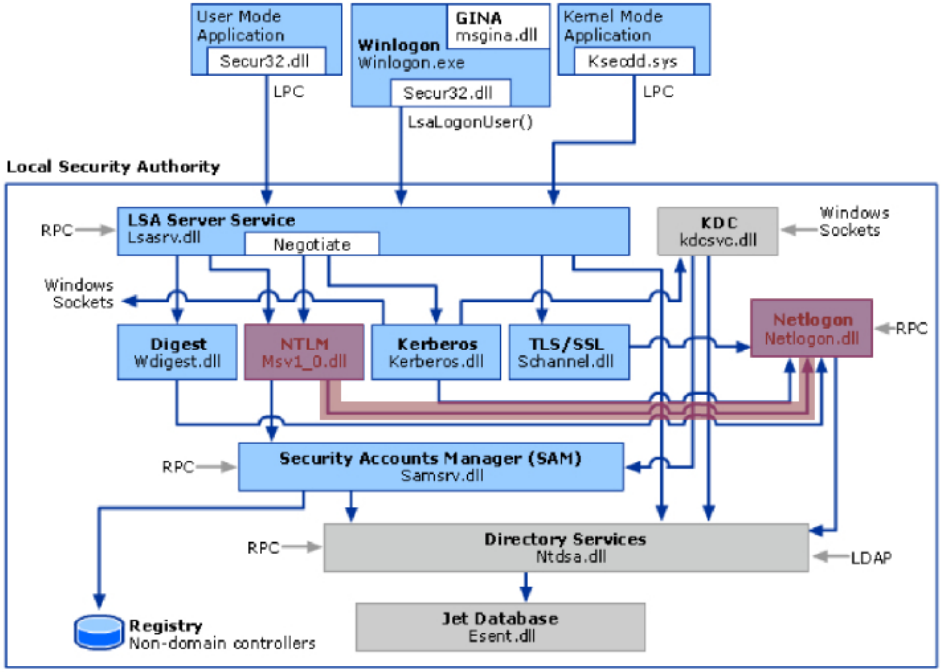


Figure – Côté client

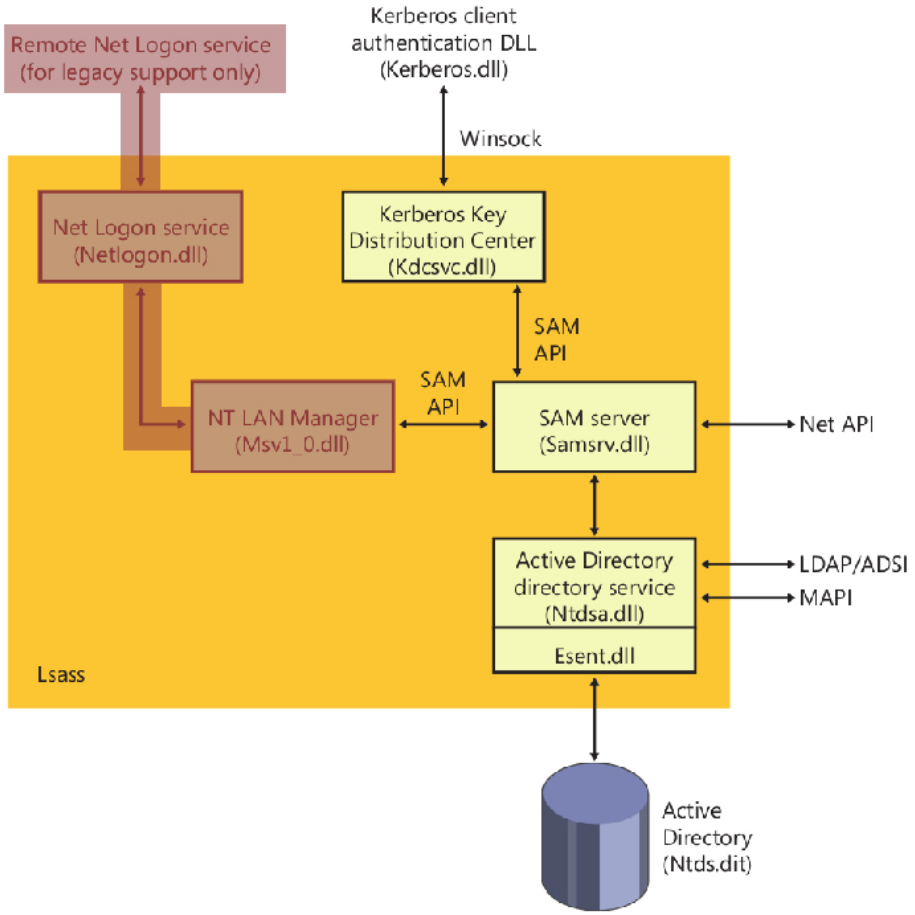


Figure – Côté serveur

NTLM

- Authentification NTLM (simplifié) :
  1. *msv1_0.dll* du client récupère les informations de l'utilisateur et génère le *hash* du mot de passe ;
     - *hash* NTLM pour NTLMv2 et LM ou NTLM pour NTLMv1 ;
  2. *msv1_0.dll* du client envoie le nom d'utilisateur au serveur ;
     - utilise *Netlogon* pour communiquer avec le serveur.
  3. *msv1_0.dll* du serveur génère et envoie au client un *challenge* ;
  4. *msv1_0.dll* du client chiffre le *challenge* avec le *hash* du mot de passe et l'envoie au serveur ;
     - le résultat est appelé la *response*.
     - chiffré en DES pour NTLMv1 et HMAC-MD5 pour NTLMv2 ;
     - NTLMv2 ajoute d'autres informations dans la réponse (nom de la machine, *timestamp*...).

NTLM

Ici il y a deux cas :

- le cas hors domaine :
  5. *msv1_0.dll* du serveur récupère le *hash* du mot de passe de sa base SAM locale ;
  6. *msv1_0.dll* du serveur chiffre le *challenge* avec le *hash* ;
  7. *msv1_0.dll* du serveur compare les deux *responses*.
- le cas du domaine :
  5. *msv1_0.dll* du serveur envoie au contrôleur de domaine :
     - le nom de l'utilisateur ;
     - le *challenge* qu'il a envoyé au client ;
     - la *response* qu'il a reçu du client.
  6. le contrôleur de domaine récupère le *hash* du mot de passe de l'utilisateur dans sa base de domaine ;
  7. le contrôleur de domaine chiffre le *challenge* avec le *hash* ;
  8. le contrôleur de domaine compare les deux *responses*.
  9. si l'authentification est un succès, le contrôleur de domaine en informe le serveur.

NTLM

Protocole	Type *hash*	Auth. client	Auth. serveur	Algo. *response*	Contenu *response*
NTLMv1	LM ou NTLM	non	oui	DES	*challenge* serveur
NTLMv2	NTLM	oui	oui	HMAC-MD5	*challenge* client et serveur, *times-tamp*, nom de machine, nom de domaine

NTLM

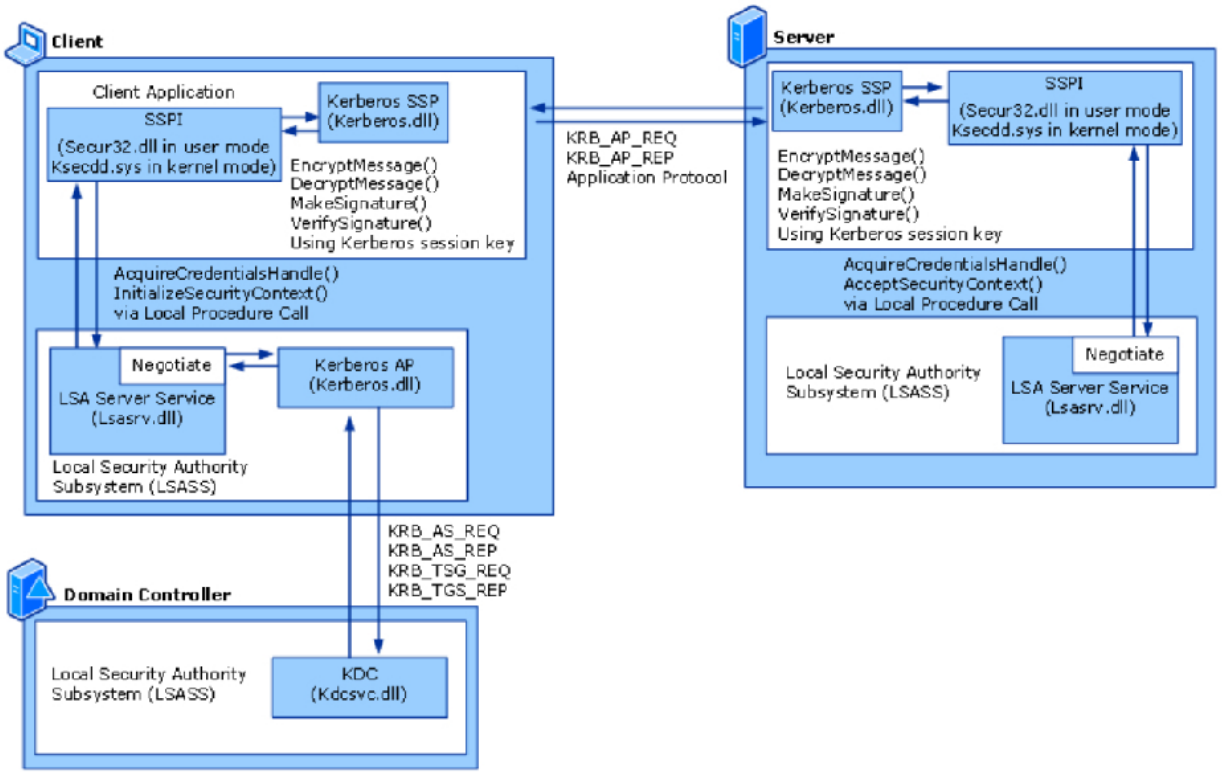
- Il est possible de faire du *pass-the-hash* ;
- Il est possible de faire des attaques par relai[2] ;
- En capturant une tentative d'authentification NTLMv1, il est possible de retrouver le *hash* du mot de passe.

2. https://byt3bl33d3r.github.io/practical-guide-to-ntlm-relaying-in-2017-aka-getting-a-foothold-in-under-5-minute html

Kerberos

- L'authentification Kerberos est l'authentification par défaut dans un domaine ;
- Implémenté dans *kerberos.dll* ;
- Apporte toutes les avantages de l'authentification Kerberos :
  - authentification mutuelle ;
  - interopérabilité avec UNIX ;
  - anti-rejeu ;
  - tickets temporaires ;
  - ...

Kerberos


Client
Client Application
SSPI
(Secur32.dll in user mode
Ksecdd.sys in kernel mode)
Kerberos SSP
(Kerberos.dll)
EncryptMessage()
DecryptMessage()
MakeSignature()
VerifySignature()
Using Kerberos session key
AcquireCredentialsHandle()
InitializeSecurityContext()
via Local Procedure Call
Negotiate
LSA Server Service
(Lsasrv.dll)
Kerberos AP
(Kerberos.dll)
Local Security Authority
Subsystem (LSASS)
KRB_AP_REQ
KRB_AP_REP
Application Protocol
Server
Kerberos SSP
(Kerberos.dll)
SSPI
(Secur32.dll in user mode
Ksecdd.sys in kernel mode)
EncryptMessage()
DecryptMessage()
MakeSignature()
VerifySignature()
Using Kerberos session key
AcquireCredentialsHandle()
AcceptSecurityContext()
via Local Procedure Call
Local Security Authority
Subsystem (LSASS)
Negotiate
LSA Server Service
(Lsasrv.dll)
KRB_AS_REQ
KRB_AS_REP
KRB_TSG_REQ
KRB_TGS_REP
Domain Controller
Local Security Authority
Subsystem (LSASS)
KDC
(Kdcsvc.dll)

Single Sign-On (SSO)

- Pour supporter des fonctions de SSO, Windows garde en mémoire (celle de LSASS) certains authentifiants ;
- Dans la mémoire de LSASS, suivant les méthodes d'authentification utilisées, on peut trouver :
  - des mots de passe en clair (*wdigest.dll*) ;
  - des *hashs* de mots de passe (*msv1_0.dll*) ;
    - utilisable pour les attaques *pass-the-hash*.
  - des tickets Kerberos (*kerberos.dll*).
- Des outils comme Mimikatz [3] permettent de les récupérer.
  - il faut avoir le droit de lire la mémoire de LSASS :
    - le compte système a les permissions nécessaires ;
    - aussi possible avec le privilège *SeDebugPrivilege* par exemple (qu'a le compte administrateur).

3. `http://blog.gentilkiwi.com/mimikatz`

Stockage des secrets

- Les secrets des comptes locaux sont stockés dans `HKLM\SAM`
  - correspond au fichier `%SystemRoot%\System32\Config\Sam`.
  - besoin d'être système ou administrateur pour y accèder.
- Les secrets des comptes de domaine sont stockés dans la base NTDS des contrôleurs de domaine ;
- Les mots de passe sont stockés sous forme de *hash* ;
  - 2 types de *hash* : LM et NTLM ;
  - attention à la confusion entre le *hash* NTLM et les protocoles d'authentification NTLMv1 et NTLMv2.

NTLM et LM

Type de *hash*	Apparition	Taille max.	Alphabet	Algo.
LM	Historique (années 1980)	14 caractères (2x7)	Restreint (OEM)	DES
NTLM	Windows NT4 SP3	255 caractères	Large (Uni-code)	MD4

LM

- L'utilisation de LM est désactivée par défaut ;
- Algorithme :
  1. conversion en *upper case* ;
  2. *padding* avec des caractères NULL jusqu'à 14 caractères ;
  3. séparation en deux parties de 7 caractères ;
  4. génération de deux clefs DES avec les deux parties ;
  5. chiffre en DES la chaîne "KGS ! @#$%" avec les deux clefs ;
  6. concatenation des deux chiffrés.
- Exemple : A8BAA1C5121BBB6FAAD3B435B51404EE ;

> Très **rapide à casser**.

NTLM

- Le *hash* NTLM est utilisé par défaut ;
- Algorithme :
  1. conversion en UTF-16 *little endian* ;
  2. *hash* avec l'algorithme MD4.
- Exemple : A2A19EAEEFE8316C622D5B8D41BA9CE5 ;
- Ici la difficulté pour casser le *hash* dépend de l'algorithme MD4 et de la force des mots de passe.

Syskey

- *Syskey* est un mécanisme de protection de la base SAM qui chiffre les champs stockant des secrets ;
- *Syskey* est activé par défaut ;
- Ce mécanisme est supprimé sur Windows 10 et Windows Server 2016 ;
  - utilisation de cryptographie obsolète (RC4) ;
  - Microsoft recommande comme alternative d'utiliser *Bitlocker* (utilitaire de chiffrement de disque).

Syskey

- 3 modes existent :
  - mode 1 (par défaut) : les secrets sont chiffrés avec une clé dérivés d'une clé stockée dans `HKLM\SYSTEM` ;
  - mode 2 : les secrets sont chiffrés avec une clé dérivée d'un mot de passe donné par l'utilisateur au démarrage du système ;
  - mode 3 : les secrets sont chiffrés avec une clé dérivée d'une clé stockée sur un support amovible.

Syskey

- Vise à protéger d'un attaquant qui aurait obtenu un accès physique au disque (dans les modes 2 et 3) :
  - un tel attaquant pourrait :
    - utiliser les *hashs* pour faire du *pass-the-hash* ;
    - en *offline*, tenter de casser les *hashs*.
  - pour le mode 1, la clé est stockée dans `HKLM\SYSTEM` auquel l'attaquant aura accès s'il a accès à `HKLM\SAM`.
- Ne protège pas contre un attaquant qui aurait obtenu les droits suffisants pour accèder à la base SAM sur le système en marche.

Sommaire

1 Introduction

2 Bases

3 Contrôle d'accès

4 Authentification

5 Mécanismes de protection

User Account Control

- UAC (*User Account Control*) permet à tous les utilisateurs de **s'exécuter avec des droits limités** (c'est-à-dire non administrateur) même s'ils possèdent des droits administrateur ;
- Sans cette fonctionnalité, les utilisateurs possèdant des droits administrateur effectuent toutes leurs actions avec des droits administrateur (comme la navigation internet).

User Account Control

Fonctionnement :

- Création de **deux jetons d'accès** pour les utilisateurs possèdant des droits administrateur :
  - un **jeton filtré** de droits administrateur ;
    - suppression des SID de groupes administrateur (*deny-only*) ;
    - suppression de privilèges administrateur.
  - un jeton avec les droits de l'utilisateur.
- Par défaut, les processus se voient attacher le jeton filtré ;
- Si besoin des droits administrateur, utilisation du jeton privilégié ;
- La demande d'utilisation du jeton privilégié passe par une *pop-up* qui doit être validée par l'utilisateur ;
  - dans le cas où l'utilisateur n'est pas administrateur, il est demandé l'identifiant et le mot de passe d'un administrateur ;
  - garantit que l'utilisateur est au courant de l'élévation ;
  - un processsus ne peut pas obtenir des droits administrateur silencieusement.

```
0:001> !token
Thread is not impersonating. Using process token...
TS Session ID: 0x1
User: S-1-5-21-1997796456-807606720-3932980816-1000
Groups:
 00 S-1-5-21-1997796456-807606720-3932980816-513
    Attributes - Mandatory Default Enabled
 01 S-1-1-0
    Attributes - Mandatory Default Enabled
 02 S-1-5-32-544
    Attributes - DenyOnly
 03 S-1-5-32-545
    Attributes - Mandatory Default Enabled
 04 S-1-5-4
    Attributes - Mandatory Default Enabled
 05 S-1-2-1
    Attributes - Mandatory Default Enabled
 06 S-1-5-11
    Attributes - Mandatory Default Enabled
 07 S-1-5-15
    Attributes - Mandatory Default Enabled
 08 S-1-5-5-0-97200
    Attributes - Mandatory Default Enabled LogonId
 09 S-1-2-0
    Attributes - Mandatory Default Enabled
 10 S-1-5-64-10
    Attributes - Mandatory Default Enabled
 11 S-1-16-8192
    Attributes - GroupIntegrity GroupIntegrityEnabled
Primary Group:   LocadDumpSid failed to dump Sid at addr 0000000001f6b6c8, 0xC0000078; try own SID dump.
s-1-0x515000000
Privs:
 00 0x000000013 SeShutdownPrivilege              Attributes -
 01 0x000000017 SeChangeNotifyPrivilege          Attributes - Enabled Default
 02 0x000000019 SeUndockPrivilege                Attributes -
 03 0x000000021 SeIncreaseWorkingSetPrivilege    Attributes -
 04 0x000000022 SeTimeZonePrivilege              Attributes -
Auth ID: 0:17c11
Impersonation Level: Anonymous
TokenType: Primary
Is restricted token: no.
```

Figure – Visualisation avec WinDBG du jeton filtré

User Account Control


```
0:001> !token
Thread is not impersonating. Using process token...
TS Session ID: 0x1
User: S-1-5-21-1997796456-807606720-3932980816-1000
Groups:
 00 S-1-5-21-1997796456-807606720-3932980816-513
    Attributes - Mandatory Default Enabled
 01 S-1-1-0
    Attributes - Mandatory Default Enabled
 02 S-1-5-32-544
    Attributes - Mandatory Default Enabled Owner
 03 S-1-5-32-545
    Attributes - Mandatory Default Enabled
 04 S-1-5-4
    Attributes - Mandatory Default Enabled
 05 S-1-2-1
    Attributes - Mandatory Default Enabled
 06 S-1-5-11
    Attributes - Mandatory Default Enabled
 07 S-1-5-15
    Attributes - Mandatory Default Enabled
 08 S-1-5-5-0-97200
    Attributes - Mandatory Default Enabled LogonId
 09 S-1-2-0
    Attributes - Mandatory Default Enabled
 10 S-1-5-64-10
    Attributes - Mandatory Default Enabled
 11 S-1-16-12288
    Attributes - GroupIntegrity GroupIntegrityEnabled
Primary Group:   LocadDumpSid failed to dump Sid at addr 0000000001e1b288, 0xC0000078; try own SID dump.
s-1-0x515000000
Privs:
 00 0x000000005 SeIncreaseQuotaPrivilege          Attributes -
 01 0x000000008 SeSecurityPrivilege               Attributes -
 02 0x000000009 SeTakeOwnershipPrivilege          Attributes -
 03 0x00000000a SeLoadDriverPrivilege             Attributes -
 04 0x00000000b SeSystemProfilePrivilege          Attributes -
 05 0x00000000c SeSystemtimePrivilege             Attributes -
 06 0x00000000d SeProfileSingleProcessPrivilege   Attributes -
 07 0x00000000e SeIncreaseBasePriorityPrivilege   Attributes -
 08 0x00000000f SeCreatePagefilePrivilege         Attributes -
 09 0x000000011 SeBackupPrivilege                 Attributes -
 10 0x000000012 SeRestorePrivilege                Attributes -
 11 0x000000013 SeShutdownPrivilege               Attributes -
 12 0x000000014 SeDebugPrivilege                  Attributes -
 13 0x000000016 SeSystemEnvironmentPrivilege      Attributes -
 14 0x000000017 SeChangeNotifyPrivilege           Attributes - Enabled Default
 15 0x000000018 SeRemoteShutdownPrivilege         Attributes -
 16 0x000000019 SeUndockPrivilege                 Attributes -
 17 0x00000001c SeManageVolumePrivilege           Attributes -
 18 0x00000001d SeImpersonatePrivilege            Attributes - Enabled Default
 19 0x00000001e SeCreateGlobalPrivilege           Attributes - Enabled Default
 20 0x000000021 SeIncreaseWorkingSetPrivilege     Attributes -
 21 0x000000022 SeTimeZonePrivilege               Attributes -
 22 0x000000023 SeCreateSymbolicLinkPrivilege     Attributes -
Auth ID: 0:17beb
Impersonation Level: Anonymous
TokenType: Primary
Is restricted token: no.
```

Figure – Visualisation avec WinDBG du jeton privilégié

User Account Control


```
0:001> !token
Thread is not impersonating. Using process token...
TS Session ID: 0x1
User: S-1-5-21-1997796456-807606720-3932980816-1000
Groups:
 00 S-1-5-21-1997796456-807606720-3932980816-513
    Attributes - Mandatory Default Enabled
 01 S-1-1-0
    Attributes - Mandatory Default Enabled
 02 S-1-5-32-544
    Attributes - Mandatory Default Enabled Owner
 03 S-1-5-32-545
    Attributes - Mandatory Default Enabled
 04 S-1-5-4
    Attributes - Mandatory Default Enabled
 05 S-1-2-1
    Attributes - Mandatory Default Enabled
 06 S-1-5-11
    Attributes - Mandatory Default Enabled
 07 S-1-5-15
    Attributes - Mandatory Default Enabled
 08 S-1-5-5-0-97200
    Attributes - Mandatory Default Enabled LogonId
 09 S-1-2-0
    Attributes - Mandatory Default Enabled
 10 S-1-5-64-10
    Attributes - Mandatory Default Enabled
 11 S-1-16-12288
    Attributes - GroupIntegrity GroupIntegrityEnabled
Primary Group:   LocadDumpSid failed to dump Sid at addr 0000000001e1b288, 0xC0000078; try own SID dump.
s-1-0x515000000
Privs:
 00 0x000000005 SeIncreaseQuotaPrivilege            Attributes -
 01 0x000000008 SeSecurityPrivilege                 Attributes -
 02 0x000000009 SeTakeOwnershipPrivilege            Attributes -
 03 0x00000000a SeLoadDriverPrivilege               Attributes -
 04 0x00000000b SeSystemProfilePrivilege            Attributes -
 05 0x00000000c SeSystemtimePrivilege               Attributes -
 06 0x00000000d SeProfileSingleProcessPrivilege     Attributes -
 07 0x00000000e SeIncreaseBasePriorityPrivilege     Attributes -
 08 0x00000000f SeCreatePagefilePrivilege           Attributes -
 09 0x000000011 SeBackupPrivilege                   Attributes -
 10 0x000000012 SeRestorePrivilege                  Attributes -
 11 0x000000013 SeShutdownPrivilege                 Attributes -
 12 0x000000014 SeDebugPrivilege                    Attributes -
 13 0x000000016 SeSystemEnvironmentPrivilege        Attributes -
 14 0x000000017 SeChangeNotifyPrivilege             Attributes - Enabled Default
 15 0x000000018 SeRemoteShutdownPrivilege           Attributes -
 16 0x000000019 SeUndockPrivilege                   Attributes -
 17 0x00000001c SeManageVolumePrivilege             Attributes -
 18 0x00000001d SeImpersonatePrivilege              Attributes - Enabled Default
 19 0x00000001e SeCreateGlobalPrivilege             Attributes - Enabled Default
 20 0x000000021 SeIncreaseWorkingSetPrivilege       Attributes -
 21 0x000000022 SeTimeZonePrivilege                 Attributes -
 22 0x000000023 SeCreateSymbolicLinkPrivilege       Attributes -
Auth ID: 0:17beb
Impersonation Level: Anonymous
TokenType: Primary
Is restricted token: no.
```

Figure – Visualisation avec WinDBG du jeton privilégié

User Account Control

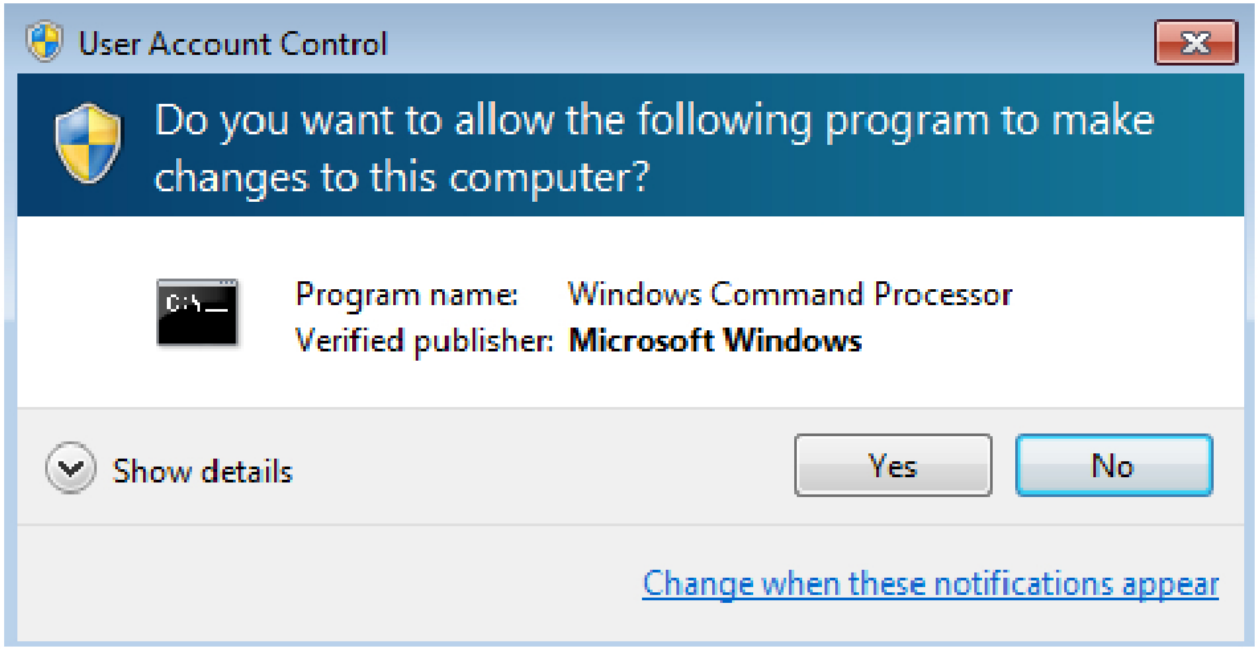


Figure – *pop-up* UAC pour un utilisateur avec des droits administrateur

User Account Control

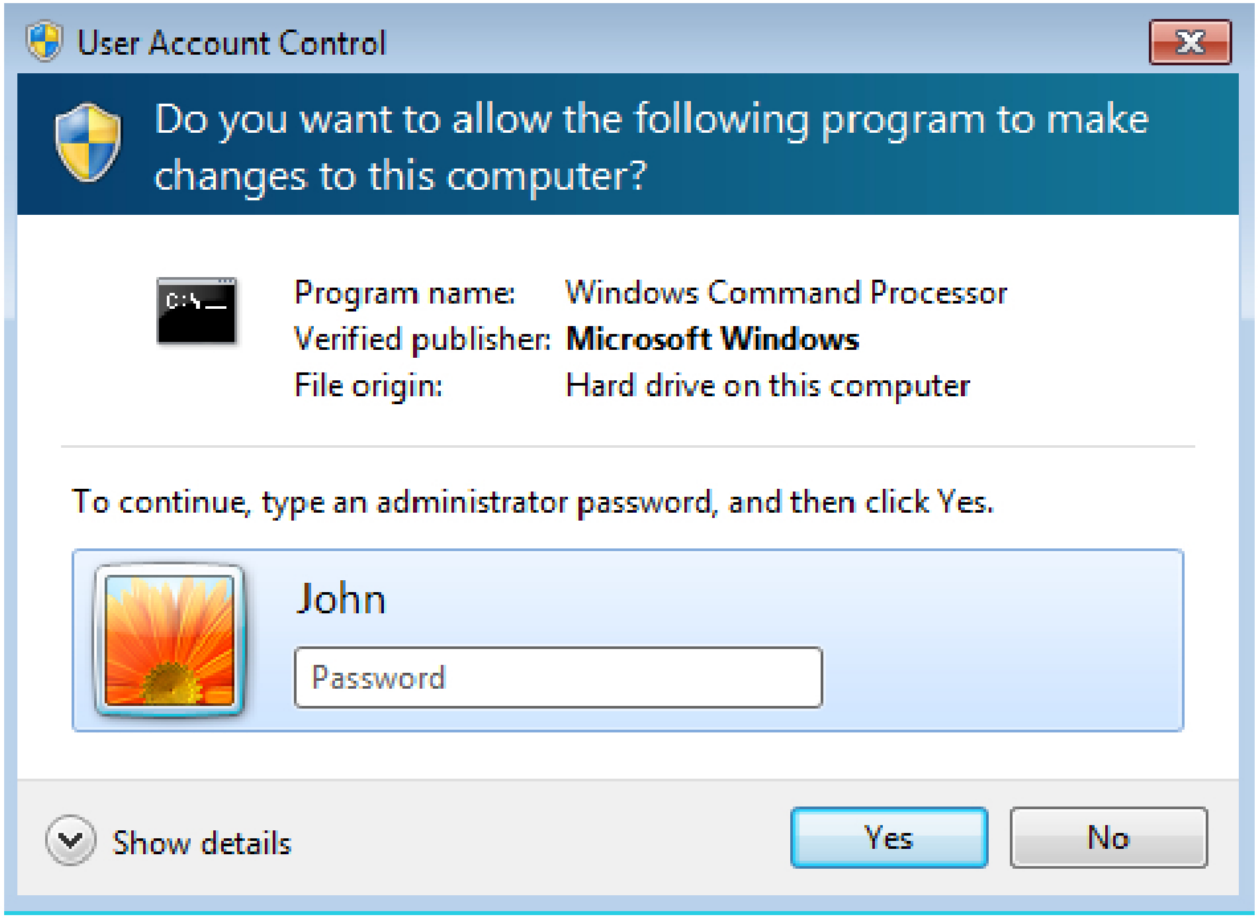


Figure – *pop-up* UAC pour un utilisateur sans droit administrateur

User Account Control

- Il existe plusieurs façons de demander une élévation de privilèges :
  - clique droit+"Exécuter en tant qu'administrateur" ;
  - heuristiques de Windows comme :
    - des mots clefs dans le nom de fichier (setup, install...) ;
    - des séquences d'octets dans l'exécutable.
  - tag *requestedExecutionLevel* dans le manifest du fichier exécutable.

User Account Control

Valeurs possibles de *requestedExecutionLevel* :

Valeur	Signification
As invoker	Ne demande jamais l'élévation de privilèges.
Highest available	Si l'utilisateur n'est pas administrateur équivalent à *As invoker* sinon demande d'élévation de privilèges.
Require administrator	Demande toujours l'élévation de privilèges.

User Account Control

```
C:\>strings c:\Windows\System32\eventvwr.exe
...
<trustInfo xmlns="urn:schemas-microsoft-com:asm.v3">
<security>
<requestedPrivileges>
<requestedExecutionLevel
level="highestAvailable"
uiAccess="false"
/>
</requestedPrivileges>
</security>
</trustInfo>
...
```

User Account Control

- Beaucoup de façons de contourner UAC ont été découvertes ;
- La recherche ”bypass UAC” sur un moteur de recherche donne beaucoup de résultats ;
- En grande partie via l’exploitation de la fonctionnalité *auto-elevate* :
  - certains binaires identifiés comme sûrs par Windows bénéficient d’une élévation sans *pop-up*.

Niveaux d'intégrité

- MIC (*Mandatory Integrity Control*) est un mécanisme de contrôle d'accès aux objets ;
- Ce mécanisme s'ajoute au contrôle d'accès basé sur les DACL et s'effectue avant ce dernier ;
- MIC se base sur des **niveaux d'intégrité**.

Niveaux d'intégrité

- Les **descripteurs de sécurité** des objets se voient ajouter :
  - un niveau d'intégrité ;
  - des ACE définissant les règles d'accès vis à vis de l'intégrité.
- Les objets sans niveau d'intégrité sont considérés comme *medium* ;
- Les objets créés recoivent le niveau d'intégrité du jeton ;

- Les **jeton d'accès** se voient ajouter :
  - un niveau d'intégrité ;
  - une *mandatory policy* définissant le comportement face aux niveaux d'intégrité.
- Les processus recoivent le niveau d'intégrité le plus bas entre celui du jeton et celui du fichier exécutable ;

Niveaux d'intégrité

SID	Niveau	Usage
S-1-16-0x0000	*Untrusted* (0)	Utilisé pour les processus démarrés avec le groupe Anonyme.
S-1-16-0x1000	*Low* (1)	Utilisé pour le mode protégé d'Internet Explorer.
S-1-16-0x2000	*Medium* (2)	Utilisé pour les applications normales.
S-1-16-0x3000	*High* (3)	Utilisé pour les applications d'administration.
S-1-16-0x4000	*System* (4)	Utilisé pour les services ou les applications système (comme *Winlogon*).

Niveaux d'intégrité

- Par exemple, un jeton d'accès avec un niveau d'intégrité *Low* ne pourra écrire dans un objet avec un niveau d'intégrité *Medium* même si les DACL de l'objet lui autorise l'accès.

Niveaux d'intégrité

- Les ACE de type `SYSTEM_MANDATORY_LABEL_ACE` des SACL associent :
  - un SID de niveau d'intégrité ;
  - avec un masque d'accès.
- Le masque d'accès peut contenir les valeurs :

Valeur	Signification
`SYSTEM_MANDATORY_LABEL_NO_WRITE_UP` (1)	Un compte avec un niveau d'intégrité plus bas ne peut pas écrire dans l'objet.
`SYSTEM_MANDATORY_LABEL_NO_READ_UP` (2)	Un compte avec un niveau d'intégrité plus bas ne peut pas lire dans l'objet.
`SYSTEM_MANDATORY_LABEL_NO_EXECUTE_UP` (3)	Un compte avec un niveau d'intégrité plus bas ne peut pas exécuter l'objet.

- Par défaut, les objets se voient assigner `SYSTEM_MANDATORY_LABEL_NO_WRITE_UP` ;

Niveaux d'intégrité

- Les jetons d'accès peuvent avoir une *mandatory policy* qui définit le comportement du jeton vis à vis des niveaux d'intégrité :

Valeur	Signification
`TOKEN_MANDATORY_POLICY_OFF` (0)	Aucun contrôle de niveau d'intégrité pour ce jeton.
`TOKEN_MANDATORY_POLICY_NO_WRITE_UP` (1)	Un processus associé à ce jeton ne permet d'écrire sur un objet de plus haut niveau d'intégrité.
`TOKEN_MANDATORY_POLICY_NEW_PROCESS_MIN` (2)	Un processus associé à ce jeton à pour niveau d'intégrité MIN(processus parent, fichier exécutable).
`TOKEN_MANDATORY_POLICY_VALID_MASK` (3)	Combinaison de `TOKEN_MANDATORY_POLICY_NO_WRITE_UP` et `TOKEN_MANDATORY_POLICY_NEW_PROCESS_MIN`.

Niveaux d'intégrité

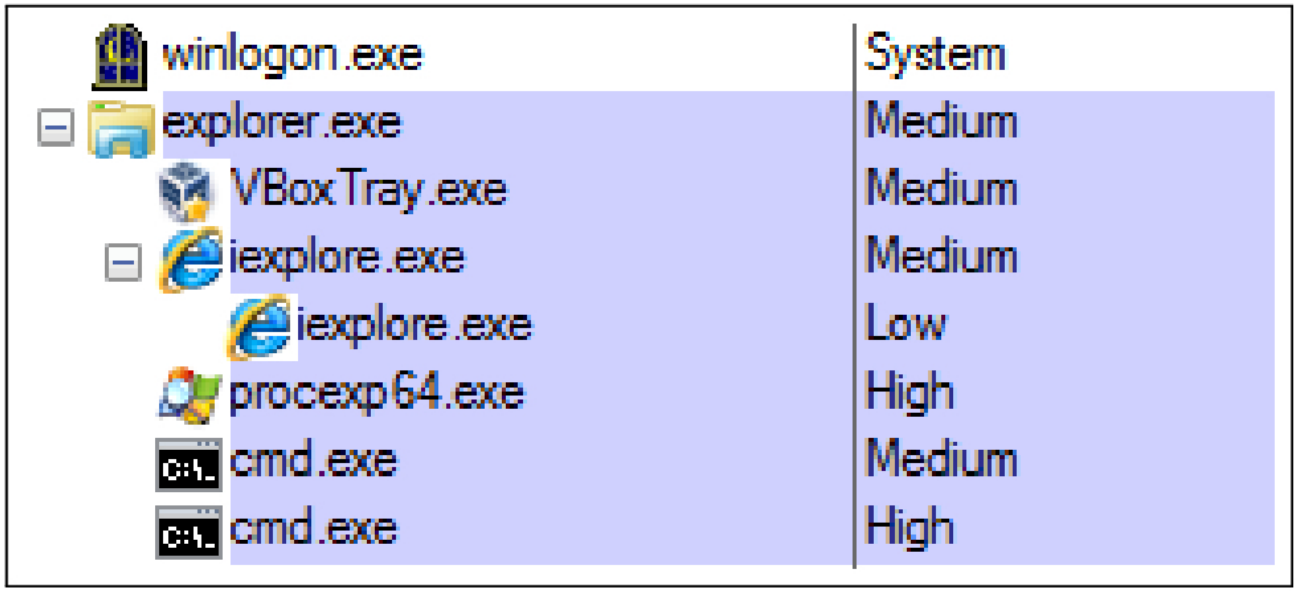


Figure – Visualisation du niveau d'intégrité de processus avec *process explorer*

Niveaux d'intégrité

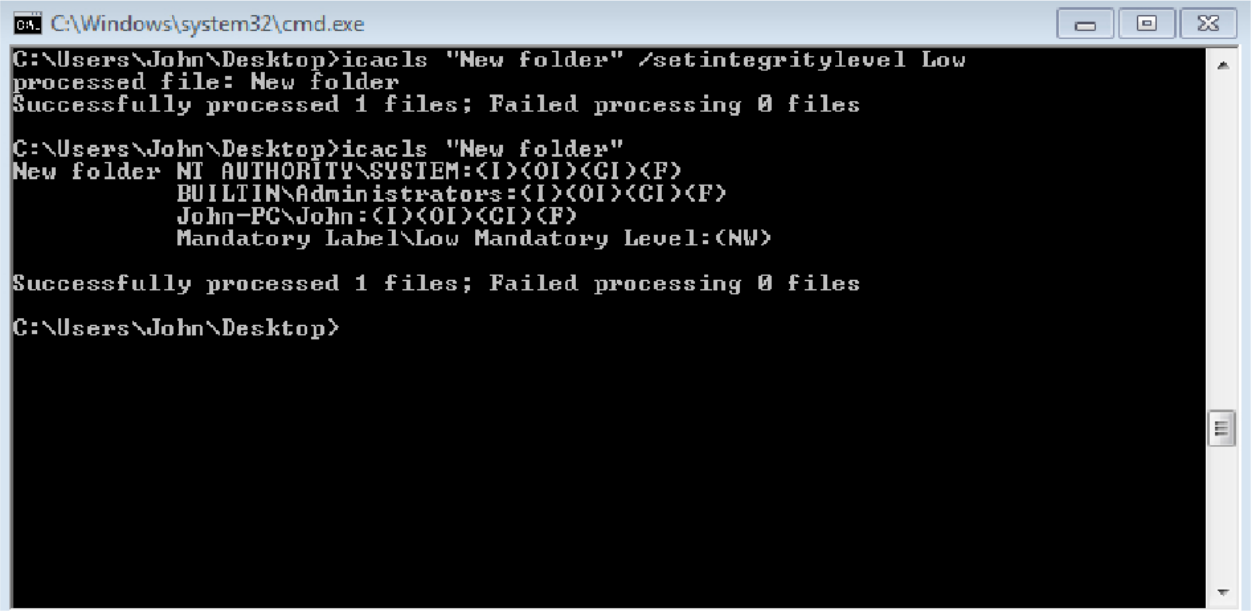


Figure – Modification et visualisation du niveau d'intégrité de fichiers avec *icacls*

Niveaux d'intégrité

```
kd> !token fffff8a0083b1a90
_TOKEN fffff8a0083b1a90
TS Session ID: 0x1
User: S-1-5-21-1997796456-807606720-3932980816-1000
Groups:
 00 S-1-5-21-1997796456-807606720-3932980816-513
    Attributes - Mandatory Default Enabled
 01 S-1-1-0
    Attributes - Mandatory Default Enabled
 02 S-1-5-32-544
    Attributes - DenyOnly
 03 S-1-5-32-545
    Attributes - Mandatory Default Enabled
 04 S-1-5-4
    Attributes - Mandatory Default Enabled
 05 S-1-2-1
    Attributes - Mandatory Default Enabled
 06 S-1-5-11
    Attributes - Mandatory Default Enabled
 07 S-1-5-15
    Attributes - Mandatory Default Enabled
 08 S-1-5-5-0-97200
    Attributes - Mandatory Default Enabled LogonId
 09 S-1-2-0
    Attributes - Mandatory Default Enabled
 10 S-1-5-64-10
    Attributes - Mandatory Default Enabled
 11 S-1-16-4096
    Attributes - GroupIntegrity GroupIntegrityEnabled
Primary Group: S-1-5-21-1997796456-807606720-3932980816-513
Privs:
 19 0x000000013 SeShutdownPrivilege                Attributes -
 23 0x000000017 SeChangeNotifyPrivilege            Attributes - Enabled Default
 25 0x000000019 SeUndockPrivilege                  Attributes -
 33 0x000000021 SeIncreaseWorkingSetPrivilege      Attributes -
 34 0x000000022 SeTimeZonePrivilege                Attributes -
Authentication ID:         (0,17c11)
Impersonation Level:       Anonymous
TokenType:                 Primary
Source: User32             TokenFlags: 0xe00 ( Token in use )
Token ID: 3412cc           ParentToken ID: 17c14
Modified ID:               (0, 3412e4)
RestrictedSidCount: 0      RestrictedSids: 0000000000000000
OriginatingLogonSession: 3e7
```

Figure – Visualisation du niveau d'intégrité d'un jeton avec *kd*

Niveaux d'intégrité

```
kd> !sd 0xfffff8a0`072946b5 & -10
->Revision: 0x1
->Sbz1    : 0x0
->Control : 0x8814
            SE_DACL_PRESENT
            SE_SACL_PRESENT
            SE_SACL_AUTO_INHERITED
            SE_SELF_RELATIVE
->Owner   : S-1-5-21-1997796456-807606720-3932980816-1000
->Group   : S-1-5-21-1997796456-807606720-3932980816-513
->Dacl    :
->Dacl    : ->AclRevision: 0x2
->Dacl    : ->Sbz1       : 0x0
->Dacl    : ->AclSize    : 0x5c
->Dacl    : ->AceCount   : 0x3
->Dacl    : ->Sbz2       : 0x0
->Dacl    : ->Ace[0]: ->AceType: ACCESS_ALLOWED_ACE_TYPE
->Dacl    : ->Ace[0]: ->AceFlags: 0x0
->Dacl    : ->Ace[0]: ->AceSize: 0x24
->Dacl    : ->Ace[0]: ->Mask : 0x001fffff
->Dacl    : ->Ace[0]: ->SID: S-1-5-21-1997796456-807606720-3932980816-1000

->Dacl    : ->Ace[1]: ->AceType: ACCESS_ALLOWED_ACE_TYPE
->Dacl    : ->Ace[1]: ->AceFlags: 0x0
->Dacl    : ->Ace[1]: ->AceSize: 0x14
->Dacl    : ->Ace[1]: ->Mask : 0x001fffff
->Dacl    : ->Ace[1]: ->SID: S-1-5-18

->Dacl    : ->Ace[2]: ->AceType: ACCESS_ALLOWED_ACE_TYPE
->Dacl    : ->Ace[2]: ->AceFlags: 0x0
->Dacl    : ->Ace[2]: ->AceSize: 0x1c
->Dacl    : ->Ace[2]: ->Mask : 0x00121411
->Dacl    : ->Ace[2]: ->SID: S-1-5-5-0-97200

->Sacl    :
->Sacl    : ->AclRevision: 0x2
->Sacl    : ->Sbz1       : 0x0
->Sacl    : ->AclSize    : 0x1c
->Sacl    : ->AceCount   : 0x1
->Sacl    : ->Sbz2       : 0x0
->Sacl    : ->Ace[0]: ->AceType: SYSTEM_MANDATORY_LABEL_ACE_TYPE
->Sacl    : ->Ace[0]: ->AceFlags: 0x0
->Sacl    : ->Ace[0]: ->AceSize: 0x14
->Sacl    : ->Ace[0]: ->Mask : 0x00000003
->Sacl    : ->Ace[0]: ->SID: S-1-16-4096
```

Figure – Visualisation des ACE d'intégrité d'un objet avec *kd*

Windows Defender

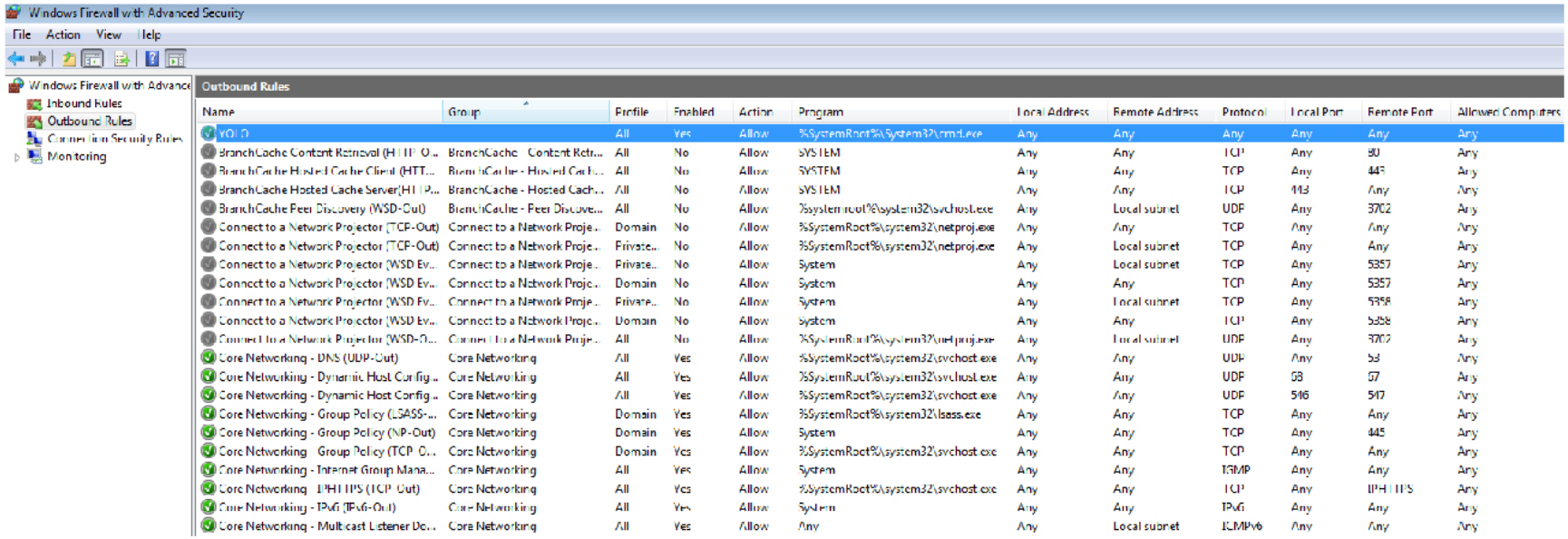
- Au départ, il s'agit d'un logiciel cherchant principalement des logiciels "espions" ;
- A évolué en anti-virus complet à partir de Windows 8 ;
  - analyse en temps réel et à la demande ;
  - analyse par signature, par heuristique et analyse comportementale ;
  - cherche à détecter des virus, des logiciels espions, des logiciels de publicité, des *trojans*...
- Avantages/incovénients :
  - \+ c'est un anti-virus natif (bonne intégration) ;
  - \+ installé sur tous les Windows (disponibilité d'un grand nombre de samples) ;
  - \- moins d'expérience que les anti-virus commerciaux
  - \- moins de moyens que les anti-virus commerciaux.
- D'après les chiffres de certaines études [4], à la hauteur des anti-virus commerciaux :
  - résultats à prendre avec des pincettes.

4. https://www.av-test.org/fr/antivirus/particuliers-windows/

Windows Firewall

- Basé sur WFP (*Windows Filtering Platform*) ;
- Permet de faire du filtrage *stateful* ;
- Peut filtrer suivant :
  - l'IP destination et source ;
  - le port destination et source ;
  - le programme ;
  - les profiles de connexion (*public*, *domain*, *private*).
- Géré graphiquement avec l'outil *Windows Firewall with Advanced Security* (*firewall.cpl*) ;
- Géré en ligne de commande avec *netsh advfirewall*.

Windows Firewall



Name	Group	Profile	Enabled	Action	Program	Local Address	Remote Address	Protocol	Local Port	Remote Port	Allowed Computers
VOID		All	Yes	Allow	%SystemRoot%\System32\cmd.exe	Any	Any	Any	Any	Any	Any
BranchCache Content Retrieval (HTTP-O...	BranchCache - Content Retr...	All	No	Allow	SYSTEM	Any	Any	TCP	Any	80	Any
BranchCache Hosted Cache Client (HTT...	BranchCache - Hosted Cach...	All	No	Allow	SYSTEM	Any	Any	TCP	Any	443	Any
BranchCache Hosted Cache Server(HTTP...	BranchCache - Hosted Cach...	All	No	Allow	SYSTEM	Any	Any	TCP	443	Any	Any
BranchCache Peer Discovery (WSD-Out)	BranchCache - Peer Discove...	All	No	Allow	%systemroot%\system32\svchost.exe	Any	Local subnet	UDP	Any	3702	Any
Connect to a Network Projector (TCP-Out)	Connect to a Network Proje...	Domain	No	Allow	%SystemRoot%\system32\netproj.exe	Any	Any	TCP	Any	Any	Any
Connect to a Network Projector (TCP-Out)	Connect to a Network Proje...	Private...	No	Allow	%SystemRoot%\system32\netproj.exe	Any	Local subnet	TCP	Any	Any	Any
Connect to a Network Projector (WSD Ev...	Connect to a Network Proje...	Private...	No	Allow	System	Any	Local subnet	TCP	Any	5357	Any
Connect to a Network Projector (WSD Ev...	Connect to a Network Proje...	Domain	No	Allow	System	Any	Any	TCP	Any	5357	Any
Connect to a Network Projector (WSD Ev...	Connect to a Network Proje...	Private...	No	Allow	System	Any	Local subnet	TCP	Any	5358	Any
Connect to a Network Projector (WSD Ev...	Connect to a Network Proje...	Domain	No	Allow	System	Any	Any	TCP	Any	5358	Any
Connect to a Network Projector (WSD-O...	Connect to a Network Proje...	All	No	Allow	%SystemRoot%\system32\netproj.exe	Any	Local subnet	UDP	Any	3702	Any
Core Networking - DNS (UDP-Out)	Core Networking	All	Yes	Allow	%SystemRoot%\system32\svchost.exe	Any	Any	UDP	Any	53	Any
Core Networking - Dynamic Host Config...	Core Networking	All	Yes	Allow	%SystemRoot%\system32\svchost.exe	Any	Any	UDP	68	67	Any
Core Networking - Dynamic Host Config...	Core Networking	All	Yes	Allow	%SystemRoot%\system32\svchost.exe	Any	Any	UDP	546	547	Any
Core Networking - Group Policy (LSASS-...	Core Networking	Domain	Yes	Allow	%SystemRoot%\system32\lsass.exe	Any	Any	TCP	Any	Any	Any
Core Networking - Group Policy (NP-Out)	Core Networking	Domain	Yes	Allow	System	Any	Any	TCP	Any	445	Any
Core Networking - Group Policy (TCP-O...	Core Networking	Domain	Yes	Allow	%SystemRoot%\system32\svchost.exe	Any	Any	TCP	Any	Any	Any
Core Networking - Internet Group Mana...	Core Networking	All	Yes	Allow	System	Any	Any	IGMP	Any	Any	Any
Core Networking - IPHTTPS (TCP-Out)	Core Networking	All	Yes	Allow	%SystemRoot%\system32\svchost.exe	Any	Any	TCP	Any	IPHTTPS	Any
Core Networking - IPv6 (IPv6-Out)	Core Networking	All	Yes	Allow	System	Any	Any	IPv6	Any	Any	Any
Core Networking - Multicast Listener Do...	Core Networking	All	Yes	Allow	Any	Any	Local subnet	ICMPv6	Any	Any	Any

Figure – utilitaire *Windows Firewall with Advanced Security*

Bitlocker

- Utilitaire de **chiffrement de disque** ;
- Basé sur des standards cryptographiques solides (SHA256, RSA, AES...) ;
- Capable d'utiliser le **TPM** ;
- Peut faire du *boot* sécurisé ;
- Peut faire de la *full-volume encryption* (FVE) ;
- Très simple d'utilisation.

Bitlocker

- La *volume master key* (VMK) peut venir de :

Source	Protection
TPM	Protège contre les attaques logicielle mais pas contre les attaques matérielles (type DMA).
TPM + PIN	Ajout de protection contre les attaques matérielles.
TPM + clef USB	Protection totale contre les attaques matérielles, mais vulnérable au vol de clef USB
TPM + PIN + clef USB	Protection totale contre les attaques matérielles.
clef USB	Protection totale contre les attaques matérielles, mais vulnérable au vol de clef USB (destiné au système sans TPM).

Bitlocker

Bitlocker to Go :

- **Chiffrement de clef USB** ;
- Pas d'utilisation de TPM dans ce cas ;
- Protection par mot de passe ou *smartcard* ;
- Outils disponibles sous Linux pour l'interopérabilité.

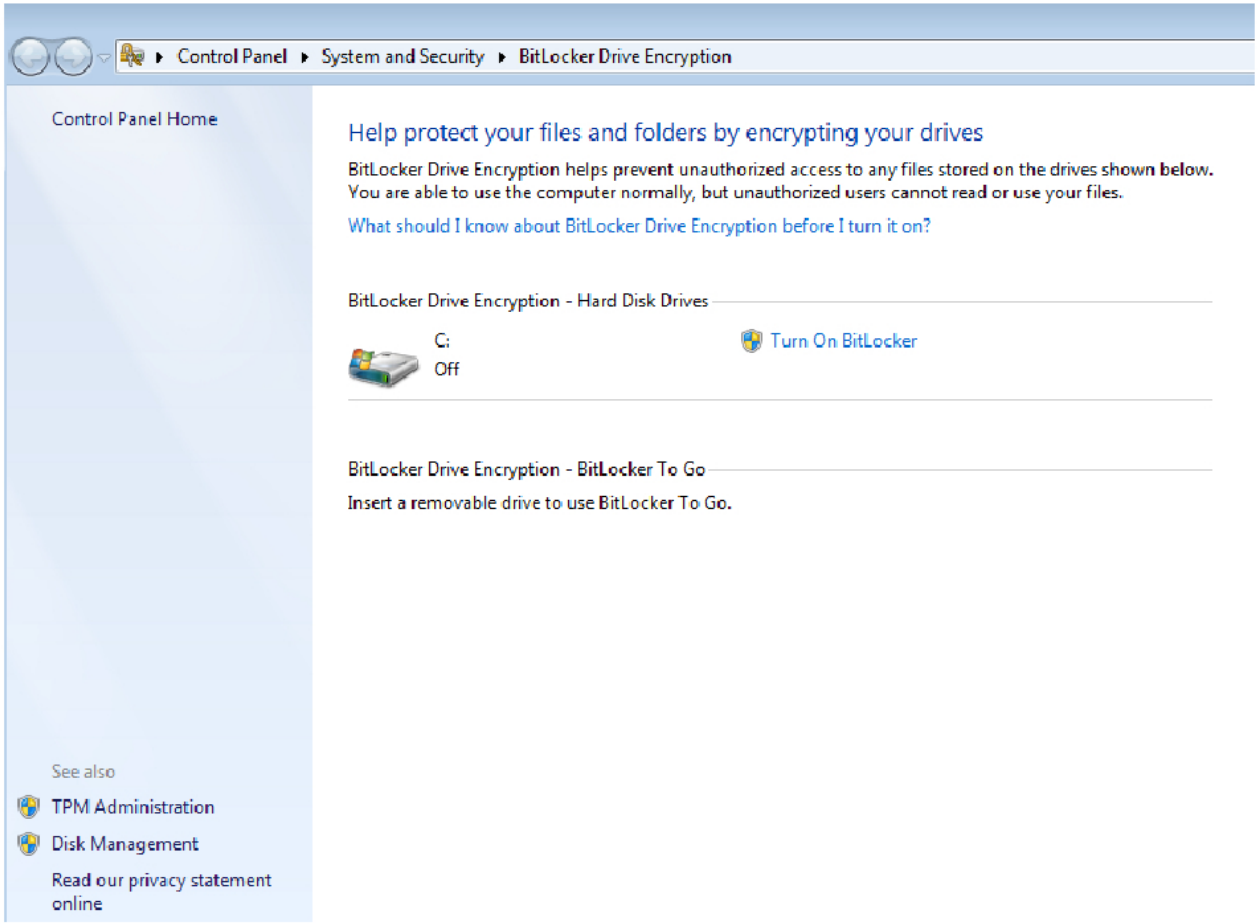


Figure – Utilitaire *Bitlocker Drive Encryption*

Applocker

- Permet d'**empêcher des programmes non autorisés de s'exécuter** ;
- Évolution de SRP (*Software Restriction Policies*) ;
- Permet de contrôler l'exécution de :
  - PE et DLL (.EXE et .DLL) ;
  - *Microsoft Software Installer* (.MSI et .MSP) ;
  - *Windows PowerShell* (.PS1) ;
  - *batch* (.BAT et .CMD) ;
  - script VisualBasic (.VBS) ;
  - javascript (.JS).

Applocker

- Les règles peuvent interdire ou autoriser l'exécution ;
- Les règles sont associées à un utilisateur ou un groupe ;
- Les règles peuvent être définies :
  - par chemin de fichier ;
  - par *hash* de fichier ;
  - par certificat de signature.
- Pour chaque règle on peut ajouter des exceptions ;
- Il est possible d'activer les règles en *enforce* ou seulement en *audit*.

Applocker

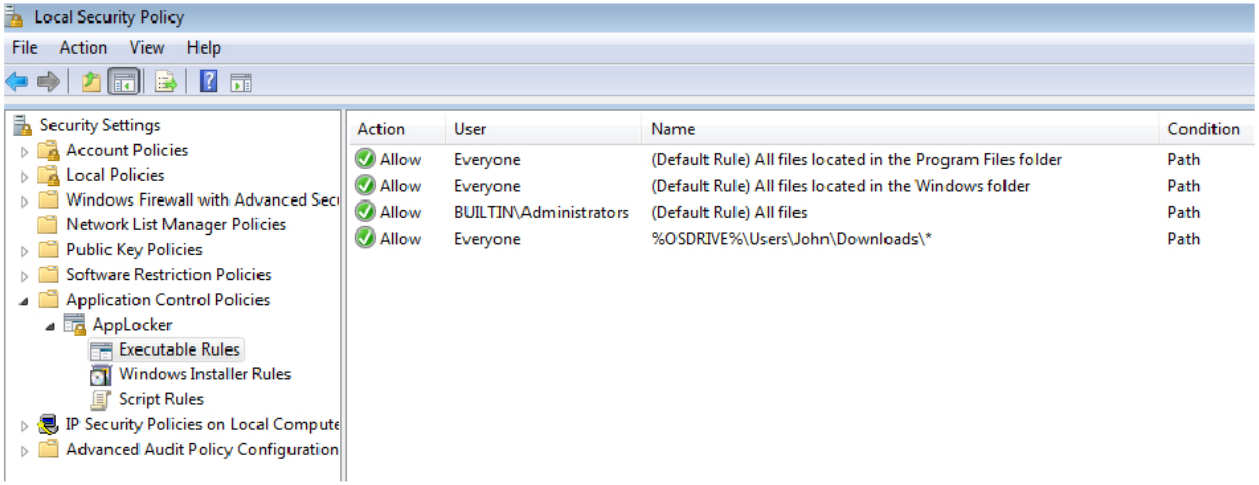


Figure – Manipulation de *Applocker* via *Local Security Policy*

Windows 10

- Windows 10 a apporté de nouveaux ménicanismes de protection comme :
  - *Virtualization-based Security* (VBS) : utilisation de la virtualisation pour isoler des parties critiques de la mémoire du système ;
  - *Credential Guard* : basé sur VBS, protège les authentifiants (stockés dans la mémoire de LSASS) ;
  - *Device Guard* : basé sur VBS, permet de n'exécuter que des applications de confiance.

Commissariat à l'énergie atomique et aux énergies alternatives CEA
Centre de Bruyères-le-Châtel │ 91297 Arpajon Cedex
T. +33 (0)1 69 26 40 00 │ F. +33 (0)1 69 26 40 00
Établissement public à caractère industriel et commercial
RCS Paris B 775 685 019