Jean-Christophe GALLARD

Sécurité sous Windows 2000 / 2003 / XP

Concepts et mise en œuvre

Historique des versions

Version	Date	Modifications
1.0	Mai 2004	Création du document
1.1	Décembre 2004	Refonte globale et changement de mise en forme
1.2	Février/Mars 2005	Corrections mineures

Table des matières

Introduction

« Les ordinateurs, plus on s'en sert moins,
moins ça a de chance de mal marcher. »

Les Shadoks.

Historique

Le système Windows NT a été développé depuis 1989 par Microsoft. Initialement prévu pour fonctionner avec un processeur Risc i860 (le N-Ten, c'est d'ailleurs de là que provient l'acronyme NT) et pour être compatible avec OS/2, rapidement le système s'orienta vers les processeurs Intel x86 et vers une compatibilité avec Windows. La première démonstration de NT eut lieu au Comdex en octobre 1991, puis la première version bêta vit le jour au printemps 1992 avant le lancement de Windows NT 3.1 en juillet 1993. La suite Office fut disponible en août 1994, immédiatement suivie d'une évolution majeure de NT qui devint NT 3.5.

L'étape suivante fut la sortie de Windows NT 3.51 en mai 1995 qui offrait la compatibilité avec Windows 3.11 et en reprenait l'interface graphique. La version actuelle, Windows NT 4.0, sortie en août 1996, reprenait l'interface graphique de Windows 95. Enfin, la première version bêta de Windows NT 5.0 fut adressée aux développeurs en septembre 1997.

En 1999, Microsoft annonçait la disparition de Windows NT 5.0 au profit de Windows 2000, censé fusionner les développements conjoints de Windows 98 et de Windows NT. Windows 2000 est actuellement commercialisé par Microsoft depuis le 17 février 2000.

Parallèlement à cette sortie tant attendue de son nouveau système d'exploitation, Microsoft poursuivait l'évolution de son offre avec le développement de Windows 2003 – ex Windows .NET -, remplaçant annoncé de Windows 2000 Serveur.

Avec cette nouvelle mouture du système d'exploitation de la firme de Redmond, et exclusivement dédiée aux serveur, le système Windows XP, évolution naturelle de Windows 2000 Professionnel, s'intègre tout naturellement dans un rôle de poste client.

Windows 2000

L'apparition de Windows 2000 constitue une véritable révolution dans le processus de développement des systèmes d'exploitation chez Microsoft. Les similitudes entre Windows NT 4.0 et Windows 2000 sont grandes, et ce parce que la technologie sous-jacente reste identique, mais le système Windows 2000 demeure bien plus qu'une version améliorée de Windows NT 4.0.

Si le système Windows NT 4.0 concevait la gestion des ressources qui lui étaient confiées comme une structure « à plat » utilisant des protocoles de gestion et de communication

propriétaires ou adaptés de développements antérieurs[1], le système Windows 2000 intègre dès sa conception une notion de forte hiérarchisation de ses composants et, surtout, une volonté d'ouverture sur le monde de par l'adoption de nombreux protocoles standardisés (Kerberos V5, IPSec, LDAP, SSL/TLS, X509…).

La plus importante (r)évolution de Windows 2000 demeure la notion d'**Active Directory** : cette nouveauté majeure constitue la pierre angulaire de tout l'édifice Windows 2000.

Windows 2003

Longtemps attendue, et maintes fois repoussée, la sortie officielle de Windows 2003 sonne le glas de Windows NT 4.0 Serveur. Si la coexistence pacifique entre Windows NT et Windows 2000 était encore possible, la conception et les choix de paramétrage par défaut de Windows 2003 sont désormais ouvertement hostiles à Windows NT 4.0

Anciennement présenté sous le nom « Windows .NET » auprès du public, Windows 2003 entérine définitivement les choix réalisés pour Windows 2000 et poursuit sa logique de développement.

Curieusement, cette nouvelle version du système ne constitue pas une réelle évolution majeure ; Windows 2003 reprend les mêmes bases que Windows 2000 et en améliore / complète / corrige certains concepts. A vrai dire, seule la notion de « framework .NET » permet à Windows 2003 de s'affirmer comme un nouveau système et non comme une simple évolution de Windows 2000 Serveur et qui en corrige ses bugs résiduels.

[1] Le système Windows NT conserve de nombreuses traces de ses prédécesseurs. Ainsi le système de gestion de fichier NTFS doit-il beaucoup au système HPFS développé à l'origine pour le système OS/2 d'IBM.

Architecture du système

« Parce que moi au départ j'ai fait informatique comme études, pas Windows ! »

Lu sur fr.comp.securite

Préambule

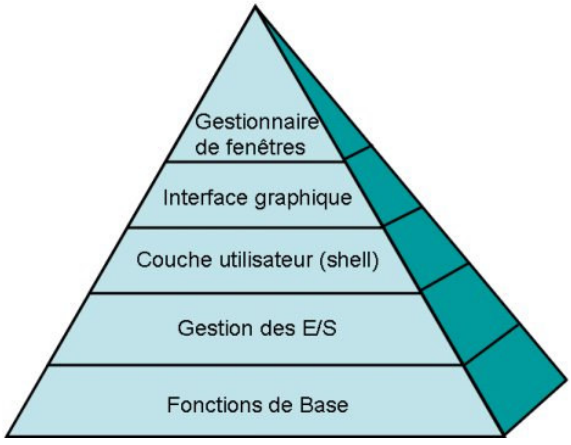
Les systèmes d'exploitation sont généralement constitués de parties qui prennent en charge chacune un certain nombre de fonctionnalités, les parties les plus complexes s'appuyant toutes sur les services plus fondamentaux.

Ainsi, la gestion de la mémoire et des threads d'exécution, et parfois même les systèmes de fichiers, se trouvent naturellement dans les fonctionnalités de base de tous les systèmes.

Au dessus de ces fonctions de base, organisées comme des couches successives de fonctionnalités, on retrouve toutes les fonctions haut niveau du système. On place généralement dans cette catégorie la gestion du réseau, la gestion des périphériques d'entrée / sortie (clavier, écran, ports de communication).

Vient ensuite la couche utilisateur, qui prend en charge l'interface du système avec ses utilisateurs. Cette couche est en général très simplifiée, et fonctionne souvent dans le mode ligne de commande. Ceci signifie que les commandes du système sont saisies au clavier et les résultats sont renvoyés à la suite de leur exécution.

L'interface graphique se place encore au dessus de la couche utilisateur. Celle-ci prend en charge la gestion de toutes les ressources graphiques, mais ne va en général pas plus loin. La gestion des fenêtres par exemple est effectuée par une couche supplémentaire, sur laquelle s'appuient les applications.



Bien que les fonctionnalités des systèmes d'exploitation soient structurées en couches logicielles, la plupart des systèmes sont relativement rigides, du fait d'un grand nombre d'interactions entre les différentes parties du système. Cette rigidité est la source de la grande difficulté que les développeurs éprouvent pour faire évoluer et maintenir les systèmes d'exploitation. C'est aussi le facteur d'instabilité numéro un : le dysfonctionnement d'un service isolé peut engendrer la mort du système complet.

Cette architecture, qui est très courante malgré ces défauts, se dit « **monolithique** ». On trouve dans cette catégorie de système certains systèmes courants, tels que Windows 95/98, et les systèmes Unix (dont GNU/Linux).

À l'opposé, on retrouve l'architecture dite des « **micronoyaux** ». Dans ce modèle de développement, seul les services de base sont exécutés avec des privilèges systèmes. Tous les autres services sont considérés comme des extensions du micronoyau, et sont de ce fait relativement indépendants les uns des autres.

Ce type d'architecture suppose de mettre en place un mécanisme de communication efficace entre les différents modules du système. Ces mécanismes de communication interprocessus ultra performants sont fournis par le micronoyau.

Cette approche micronoyau apporte, du moins en théorie, un grand nombre d'avantages dont le premier d'entre eux demeure la portabilité du système. En effet, dans la mesure où c'est le micronoyau qui contient les parties les plus spécifiques à la plate-forme matérielle, les couches logicielles supérieures ont alors la possibilité d'utiliser un niveau d'abstraction tel qu'elles deviennent moins sujettes à des problèmes de compatibilité. Porter sur une autre architecture matérielle un système utilisant une approche micronoyau devient alors plus facile, puisque seul un noyau réduit doit alors être adapté pour se conformer aux nouvelles spécificités matérielles. D'ailleurs, c'est là la principale raison historique pour laquelle les chercheurs se sont intéressés à cette technique.

Le second intérêt d'une telle approche réside dans la sécurité et surtout la stabilité du système. Si seule une portion réduite de code (le micronoyau) est exécutée avec des privilèges systèmes, le système d'exploitation gagne nécessairement en stabilité.

En effet, quand un processus s'exécutant avec des privilèges systèmes plante, la probabilité pour que tout le système soit affecté par cette erreur d'exécution est grande. A l'inverse, un processus avec peu de privilèges peut difficilement gêner tout le système, d'une part parce son jeu d'instruction réduit lui interdit un certain nombre d'opérations critiques et d'autre part parce qu'il s'exécute sous le contrôle du noyau.

Dès les premiers développements de Windows NT, Microsoft s'était engagé dans la voie du micronoyau en s'inspirant largement de l'architecture du vénérable système MACH et Windows 2000 conserve les traces de ce type d'approche même si celle-ci a due être largement adaptée comme on le verra plus loin.

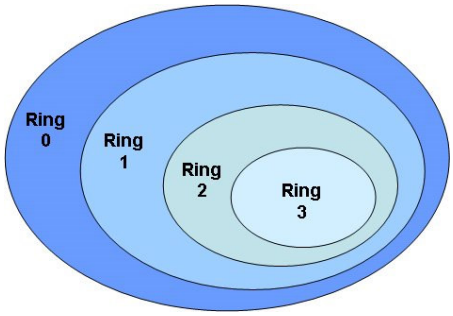
Le seigneur des anneaux

Sur les processeurs modernes, les processus ont la possibilité de s'exécuter selon des modes d'utilisation différents. Les différences entre ces modes portent essentiellement sur le jeu d'instruction disponible. Ainsi, les processeurs Intel de génération x86 (depuis le 80386) supportent jusqu'à 4 niveaux d'exécution, appelés « **rings** » et numérotés de 0 à 3.

En « ring 0 », un processus dispose de l'intégralité du jeu d'instruction du processeur. Il peut adresser directement certains registres et surtout la mémoire physique du système.

En « ring 3 », un processus ne peut directement adresser la mémoire physique et, surtout, il ne peut plus directement revenir en « ring 0 ».

Ainsi, plus le niveau de « ring » augmente, moins le processus dispose d'instructions, ce qui définit les niveaux d'exécution comme des niveaux de privilèges.



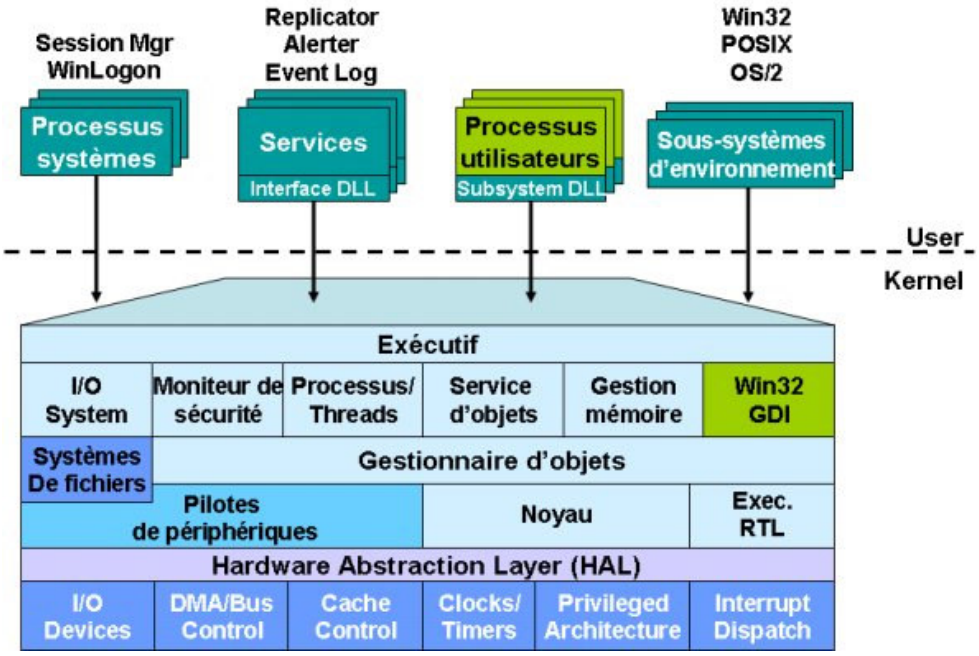
Sous Windows 2000, le système autorise l'exécution des processus selon deux modes distincts : le mode Utilisateur (**User Mode**) et le mode Noyau (**Kernel Mode**), correspondant respectivement aux rings 0 et 3 des architectures Intel.

Pourquoi Microsoft n'a t-il alors pas utilisé au maximum les possibilités du processeur Intel et s'est-il limité aux seuls Rings 0 et 3 ? Parce que le système Windows NT, père direct de Windows 2000, a toujours été conçu pour être porté sur de nombreuses plates-formes matérielles dont des systèmes à processeur RISC. Or, l'immense majorité des processeurs RISC ne dispose que de 2 niveaux d'exécution et non de 4 comme les systèmes Intel x86. Dans un souci de portabilité, mais également de simplicité, le choix a donc été fait de se limiter à deux niveaux d'exécution distincts.

Toute la sécurité du système Windows 2000 repose alors sur ces deux modes d'exécution. Le système tourne en Kernel Mode et fournit alors des services aux processus en User Mode. Un processus en User Mode ne peut passer en Kernel Mode puisque les instructions nécessaires à ce passage lui sont interdites (mieux : elles n'existent pas !). Ne pouvant accéder aux ressources matérielles et à la mémoire physique, ces processus sont donc obligés de passer par un service du système, fonctionnant lui en Kernel Mode et généralement appelé « service dispatcher ».

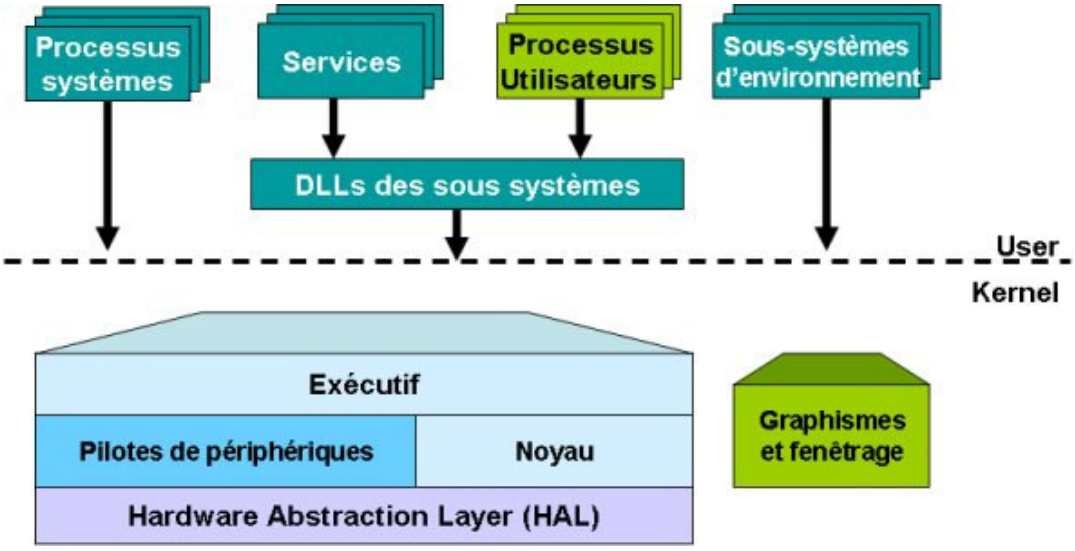
Architecture globale

Le noyau de Windows 2000 est architecturé suivant le schéma présenté ci-dessous :



Au dessus d'une couche d'abstraction matérielle, créée à l'origine pour simplifier le portage du système, on trouve les pilotes de périphériques, le noyau et l'ensemble des gestionnaires du système, organisés par modules distincts.

On peut cependant simplifier cette architecture de la façon suivante :



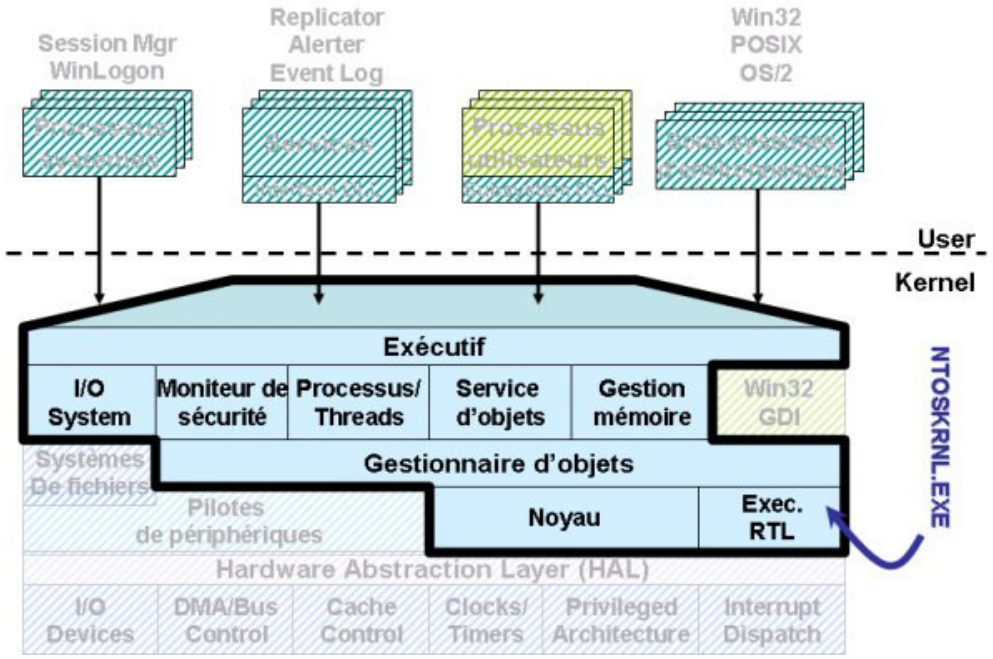
Un des modules fondamentaux de l'exécutif est le gestionnaire d'objet. En effet, le concept de base utilisé dans Windows 2000 est l'objet. Il n'existe pas moins de 27 types d'objet gérés par le système, parmi lesquels :

- les événements (événement système, interruption…),
- les fichiers (fichier réel sur disque ou fichier virtuel correspondant à un périphérique),
- jeton d'accès (identification d'un utilisateur),
- port LPC (utilisé pour la communication inter processus),
- processus,
- sémaphore,
- thread,
- timer.

Le système d'exploitation a la charge de créer et détruire ces objets et, pour ce qui nous intéresse, de gérer des « Security Descriptor » pour chacun d'entre eux : ceux-ci indiquent qui a le droit de faire quoi sur ces objets.

Communication avec le noyau

Le cœur du système Windows 2000 réside dans le fichier NTOSKRNL.EXE. Ce seul fichier contient à la fois le noyau et l'exécutif du système. Les fonctions systèmes disponibles dans le mode utilisateur sont quant à elles exportées par la NTDLL.DLL et les autres sous-systèmes d'environnement.



Notons qu'il existe 4 versions distinctes de ce noyau :

- NTOSKRNL.EXE Version Monoprocesseur.
- NTKRNLMP Version Multiprocesseur.
- NTKRNLPA Version Monoprocesseur avec PAE[1].
- NTKRPAMP Version Multiprocesseur avec PAE.

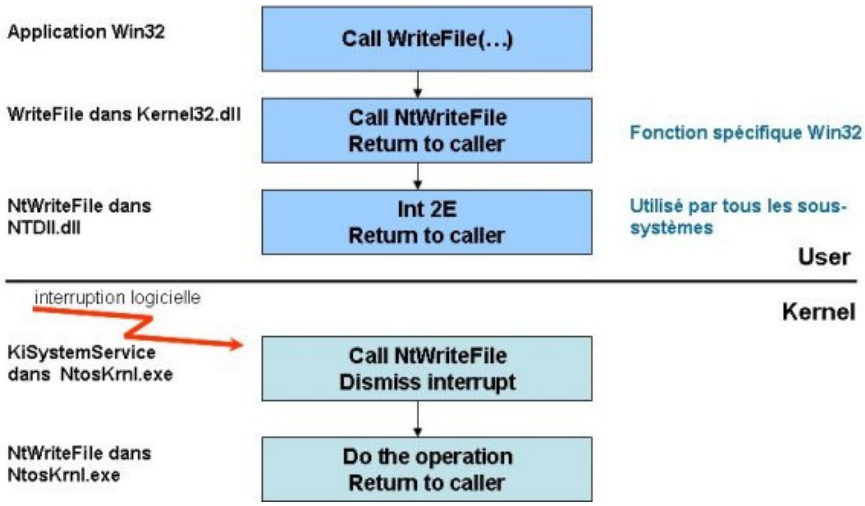
Lorsque l'on souhaite faire appel à une fonction du noyau (écrire dans un fichier par exemple), il suffit donc d'appeler la fonction interne correspondante. La liste des fonctions disponibles du noyau se nomme la « Native API ». On peut reconnaître le composant touché par chaque fonction de la Native API par le préfixe du nom de la fonction.

Préfixe	Composant
Cc	Cache Manager
Ex	Routines de support de l'exécutif
Exp	Routines privées de support de l'exécutif (non exportées)
FsRtl	File System Run-Time Library
Hal	Hardware Abstraction Layer (couche d'abstraction matérielle)
Io	Sous-système d'Entrées/Sorties
Ke	Kernel (Noyau)
Ki	Kernel Internal (non disponible en dehors du noyau)
Lsa	Sous-systèmes d'authentification
Mm	Gestionnaire de mémoire
Ob	Object Manager
Po	Power Management
Ps	Support des processus
Ps	Process Structure
Rtl	Run-Time Library
Se	Sécurité
Wmi	Windows Management Instrumentation
Zw	Accès aux fichiers et au registre

[1] Physical Address Extension

La dernière fonction appelée en mode utilisateur par ces « fonction natives » est alors la fonction « Change Mode to Kernel ». Cet appel génère une interruption logicielle (l'interruption 2E sur les machines x86) qui est traitée par le « service dispatcher » en mode noyau. A l'issue du traitement, le système retourne au User Mode et rend la main.

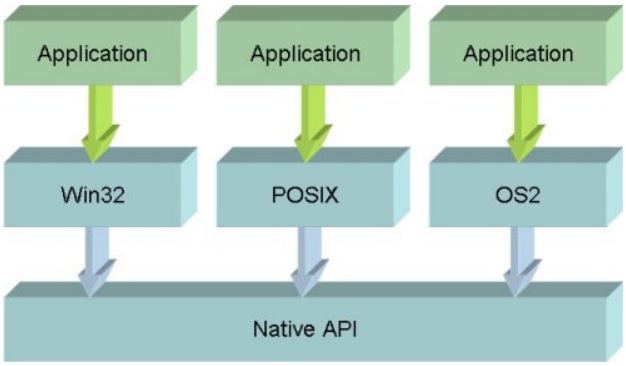
Dans les faits, aucun programme n'est censé accéder directement à cette Native API pour la bonne et simple raison que Microsoft ne la documente pas ! Au lieu d'accéder à cette API native, les développeurs sont invités à utiliser une autre API, dénommée API Win 32 et fournie dans KERNEL32.DLL. Cette API encapsule la native API en réarrangeant et en simplifiant parfois ses paramètres. Un simple appel système d'écriture dans un fichier se transforme alors en une suite de passages de témoins comme l'indique la figue suivante.



L'intérêt principal de ne pas documenter l'API native et de devoir recourir à une API de plus haut niveau comme la Win32 API se fait surtout sentir pour les développeurs du système d'exploitation:

- En premier lieu, les développeurs du noyau peuvent modifier les fonctions internes du noyau sans avoir besoin d'en avertir les développeurs, puisque les appels systèmes ne sont pas censés être directement utilisés.

- Ensuite, avec une telle architecture, il est possible de supporter autant de sous-systèmes différents que nécessaire, tout en faisant appel *in fine* aux mêmes fonctions natives ; une application Win32 pourra lancer un nouveau processus avec l'appel à CreateProcess(), tandis qu'une application POSIX appellera exec() ou fork(), mais, au final, c'est toujours la même fonction native qui sera appelée.



Un micronoyau pas si micro

Dans le milieu des années 1980, vers 1986-87, un groupe d'ingénieurs de Microsoft, dont **Dave Cutler** transfuge de Digital où il exerçait en tant qu'architecte du système d'exploitation VMS, mirent en place un groupe de travail dont le but était de déterminer ce que serait le système d'exploitation du futur. A l'époque, une théorie arrêtée définissait la manière d'écrire un système d'exploitation portable et sécurisé ; tout le monde préconisait d'utiliser une architecture fondée sur les micronoyaux.

C'est sur ces bases que furent conçus les premiers concepts fondateurs de Windows NT. Et pourtant, à y regarder de plus près, le système Windows 2000, bien que très largement inspiré de ce modèle, **n'est pas** un système à micronoyau si l'on se réfère à la stricte définition du concept d'origine.

Dans les noyaux monolithiques, comme celui de Linux, la mémoire est divisée entre l'espace utilisateur et l'espace noyau. L'espace noyau est l'endroit où le code réel du noyau réside après son chargement, et où la mémoire est allouée pour les opérations qui prennent place à son niveau. Ces opérations incluent l'ordonnancement, la gestion des processus et des signaux, des entrées/sorties assurées par les périphériques, de la mémoire et de la pagination.

Un micronoyau effectue un nombre bien plus restreint d'opérations, et sous une forme bien plus limitée ; la communication entre processus, une gestion limitée des processus et de l'ordonnancement ainsi qu'une partie des entrées/sorties de bas niveau. Une architecture à micronoyau est en quelque sorte une manière de s'éloigner des détails du contrôle des processus.

Bien que séduisant sur le papier, les architectures à micronoyau se sont rapidement heurtées à des problèmes de performances ; comme les différents processus tournent dans des environnements cloisonnés, le facteur de performance essentiel de ce type d'architecture demeurent les mécanismes de communications interprocessus et surtout les mécanismes d'appels systèmes, extrêmement gourmands en ressources temporelles. Au fur et à mesure de sa conception, le système Windows NT a ainsi vu son noyau grossir de plus en plus, certains composants se retrouvant basculés du mode utilisateur au mode noyau, de telle sorte qu'il n'est aujourd'hui plus possible de considérer le système comme un micronoyau.

L'exemple type de ce changement demeure la mini révolution qui accompagna le système lors de la sortie de Windows NT 4.0 en 1996. Parmi les nombreuses différences entre Windows NT 3.51 et Windows NT 4.0, il apparaît que l'ensemble des mécanismes de gestion des graphismes (matérialisés par les fichiers USER.EXE et GDI.EXE) fut alors basculé, pour des raisons de performance, du mode User au mode Kernel.

Une GUI dans le noyau

Microsoft a eu une raison valable pour avoir déplacé les fichiers USER.EXE et GDI.EXE vers le mode noyau. Avec la nouvelle interface graphique de Windows 95, Windows NT 4.0 se voyait alourdi davantage que ses prédécesseurs. L'une des conséquences majeures de l'apport d'une interface graphique évoluée sous Windows NT 3.51 a en effet été un ralentissement considérable du système, comparativement aux autres systèmes d'exploitations semblables de l'époque (Windows 3.11 notamment).

Tout est dû aux très nombreux appels systèmes nécessaires pour animer une interface graphique. Avec un Pentium 90, l'accès au matériel depuis le mode utilisateur s'effectue en 70 microsecondes. Depuis le mode noyau, ce même accès s'exécute en quelques microsecondes. Le nombre d'accès a augmenté sans cesse dans la nouvelle interface et le besoin en ressources s'est accru d'autant. Sans le déplacement des fichiers USER.EXE et GDI.EXE vers le mode noyau, Windows NT aurait sans conteste accusé une extrême lenteur.

Effet secondaire du déplacement de ces fichiers vers le mode noyau : la diminution du besoin en mémoire de ces deux applications. L'espace ainsi économisé est rendu aussitôt à l'Explorateur de Windows 95. Tout compte fait, Windows NT 4.0 ne consomme pas plus d'espace que Windows NT 3.51.

L'un des dangers résultant de cette nouvelle organisation est que les routines des fichiers USER.EXE et GDI.EXE peuvent écrire désormais dans les plages mémoire des autres applications noyau. Risque principal : la paralysie de GDI.EXE peut conduire à la mort le système entier !

Sous Windows NT 3.51, l'organisation était telle que GDI.EXE et USER.EXE ne pouvaient pas écrire dans le mode noyau et ne nuisaient pas à la stabilité du système d'exploitation. Pour l'utilisateur dont l'application s'est bloquée, les conséquences sont graves : jusqu'à présent, seules les données de l'application ayant provoqué l'erreur étaient vouées à la perte. Désormais, les données non sécurisées de toutes les applications sont perdues car le système entier est paralysé.

Les pilotes des cartes graphiques, qui étaient à l'époque implémentés en mode utilisateur fonctionnent maintenant directement en mode noyau. Ils peuvent donc influencer la stabilité du système entier. Bien que ces pilotes soient les mêmes qu'autrefois, des dangers supplémentaires surgissent en mode noyau en raison de l'accès direct au matériel effectué par ces pilotes. Et les tests de sécurité n'ont été réalisés que sur les pilotes des composants figurant dans la liste de compatibilité de Microsoft…

Modèle de Sécurité de Windows 2000

«
```
#define P(X)j=write(1,X,1)
#define C 39
int M[5000]={2},*u=M,N[5000],R=22,a[4],l[]={
0,-1,C-1,-1},m[]={1,-C,-1,C},*b=N,*d=N,c,e,f
,g,i,j,k,s;main(){for(M[i=C*R-1]=24;f|d>=b;)
{c=M[g=i];i=e;for(s=f=0;s<4;s++)if((k=m[s]+g
)>=0&&k<C*R&&l[s]!=k%C&&(!M[k]||!j&&c>=16!=M
[k]>=16))a[f++]=s;if(f){f=M[e=m[s=a[rand()/(
1+2147483647/f)]]+g];j=j<f?f:j;f+=c&-16*!j;M
[g]=c|1<<s;M[*d++=e]=f|1<<(s+2)%4;}else e=d>
b++?b[-1]:e;}P(" ");for(s=C;--s;P("_"))P(" "
);for(;P("\n"),R--;P("|"))for(e=C;e--;P("_ "
+(*u++/8)%2))P("| "+(*u/4)%2);}
```
 »

Carl Shapiro – Grand prix de l'IOCCC[1] 1985

Principes généraux

Chaque **objet** du système est protégé par un **descripteur de sécurité** qui va définir quels types d'accès vont être autorisés de la part des différentes entités.

Une entité n'accède pas directement à un objet, elle le fait via un processus qui fonctionne sous son identité : c'est donc au niveau du processus que va être définit le contexte de sécurité qui va permettre, ou non, d'accéder à des objets. Cette structure attachée aux processus s'appelle un **jeton**.

Avant de pouvoir lancer un processus, une entité doit **s'authentifier** auprès du système. L'ouverture de session définit alors le contexte de sécurité d'une entité connectée à un système donné.

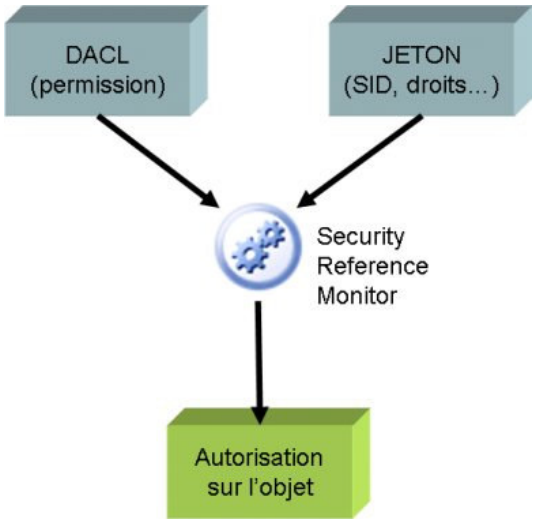
Le modèle de sécurité de Windows 2000 repose donc sur des mécanismes d'autorisation, que l'on positionne sur les objets gérés par le système.

L'ouverture de session est gérée par la **Local Security Authority**, ou **LSA,** matérialisée par le processus LSASS.EXE ; ce processus fonctionne en mode utilisateur et assure une communication directe avec le module noyau appelé SRM (Security Reference Monitor).

[1] http://www.ioccc.org/. L'Internet Obfuscated C Code Contest, ou IOCCC, est un concours de programmation en langage C dont la règle principale consiste à écrire un programme le plus obscurci possible.

Le rôle du SRM est de valider les accès aux objets et d'effectuer la journalisation des évènements de sécurité. La politique locale de sécurité d'une machine (droits et privilèges des utilisateurs, classes d'évènements à auditer…) est stockée dans la base de registre sous HKLM\SECURITY.

L'accès à un objet fait donc intervenir plusieurs paramètres : les permissions sur l'objet, le jeton du processus requérant, et le composant de validation de l'accès :



Enfin, et afin d'assurer le **cloisonnement** (et donc la sécurité) des processus entre eux, chaque processus évolue dans un espace mémoire qui lui est propre ; le système garantit alors le fait qu'un processus ne puisse accéder à l'espace mémoire d'un autre processus, sauf si un privilège particulier le lui autorise.

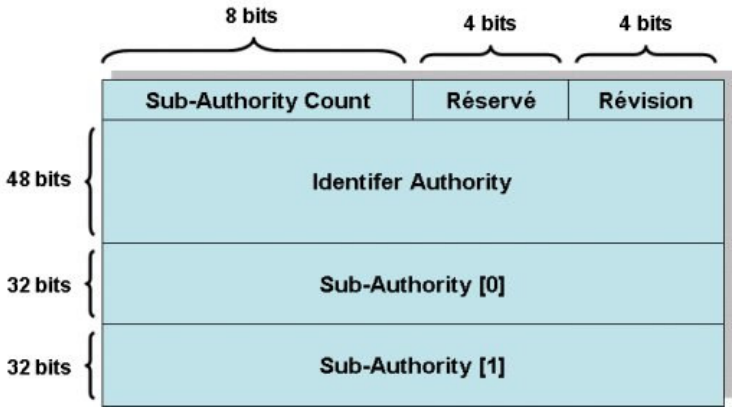
Notion de SID

Au lieu d'utiliser des noms (qui peuvent ne pas être uniques) pour identifier les entités qui effectuent des actions sur un système, Windows utilise des Security Identifiers (SID).

Les utilisateurs, les groupes, les machines, les domaines, les membres des domaines,… sont donc identifiés de façon unique par des SID.

Un SID est une valeur numérique de longueur variable qui est constituée :

- D'un numéro de révision de structure de SID
- D'un identificateur d'autorité sur 48 bits
- D'un nombre variable de sous autorité sur 32 bits, appelées également RID (Relative Identifier)



La valeur de l'autorité (identifier authority) identifie l'agent qui a émis le SID ; cet agent est typiquement un système local Windows 2000/3 ou un domaine.

Les valeurs de sous autorité identifient les « trustees » relativement à l'autorité d'émission et les RID sont simplement des moyens simples de créer des SID uniques sur la base d'un SID de base. Etant donnée la longueur d'un SID et le fait que Windows génère des valeurs aléatoires au sein de chaque SID, il est **virtuellement impossible** pour Windows d'émettre le même SID deux fois sur des machines ou des domaines de par le monde.

Lors de leur mise en forme sous forme de texte, chaque SID a un préfix sous la forme de la lettre « S » et ses différents composants sont séparés par des tirets (« - ») :

S-1-5-21-583907252-1078145449-1957994488-500

Dans ce SID, le numéro de révision est 1, la valeur d'identificateur d'autorité est 5 (l'autorité de sécurité de Windows 2000), on a 4 valeurs de sous autorité et un RID (500)

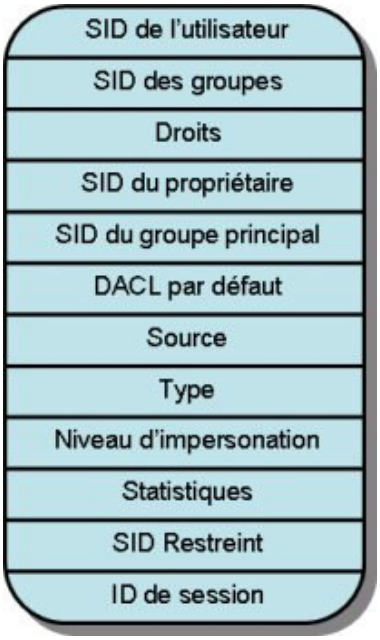
Voici quelques règles de construction des SID :

- Quand on installe Windows, le programme d'installation affecte à la machine un SID
- Windows affecte également un SID aux comptes locaux de la machine
- Chaque SID de compte local est basé sur le SID de la machine avec un RID différent à la fin : les RID pour les comptes utilisateur et les groupes démarrent à 1000 et sont incrémentés de 1 pour chaque nouvel utilisateur ou groupe
- De même Windows affecte un SID pour chaque nouveau domaine Windows.
- Chaque SID de domaine est basé sur le SID du domaine avec un RID différent à la fin : les RID pour les comptes utilisateur et les groupes démarrent à 1000 et sont incrémentés de 1 pour chaque nouvel utilisateur ou groupe.

Certains SIDs sont réservés à des comptes (systèmes, groupes, machines…) connus. La liste exhaustive de ces SIDs pour le système Windows 2000 est présentée en annexe de ce document.

Jetons d'accès

Une fois qu'un utilisateur a été authentifié au sein de sa session, celui-ci se voit remettre un jeton d'accès. Ce jeton est utilisé pour toute la durée de la session est n'est libéré que lorsque l'utilisateur ferme sa session. Il est utilisé pour représenter l'utilisateur dans toutes les demandes d'accès aux ressources du système.



Notons que ce jeton contient un certain nombre d'informations additionnelles sur l'utilisateur, et utilisées par le sous-système de sécurité de Windows NT.

La table suivante précise les rôles de chacune de ces informations :

Composant du jeton	**Description**
SID de l'utilisateur	Représentation numérique du compte utilisateur pour identifier l'utilisateur sur le système.
SID des groupes	Représentation numérique des groupes (locaux et globaux) auxquels appartient l'utilisateur.
Droits	Les droits qu'un utilisateur possède.
SID du propriétaire	Le SID de l'utilisateur ou du groupe qui, par défaut, devient le propriétaire des objets créés avec ce jeton. Ce SID est généralement identique au SID de l'utilisateur, sauf pour les comptes de type administrateur ; dans ce cas, ce SID correspond à celui du groupe « Administrateurs »
SID du groupe principal	Le SID du groupe principal d'appartenance de l'utilisation. Ce champ n'est utilisé que par le sous-système POSIX.
DACL par défaut	Une liste d'ACLs par défaut que le système applique aux objets créés avec ce jeton et si aucune ACL n'a été précisée lors de la création. L'ACL par défaut accorde le Contrôle Total au créateur Propriétaire.
Source	Le processus qui a généré ce jeton (par exemple le Session Manager, le LAN Manager, ou le serveur Remote Procedure Call - RPC).
Type	Une valeur indiquant s'il s'agit d'un jeton primaire ou d'un jeton d'impersonation. Un jeton primaire est un jeton qui représente le contexte de sécurité d'un processus. Un jeton d'impersonation est un jeton qu'un thread peut utiliser au sein d'un processus de service afin d'adopter temporairement un autre contexte de sécurité (généralement le contexte de sécurité du client du service)
Niveau d'impersonation	Fixe le niveau d'impersonation pour le jeton d'accès, et détermine les informations qu'un autre processus dispose pour le compte d'un client (voir le chapitre

Composant du jeton	Description
	traitant de l'impersonation, plus loin dans ce document).
Statistiques	Informations utilisées en interne du système et comportant des statistiques d'utilisation.
SIDs restreints	Une liste optionnelle de SID ajoutée par un processus disposant des privilèges de créer des jetons restreints. Un SID restreint permet de limiter les accès d'un thread à un niveau de privilège inférieur à celui du processus père.
ID de session	Une valeur indiquant si le jeton est associé à une session Terminal Server.

Les jetons contiennent aussi un champ « Expiration time » qui n'est pas utilisé pour le moment. Ce champ pourrait permettre à terme la mise en place réelle de l'expiration d'un compte : aujourd'hui, si l'utilisateur reste connecté après la date d'expiration du compte, le système laisse l'utilisateur continuer d'accéder aux ressources. Aujourd'hui, la seule solution pour empêcher l'utilisateur de continuer d'accéder aux ressources est donc de forcer une déconnexion.

Note :

Windows 2000 introduit un nouveau type de jeton appelé « jeton restreint » (restricted token). Un jeton restreint est un jeton « copie » dérivé d'un jeton primaire ou d'impersonation, avec cette différence que certains privilèges peuvent être révoqués dans ce jeton.

Ce mécanisme peut être utilisé par un thread qui souhaiterait réaliser une impersonation du client mais avec des privilèges réduits, par exemple pour du code venant d'un domaine n'étant pas de confiance.

Il peut également être utilisé pour lancer un processus sensible avec le juste niveau de privilège requis pour son fonctionnement.

```
kd> dt _TOKEN
   +0x000 TokenSource      : _TOKEN_SOURCE
   +0x010 TokenId          : _LUID
   +0x018 AuthenticationId : _LUID
   +0x020 ParentTokenId    : _LUID
   +0x028 ExpirationTime   : _LARGE_INTEGER
   +0x030 TokenLock        : Ptr32 _ERESOURCE
   +0x034 ModifiedId       : _LUID
   +0x03c SessionId        : Uint4B
   +0x040 UserAndGroupCount : Uint4B
   +0x044 RestrictedSidCount : Uint4B
   +0x048 PrivilegeCount   : Uint4B
   +0x04c VariableLength   : Uint4B
   +0x050 DynamicCharged   : Uint4B
   +0x054 DynamicAvailable : Uint4B
   +0x058 DefaultOwnerIndex : Uint4B
   +0x05c UserAndGroups    : Ptr32 _SID_AND_ATTRIBUTES
   +0x060 RestrictedSids   : Ptr32 _SID_AND_ATTRIBUTES
   +0x064 PrimaryGroup     : Ptr32 Void
   +0x068 Privileges       : Ptr32 _LUID_AND_ATTRIBUTES
   +0x06c DynamicPart      : Ptr32 Uint4B
   +0x070 DefaultDacl      : Ptr32 _ACL
   +0x074 TokenType        : _TOKEN_TYPE
   +0x078 ImpersonationLevel : _SECURITY_IMPERSONATION_LEVEL
   +0x07c TokenFlags       : UChar
   +0x07d TokenInUse       : UChar
```

```
+0x080 ProxyData        : Ptr32 _SECURITY_TOKEN_PROXY_DATA
+0x084 AuditData        : Ptr32 _SECURITY_TOKEN_AUDIT_DATA
+0x088 VariablePart     : Uint4B
```

Structure exacte d'un Jeton, telle que révélée par le Kernel Debugger

Descripteurs de sécurité

Des descripteurs de sécurité (Security Descriptors - SD) sont assignés à tous les objets sécurisables de Windows 2000, lors de leur création. Ces descripteurs sont utilisés, en interne du système d'exploitation, afin de déterminer les permissions qu'un utilisateur dispose sur ces objets.

Les principaux objets sécurisables du système sont les suivants :

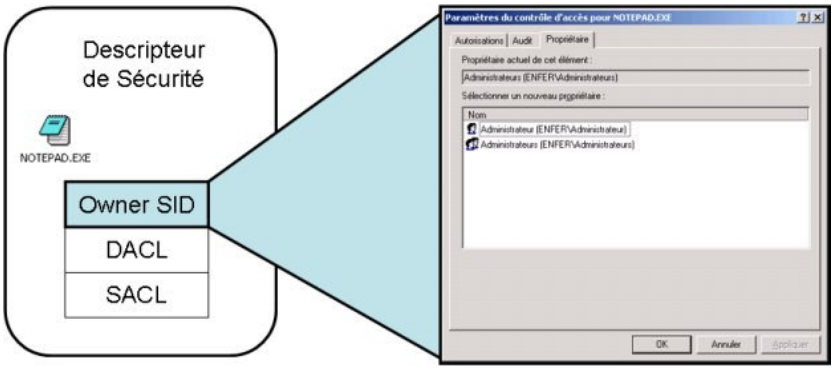
- Fichiers
- Répertoires
- Mappings de fichiers
- Pipes de communications interprocessus
- Mailslots
- Processus
- Threads
- Jetons d'accès
- Machines Windows
- Bureaux
- Clefs de registre
- Objets Services
- Périphériques
- Objets de Synchronisation (événements, mutexs, sémaphores)
- Objets en mode Utilisateurs
- Objets en mode Noyau

Un descripteur de sécurité contrôle donc qui a accès à un objet, il comporte les attributs suivants :

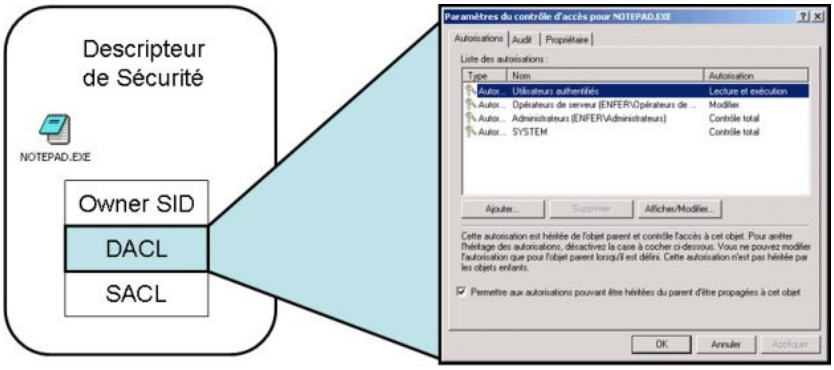
Attributs d'un Descripteur de Sécurité	Description
SID du propriétaire (Owner SID)	L'identificateur de sécurité du propriétaire de l'objet.
SID du groupe	Identificateur de sécurité du groupe principal pour l'objet (utilisé uniquement par le sous système POSIX)
Discretionary ACL (DACL)	Le DACL détermine quels utilisateurs et/ou groupes ont accès à cet objet ainsi que le type d'accès autorisé.
System ACL (SACL)	Le SACL est utilisé pour déterminer ce qui doit être audité sur cet objet.

Quand un utilisateur ou un processus requiert un accès à un objet sécurisé, le processus requérant présente d'abord son jeton d'accès au SRM qui vérifie alors les permissions sur cet objet en interrogeant le Security Descriptor de l'objet. Si les permissions positionnées ne permettent pas l'accès à cet objet ou si l'utilisateur (ou le processus) appelant ne dispose pas d'un privilège lui donnant accès à cet objet, la requête est rejetée. Dans le cas contraire, la requête est acceptée et éventuellement auditée si le SACL le précise

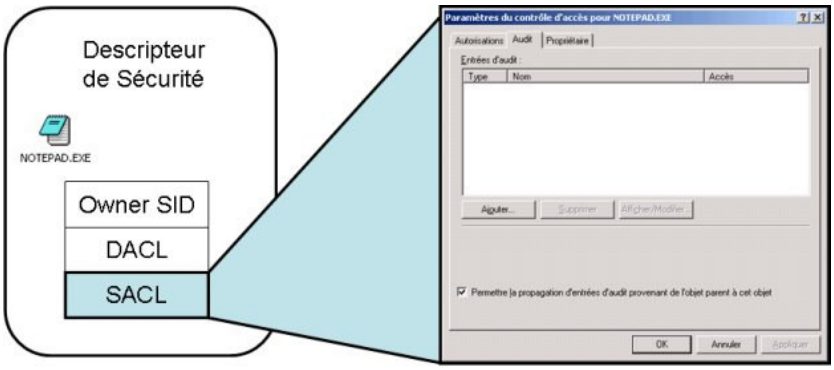
L'exemple qui suit donne les différents éléments pour le programme NOTEPAD.EXE. Ces trois fenêtres sont accessibles en cliquant sur le bouton droit de la souris sur le fichier correspondant et en sélectionnant l'option "Propriétés" ; le bouton "Avancé" propose 3 onglets, correspondant chacun aux trois figures qui suivent.



Propriétaire



Autorisations



Audit

L'emplacement précis du stockage des Security Descriptors dans le système de fichiers sera décrit plus loin, dans le chapitre relatif au système de gestion de fichier NTFS.

Droits des utilisateurs

Sous Windows 2000, on réalise une distinction sémantique entre « **droits** » et « **permissions** ».

Une *permission* correspond à une autorisation d'accès à un objet du système ; une ACL représente une permission (on parle également d'autorisations).

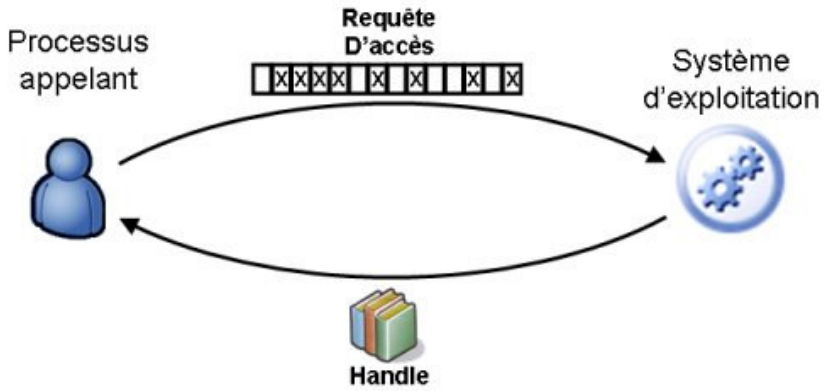
Un *droit* est un privilège accordé à un utilisateur ou à un groupe, indépendamment de tout objet du système : par exemple, un opérateur de sauvegarde peut sauvegarder tout un répertoire même si les ACLs positionnées lui en interdisent l'accès, et ce parce qu'il dispose du droit « sauvegarder des fichiers et des répertoires ».

Lorsque l'on relève un problème d'accès à une ressource, si il ne s'agit pas d'un problème de permissions, il y a de fortes chances qu'il s'agisse d'un problème de droits insuffisants.

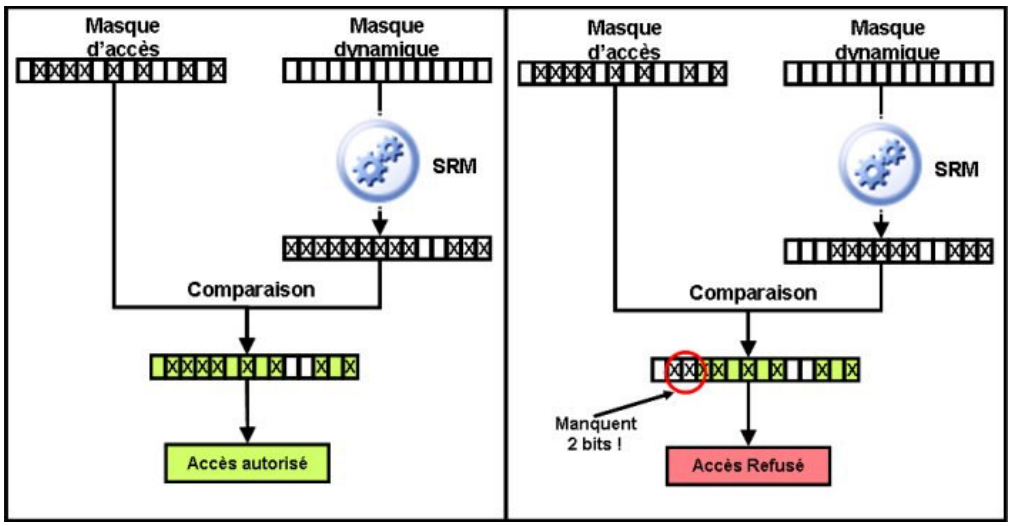
Détermination de l'accès à un objet

Un processus n'accède **jamais** directement à un objet ; il le fait de manière indirecte en réalisant une requête auprès du noyau, qui ne renvoie pas un pointeur sur l'objet mais une référence interne appelée « handle ». Tous les accès ultérieurs à l'objet seront alors effectués en précisant ce « handle ». Ce mécanisme de « handle » permet de protéger les objets contre des accès directs à leurs structures internes.

La requête initiale permettant de récupérer un handle sur un objet doit alors préciser quel type d'accès il est demandé (lecture, écriture…), en passant au système un masque d'accès (champ de bits dans lequel on positionne certains bits à 1 en fonction de l'accès requis). Le noyau raccroche alors à ce handle le masque d'accès utilisé lors de l'appel initial ; il n'est donc pas possible de requérir un handle sur un objet en **lecture** et d'utiliser ce handle pour accéder en **écriture** à l'objet

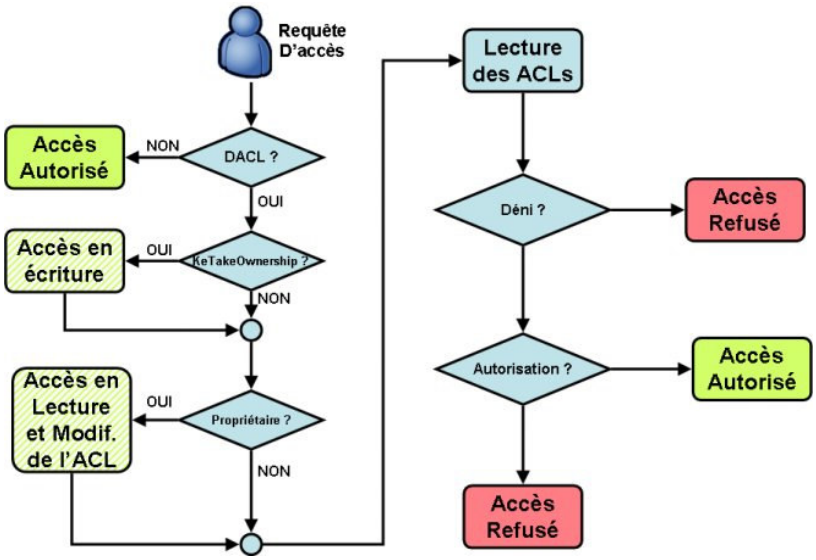


Le sous-système de sécurité détermine alors si l'appelant dispose des privilèges nécessaires à l'exécution de sa requête. Le principe général consiste à partir d'un masque dynamique vide (tous les bits à 0), puis à mettre à 1 les bits d'accès de ce masque au fur et à mesure que les autorisations sont rencontrées. A la fin de l'opération, ce masque dynamique est comparé au masque d'accès fourni lors de l'appel. Si les autorisations sont suffisantes, l'accès est garanti, le handle est créé et transmis à l'appelant. Précisons en outre qu'une autorisation de type « déni d'accès » est considérée comme une opération *absorbante*, annulant toute permission sur le bit considéré.



L'algorithme de détermination de l'accès est précisé ci-dessous :

- Si l'objet ne dispose pas de DACL (pointeur nul), l'objet n'est pas protégé et l'accès est garanti.
- Si l'appelant dispose du droit « Prendre possession des objets », l'accès est autorisé en écriture.
- Si l'appelant est le propriétaire de l'objet, l'accès est autorisé pour « Lire l'ACL » (READ_CONTROL) et « Ecrire l'ACL » (Write DACL)
- Chaque ACE dans l'ACL est examinée :
    - Si un déni d'accès est positionné pour le(s) SID(s) de l'appelant, l'accès est refusé
    - Si un accès est autorisé pour le SID de l'appelant, l'accès est autorisé
    - Si la fin de l'ACL est atteinte sans qu'une autorisation ou un déni d'accès soit rencontré, l'accès est refusé



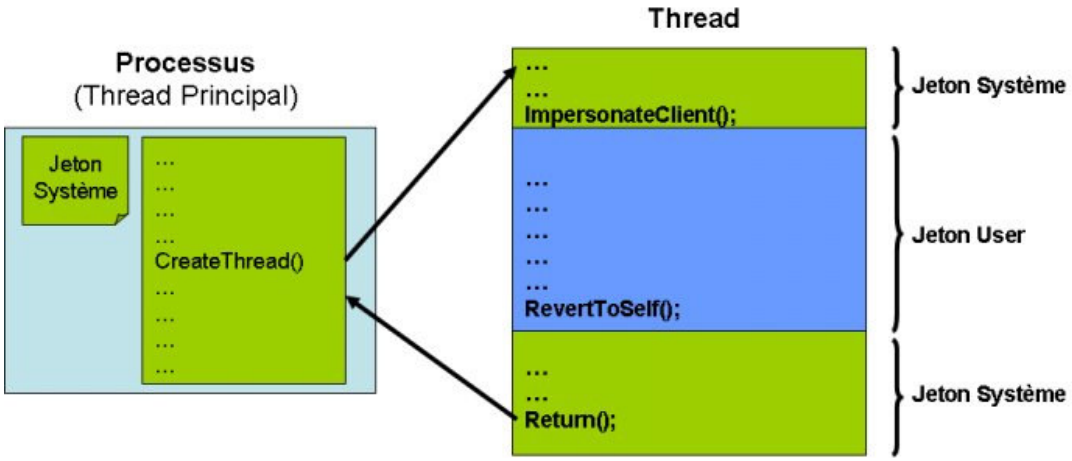
L'impersonation

Principe

Lorsqu'un utilisateur accède à un programme, celui-ci s'exécute dans le contexte de sécurité de l'utilisateur. La combinaison du jeton d'accès et du programme lancé s'appelle un « Sujet ». Souvent, un même « Sujet » peut être amené à appeler d'autres processus pour réaliser sa tâche, comme un processus serveur.

Il peut parfois être utile qu'un processus serveur utilise le contexte de sécurité de l'utilisateur ayant fait appel à lui, plutôt que son propre contexte de sécurité, tout simplement parce qu'un processus serveur peut disposer de privilèges moindres ou supérieurs à ceux de l'utilisateur ayant fait appel à lui. Si le procédé d'impersonation n'existait pas, les requêtes à des serveurs dépendraient directement du contexte de sécurité dans lequel a été lancé le processus serveur.

Ainsi, dans le cas d'une requête d'accès à un fichier, le processus serveur de fichiers peut disposer de privilèges systèmes. Si l'appel a lieu dans le contexte de sécurité du serveur de fichier, l'utilisateur pourra donc avoir accès à TOUT le système de fichiers, outrepassant ainsi les permissions restrictives qui auraient pu être positionnées sur les fichiers. Le mécanisme d'impersonation permet de contourner la difficulté en effectuant la requête dans le contexte de sécurité de l'utilisateur et donc en obligeant le serveur à réagir comme s'il était lui-même l'utilisateur ayant réalisé la requête.



Sécurité du mécanisme

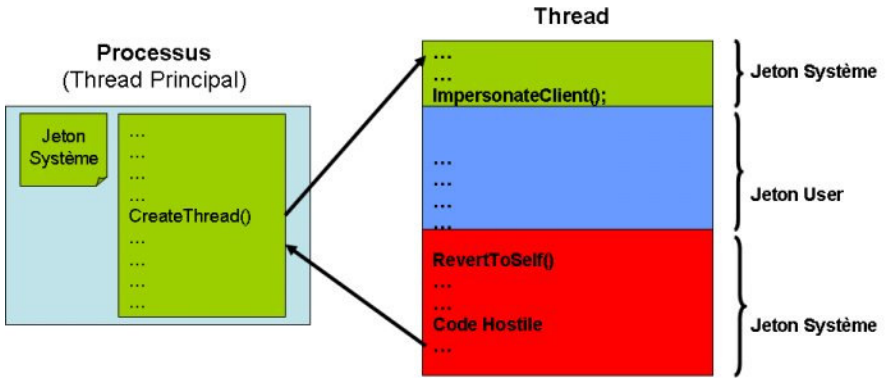
Il est toutefois nécessaire de noter que le procédé d'impersonation s'effectue **dans** le thread et qu'il est possible de mettre fin **à tout moment** à ce mécanisme. En d'autres termes, le procédé ne permet pas de se protéger, par exemple, en cas de buffer-overflow sur le thread en impersonation.

En effet, si l'on traite au sein d'un processus serveur (un service par exemple, avec des privilèges SYSTEM) une requête d'un utilisateur en utilisant le mécanisme d'impersonation, le traitement de la requête doit être exemplaire. Dans le cas contraire, le client peut alors forger une requête, en profitant par exemple d'un buffer-overflow dans le passage des paramètres, et faire exécuter, par le thread d'impersonation, du code injecté.

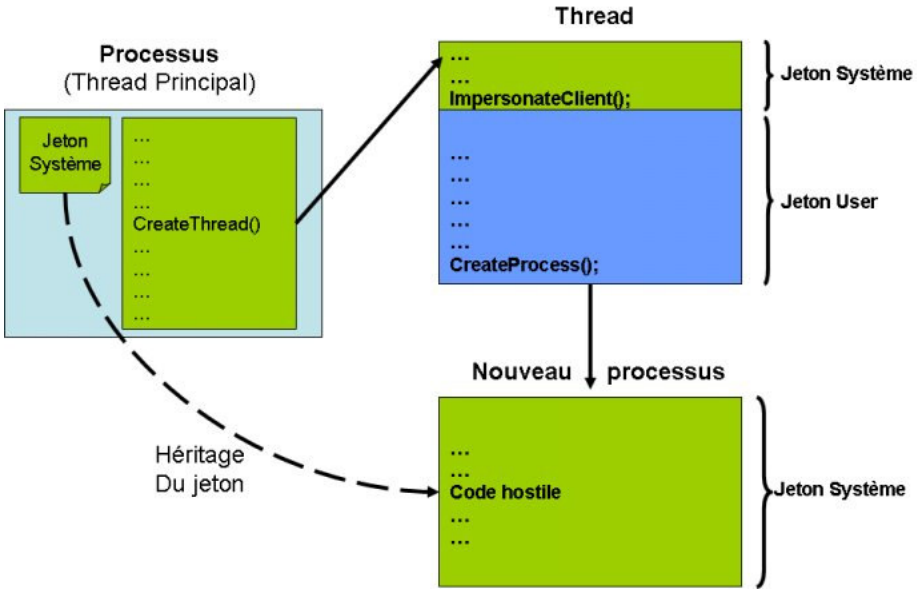
L'attaquant dispose alors de deux techniques pour sortir du contexte « utilisateur » du thread :

- Soit il provoque une sortie explicite de l'impersonation en réalisant un appel à *RevertToSelf(),*
- Soit il lance un nouveau processus en réalisant un appel à *CreateProcess()*.

Dans le premier cas, on se retrouve dans le thread d'impersonation, avec les privilèges du processus principal.



Dans le second cas, un appel à *CreateProcess()* génère un nouveau processus qui, et c'est le comportement par défaut du système, va alors hériter du jeton du processus principal et non du jeton du thread appelant.



Les niveaux d'impersonation

Le jeton d'accès accroché à tout processus dispose d'un champ dans lequel se trouve stocké le « niveau d'impersonation » que le processus propriétaire est autorisé à réaliser.

Ce niveau, positionné par le système d'exploitation, autorise donc le processus en cause à réaliser l'impersonation selon les quatre méthodes suivantes :

Niveau d'Impersonation	Description
SecurityAnonymous	Le processus serveur ne peut identifier le client. Pas de possibilité d'impersonation
SecurityIdentification	Le processus serveur ne peut qu'identifier le client (et donc accéder à son jeton d'accès) mais ne peut pas réaliser d'impersonation. Utile pour déterminer au niveau du serveur si un utilisateur a le droit d'effectuer telle ou telle requête : c'est alors le processus serveur qui gère les permissions et non le système d'exploitation.
SecurityImpersonation	Le processus serveur peut identifier le client (et donc accéder à son jeton d'accès) et peut réaliser une impersonation en local. Il ne peut pas réaliser d'impersonation pour un client distant.
SecurityDelegation	Disponible uniquement sous Windows 2000 et 2003. Le processus serveur peut identifier le client (et donc accéder à son jeton d'accès) et peut réaliser une impersonation pour un client local **et** distant.

Les différents types de groupes

Afin de pouvoir regrouper efficacement les utilisateurs présentant les mêmes besoins, le système Windows 2000 offre la notion de groupes de sécurité. Windows 2000 définit 4 types de groupes :

- Groupe **Local**
- Groupe de **Domaine Local**
- Groupe **Global**
- Groupe **Universel** (disponible uniquement en mode natif)

Dans une utilisation standard, les groupes globaux et universels servent à définir des ensembles de personnes et n'ont pas de droits et de permissions directement associées. Ils sont ensuite ajoutés à des groupes locaux, qui eux définissent les droits et permissions relatifs au domaine concerné.

On trouve également la notion de **groupe de distribution** : il ne s'agit pas d'un type de groupe auquel on peut se référer dans le positionnement d'ACLs, mais d'un groupe particulier permettant de gérer des listes de diffusion de messagerie (avec Microsoft Exchange par exemple).

Les différences entre ces groupes sont présentées dans le tableau suivant :

	Peut contenir…	Est visible de…	S'applique à…
Local	Utilisateur local Utilisateur global Groupe global	La machine	La machine
Domaine Local	Utilisateur global Machine Contact Groupe global Groupe universel	Du domaine	Au domaine
Global	Utilisateur global Machine Contact	De la forêt	Au domaine
Universel	Utilisateur Machine Contact Groupe global Groupe universel	De la forêt	La forêt

L'API AuthZ

L'API Windows AuthZ a été introduite avec Windows XP et implémente le même modèle de sécurité que le SRM mais en mode user (\Windows\System32\Authz.dll).

Ceci donne aux applications qui veulent protéger leurs propres objets privés (tels que des tables de base de données) la possibilité de bénéficier du modèle de sécurité de Windows sans avoir à payer le coût des transitions en mode noyau.

L'API AuthZ utilise des structures de données standard de security descriptor, de SID et de privilèges. Au lieu d'utiliser des jetons pour représenter des clients, AuthZ utilise un contexte de sécurité particulier (le AUTHZ_CLIENT_CONTEXT).

AuthZ utilise les équivalents en mode user pour tous les contrôles d'accès et les fonctions de sécurité de Windows.

Par exemple, AuthzAccessCheck() est l'équivalent de l'API AccessCheck() qui utilise la fonction SeAccessCheck() du SRM.

Un autre avantage pour les applications utilisant AuthZ est qu'elles peuvent demander à AuthZ de cacher les résultats des contrôles de sécurité pour améliorer les performances des contrôles suivants qui utilisent le même contexte client et le même SD.

AuthZ est documenté dans le « Platform SDK », disponible sur le site Internet de Microsoft.

Mécanismes d'authentification

« Sésame, ouvre-toi ! »

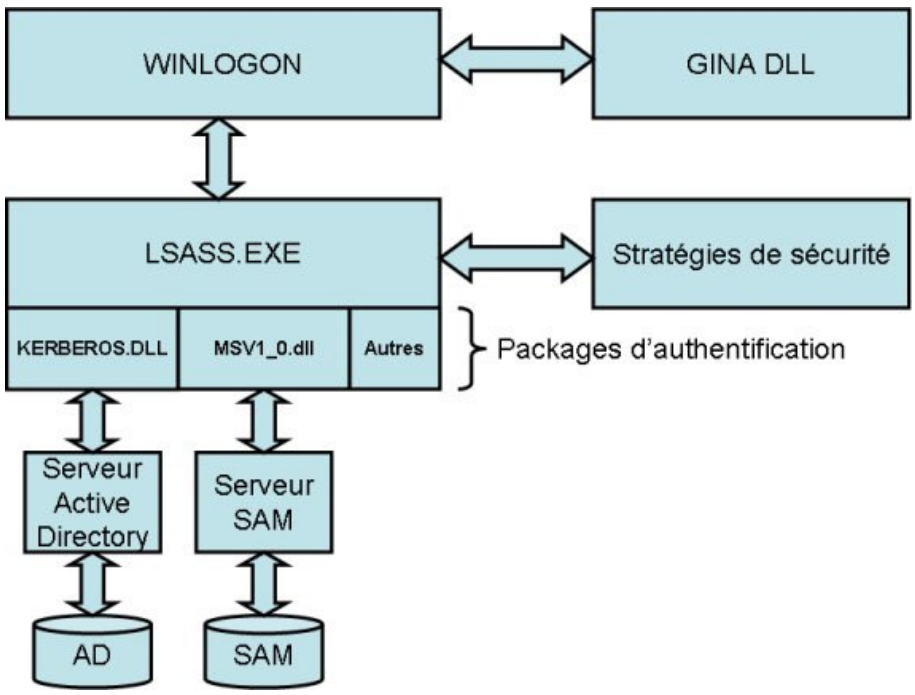
Ali Baba – Les Milles et une nuits

Principes de l'ouverture de session

Architecture

Le « logon » ou ouverture de session interactive, par opposition à l'ouverture de session réseau (network logon), est une procédure **obligatoire** mettant en jeu de nombreux composants du système parmi :

- Le processus Winlogon
- La « Local Security Authentication Authority », implémentée par le processus LSASS.EXE,
- Un ou plusieurs packages d'authentification « authentication packages »
- La SAM ou l'Active Directory.



Chaque utilisateur doit donc procéder à une ouverture de session afin d'utiliser les ressources de cette machine ou du réseau sur laquelle elle est installée. Une fois ce processus réalisé, un jeton d'accès est généré, qui contient des informations de sécurité spécifique à l'utilisateur concerné, entre autres : identificateur de sécurité (SID), identificateur de groupes, droits et permissions utilisateur. L'utilisateur, ainsi que tout processus lancé par celui-ci, est alors identifié par ce jeton.

Les packages d'authentification

Les packages d'authentification sont présents dans le système sous la forme de DLLs implémentant les mécanismes de vérification de l'authentification.

Le protocole Kerberos (KERBEROS.DLL) est le package d'authentification par défaut sous Windows 2000 lors d'une ouverture de session interactive dans un domaine, tandis que le MSV1_0 (présent dans la MSV1_0.DLL) assure les vérifications d'identité dans le cas d'une ouverture de session interactive locale, lors d'une ouverture de session dans un domaine pré Windows 2000 (un domaine Windows NT 4.0 par exemple) ou lorsque qu'aucun contrôleur de domaine n'est joignable.

La GINA Dll

Le Winlogon ne gère pas directement la partie graphique de l'ouverture de session, qui se trouve déportée dans une librairie spécialisée appelée une GINA (Graphical Identification and Authentication).

La GINA par défaut de chaque système Windows est la MSGINA.DLL, mais il est possible de coder soi-même sa propre GINA pour remplacer celle de Microsoft ; c'est par ailleurs ce que font certains éditeurs de logiciels comme Novell, et ce afin d'assurer l'authentification depuis des machines Windows sur leur serveur Novell Netware.

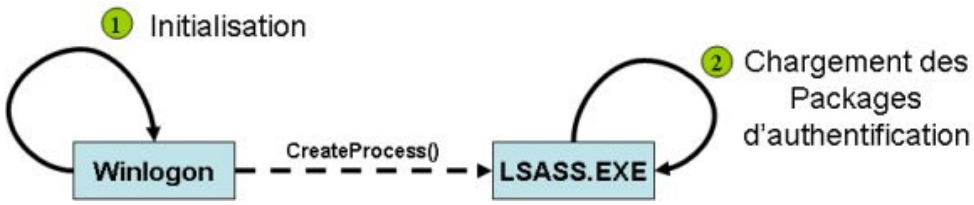
Fonctionnellement une GINA se présente sous la forme d'une DLL exportant certaines fonctions de façon standardisée.

Le processus Winlogon

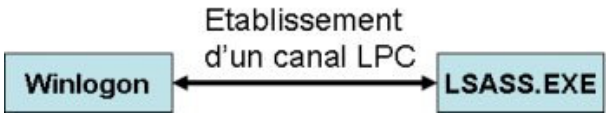
Le processus Winlogon est un processus de confiance initial pour toute authentification sur le système ; il assume un rôle de coordinateur des ouvertures de session, est responsable du lancement du Shell utilisateur, des changements de mots de passe et du verrouillage déverrouillage des écrans de veille.

Au cours du démarrage de Windows 2000, le processus WINLOGON est lancé et crée un environnement contrôlé d'ouverture de session. Les 4 étapes décrites par la suite sont celles qui sont effectuées avant une quelconque interaction avec l'utilisateur.

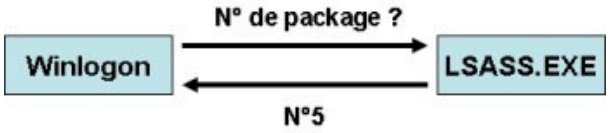
1. WINLOGON crée une fenêtre sécurisée ayant un accès exclusif au système et ouvre trois environnements de travail séparés (« application », « écran de veille » et « environnement du Winlogon »), puis lance le processus LSA qui charge les packages d'authentification enregistrés.



2. WINLOGON ouvre ensuite une connexion LPC[1] vers la Local Security Authority (LSA) du système local. Cette connexion, établie par le biais des API LsaAuthenticationPort() et LsaRegisterLogonProcess(), est un canal sécurisé au travers duquel les informations d'authentification seront passées aux packages d'authentification (par défaut Kerberos.dll sous Windows 2000) à des fins de validation.



3. WINLOGON interroge alors le package d'authentification et obtient alors un numéro d'identifiant pour le package installé, numéro qui lui servira pour le passage des informations d'authentification.



4. Après avoir enregistré les informations d'environnement récupérées durant les précédentes phases, la **Security Attention Sequence** (**SAS** - par défaut la combinaison de touches CTRL-ALT-DEL) est activée et l'environnement de travail est verrouillé.

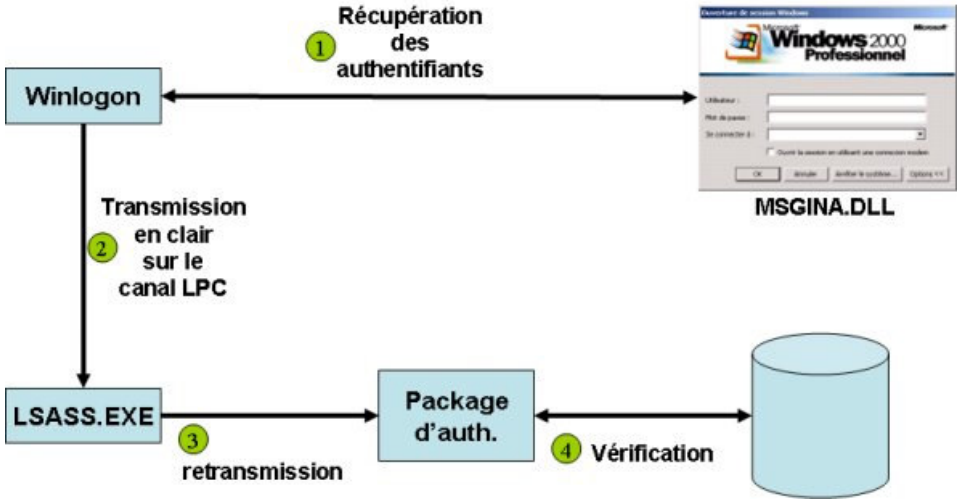
Une fois activée, un utilisateur peut dès lors utiliser la combinaison de touches du SAS.

Ce SAS génère une interruption qui est envoyée au système ; le système réagit alors en amorçant la séquence de Logon. Dès qu'un utilisateur active le SAS, plus aucun processus ne peut alors accéder à l'environnement de travail. A ce point de la procédure, l'interface graphique d'ouverture de session est lancée et permet alors à l'utilisateur de saisir les informations relatives à l'authentification (couple login/password et type de connexion - distante ou locale).

WINLOGON récupère alors ces informations et les transmet en clair, via la fonction *LSALogonUser(),* auprès de la LSA (LSASS.EXE) en même temps que le numéro du package d'authentification. La LSA envoie alors les données d'authentification en clair au package en question, pour vérification.

Le package d'authentification chiffre le mot de passe avec un algorithme différent selon le cas de figure (KERBEROS ou MSV1_0), puis interroge la base de donnée des comptes (Active Directory ou SAM).

[1] Local Procedure Call – Appel de Procédure Locale



Dans les deux cas, le package récupère et vérifie les données d'authentification avec le contenu de la base correspondante (locale ou distante) afin de déterminer si ces informations sont valides et si l'utilisateur a le droit d'ouvrir une session du type demandé. Le compte est scruté régulièrement par la suite afin de vérifier dans le temps que de nouvelles restrictions ne s'appliquent.

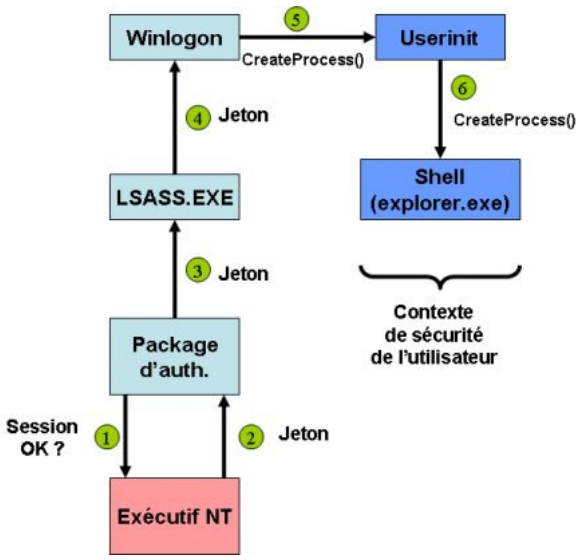
En cas de validation, le package d'authentification poursuit son action et collecte les informations concernant le compte utilisateur parmi lesquelles :

- L'identificateur de sécurité de l'utilisateur (SID)
- Les SIDs de chaque groupe auquel appartient l'utilisateur,
- Le SID du propriétaire par défaut,
- Le DACL (Discretionnary Access Control List) par défaut

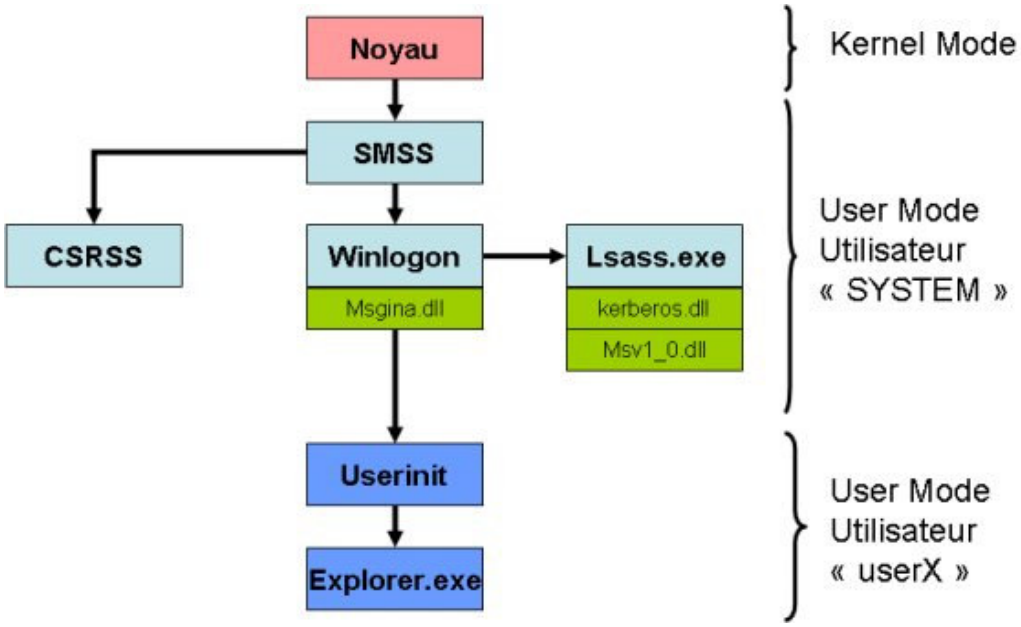
Une fois ces opérations effectuées, le package d'authentification passe ces informations à l'exécutif de Windows 2000 pour la création d'un jeton d'accès. Selon que la demande d'ouverture de session est locale ou distante, l'exécutif créera soit un jeton Primaire (ouverture de session locale) soit un jeton dit « d'impersonation » (ouverture de session distante). Ce jeton est alors renvoyé au package d'authentification puis au LSASS.EXE afin d'initier la session utilisateur.

Enfin, le processus WINLOGON crée une tâche (par défaut il s'agit de « userinit.exe »), qui lance un nouveau Shell utilisateur ; par défaut, le Shell utilisateur est le programme EXPLORER.EXE[1].

[1] L'explorateur Windows est surtout utilisé en tant qu'explorateur du système de fichier, mais sa tâche première consiste à construire le bureau utilisateur (barre des tâches, icônes, menu démarrer etc.). Pour s'en convaincre, il suffit de lancer un gestionnaire des tâches et de tuer tous les processus explorer.exe ; le bureau finit alors par disparaître !



Ce jeton ne sert pas seulement à valider les accès aux objets du système, il est également utilisé pour identifier les actions de l'utilisateur dans les journaux d'événement et pour accéder aux ressources réseau.



Résumé de l'arborescence des processus

Stockage des mots de passe dans la SAM

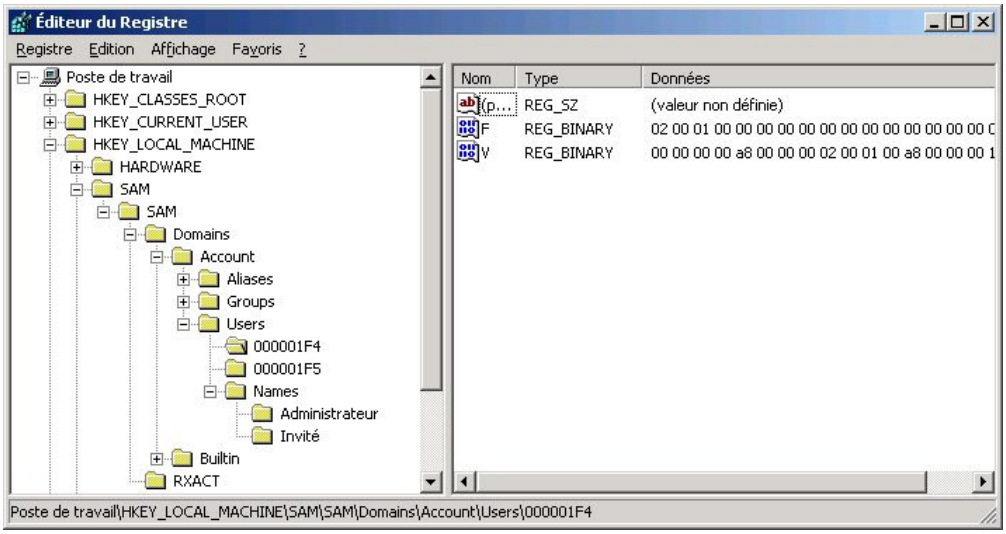
Le système d'exploitation Windows 2000 stocke les mots de passe des utilisateurs dans une base de données spécifique.

Pour stocker les mots de passe des utilisateurs locaux (sous Windows 2000 Pro, Windows XP et Windows NT 4.0 Workstation) c'est une base de donnée appelée SAM qui héberge ces mots de passe. Dans le cadre de l'utilisation d'un système Windows 2000 en domaine Windows 2000 les mots de passe des utilisateurs de domaines sont transférés dans l'annuaire Active Directory. **Quel que soit le cas de figure, les mots de passe ne sont jamais stockés en clair mais toujours sous une forme chiffrée**.

Contenu de la SAM

La base SAM contient tous les noms des utilisateurs référencés sur la machine, leur mot de passe sous sa forme chiffrée, les groupes auxquels ils appartiennent, ainsi que la description des groupes.

Cette base SAM est protégée par le mécanisme de contrôle d'accès fourni par le système d'exploitation. Elle est accessible, pour peu de disposer des droits suffisants, soit directement par la lecture des fichiers SAM (situés dans %Systemroot%\system32\config\SAM), soit par interrogation de la base des registres (la SAM est visualisable sous HKLM\SAM).



La structure des données utilisateurs en SAM dispose de deux champs disponibles pour stocker les mots de passe :

1. le premier de ces champs permet le stockage du mot de passe au format Lan Manager (pour d'historiques raisons de compatibilité Win95 et Win3.11, qui ne peuvent s'authentifier qu'avec ce mécanisme),

2. le second champ permet de stocker le mot de passe selon le format natif de Windows NT.

Dans les deux cas, le mot de passe n'est pas stocké en clair, mais sous sa forme chiffrée. La différence entre ces deux mécanismes est l'algorithme de chiffrement utilisé.

Structure Lan Manager (LM hash)

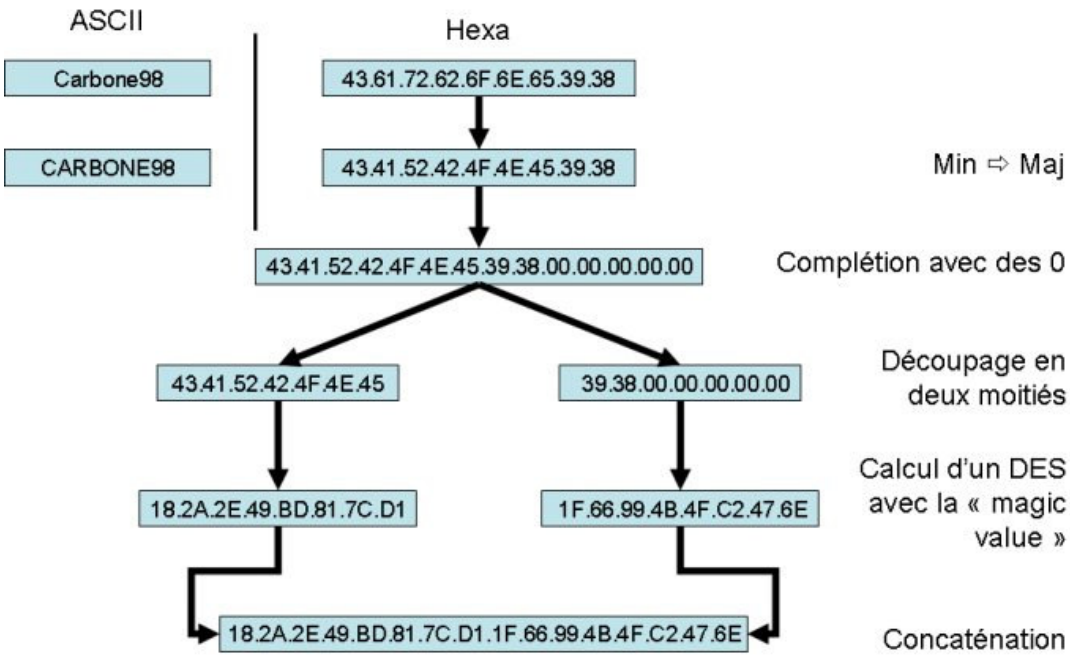
L'algorithme de chiffrement du « LANMAN password » est le suivant :

1. Passer le mot de passe en clair en majuscule.

2. Si le mot de passe en clair est plus grand que 14 caractères, alors le tronquer à 14 caractères, sinon le compléter avec des 0.

3. Former deux clefs de chiffrement K1 et K2 avec les deux moitiés (7 octets) du mot de passe résultant du point précédent.

4. Chiffrer la « magic value » 4B.47.53.21.40.23.24.25h avec K1 et K2 avec l'algorithme de chiffrement DES. Le résultat du chiffrement d'avec K1 forme les 8 octets de poids faible du « LANMAN password ». Le résultat du chiffrement avec K2 forme les 8 octets de poids fort.



Exemple reprenant la Figure ci-dessus :

Avec le mot de passe suivant « Carbone98» ;

1. Carbone98 ⇨ CARBONE98 (passage en majuscules)
2. CARBONE98 ⇨ `43.41.52.42.4F.4E.45.39.38.00.00.00.00.00` (complétion avec des zéros)
3. `43.41.52.42.4F.4E.45.39.38.00.00.00.00.00` ⇨ `43.41.52.42.4F.4E.45` + `39.38.00.00.00.00.00` (découpage en deux parties de 7 octets chacune)
4. `43.41.52.42.4F.4E.45` ⇨ `18.2A.2E.49.BD.81.7C.D1` et `39.38.00.00.00.00.00` ⇨ `1F.66.99.4B.4F.C2.47.6E` (chiffrement de la « magic value » avec K1 et K2)
5. Finalement, le LM hash sera : `18.2A.2E.49.BD.81.7C.D1.1F.66.99.4B.4F.C2.47.6E.`

Ainsi, découvrir un mot de passe stocké sous la forme LM Hash revient à découvrir 2 mots de passe d'une taille de 7 caractères maximum (insensibles majuscules / minuscules).

D'autre part, dans la mesure où il n'existe pas de mécanisme de « salage », deux mots de passe identiques auront le même cryptogramme LM Hash.

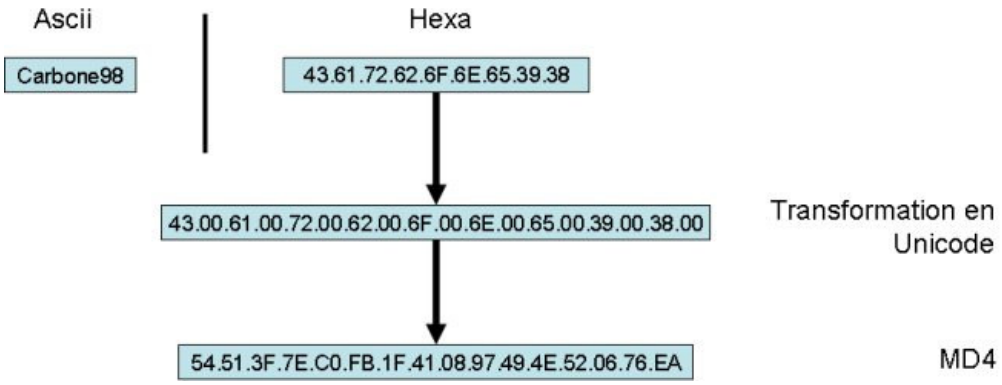
Il est également facile de déterminer si un mot de passe a une taille inférieure à 8 caractères. En effet, la seconde partie du LM Hash sera alors toujours `0xAAD3B435B51404EE` (résultat du chiffrement de `0x4B47532140232425` avec `0x00000000000000`).

Structure native Windows NT (NT hash)

L'algorithme de chiffrement du « NT password » est le suivant :

- Le mot de passe est convertit au format UNICODE.

- Un MD4 (algorithme de hachage permettant de calculer une empreinte d'un motif binaire) est appliqué sur le mot de passe résultant pour former les 16 octets du « NT password ».



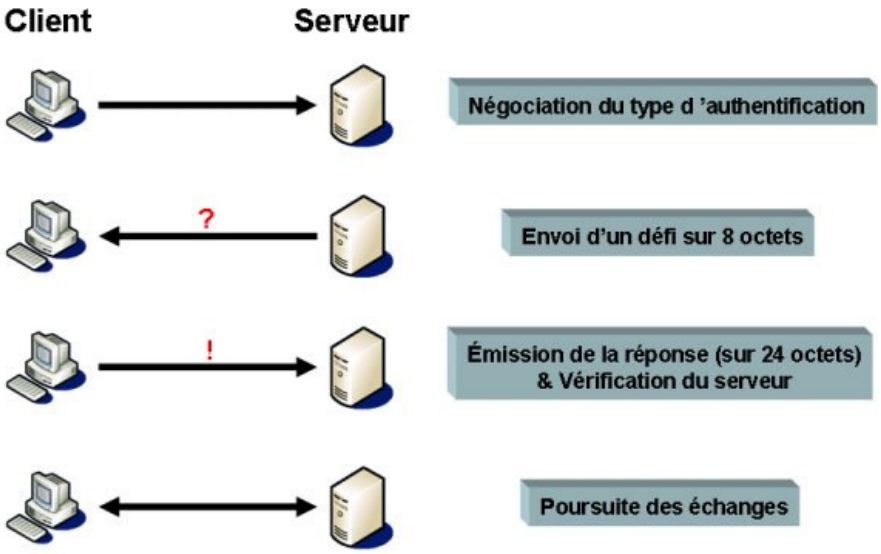
D'autre part, dans la mesure où il n'existe pas de mécanisme de « salage », deux mots de passe identiques auront le même cryptogramme NT Hash.

L'authentification réseau sous Windows NT 4.0

Principe du « défi-réponse »

Le principe d'authentification entre un client et un serveur Windows NT repose sur un mécanisme dit de « défi-réponse » dont la description suit :

- Le client initie une session sur le serveur,
- Le serveur et le client négocient les types d'authentification,
- Le serveur envoie un défi sur 8 octets,
- Le client calcule sa réponse et l'envoie au serveur,
- Le serveur calcule la réponse correcte (LanManager et NTLM) et compare son résultat avec celui du client,
- Si son résultat NTLM n'est pas correct, il vérifie son résultat LanManager avec celui fourni par le client (pour compatibilité Windows 3.11 et Windows 95/98)
- Le serveur valide l'authentification puis poursuit les échanges.



La réponse R au défi est caractérisée par le calcul d'une fonction f prenant en compte deux paramètres ; le défi proposé et le cryptogramme du mot de passe de l'utilisateur.

$$R = f(Défi, K_{mdp})$$

La mécanique de défi réponse sous Windows est implémentée de deux manières différentes :

- L'implémentation **Lan Manager**,
- Et l'implémentation **NTLM**

La première implémentation demeure pour des raisons de compatibilité ascendante : les clients Windows antérieurs à Windows NT (Windows 3.11, 95, 98) ne savent s'authentifier qu'avec ce schéma. La seconde implémentation a été développée spécifiquement pour Windows NT.

Par défaut, sous Windows NT 4.0, les deux implémentations sont utilisées conjointement : le client renvoie donc 2 réponses au défi proposé, chacune de ces réponses correspondant aux schémas décrits ici.

Implémentation LanManager

Le protocole d'authentification entre un client et un serveur sur un réseau Windows NT, en suivant le schéma d'authentification LanManager, est le suivant :

1. Supposons que l'utilisateur à authentifier dispose d'un LM Hash de valeur 0x182A2E49BD817CD11F66994B4FC2476E (ce qui est le LM Hash du mot de passe « Carbone98 »).

2. Le serveur envoie un défi de 8 octets à l'utilisateur.

3. L'utilisateur prend son LM Hash (0x182A2E49BD817CD11F66994B4FC2476E), et ajoute 5 octets de valeur nulle à celui-ci (le LM Hash devient alors 0x182A2E49BD817CD11F66994B4FC2476E0000000000)

4. La chaîne 0x182A2E49BD817CD11F66994B4FC2476E0000000000 est scindée alors en trois groupes de 7 octets : 182A2E49BD817C + D11F66994B4FC2 + 476E0000000000

5. Chacune de ces chaînes est ensuite modifiée pour former une clef DES de parité paire de 8 octets (on prend chaque bloc de 7 bits auquel on rajoute un bit de parité paire)

6. Nous avons donc maintenant 3 clefs DES de 8 octets (clef1, clef2 et clef3)

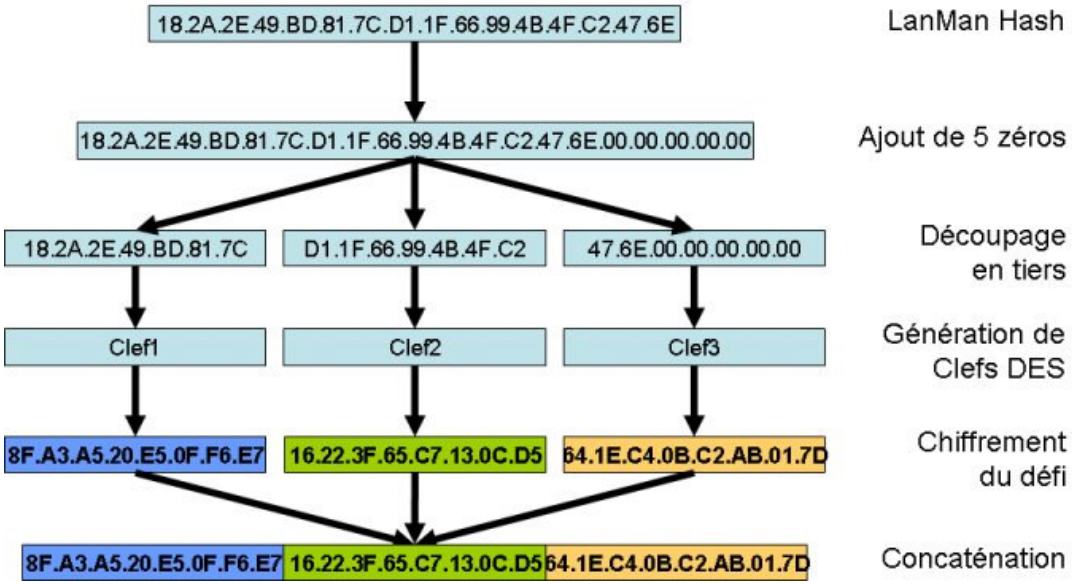
7. clef1 est utilisée pour chiffrer le défi. Le résultat est 0x8fa3a520e50ff6e7.

8. clef2 est utilisée pour chiffrer le défi. Le résultat est 0x16223f65c7130cd5.

9. clef3 est utilisée pour chiffrer le défi. Le résultat est 0x641ec40bc2ab017d.

10. Ces trois valeurs sont concaténées et envoyées au serveur sous la forme d'une réponse de 24 octets (0x8fa3a520e50ff6e716223f65c7130cd5641ec40bc2ab017d)

11. De son coté, le serveur a réalisé la même opération : il n'a plus qu'à comparer le résultat de son calcul avec celui renvoyé par l'utilisateur ; en cas de résultat identique, l'utilisateur est authentifié au sein de sa session.



Implémentation NTLM

Lors d'une authentification NT pure, le défi est chiffré avec comme clef le Hash Windows NT, puis cette réponse de 24 octets est émise auprès du serveur.

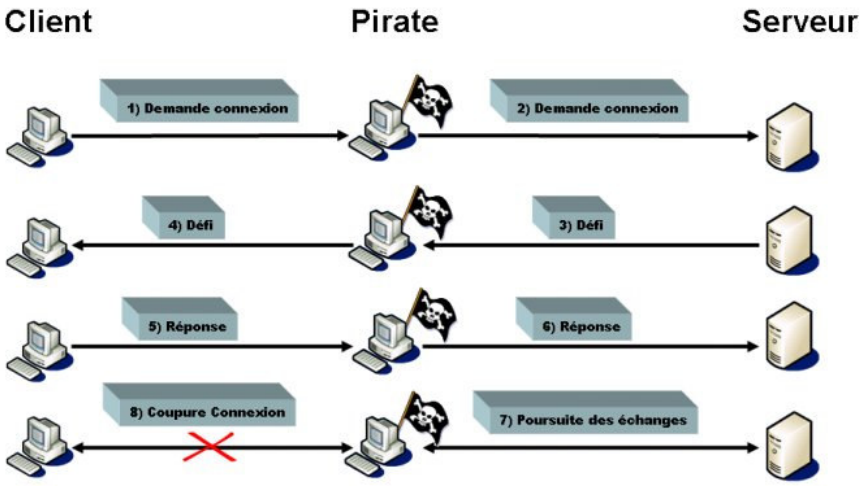
Vulnérabilités de l'authentification Windows NT

Ce mécanisme de « défi réponse » souffre cependant de quelques limitations et vulnérabilités, inhérentes à ce type de protocole. En particulier ce mécanisme d'authentification est vulnérable aux attaques de type « Man in the Middle » (également appelées attaques « monkey in the middle » ou « attaque du singe répéteur »).

Attaques « Man in the Middle »

Les attaques de type « Man in the Middle » sont communes à tous les protocoles qui n'implémentent pas de mécanisme de signature (elles fonctionnent donc pour le protocole SMB de base, mais aussi pour les protocoles Telnet, FTP, HTTP, etc.).

Le principe de l'attaque nécessite pour l'attaquant de se positionner en coupure entre un client et un serveur. Lorsque le client émet une requête auprès du serveur, la demande de connexion est interceptée par l'attaquant qui la réémet au serveur en changeant l'adresse source du client (il met la sienne à la place). Le serveur demande une authentification, qui est réémise au client par l'attaquant. Le client s'authentifie, cette authentification est alors retransmise au serveur par l'attaquant qui coupe alors la connexion entre lui et le client. Comme c'est l'attaquant qui s'est connecté au serveur (par un simple mécanisme de relayage de trames), c'est lui qui dispose désormais d'une session ouverte sur le serveur.



Attaque par utilisation brute du cryptogramme utilisateur

Une autre vulnérabilité importante du mécanisme repose sur la conception même de l'algorithme de calcul de la réponse. En effet, la réponse fournit prend en compte non pas le mot de passe de l'utilisateur, mais le cryptogramme de ce mot de passe.

En d'autres termes, **la seule connaissance du cryptogramme du mot de passe d'un utilisateur suffit à s'authentifier à distance sur une machine**, ce qui signifie qu'un attaquant peut tout à fait éviter d'avoir à casser durant de longues journées, voire de longs mois, les mots de passe d'une base SAM.

Il existe deux possibilités pour réaliser cet « exploit ».

- La première consiste à modifier un client Samba pour Unix pour que le calcul de la réponse à un défi s'effectue non pas à partir d'un mot de passe mais d'un fichier de cryptogrammes. Cette solution demeure contraignante car elle nécessite d'atteindre une machine Windows depuis une station Unix, ce qui implique la perte de certains outils natifs dans Windows NT.

- L'autre solution a été décrite dès avril 2000 par un certain **Herñan Ochoa**, dans un avis intitulé « **Modifying Windows NT Credentials** » et publié sur le site de SecurityFocus. Le principe de cette attaque consiste, sur un poste Windows NT local maîtrisé par l'attaquant, à **modifier directement, dans l'espace mémoire du processus LSASS.EXE, les cryptogrammes de l'utilisateur** en cours de session locale, et ce en utilisant une technique d'injection de DLL[1]. L'attaquant n'a alors plus qu'à tenter une connexion réseau vers sa cible, cette dernière ne lui demandera pas de mot de passe puisque les authentifiants en cours ont suffit à réaliser l'authentification.

Cette dernière technique offre de nombreux avantages dont celle de conserver un poste Windows disposant de tous les outils natifs du système Windows. Cependant, aucune exploitation de cette « faille » n'a jamais été rendue publique à ce jour…

Ce sont les nombreuses vulnérabilités des mécanismes d'authentification de Windows NT 4.0 qui ont amené les développeurs de Windows 2000 à intégrer dans ce système un autre mécanisme d'authentification beaucoup plus robuste : le système Kerberos.

[1] Voir en annexe de ce document pour la description de la technique d'injection de DLL.

Kerberos V5

« Rrrrrrr ! »

Alain Chabat / Les Robins des bois

Préambule

Le protocole Kerberos fut créé à l'origine au sein du MIT en 1983 par les ingénieurs travaillant sur le Projet Athena. Kerberos V5 est désormais un standard de l'IETF (Internet Engineering Task Force) dont les spécifications sont décrites par la RFC 1510. Selon Microsoft, l'implémentation de Kerberos V5 suit les recommandations de cette RFC, et les mécanismes et les formats utilisés pour le passage des jetons de sécurité suivent la RFC 1964.

Dans les faits, et selon la documentation Microsoft, la GSS-API définie dans la RFC 1964 n'est utilisée que dans le cas où l'un des deux cotés, client ou serveur, n'est pas un système Windows 2000. Dans le cas contraire, c'est la SSPI, mécanisme propriétaire de Microsoft semblable à la GSS-API, qui est utilisée. La SSPI n'est qu'une implémentation particulière de la RFC 1964 dans laquelle Microsoft utilise des champs inutilisés du protocole pour passer des informations spécifiques à Windows 2000 ; de fait, les formats utilisés par la SSPI n'étant pas documentés (la SSPI est présentée comme une boite noire, regroupant des API communes), il apparaît délicat de faire fonctionner ensemble un système Windows 2000 avec un autre système compatible Kerberos V5, tout en bénéficiant de l'ensemble des fonctionnalités offertes par l'utilisation de Kerberos dans Windows 2000.

Pourtant, on trouve sur l'Internet l'ensemble des méthodes nécessaires :

1. pour faire tourner une machine Unix en tant que Serveur Kerberos pour une architecture Windows 2000,

2. pour utiliser un serveur Windows 2000 en tant que serveur Kerberos dans une architecture Kerberos classique

Cette implémentation particulière de la RFC 1964 a d'ailleurs provoqué l'ire de la communauté Internet et en particulier de l'IETF qui, bien évidemment, n'a pas été consulté.

Dans son principe, Kerberos est un mécanisme d'authentification mutuel entre clients et serveurs ; un client réalise une authentification sur un serveur Kerberos afin d'obtenir un jeton d'accès pour une ressource tierce. Ce jeton d'accès, à validité limitée dans le temps, sert alors de moyen d'authentification pour accéder à la ressource considérée.

Dans les années qui ont précédé l'arrivée de Windows 2000, l'authentification dans le monde Windows NT était assurée par le mécanisme d'authentification NTLM, décrit dans

le chapitre précédent. Ce mécanisme a montré ses limites en termes de performances et d'efficacité et c'est donc vers Kerberos V5 que s'est tourné Microsoft pour le mécanisme d'authentification par défaut de Windows 2000. Cependant, Windows 2000 continue à assurer une compatibilité ascendante avec Windows NT, puisqu'il est capable de réaliser des authentifications NTLM dans le cas de communications avec des machines Windows NT.

Pour ce qui concerne l'ouverture de session sous Windows 2000, le comportement par défaut lorsque l'on tente de s'authentifier depuis une machine Windows 2000 Pro sur un serveur Windows 2000 est d'utiliser le protocole d'authentification Kerberos.

Principes et Terminologie

Le système Kerberos est principalement un serveur d'authentification externe, dont le protocole est fondé sur le modèle de Needham et Schroeder publié en 1978 (« Using Encryption for Authentication in Large Networks of Computers »).

L'architecture de Kerberos constitue une architecture tripartite :

- Le client
- Le serveur de ressources
- Une (ou plusieurs) autorité(s) approuvée(s).

L'autorité approuvée (AA) est un serveur dit « de confiance », et reconnu comme tel à la fois par le client et le serveur. On présuppose par ailleurs que l'autorité approuvée ne constitue pas le maillon faible du système, c'est-à-dire qu'il n'est vulnérable à aucune attaque connue.

Avant de poursuivre plus avant dans la description du schéma d'authentification, il est nécessaire d'introduire quelques notions de terminologie.

- Un « **principal** » Kerberos désigne un client du protocole, identifiable par un nom unique. Un client ou un serveur constitue un principal Kerberos
- Un « **Key Distribution Center »** (**KDC**) est une autorité approuvée qui stocke les informations de sécurité relatives aux principaux. En outre, il génère et gère les clefs de session.
- Un « **royaume »** (ou « **realm** ») Kerberos est une organisation logique dans laquelle il existe au moins une autorité approuvée et qui est capable d'authentifier les principaux déclarés sur ce serveur.
- Un « **ticket »** est une structure de données constituée d'une partie chiffrée et d'une partie claire. Les tickets servent à authentifier les requêtes des principaux. Il existe par ailleurs deux types de ticket :
    - Les tickets **TGT** (Ticket Granting Ticket)
    - Les tickets **ST** (Service Ticket)

Un système Kerberos assure deux types de service, par ailleurs non nécessairement hébergés sur la même machine ; un service d'authentification (**AS** ou « Authentication Service ») et un service d'octroi de tickets (**TGS** ou « Ticket Granting Service »).

Dans Kerberos, une AA (ie un KDC) génère et stocke les clefs secrètes (K_{sec}) des principaux qui lui sont rattachés. Sous Windows 2000, K_{sec} est directement dérivée du mot de passe de l'utilisateur.

Pour des raisons de sécurité, ces clefs secrètes ne servent que lors de la phase initiale d'authentification : dans toutes les autres phases, on utilise des clefs de session « jetables ».

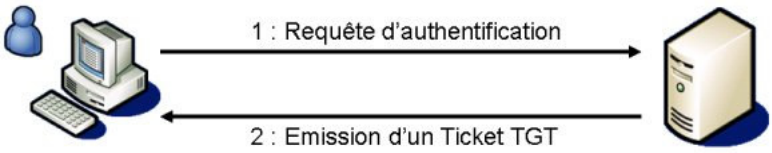
Le système Kerberos V5, tel que défini dans la RFC 1510, n'utilise pas d'algorithmes à clefs asymétriques ; ce sont des clefs partagées qui sont utilisées pour l'authentification.

Cependant, Microsoft a apporté une extension au système Kerberos permettant l'utilisation de la cryptographie asymétrique. Cette extension n'est utilisée dans Windows 2000 que lors d'une authentification locale mettant en jeu une carte à puce.

Détails du protocole

Dans un premier temps, le client désirant accéder à une ressource réalise une première phase visant à s'authentifier auprès du service AS d'un KDC.

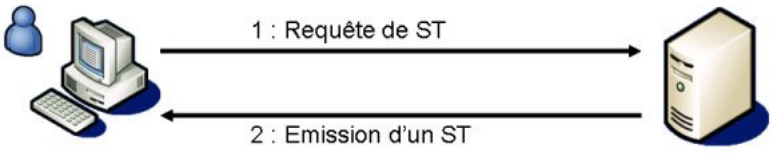
Ce premier échange va permettre au client de récupérer un TGT auprès du service AS. La requête initiale contient alors, en clair, l'identité du requérant et le serveur pour lequel on demande un ticket. La partie chiffrée du TGT l'est avec la clef secrète du client ce qui implique que seul le bon utilisateur pourra déchiffrer ce TGT et donc s'en servir correctement.



Requête d'authentification sur un service d'AS

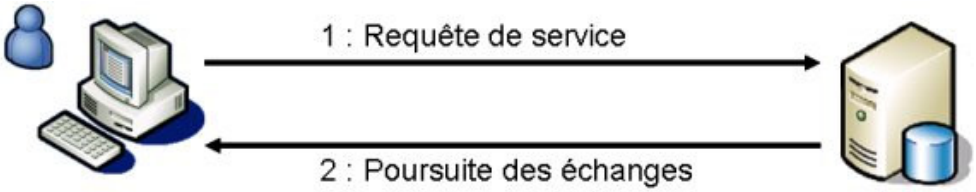
Précisons que, **l'authentification mutuelle n'est pas disponible lors de ce premier échange** client/AS, c'est-à-dire que le client n'est pas en mesure d'identifier avec certitude le serveur d'authentification. En effet, celui-ci ne renvoie au client que de l'information sous la forme de clés et de tickets, et lorsque le client déchiffre le message, il n'a aucun moyen de vérifier si les données en clair sont cohérentes.

Ce TGT ne servira par la suite que pour récupérer un ST auprès du service TGS, au terme d'un second échange client/TGS.



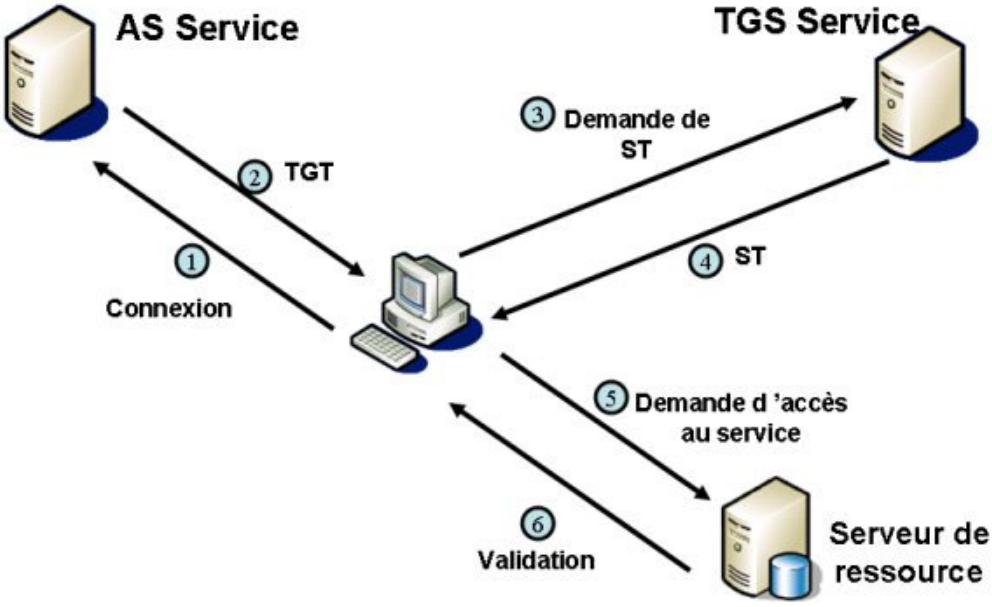
Requête de Service Ticket sur un service TGS

Le ST ainsi obtenu est alors présenté au serveur de ressource qui valide ou non la requête.



Requête d'accès à une ressource

Au final, la chronologie des échanges nécessaires pour atteindre un service donné est représentée sur la figure suivante :



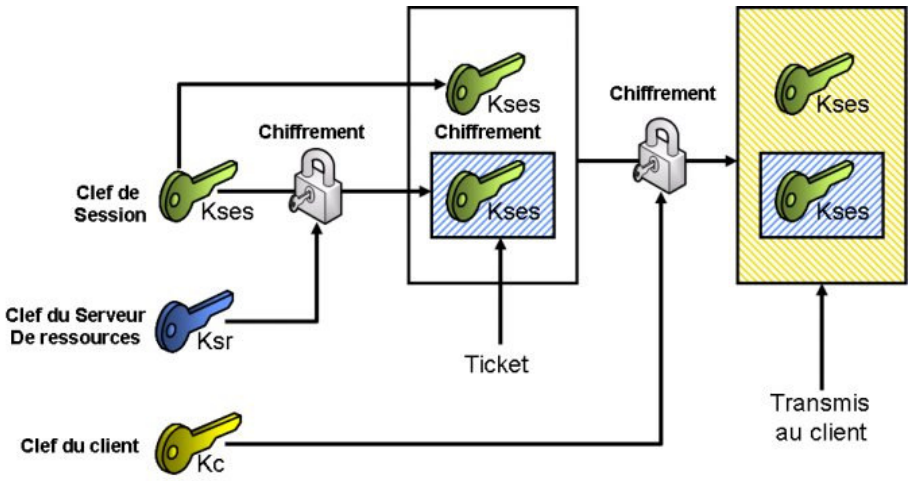
Génération et traitement des tickets

L'accès à une ressource est ainsi réalisé en trois passes distinctes :

1. Génération du ticket ST par le serveur et transmission au client,
2. Traitement du ticket ST par le client et préparation de la requête au serveur,
3. Traitement de la requête par le serveur et poursuite des échanges.

Suite à la requête initiale du client, le serveur lui renvoie une structure de données chiffrée avec sa clef secrète et contenant :

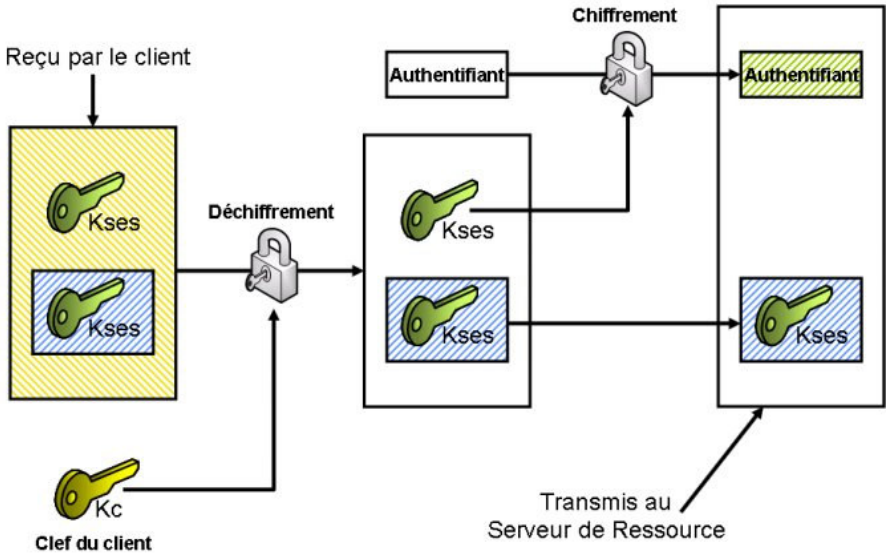
- Une clef de session en clair
- La même clef de session, chiffrée avec la clef secrète du serveur de ressources
- Un horodatage



Génération d'un ST

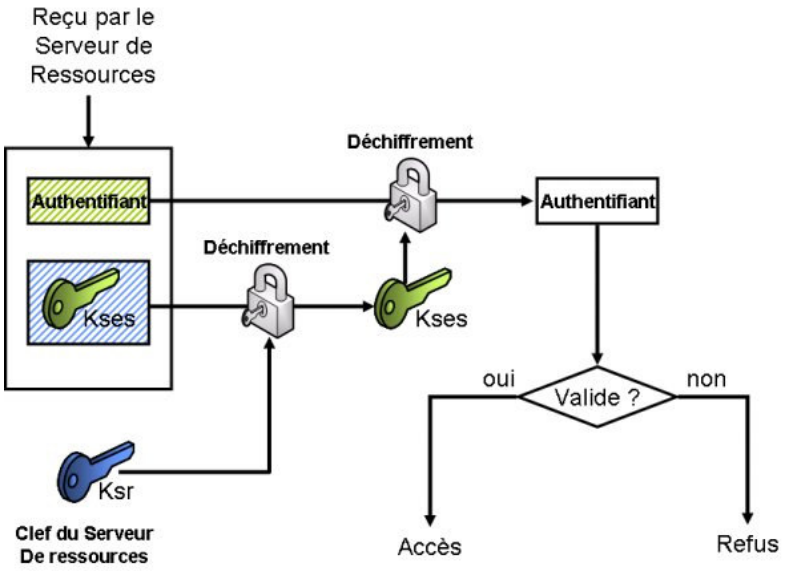
Cette structure de données est déchiffrée par le client, qui se sert alors de son contenu pour préparer une requête à destination du serveur de ressource. Cette requête est composée :

- de la clef de session chiffrée avec la clef secrète du serveur de ressource (tel que nous l'a transmis le KDC)
- d'un authentifiant, chiffré avec la clef de session.



Traitement du ticket ST par le client

Lorsque le serveur de ressource réceptionne cette requête, il déchiffre la clef de session avec sa clef secrète, puis utilise cette clef de session pour déchiffrer l'authentifiant.



Traitement de la requête d'accès par le serveur de ressources

L'authentifiant fourni par le client contient une structure de données dont la cohérence est vérifiée après déchiffrement par le serveur de ressource : si cet authentifiant est correctement déchiffré le serveur de ressource valide alors la requête utilisateur.

Structures de données utilisées

A titre d'information, le contenu des structures de données des authentifiants et des tickets est donné ci après.

Structure d'un ticket Kerberos générique (TGT et ST)

Champ	Description	
tkt-vno	Version (5)	Clair
realm	Royaume d'origine du ticket	
sname	Nom de l'AA ayant délivré le ticket	
flags	Drapeaux d'états du ticket	Chiffré
key	Clef de session pour l'échange futur	
crealm	Royaume d'origine du client	
cname	Nom du client	
transited	Liste des royaumes ayant pris part dans le schéma d'authentification	
authtime	Horodatage de l'authentification	
starttime	Indique à partir de quand le ticket est valide	
endtime	Indique l'expiration du ticket	
renew-till	Pour ticket renouvelables ; indique jusqu'à quand le ticket peut être renouvelé	
caddr	Contient 0 ou une liste d'adresses depuis lesquelles le ticket est utilisable	
authorization-data	Champ utilisé par les applications pour passer des données spécifiques	

Structure d'un authentifiant Kerberos

Champ	Description
authenticator-vno	Version (5)
crealm	Royaume d'origine du client
cname	Nom du client
chksum	Somme de contrôle d'intégrité (optionnel)
cusec	Contient la partie en microsecondes de l'horodatage client
ctime	Horodatage client
subkey	Peut préciser une clef de session pour protéger l'échange (optionnel. Par défaut, contient la clef de session fournie par l'AA)
seq-number	Numéro de séquence (optionnel)
authorization-data	Champ utilisé par les applications pour passer des données spécifiques

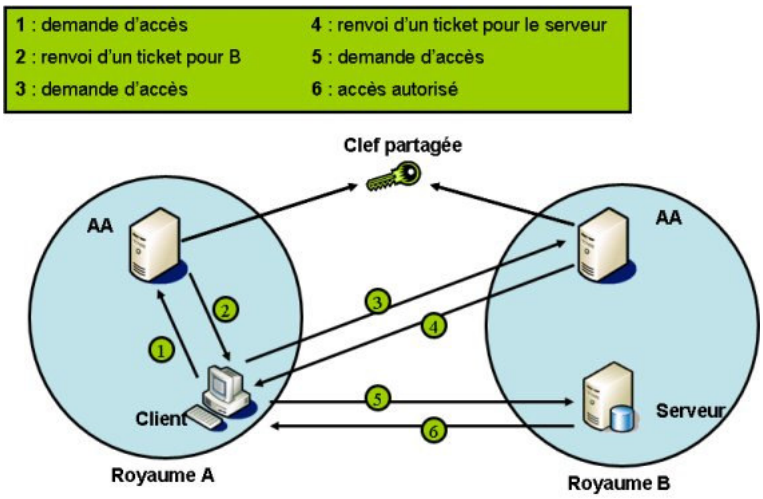
Authentification entre royaumes

On a vu qu'un royaume était une limite de sécurité, au sein de laquelle il était possible d'authentifier les utilisateurs qui y étaient déclarés. Kerberos dispose en outre d'un mécanisme permettant à un utilisateur d'accéder à des ressources n'appartenant pas à son propre royaume.

Le principe de base d'un tel mécanisme consiste à faire partager entre deux royaumes une même clef secrète.

Quand un utilisateur d'un royaume A souhaite atteindre un serveur d'un royaume B :

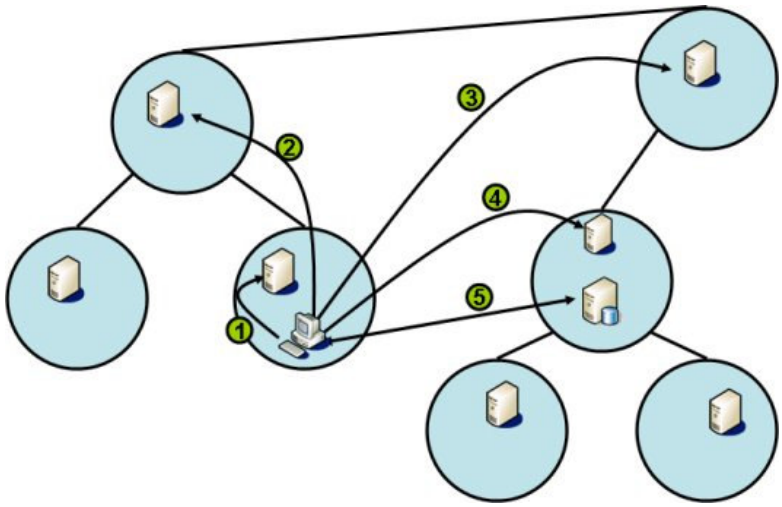
- il contacte son propre KDC,
- qui lui transmet un « Refferal Ticket » (TGT chiffré avec une clef partagée inter royaume)
- qui servira à obtenir auprès de l'AA de B un ST pour le serveur souhaité.



Ce type de mécanisme fonctionne relativement bien lorsque l'on doit traiter un nombre peu important de royaumes. Mais sachant que la distribution des clefs partagées suit une

difficulté croissante[1], on a alors recours à l'astuce suivante : **on définit une structure hiérarchique des royaumes entre eux, autorisant ainsi l'accès aux ressources par des rebonds successifs.**

Dans le schéma qui suit, chaque lien entre royaumes indique le partage d'une clef inter royaume qui servira a émettre des « referral tickets » auprès du client. Pour atteindre la ressource considérée, on a ici besoin de 4 tickets successifs pour obtenir l'accès demandé :



Accès à une ressource d'un autre royaume

Les avantages d'une telle architecture hiérarchique sont multiples :

- Elle préserve l'isolement des royaumes entre eux,
- Tout client d'un royaume peut accéder aux ressources de n'importe quel serveur (si ce dernier l'autorise)…
- …même si ce serveur ne fait pas partie du royaume du client,
- Les relations entre royaumes sont transitives et bidirectionnelles ce qui évite, lors de l'ajout d'un nouveau royaume, de redistribuer N clefs.

L'Emprunt d'identité

Le système Kerberos V5 a été conçu pour autoriser le fonctionnement d'emprunt d'identité parfois requis pour les applications n-tiers.

Dans une application n-tiers, le client accède généralement à un « portail » applicatif. Il ne voit que ce portail qui relaie ses demandes auprès des serveurs de ressources adéquats. Dans cette architecture, le portail agit directement en lieu et place du client.

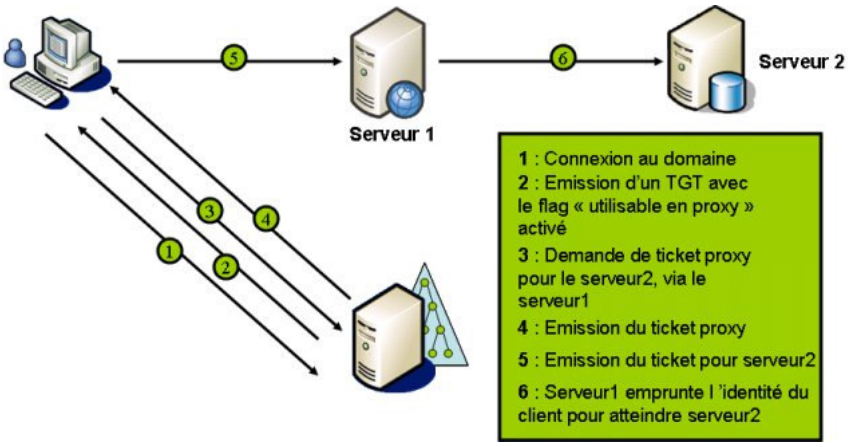
Deux méthodes peuvent être utilisées dans Kerberos pour l'exploitation de ce mécanisme :

- Le mode « proxy »
- Le mode « transfert »

[1] Si on a n royaumes, le nombre Nb de clefs partagées à distribuer est donné par la formule Nb = n(n-1)

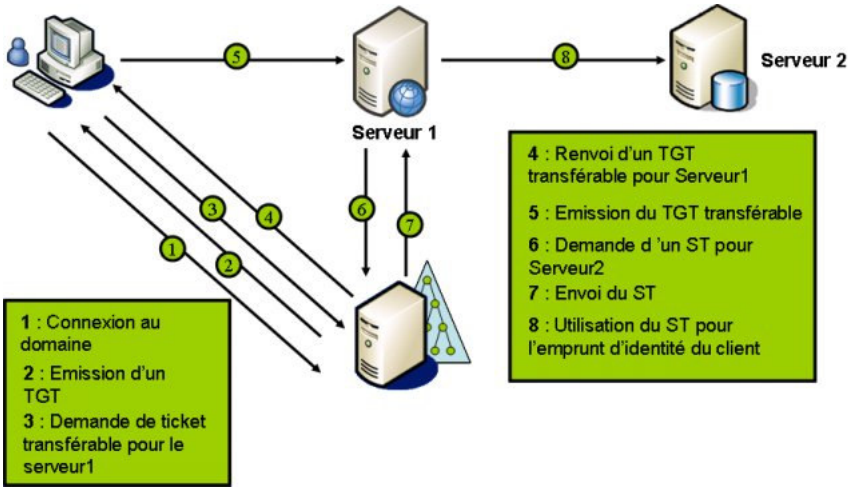
Mode Proxy

En mode proxy, le client récupère un ST depuis son KDC et le transmet au serveur portail. Ce serveur utilise alors ce ST pour se connecter à la ressource comme s'il était lui-même le client.



Mode Transfert

En mode transfert, le client obtient un TGT qu'il peut transférer à un serveur ; le serveur agit alors comme le client, en effectuant une demande de ST au KDC, comme s'il était le client.



Le **mode proxy** constitue le mode de fonctionnement **le plus sécurisé** puisque le ST transmis au serveur n'est valable que pour le serveur de ressource considéré. En **mode transfert** le TGT transmis au serveur portail est **une véritable délégation de pouvoir** qui nécessite donc une confiance absolue dans ce serveur (il pourrait utiliser ce TGT pour accéder à d'autres ressources en lieu et place du client !).

Précisons que Windows 2000 ne gère que le mode transfert, le mode proxy n'étant disponible qu'avec Windows 2003.

Limitations

Bien que fondé sur des bases solides, Kerberos possède un certain nombre de points faibles :

- Il ne supporte que les mécanismes de chiffrement symétriques, qui nécessitent un partage et une mise à jour des clefs entre les différents serveurs d'administration et les clients. De plus, en cas de piratage des clefs, tous les clients peuvent être usurpés

- Si un pirate parvient à déterminer une clef d'un client, même ancienne, et a réussi à obtenir une capture de l'ensemble des messages de changement de mots de passe, il pourra en déduire la clef en cours, puisque les messages de changement de clef utilisent l'ancienne clef à chaque fois.

- Kerberos ne prend pas en compte les aspects d'autorisation : c'est à chaque système de s'adapter à Kerberos pour traiter la problématique de l'accès aux ressources.

- L'utilisation des horodatages permet d'éviter le rejeu sauf si les horloges locales sont trop désynchronisées, ou si le service d'horloge est piraté. Dans ce cas, il y a un risque de rejeu ou de refus de service de la part du serveur. Kerberos nécessite donc un service de temps fiable.

- Il n'existe pas d'authentification mutuelle lors du protocole d'authentification initial. Le ticket délivré par le serveur est chiffré avec le K_{sec} de l'utilisateur. Le serveur est supposé comme authentique si K_{sec} est correct. Or, si K_{sec} est incorrect, le client déchiffrera le ticket de façon incorrecte et n'aura pas moyen de s'en apercevoir. C'est uniquement lors d'une requête auprès d'un serveur de ressources et lorsque ce dernier lui refusera l'accès (les informations contenues dans les tickets n'ont alors aucune chance d'être cohérentes) qu'il pourra soupçonner que le serveur d'authentification est un leurre.

Spécificités de Kerberos sous Windows 2000

Le RFC 1510 précise que Kerberos utilise **toujours** le port UDP 88. Dans son implémentation, Microsoft utilise le port UDP 88 uniquement pour les messages Kerberos de taille (MTU) inférieure à 1472 octets.

De 1473 à 2000 octets, Microsoft utilise la fragmentation UDP sur le port 88. Au delà de 2000 octets, Microsoft utilise le port TCP 88. En outre, les messages Kerberos de Microsoft contenant des informations « credential » (SIDs d'utilisateurs et de groupes), ils dépassent majoritairement la taille de 2000 octets et utilisent donc le port TCP 88.

Microsoft a proposé une révision de la RFC 1510 sur ce sujet. Cette modification ne concerne a priori que les échanges entre systèmes Windows 2000. Mais l'interopérabilité avec d'autres Kerberos doit alors être testée.

En outre, Microsoft a proposé un certain nombre d'évolution à l'IETF parmi lesquelles :

- Kerberos change password protocol
- Kerberos set password protocol
- RC4-HMAC Kerberos Encryption Type
- PKINIT (support des clés asymétriques : utilisé dans Windows 2000 pour le support de l'authentification par cartes à puces)

La Base de Registres

« Pour moi, une seule chose est claire. Avant d'effectuer toute modification des règles, il faut définir des règles de modification des règles. »

TM in Guide du Cabaliste Usenet

Pourquoi une base de registres ?

La base des registres fournit, sous Windows 9x et NT/2000/2003, une base de données unique dans laquelle sont stockées toutes les informations de configuration, au sein d'une structure hiérarchique. Avant Windows 95 et dans le monde Windows, la seule façon de configurer des applications et les fonctions du système d'exploitation était d'utiliser des fichiers de configuration (les fameux .INI dans le meilleur des cas, d'autres fichiers plus exotiques dans d'autres cas).

Dans les environnements Windows 9x et NT, la base des registres remplace définitivement ce fonctionnement à base de fichiers de configuration, du moins pour les applications 32 bits. Chaque clef dans le registre est similaire, dans son fonctionnement, aux en-têtes entre crochetés des fichiers .INI.

Les problèmes liés à l'utilisation de ces fichiers .INI étaient nombreux. Entre autre, il était impossible d'utiliser ces fichiers pour y stocker autre chose que du texte pur, ce qui peut être gênant quand on doit stocker des valeurs de type hexadécimales ou binaires (on doit alors avoir recours à une forme de codage de ces valeurs), de plus la structure de ces fichiers interdisait la possibilité de hiérarchiser leur contenu.

Une clef de registre peut donc contenir un ensemble de types de valeurs autres que du texte pur, et il est possible d'utiliser des sous-clefs (donc de hiérarchiser le contenu de la base).

Pour visualiser le contenu de la base des registres on peut utiliser les deux utilitaires fournis avec le système d'exploitation ; regedit.exe et regedt32.exe. Seul le second programme, d'ailleurs absent dans Windows 9x, permet de positionner des ACLs sur des clefs de registre.

Structure de la base des registres

Sous Windows 2000, la base des registres est divisée en 5 structures distinctes :

HKEY_CURRENT_USER

Cette clef contient les informations de configuration liées à l'utilisateur en cours de session. Les préférences d'écran, l'état du bureau, les volumes montés etc. sont stockés

dans cette clef. Dans les faits, cette clef n'est qu'un alias, initialisé à l'ouverture de session, vers une ruche de la clef HKEY_USERS.

HKEY_USERS

Cette clef contient les informations de configuration liées aux utilisateurs du système. Les ruches de cette clef correspondent aux profils utilisateurs que l'on trouve dans le répertoire %systemroot%\profiles (plus précisément, le fichier Ntuser.dat (ou Ntuser.man) situé dans chaque profil utilisateur).

HKEY_LOCAL_MACHINE

Cette clef contient les informations de configuration spécifique à la machine, indépendamment de l'utilisateur en cours de session. Ces informations sont stockées dans le répertoire %systemroot%\system32\config comme une série de fichiers systèmes, à l'exception de la clef volatile « hardware ».

De par les informations qui s'y trouvent, cette clef est probablement la plus importante et la plus sensible de toute la base des registres. Elle contient cinq sous-clefs :

Hardware : base de données décrivant le matériel présent sur l'ordinateur, la manière dont les pilotes de périphérique doivent utiliser le matériel etc. une grande partie des données situées dans cette sous-clef est régénérée à chaque démarrage par le système d'exploitation.

SAM : Security Account Manager, contient toutes les informations relatives aux utilisateurs et aux groupes déclarés sur la machine.

Security : base de données contenant la politique de sécurité locale à la machine, comme les droits spécifiques des utilisateurs. Le contenu de cette clef n'est utilisé que par le sous-système de sécurité du système d'exploitation.

Software: Cette clef contient des informations sur les logiciels installés ainsi que des informations de configuration du système d'exploitation.

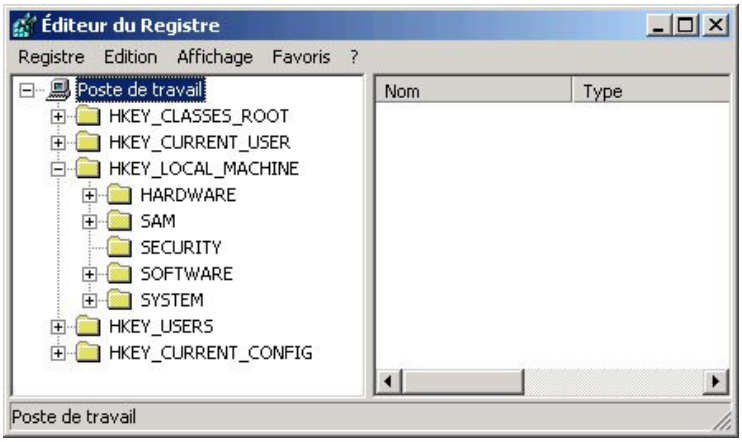
System: Base de données contrôlant le démarrage de Windows NT, le chargement des pilotes de périphérique et le comportement du système d'exploitation.

HKEY_CLASSES_ROOT

Les informations stockées ici sont utilisées par le système d'exploitation pour savoir quelle application lancer quand un fichier est ouvert par lien OLE (double click sur un fichier par exemple), et pour stocker les informations concernant les liens et les encapsulations entre objets. C'est en fait un alias vers une sous-clef de HKEY_LOCAL_MACHINE\Software

HKEY_CURRENT_CONFIG

Les informations stockées dans cette clef sont utilisées pour déterminer quels logiciels et pilotes de périphérique doivent être chargés, ou pour la résolution d'écran à utiliser. Cette clef contient des sous-clefs System et Software qui conservent des traces des précédentes configurations.



Chaque branche peut contenir un certain nombre de clés et de sous clés. Chaque clé/sous-clé contient un certain nombre de valeurs constituées de trois éléments :

- Un **nom**, exemple Wallpaper
- Un **type**, exemple REG_SZ (chaîne de caractère)
- Une **valeur courante** ou **contenu** exemple "c:\winnt\lanma256.bmp"

Types

On a vu que les valeurs dans la base de registres peuvent être typés. Le tableau suivant indique les différents types existant sous Windows 2000.

Nom	Description
REG_NONE	Valeur non typée.
REG_SZ	Chaîne de taille fixe codée en UNICODE.
REG_EXPAND_SZ	Chaîne de taille variable, codée en UNICODE et pouvant contenir des variables d'environnement.
REG_BINARY	Type binaire
REG_DWORD	Nombre sur 32 bits
REG_DWORD_LITTLE_ENDIAN	Nombre sur 32 bits, l'octet de poids faible étant en premier. Equivalent à REG_DWORD.
REG_DWORD_BIG_ENDIAN	Nombre sur 32 bits, l'octet de poids fort étant en premier.
REG_LINK	Lien symbolique, codé en UNICODE
REG_MULTI_SZ	Tableau de chaînes en UNICODE.
REG_RESOURCE_LIST	Hardware resource description
REG_FULL_RESOURCE_DESCRIPTOR	Hardware resource description
REG_RESOURCE_REQUIREMENTS_LIST	Resource requirements

Fichiers de la base de registre

Les fichiers contenant les informations de la base de registre sont stockés dans le répertoire %SYSTEMROOT%/SYSTEM32/CONFIG. Ce sont :

- **SAM** pour la branche HKEY_LOCAL_MACHINE\SAM
- **SECURITY** pour la branche HKEY_LOCAL_MACHINE\Security
- **SOFTWARE** pour la branche HKEY_LOCAL_MACHINE\Software

- **SYSTEM** pour les branches HKEY_LOCAL_MACHINE\System & HKEY_CURRENT_CONFIG
- **DEFAULT** pour la branche HKEY_USERS\.DEFAULT
- **Ntuser.dat** pour la branche HKEY_CURRENT_USER (ce fichier est stocké dans le répertoire %SYSTEMROOT%\PROFILES\%USERNAME%).

Ces fichiers sont aussi appelés RUCHES et sont accompagnés de compagnons d'extensions différentes :

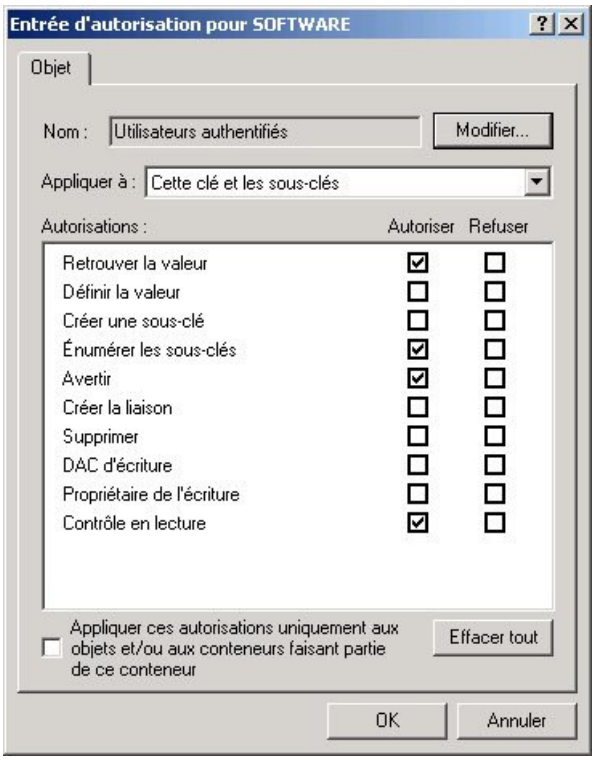
- **system.alt** qui contient une copie de sauvegarde de HKEY_LOCAL_MACHINE\System.
- **.log** - le journal des modifications apportées sur le fichier initial.
- **.sav** - une copie du fichier initial après la configuration en mode texte.

Sécurité de la base des registres

Compte tenu du rôle capital que joue la base des registres au sein de la configuration du système d'exploitation, il est possible de positionner des ACLs (Access Control List, ou permissions) sur tout ou partie de la base des registres.

Ainsi, la ruche HKEY_LOCAL_MACHINE\SAM est elle protégée contre une lecture par les utilisateurs et pour cause : cette structure ne contient rien de moins que les cryptogrammes représentant les mots de passe chiffrés des utilisateurs. On verra que même les Administrateurs n'ont pas accès à cette structure, bien qu'ils aient la possibilité de la déprotéger.

D'autres clefs, toutes aussi sensibles que la SAM, sont également protégées par défaut au cours de l'installation d'un système Windows 2000. Notons que cette protection par ACLs ne peut être réalisée efficacement qu'à la condition sine qua non d'utiliser le système de fichier NTFS sur le disque accueillant le système d'exploitation. Le positionnement d'ACLs sur des clefs de la base des registres peut se faire en utilisant l'utilitaire regedt32.exe.



Les ACLs que l'on peut positionner sur les objets de la base des registres sont les suivantes :

Requête sur une valeur Autorisation de lire une entrée à partir d'une clef de registre.

Définir la valeur Autorisation de définir les entrées dans une clef de registre.

Créer une sous-clef Autorisation de créer une sous-clef sur une clef de registre.

Enumérer les sous-clefs Autorisation de lister les sous-clefs d'une clef de registre.

Avertir Autorisation d'auditer les événements de notification.

Créer la liaison Autorisation de créer une liaison symbolique dans une clef (alias).

Supprimer Autorisation de supprimer la clef sélectionnée.

Accès en écriture à la liste de contrôle d'accès

Autorisation de positionner des ACLs sur une clef.

Accès en écriture du propriétaire

Autorisation d'affecter un propriétaire à une clef.

Contrôle en lecture Autorisation de lister les informations de sécurité (ACLs) d'une clef.

Le Système de Fichiers NTFS

« - J'ai lu quelque part qu'il y avait une manière d'avoir des ACL avec Ext2fs. Ca se passe comment ?
- Ca se passe bien. »

Le guide du linuxien pervers

Le système de gestion de fichiers natif de Windows NT est NTFS (New Technology File System), largement inspiré des développements précédents pour OS/2 (HPFS - High Performance File System). Ce système de fichier permet de gérer des listes de contrôles d'accès afin de déterminer qui a accès aux objets du système de fichiers.

Fonctionnement Interne

Avantages

Par rapport aux systèmes FAT issus des technologies DOS, Windows 3.x et Windows 9x, les volumes NTFS disposent en plus des fonctionnalités suivantes :

- Capacité à gérer des partitions de plus de 2Go
- Support des noms longs et des caractères Majuscules/Minuscules
- Positionnement de listes à contrôles d'accès sur les objets gérés par le système de gestion de fichiers,
- Audits des différents accès aux objets,
- Procédé d'accès aux objets gérés par des mécanismes transactionnels et permettant d'assurer des fonctions d'intégrité, et de tolérances aux pannes de niveau élémentaire (mécanismes de journaux-avant / journaux-après)
- Possibilité de gérer du mirorring entre disques, et des agrégats de partitions par bandes.
- Procédé de défragmentation "à la volée".

Tolérance aux pannes

NTFS utilise un mécanisme transactionnel pour les accès aux objets dont il a la responsabilité. Lorsqu'une donnée est accédée, le sous-système de disques vérifie les informations relatives aux fichiers accédés dans une table appelée Master File Table (MFT).

Cette table, qui est un fichier caché du système, nommé $MFT et situé à la racine du disque, contient toutes les informations relatives aux données enregistrées sur le disque, dont :

- des informations utilisées pour monter le système de fichier,
- la liste des mauvais clusters,
- les attributs de chaque fichier.

Lors d'une requête d'accès à un fichier, la MFT est interrogée pour vérifier l'existence du fichier et, en cas de succès, les attributs de sécurité (SIDs) sont chargés et comparés au jeton d'accès du processus appelant.

Lorsque le fichier est ouvert, une entrée est créée par le Log File Service (LFS) et stockée sur le disque dans le fichier caché $LogFile. Ce fichier est utilisé par le LFS pour enregistrer les modifications réalisées sur le fichier. Ce qui est enregistré dans ce fichier ne concerne cependant que les modifications sur le système de fichiers (emplacement sur le disque des blocs modifiés par exemple) et non pas les modifications sur le fichier. Dans la mesure où c'est un mécanisme transactionnel qui est utilisé pour l'écriture sur le disque, les données sont écrites au sein d'une transaction ininterruptible ; ce qui permet, en cas de plantage d'une entrée / sortie sur le disque, de revenir à une situation antérieure sans difficulté. L'écriture est programmée à intervalles réguliers par un mécanisme de « write back », toute modification d'un fichier étant d'abord réalisée en mémoire avant d'être effective. Grâce à ce mécanisme, les altérations de données qui auraient pu avoir lieu en mémoire peuvent être éventuellement réparées par consultation du fichier $LogFile par le système.

Les fichiers cachés de Windows 2000

On a vu que NTFS gérait au moins deux fichiers cachés ($MFT et $LogFile). Ces attributs « cachés » des fichiers vont plus loin que le simple bit H hérités des systèmes FAT, puisqu'il est effectivement impossible de visualiser ces fichiers en utilisant les APIs utilisateurs standard de Windows 2000.

La MFT constitue le cœur du système de fichiers NTFS. Elle est implémentée comme un tableau d'enregistrements de fichiers, chaque enregistrement ayant une taille fixe de 1 Ko. La MFT contient autant d'entrées qu'il existe de fichiers sur le volume.

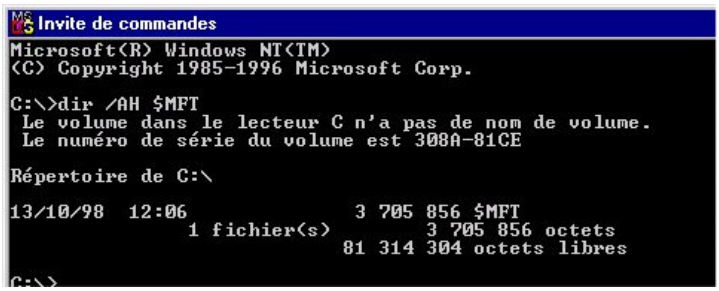
En plus du fichier MFT, chaque volume NTFS contient un jeu de fichiers spéciaux (les « Metadata Files ») qui servent à renseigner le système d'exploitation sur sa structure. Ces fichiers ont tous un nom commençant par le caractère $ et sont tous également cachés à l'utilisateur final.

La MFT réserve ainsi ses 16 premières entrées à la constitution des Metadata Files :

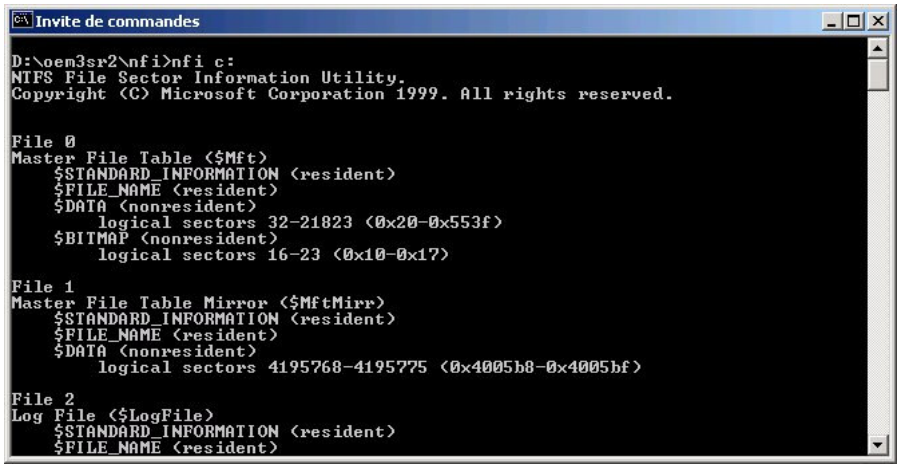
N° d'entrée dans la MFT	Nom du fichier	Description
0	**$Mft**	Master File Table : stocke toutes les informations sur toutes les données enregistrées sur le disque (y compris les autres fichiers Metadata)
1	**$MftMirr**	MFT Mirror File : contient les 16 premiers enregistrements de la MFT et est stocké en milieu de volume. Ce fichier est utilisé pour réparer des dommages à la MFT.
2	**$LogFile**	Ce fichier critique est utilisé par le Log File Service pour gérer les modifications sur les fichiers et répertoires de la structure du système de fichiers NTFS. Utilisée en cas de crash du système.
3	**$Volume**	Ce fichier contient la version de NTFS utilisée et le nom du

N° d'entrée dans la MFT	Nom du fichier	Description
		volume.
4	**$AttrDef**	Attribute Definition File. Contient de l'information sur les types d'attributs autorisés pour l'utilisation du volume, comme le fait que le disque soit ou non recouvrable en cas de panne.
5	\	Répertoire Racine du volume
6	**$Bitmap**	Ce fichier contient, pour chaque cluster, une information indiquant si celui-ci est utilisé ou occupé. L'information est stockée sous la forme d'un tableau de bits.
7	**$Boot**	Ce fichier contient des informations de bootstrap pour accéder au volume NTFS.
8	**$BadClus**	En cas de détection de mauvais clusters, ceux ci sont référencés dans ce ficher afin de ne plus être utilisés ultérieurement. Si un mauvais cluster est détecté, NTFS positionne un bit dans le fichier $Volume afin de provoquer un CHKDSK au prochain redémarrage en vue de l'éventuel réparation des clusters.
9	**$Secure**	**Nouveauté apparue sous Windows 2000**, ce fichier contient les descripteurs de sécurité de tous les fichiers présents sur le volume. Dans les fait, le « Security Descriptor » associé à chaque fichier pointe vers des entrées de ce fichier, ce qui permet d'économiser de la place pour des objets ayant les mêmes paramètres de sécurité.
10	**$UpCase**	Ce fichier est utilisé pour mapper les lettres majuscules avec les lettres minuscules pour l'accès aux fichiers et répertoires
11	**$Extend**	Répertoire de fichiers MetaData additionnels ; on peut y trouver le fichier $Quota, déjà présent sous NT 4.0 mais inutilisé sous cette version, et destiné à recevoir les données de quotas de disques.
12 à 15	**N/A**	Inutilisés
15 à xxx		Fichiers et répertoires utilisateurs

Il n'est pas possible d'ouvrir ces fichiers avec un quelconque éditeur, par contre on peut visualiser leur existence et leur taille (**sous Windows NT 4.0 uniquement**) par la simple frappe de la commande "DIR /AH NomdeFichier" à la racine du disque (C:\).



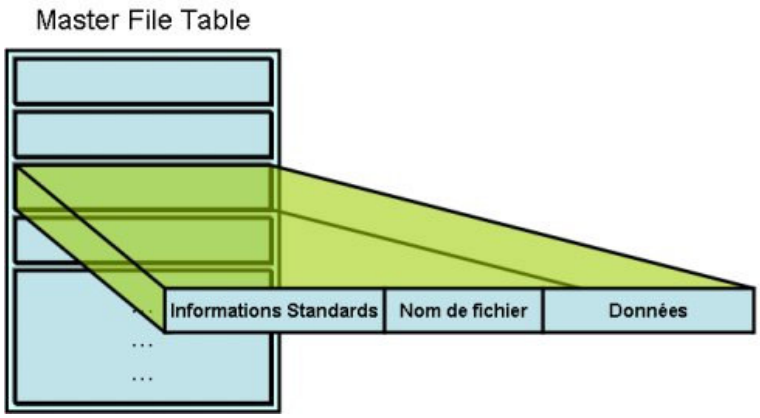
Sous Windows 2000, il est possible de lister ces Metadata files avec l'utilitaire « **nfi.exe** » fourni avec les « OEM support tool » de Windows 2000, téléchargeables gratuitement sur le site Internet de Microsoft.



Structure d'un fichier NTFS

Au lieu de voir les fichiers sous la forme d'un annuaire de données typées (textuelles, binaires…), le système NTFS stocke les fichiers comme une collection de couples attributs/valeurs dont l'un d'entre eux est le contenu même du fichier (*unnamed data attribute*).

Ainsi, à chaque enregistrement de la MFT correspond un fichier ou un répertoire, comme l'illustre la figure suivante :



Un fichier NTFS constitue une structure de données composée de différents attributs, chaque attribut étant stocké comme un flux d'octets (stream) distinct. A strictement parler, NTFS ne réalise donc pas de lecture/écriture sur des fichiers, il opère des opérations de lecture/écriture sur des flux d'attributs.

Par défaut, les opérations de lecture et d'écriture classiques sont réalisées sur le « *unnamed data stream* », mais il est possible de gérer autant d'attributs que nécessaire, et y compris de créer et de manipuler d'autres data streams comme on le verra dans la section suivante.

Les attributs d'un fichier sont désignés par un nom précédé d'un caractère « $ ». Le tableau suivant liste les principaux attributs des fichiers sous NTFS (tous ne sont pas nécessairement présents selon le type de fichier retenu).

Attribut	Nom d'Attribut	Description
Volume information	$VOLUME_INFORMATION $VOLUME_NAME	Ces attributs ne sont présents que dans le fichier de metadata $Volume. Ils permettent de stocker la version et le nom de Volume.

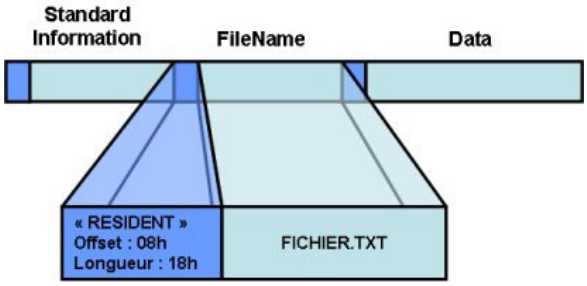
Attribut	Nom d'Attribut	Description
Standard information	$STANDARD_INFORMATION	Contient les propriétés élémentaires des fichiers : • bits Read Only, Archives etc. • dates diverses (création, modification…) • compte de hard-links • …
Filename	$FILE_NAME	Le nom du fichier en Unicode. Un fichier peut avoir plusieurs attributs de ce type (par exemple le hard-link, ou le nom court MSDOS du fichier pour les applications DOS 16 bits)
Security descriptor	$SECURITY_DESCRIPTOR	Cet attribut n'est présent que pour des raisons de compatibilité ascendante : sous Windows NT 4.0, il contenait le descripteur de sécurité des fichiers
Data	$DATA	Le contenu du fichier. Un fichier a au moins un attribut de ce type et peut même en avoir plusieurs. Un répertoire n'a pas d'attribut de ce type par défaut, mais peut en avoir éventuellement un ou plusieurs.
Index root, index allocation, and index bitmap	$INDEX_ROOT, $INDEX_ALLOCATION, $BITMAP	Ces attributs sont utilisés pour l'allocation des noms de fichiers et les index de bitmap pour de grands répertoires (attribut de répertoires uniquement).
Attribute list	$ATTRIBUTE_LIST	Une liste d'attributs complémentaires qui n'est présente que si un fichier nécessite plus d'un enregistrement dans la MFT.
Object ID	$OBJECT_ID	Un identificateur, codé sur 64 octets, les 16 premiers octets étant uniques sur le volume d'accueil. Il est utilisé par les raccourcis et certains programmes (comme les liens OLE) utilisant des APIs non documentées pour pouvoir appeler un fichier sans avoir besoin de spécifier son nom et son emplacement.
Reparse information	$REPARSE_POINT	Cet attribut permet de gérer les points de montage.
Extended attributes	$EA, $EA_INFORMATION	Attributs étendus, utilisés uniquement pour des raisons de compatibilité avec les applications OS/2.
EFS information	$LOGGED_UTILITY_STREAM	Utilisé par le système EFS (Encrypted File System). Cet attribut stocke le cryptogramme de la clef de chiffrement utilisée pour le fichier ainsi que la liste des utilisateurs autorisés à le déchiffrer.

Si un fichier est de petite taille, il est possible que tous ses attributs tiennent dans un seul enregistrement de la MFT[1]. Lorsqu'une valeur d'un attribut est directement stockée dans la MFT, cet attribut est alors dit « résident ». Certains attributs, comme par exemple le $STANDARD_INFORMATION ou le $FILE_NAME, doivent impérativement être résidents.

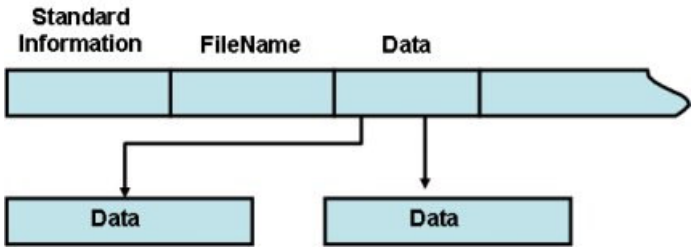
[1] Rappelons que, dans la MFT, chaque enregistrement a une taille de 1 Ko.

Chaque attribut commence par un en-tête spécifique donnant à NTFS des informations sur la façon de gérer cet attribut. L'en-tête, qui est toujours résident, précise alors si cet attribut est résident ou non.

Pour un attribut résident, l'en-tête désigne l'offset de début des données ainsi que la taille de l'attribut.



Lorsqu'un attribut est trop grand pour tenir dans un seul enregistrement de la MFT (par exemple l'attribut Data pour un fichier de plus de 1Ko) NTFS alloue alors les clusters nécessaires sur le disque, en dehors de la MFT donc, pour y stocker l'information, et l'en-tête de l'attribut contient toutes les informations nécessaires pour reconstituer le fichier.



Les « Alternate Data-Streams »

Microsoft a choisi comme structure interne du fichier NTFS de manipuler les fichiers sous forme de « streams » (littéralement : des « Flux »). Un fichier est généralement constitué d'une seule Stream dite « de données » (unamed data-stream) dans laquelle on retrouve le contenu des fichiers. Cependant, il est possible d'ajouter des « data-streams » additionnelles à un fichier NTFS afin d'y stocker des informations diverses. C'est grâce à ce mécanisme que Windows NT peut par exemple gérer des partages AppleShare pour les clients Macintosh : le système stocke dans des data-streams additionnelles les informations que les Macintosh utilisent sous forme de ressources - associations entre le fichier et l'applicatif à lancer pour l'ouvrir, icônes associées, etc.

Ces data-streams sont « invisibles », la seule façon de les lister est d'utiliser un programme spécifique (LADS par exemple) non fourni en standard avec le système d'exploitation.

Pour créer une data-stream additionnelle dans un fichier TOTO.TXT, il suffit d'ouvrir ce fichier avec un programme sachant gérer les data-streams comme NOTEPAD.EXE, et en précisant le nom de la data-stream à la suite du nom de fichier, séparé par le caractère « : ».

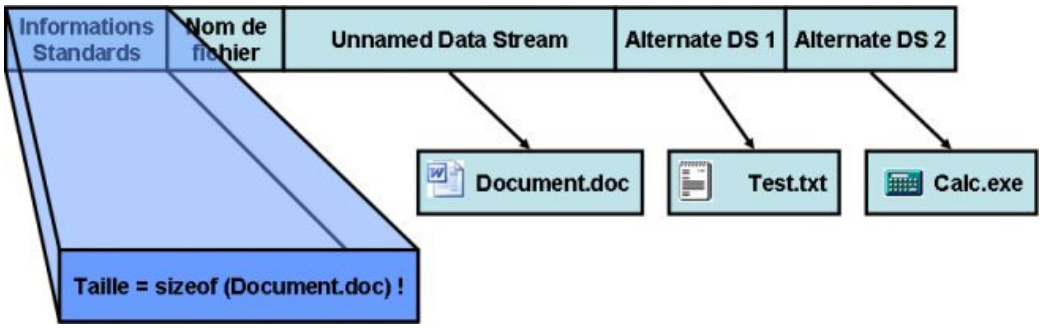
Exemple :

La commande « `NOTEPAD.EXE DOCUMENT.DOC:TEST.TXT` », créera (ou ouvrira si la data-stream existe) une data-stream TEST.TXT dans le fichier DOCUMENT.DOC.

De même, il est possible de copier un binaire dans une data-stream additionnelle avec la commande « type » et les possibilités de redirection de l'invite de commande :

```
TYPE CALC.EXE > DOCUMENT.DOC:CALC.EXE
```

Note : on peut ajouter autant de data-streams que l'on veut dans un fichier, et de la taille que l'on veut, sans que la taille apparente du fichier ne soit affectée.



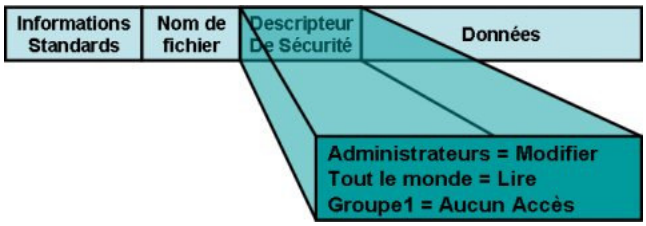
Lors de la copie de ce fichier, ces data-streams additionnelles ne sont copiées que si le volume de destination est bien au format NTFS et que l'on utilise la pile protocolaire Windows NT (copie par copier / coller sous l'explorateur par exemple), elles ne sont pas copiées lors d'un transfert FTP ou HTTP.

Stockage des Descripteurs de Sécurité

Comme on l'a vu précédemment, les Descripteurs de Sécurité des objets gérés par le système NTFS sont tous stockés dans le fichier caché **$Secure**.

Il s'agit ici d'une nouveauté, apparue avec Windows 2000, dans la gestion de la sécurité.

Sous Windows NT 4.0 en effet, les Descripteurs de Sécurité se trouvaient intégrés à chaque fichier sous la forme d'un attribut particulier (l'attribut $SECURITY_DESCRIPTOR).



Sous Windows 2000, la procédure diffère très nettement :

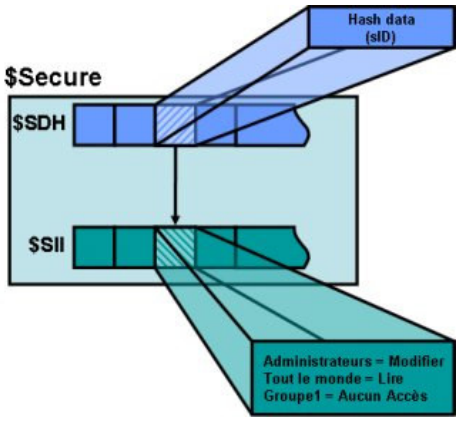
- Les gestionnaire NTFS assigne à chaque unique Descripteur de Sécurité un ID de sécurité (cet ID est différent du concept de SID décrit dans les chapitres précédents).

- Lorsque l'on souhaite appliquer un descripteur de sécurité à un fichier ou à un répertoire, le gestionnaire NTFS génère un hash sur 32 bits de ce descripteur et vérifie si ce dernier est déjà présent dans un index $SDH du fichier $Secure.

- Comme de nombreux descripteurs peuvent avoir le même hash, NTFS compare le descripteur en cours avec chaque descripteur ayant le même hash, en interrogeant la table des descripteurs $SII, afin d'être sûr du résultat.

- Si NTFS trouve un Descripteur de Sécurité identique, NTFS associe ce descripteur au fichier.

- Dans le cas contraire, NTFS alloue un nouvel ID de sécurité pour le descripteur, met à jour le $SDH avec le hash, stocke ce nouveau descripteur dans le $SII et associe ce descripteur au fichier.



Sécurité du Système de Fichiers NTFS

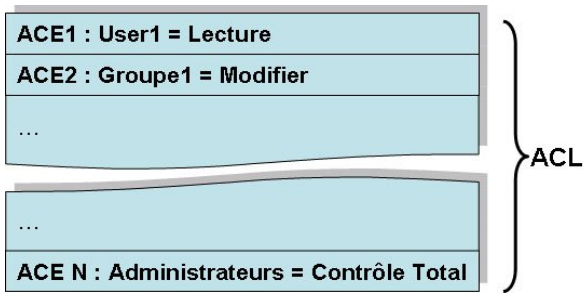
« Welcome…to the real world! »

Morpheus – The Matrix

Notion de Liste à Contrôle d'Accès

Une Liste à Contrôle d'Accès, ou ACL – Access Control List – peut être représentée comme un tableau contenant des informations sur ce que l'on a le droit de faire sur un objet.

Fonctionnellement, une ACL est composée d'une ou plusieurs ACE (Access Control Entry).



Les ACLs de Windows NT 4.0

Les ACLs sont des listes d'utilisateurs et de groupes qui disposent d'un ensemble de permissions élémentaires pour accéder ou agir sur les objets du système de fichiers. Chaque objet Windows NT dispose d'un descripteur de sécurité dans lequel sont stockées les ACLs.

Une ACL se compose d'un ensemble d'ACEs (Access Control Entry), chaque ACE constituant une autorisation élémentaire sur un objet.

Chaque permission listée ci dessous peut être associée à un utilisateur ou à un groupe. Les permissions élémentaires définies sous Windows NT4.0 sont alors les suivantes :

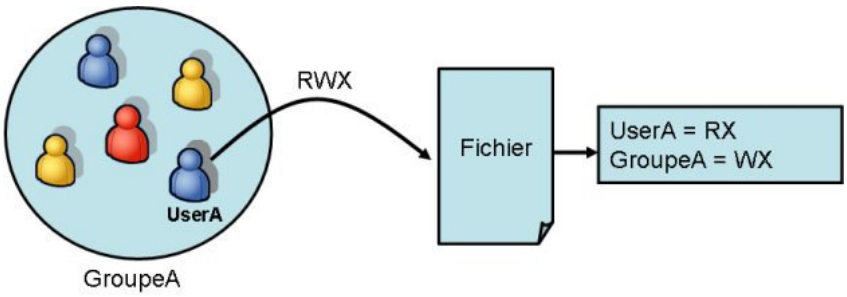
R	Répertoire : permet de lister le contenu d'un répertoire Fichier : permet de lire le contenu d'un fichier
W	Répertoire : permet d'ajouter des fichiers et de créer des sous répertoires Fichier : permet de modifier le contenu d'un fichier
X	Répertoire : permet de traverser un répertoire (cd) Fichier : permet d'exécuter un fichier si c'est un programme
D	Répertoire : permet d'effacer répertoire et sous répertoire Fichier : permet d'effacer un fichier
P	Répertoire : permet de modifier les permissions d'un répertoire Fichier : permet de modifier les permissions d'un fichier
O	Répertoire : permet de s'approprier un répertoire Fichier : permet de s'approprier un fichier

Afin de manipuler les ACLs de façon plus ergonomique, Windows NT fournit en outre une liste de permissions standards, prédéfinies, parmi lesquelles :

Nom	**Répertoire**	**Fichier**
Aucun Accès	Aucun	Aucun
Lister	RX	*Non spécifié*
Lire	RX	RX
Ajouter	WX	*Non spécifié*
Ajouter et lire	RWX	RX
Modifier	RWXD	RWXD
Contrôle total	RWXDPO	RWXDPO

Le positionnement des ACLs sur des objets du système suit une logique **cumulative** : les permissions accordées grâce à des ACLs sont cumulées au profit des utilisateurs et des groupes, **à l'exception de la permission « Aucun Accès »**.

Ainsi, si un utilisateur UserA dispose sur un objet de la permission RX, que ce même utilisateur appartient au groupe GroupeA et que ce groupe dispose de la permission WX sur cet objet, alors l'utilisateur UserA disposera au final des permissions RWX sur cet objet, résultants des permissions cumulées de UserA et de GroupeA.

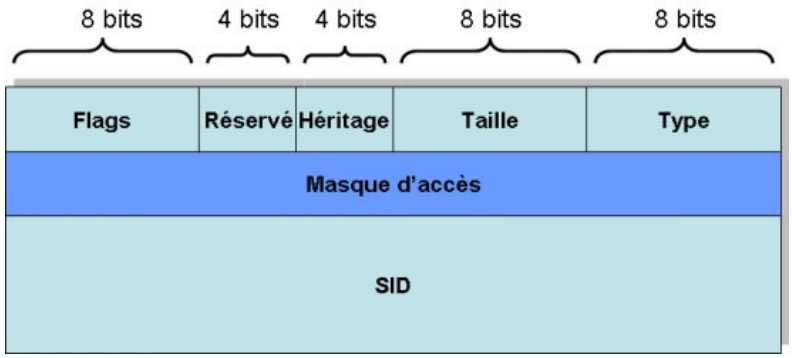


Note importante : la permission « **Aucun Accès** » a priorité sur les autres permissions, Attribuer cette permission au groupe « Tout le monde » revient alors à interdire tout accès à cet objet, y compris pour les Administrateurs).

Les ACLs de Windows 2000

L'apparition de Windows 2000 a quelque peu changé la donne en termes de positionnement d'ACLs sur les objets du système et en particulier sur le système de fichiers. Bien que complètement compatible avec les ACLs de Windows NT 4.0, la gestion des autorisations sous Windows 2000 s'est considérablement compliquée.

Comme sous Windows NT 4.0, une ACL est composée d'une ou plusieurs ACEs (Access Control Entry).

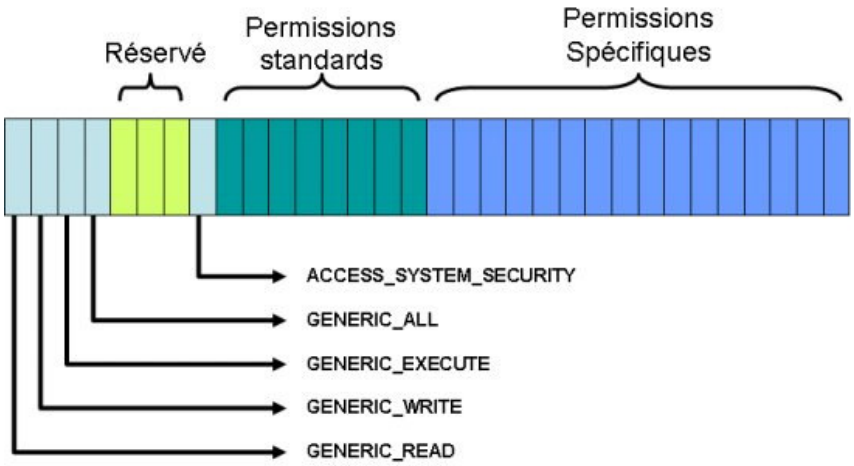


Structure d'une ACE

Chaque ACE se voit alors affecter un **type** et un **masque d'accès**. Les masques d'accès sont des entiers de 32 bits représentant ce que l'on appelle un champ de bits : chaque bit dans un masque d'accès précise une permission élémentaire.

Ce champ de 32 bits représentant une ACE a une structure commune à tous les objets : il est divisé en trois parties :

- Une partie « **générique** », indiquant un jeu de permissions des parties suivantes (le contenu du jeu peut différer selon le type d'objet),
- Une partie « **standard** » commune à tous les objets,
- Une partie « **spécifique** », dont l'existence et la signification diffère selon le type d'objet auquel on applique une ACL.



Structure d'un masque d'accès

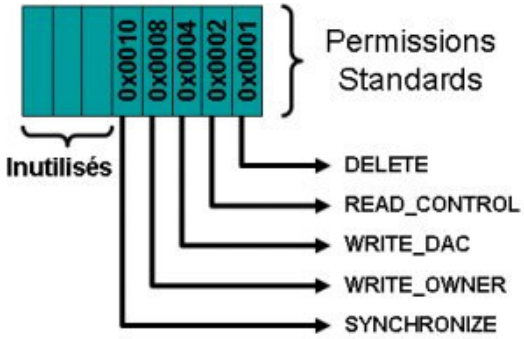
La partie générique est utilisée pour positionner ou récupérer rapidement un jeu de permissions élémentaire.

Il existe **deux principaux types d'ACE** : ACCESS_ALLOWED et ACCESS_DENY, précisant respectivement une Autorisation et un Refus d'Accès.

Pour le système de fichiers NTFS, les bits de protection disponibles dans un masque d'accès sont décrits ci-après[1].

Permissions Standards

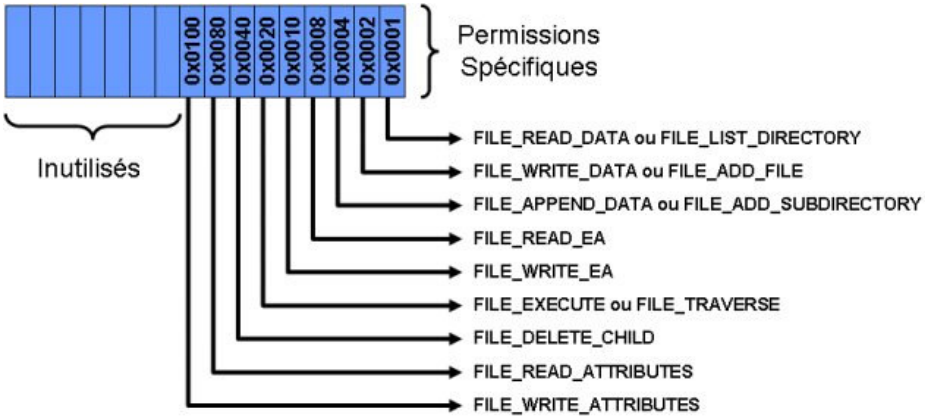
Nom	Description
DELETE	Permet de supprimer le fichier.
READ_CONTROL	Permet de lire l'ACL.
WRITE_DAC	Permet de modifier l'ACL du fichier.
WRITE_OWNER	Permet de modifier le propriétaire du fichier.
SYNCHRONIZE	Permet d'attendre un évènement de synchronisation sur le fichier.



[1] Les valeurs hexadécimales présentes dans les schémas qui suivent représentent la valeur numérique associée à une permission, donc à sa position dans le champ de bits.

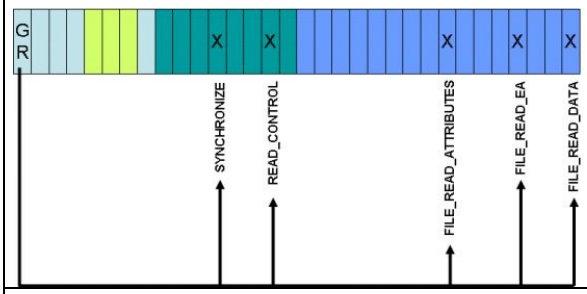
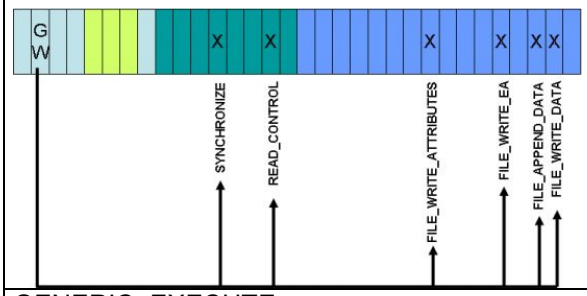
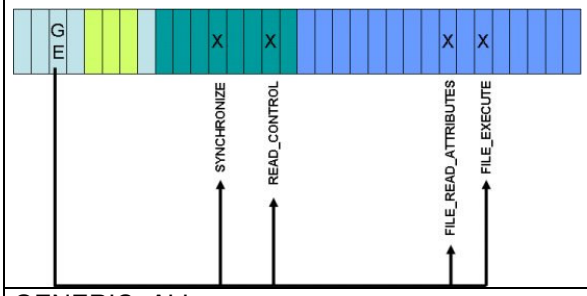
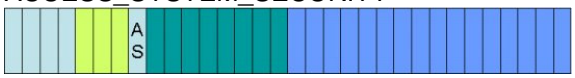
Permissions Spécifiques

Nom	Description
FILE_READ_DATA FILE_LIST_DIRECTORY	Fichier : Lire le contenu. Répertoire : Lister le répertoire.
FILE_WRITE_DATA FILE_ADD_FILE	Fichier : Modifier le contenu. Répertoire : Ajouter un fichier.
FILE_APPEND_DATA FILE_ADD_SUBDIRECTORY	Fichier : Ajouter des données (?). Répertoire : Ajouter un répertoire.
FILE_READ_EA	Lire les Attributs Etendus.
FILE_WRITE_EA	Ecrire les Attributs Etendus.
FILE_EXECUTE FILE_TRAVERSE	Fichier : Exécution. Répertoire : Traverser le répertoire.
FILE_DELETE_CHILD	Répertoire : Autorise l'effacement du répertoire même si les objets enfants ont une ACL interdisant l'effacement[1].
FILE_READ_ATTRIBUTES	Lire les Attributs.
FILE_WRITE_ATTRIBUTES	Ecrire les Attributs.



[1] Ce flag était déjà présent sous Windows NT 4.0. La permission standard « Contrôle Total » positionne par défaut ce flag.

Permissions Génériques

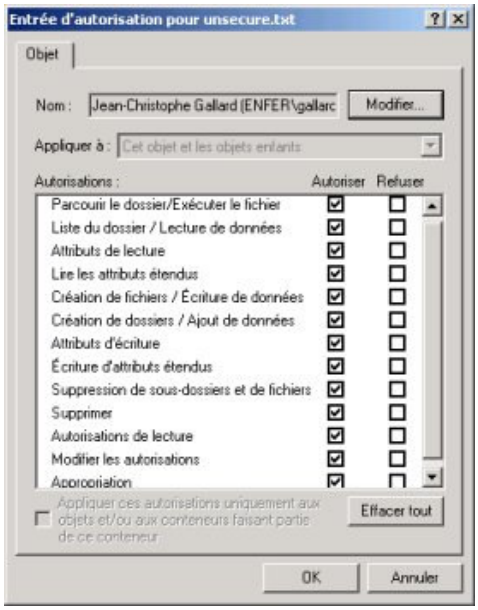
Nom	Description
GENERIC_READ 	Combinaison de: READ_CONTROL, FILE_READ_DATA, FILE_READ_ATTRIBUTES, FILE_READ_EA, SYNCHRONIZE
GENERIC_WRITE 	Combinaison de : READ_CONTROL, FILE_WRITE_DATA, FILE_WRITE_ATTRIBUTES, FILE_WRITE_EA, FILE_APPEND_DATA, SYNCHRONIZE
GENERIC_EXECUTE 	Combinaison de : READ_CONTROL, FILE_EXECUTE, FILE_READ_ATTRIBUTES, SYNCHRONIZE
GENERIC_ALL 	A priori, combinaison de toutes les permissions[1].
ACCESS_SYSTEM_SECURITY 	Permet d'accéder en écriture à l'ACL d'audit.

Correspondances Noms de Permissions / Actions du gestionnaire d'ACLs

Les noms des permissions élémentaires (FILE_READ_DATA, READ_CONTROL…) qui ont été vues jusqu'ici ne sont pas directement accessibles au travers de l'interface de Windows 2000 ; elles portent dans cette interface un nom plus explicite sur leur fonction, comme l'indique le tableau ci-dessous :

[1] La documentation de Microsoft n'est malheureusement pas claire sur ce point…

Nom Interne	Nom dans l'IHM
DELETE	Supprimer
READ_CONTROL	Autorisations de lecture
WRITE_DAC	Modifier les autorisations
WRITE_OWNER	Appropriation
SYNCHRONIZE	**PAS DE CORRESPONDANCE**
FILE_READ_DATA **FILE_LIST_DIRECTORY**	Liste du dossier / Lecture de données
FILE_WRITE_DATA **FILE_ADD_FILE**	Création de fichier / Ecriture de données
FILE_APPEND_DATA **FILE_ADD_SUBDIRECTORY**	Création de dossiers / Ajout de données
FILE_READ_EA	Lire les attributs étendus
FILE_WRITE_EA	Ecriture d'Attributs étendus
FILE_EXECUTE **FILE_TRAVERSE**	Parcourir le dossier / exécuter le fichier
FILE_DELETE_CHILD	Suppression de sous-dossier et de fichiers
FILE_READ_ATTRIBUTES	Attributs de lecture
FILE_WRITE_ATTRIBUTES	Attributs d'écriture

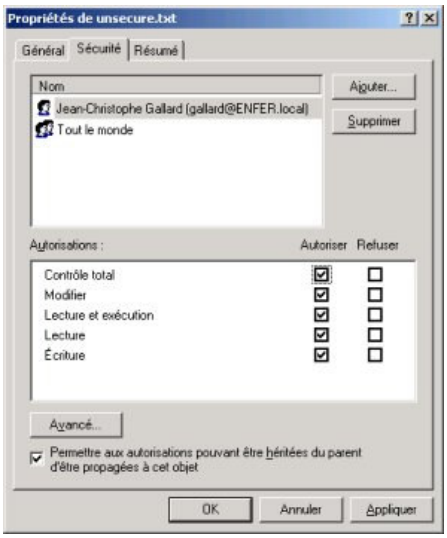


Permissions de Base

Afin d'éviter de devoir gérer trop finement les ACLs sur les objets du système de fichier, le système Windows 2000 est livré avec un jeu de permissions de base :

- Contrôle Total
- Modifier
- Lecture

- Lecture et Exécution
- Afficher le contenu du dossier (répertoires uniquement)
- Ecriture



La composition de chaque jeu de permission est précisée dans le tableau suivant :

	Contrôle Total	Modifier	Lecture	Lecture et Exécution	Afficher le contenu…	Ecriture
Parcourir le dossier / exécuter le fichier	●	●		●	●	
Liste du dossier / Lecture de données	●	●	●	●	●	
Attributs de lecture	●	●	●	●	●	
Lire les attributs étendus	●	●	●	●	●	
Création de fichier / Ecriture de données	●	●				●
Création de dossiers / Ajout de données	●	●				●
Attributs d'écriture	●	●				●
Ecriture d'Attributs étendus	●	●				●
Suppression de sous-dossier et de fichiers	●					
Supprimer	●	●				
Autorisations de lecture	●	●	●	●	●	●
Modifier les autorisations	●					
Appropriation	●					
SYNCHRONIZE	●	●	●	●	●	●

Correspondance entre ACLs Windows NT 4.0 et 2000

Windows NT 4.0	Windows 2000
R	Liste du dossier / Lecture de données + Attributs de lecture + Lire les attributs étendus + Autorisations de lecture
W	Création de fichier / Ecriture de données + Création de fichier / Ecriture de données + Création de dossiers / Ajout de données + Attributs d'écriture + Ecriture d'attributs étendus
X	Parcourir le dossier / exécuter le fichier
D	Supprimer
P	Modifier les autorisations
O	Appropriation
Contrôle Total	+ Parcourir le dossier / exécuter le fichier + Liste du dossier / Lecture de données + Attributs de lecture + Lire les attributs étendus + Création de fichier / Ecriture de données + Création de fichier / Ecriture de données + Création de dossiers / Ajout de données + Attributs d'écriture + Ecriture d'attributs étendus + **Suppression de sous-dossier et de fichiers**[1] + Supprimer + Autorisations de lecture + Modifier les autorisations + Appropriation

Le mécanisme d'héritage de Windows 2000

Sous Windows 2000, quelques différences existent avec Windows NT 4.0 dans les propriétés de sécurité des objets du système de fichier. Ces nouvelles fonctionnalités ont en fait été présentes dès l'arrivée du Service Pack 4 de Windows NT 4.0, puisque ce Service Pack mettait à jour le système de fichier NTFS de façon transparente pour l'utilisateur / administrateur.

L'interface utilisateur de Windows NT ne permettait pas leur manipulation, sauf à installer l'utilitaire SecEdit, présent sur le CD-Rom du Service Pack 4 distribué par Microsoft. Cette installation avait pour effet secondaire de fournir l'interface graphique autorisant la manipulation des ACLs à la mode de Windows 2000 (dont le mécanisme d'héritage).

Sous Windows 2000, ces nouvelles fonctionnalités sont natives et, parmi celles-ci, on trouve :

- La possibilité de propager les permissions à des objets enfants (héritage implicite),
- La possibilité d'interdire l'héritage des permissions.

[1] Bit FDC (File Delete Child).

Ce mécanisme d'héritage est rigoureusement identique à la notion d'héritage dans le monde de la programmation objet ; un objet parent (un répertoire par exemple), disposant d'une liste à contrôle d'accès, propagera ces permissions aux objets enfants et ce de manière automatique. Ainsi, en cas de modification des permissions de l'objet parent, les objets enfant se verront automatiquement appliquer ces modifications tout en conservant leurs propres permissions.

Exemple :

Un répertoire PERE dispose de l'ACL suivante pour les fichiers :

```
Tout le monde = Lire (RX)
```

Tout fichier ou répertoire ENFANT créé sous ce répertoire se verra alors attribuer l'ACL du répertoire PERE.

Supposons alors que l'on ajoute une entrée à l'ACL du fichier ENFANT :

```
Utilisateur Martin = Ecrire
```

L'ACL du fichier ENFANT sera donc :

```
Tout le monde =  Lire (RX)
Utilisateur Martin = Ecrire
```

Si l'administrateur modifie l'ACL du répertoire PERE en rajoutant la permission « modifier » au groupe GESTIONNAIRES :

```
Tout le monde = Lire (RX)
Groupe GESTIONNAIRES = Modifier
```

Sans qu'il soit nécessaire de faire quoi que ce soit, le fichier ENFANT disposera alors de l'ACL suivante :

```
Tout le monde =  Lire (RX)
Utilisateur Martin = Ecrire
Groupe GESTIONNAIRES = Modifier
```

Si, par contre, le fichier ENFANT était situé dans un répertoire FILS du répertoire PERE et que ce répertoire FILS ait été paramétré pour le blocage de l'héritage des permissions, alors aucun des fichiers du répertoire FILS n'héritera de la nouvelle ACL tandis que tous les autres répertoires et fichiers fils de PERE disposeront de leur nouvelle ACL.

Les partages Windows

Depuis les premières versions de Windows NT, le système d'exploitation de Microsoft prend en charge de façon native la gestion des partages réseaux. Ces partages réseaux sont mis en œuvre via un protocole propriétaire autrefois appelé SMB (Server Message Block) mais qui est désormais connu sous l'acronyme CIFS (Common Internet File System).

L'accès par le réseau à ces partages satisfait à la convention de nommage UNC[1], de la forme **\\NomDeMachine\NomDePartage**.

Il est possible d'accéder à un partage soit en réalisant une connexion à un lecteur réseau (ie on affecte à une lettre de lecteur inutilisée le chemin UNC du partage : `NET USE Z:`

[1] Universal Naming convention.

`\\MACHINE\PARTAGE1`) soit directement en précisant le chemin UNC (par exemple : `COPY TEST.TXT \\MACHINE\PARTAGE1\REPERTOIRE`).

Les partages par défaut

Le cœur de la gestion des partages Windows réside dans le service « Serveur » de toute machine sous Windows NT / 2000 / 2003 / XP. Ce service est, par défaut, actif et met en œuvre un certain nombre de partages de façon automatique :

Les partages C$, D$, E$...

Ces partages sont créés systématiquement pour tout volume physique non amovible présent sur le système. Chacun de ces partages pointe vers la racine de la lettre de lecteur correspondant. Ainsi, C$ pointe sur C:\, D$ sur D:\, etc. Les permissions sur ces partages n'autorisent que les administrateurs à s'y connecter.

Le partage Admin$

Ce partage pointe sur le répertoire contenant le système d'exploitation, soit généralement « C:\winnt ». Plus précisément, ce partage pointe en fait sur le contenu de la variable d'environnement %SystemRoot% du serveur.

Le partage IPC$

Il s'agit d'un partage virtuel, donc non raccroché à un répertoire existant (IPC signifiant Inter Process Communication). Ce partage permet de s'authentifier sur une machine distante, même si cette dernière ne propose aucun partage réseau. En d'autres termes, il est le point de d'écoute des appels RPC.

Le partage NETLOGON

Ce partage n'est disponible que sur les contrôleurs de domaine Windows NT et sur les contrôleurs de domaines Windows 2000 disposant du rôle d'émulateur de PDC. Sous Windows NT 4.0, ce répertoire pointe sur `%SystemRoot%\System32\Repl\Import\Scripts`. Ce répertoire héberge les scripts d'ouverture de session ainsi que les stratégies systèmes du domaine.

Sous Windows 2000, ce répertoire pointe sur `%SystemRoot%\SYSVOL\sysvol\(domaine)\SCRIPTS`.

Le répertoire vers lequel pointe ce partage est spécifié dans la valeur **Scripts** sous **`HKLM\SYSTEM\CurrentControlSet\Services\Netlogon\Parameters`**.

Le partage SYSVOL

Ce partage n'est présent que sur les contrôleurs de domaines Windows 2000. Il contient l'ensemble des GPOs d'un domaine.

Ce partage pointe sur `%SystemRoot%\SYSVOL\sysvol`. Le répertoire vers lequel pointe ce partage est spécifié dans la valeur **SysVol** sous **`HKLM\SYSTEM\CurrentControlSet\Services\Netlogon\Parameters`**.

Les partages Admin$, C$, D$... sont appelés « partages administratifs » (« admin shares »). Il est possible de les désactiver en départageant les répertoires concernés, mais ils redeviendront actifs lors du prochain redémarrage de la machine. Pour les supprimer définitivement, un paramètre de la base de registre situé sous HKLM peut être utilisé (il est différent selon le type de machine, serveur ou station de travail).

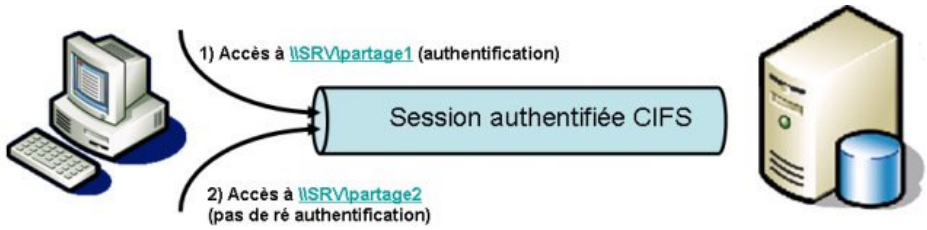
Pour désactiver totalement l'ensemble de ces partages, l'arrêt du service Serveur s'impose mais la machine ne pourra alors plus être administrée à distance.

Accès authentifiés aux partages

Lorsque l'on se connecte à un partage réseau depuis une machine cliente, cette dernière tente une connexion au partage en utilisant les pièces justificatives de l'identité en cours ; c'est ce comportement qui explique que lorsque l'on se connecte à un partage dans son domaine Windows, le système ne demande pas de se ré-authentifier. Cette authentification demeure donc totalement transparente aux yeux de l'utilisateur.

En cas d'échec de l'authentification (par exemple parce que le serveur est hors domaine), le système présente alors une bannière de connexion. A partir de cet instant, il convient de ne pas oublier que l'on vient de subir un premier échec d'authentification avec le login en cours, ce qui signifie que si la stratégie de sécurité de la machine hébergeant le partage veut que le compte soit verrouillé au bout de 3 connexions infructueuses et que si le nom du compte de domaine est identique au nom du compte requis pour l'accès au partage, il ne reste alors plus que 2 tentatives et non 3.

Dès l'authentification réussi, la machine cliente a établi un contexte de session entre elle même et le serveur : toute autre communication ultérieure avec le serveur utilisera par cette session. Ainsi, la connexion ultérieure à un autre partage sur le même serveur ne nécessitera plus de nouvelle authentification.



Ce fonctionnement, certes pratique, se révèle cependant limitatif **puisqu'il est alors impossible de monter deux partages différents sur un même serveur en utilisant deux comptes différents**. Il existe cependant une « astuce » pour outrepasser cette règle :

- On monte un premier volume en utilisant le **NOM** de la machine et le compte X
- On monte le second volume en utilisant **l'Adresse IP** de la machine et le compte Y

Partages et permissions de partage

Windows 2000 permet de gérer des volumes partagés sur un réseau et de positionner des permissions d'accès sur ces partages.

A la création d'un partage, on peut définir des permissions à deux niveaux distincts : les permissions sur le partage lui-même et les permissions NTFS sur le contenu du partage.

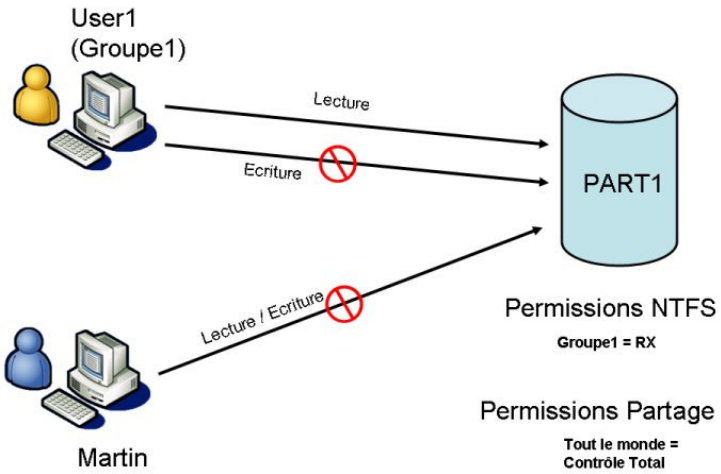
Les permissions sur le partage sont similaires aux permissions que l'on peut positionner sur les répertoires NTFS, à ceci près que ces permissions ne concernent que la façon dont le partage sera vu sur le réseau. Les permissions NTFS sont les permissions que l'on peut positionner sur les répertoires et fichiers.

Lorsque les deux mécanismes sont mis en œuvre, les permissions sur le partage donnent les permissions maximales applicables sur le répertoire partagé à travers le réseau.

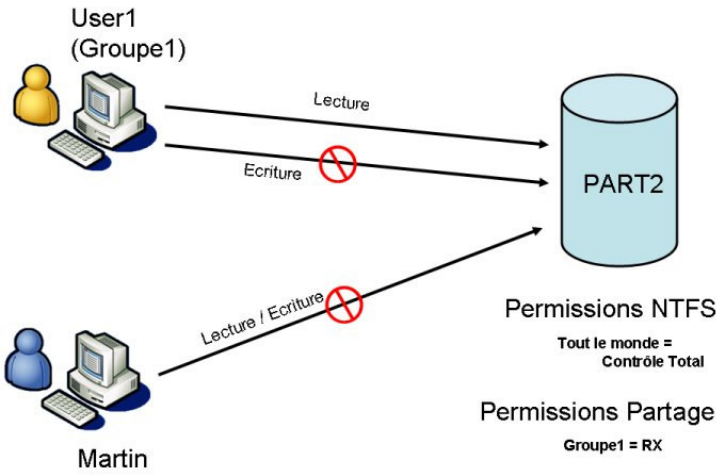
Les exemples qui suivent donnent deux techniques différentes pour autoriser l'accès à un partage à un groupe d'utilisateur en lecture / exécution (permissions RX).

Exemple 1 : on définit un partage **PART1** sur un répertoire avec la permission NTFS **« Groupe1 » = « RX »**, et la permission de partage **« Tout le monde » = « Contrôle Total »**. L'utilisateur distant « User1 », appartenant au groupe « Groupe1 » n'a pas accès

en Ecriture au partage, malgré la permission NFTS lui attribuant le contrôle total sur le répertoire. Cependant, les utilisateurs du groupe « Groupe1 » accédant au répertoire en local (donc hors connexion réseau) y auront accès en Contrôle Total. Les autres n'y accéderont qu'en lecture.



Exemple 2 : on définit un partage **PART2** sur un répertoire avec la permission NTFS « **Tout le monde** » = « **Contrôle Total** », et la permission de partage « **Groupe1** » = « **RX** ». L'utilisateur distant « User1 », appartenant au groupe « Groupe1 » n'aura de la même façon accès au répertoire PART2 qu'en lecture / exécution, du fait des restrictions sur les permissions du partage.



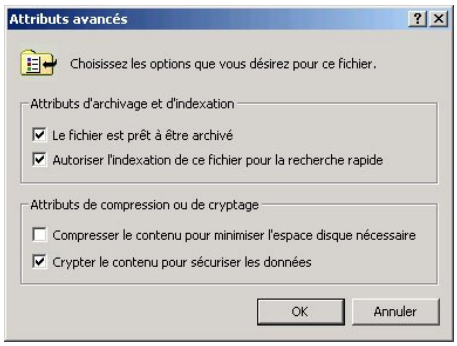
Il est également possible d'interdire la visualisation des partages via le voisinage réseau, en ajoutant à la fin du nom de partage le caractère $. C'est par exemple le cas du partage Admin$, qui est le partage administratif actif par défaut sur toutes les machines Windows 2000.

Encrypted File System (EFS)

Sous Windows NT, le système de fichier permettait de gérer des permissions sur les objets gérés par celui-ci (répertoires, fichiers).

Cependant les permissions positionnées sur ces objets pouvaient être outrepassées par un accès physique au média de stockage. Ainsi de nombreux utilitaires ont vu leur apparition, permettant depuis une disquette de démarrage MS-DOS ou Linux d'accéder directement au disque dur sans être contraint par les éventuelles permissions positionnées sur le système de gestion de fichiers.

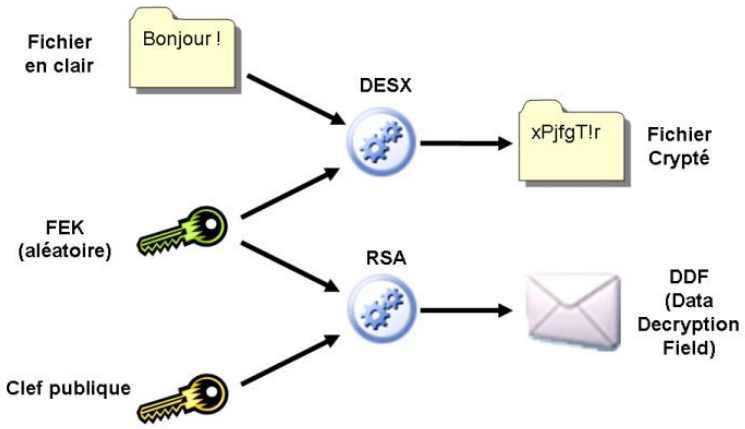
Avec Windows 2000 apparaît la notion de système de fichiers chiffrés ou EFS (Encrypted File System). Il s'agit d'une simple surcouche transparente à NTFS, utilisant les nouvelles fonctionnalités apparues avec NTFS V2, qui devient de facto le standard de système de gestion de fichiers sous Windows 2000.



EFS a pour principale caractéristique d'intervenir au niveau d'un fichier ou d'une arborescence. Chaque fichier est chiffré à l'aide d'une clef différente tirée aléatoirement. Le mécanisme de chiffrement de fichiers permet, de plus, la capacité de récupération de fichiers chiffrés par un administrateur.

Le mécanisme de chiffrement utilisé est le suivant :

- Une clef (**FEK** - **File Encryption Key**) de 128 bits est tirée aléatoirement par le système.
- Cette FEK sert à chiffrer, par un algorithme DESX[1], le fichier considéré
- Cette FEK est alors chiffrée par un algorithme de chiffrement à clef publique ; la FEK est chiffrée à l'aide de la clef publique de l'utilisateur (donc connue de tous) et le résultat est stocké dans NTFS sous un champ particulier associé au fichier (champ **DDF** - **Data Decryption Field**).
- Pour déchiffrer le fichier, l'utilisateur utilise une clef privée (connue de lui seul) associé à son compte, afin de déchiffrer le champ DDF
- La FEK récupérée dans ce champ permet alors de déchiffrer le fichier
- En outre, un mécanisme permet à un administrateur système de déchiffrer un fichier par un utilisateur.



Il est théoriquement possible de chiffrer un fichier en faisant en sorte que plusieurs utilisateurs puissent le déchiffrer. Il suffit pour cela de générer autant de DDF qu'il existe d'utilisateurs capables de déchiffrer le fichier. Cependant, cette possibilité n'est offerte que dans Windows 2003.

[1] DESX est l'implémentation « Microsoft » du DES…

En plus des champs DDF, un ou plusieurs champs dits **DRF** (**Data Recovery Field**) sont ajoutés au fichier, permettant à des agents de récupération (paramétrables par l'administrateur) de déchiffrer le FEK du fichier, rendant ainsi possible la récupération par l'administrateur d'un fichier chiffré. Par défaut, l'administrateur du système est un agent de récupération.

Sous Windows 2003, le nombre d'agent de récupération peut être de zéro, contrairement à Windows 2000 ou au moins un agent de récupération devait être défini, par défaut le compte **BUILT_IN\Administrateur**. En outre, Windows 2003 autorise l'utilisation de l'algorithme de chiffrement AES en lieu et place du DES.

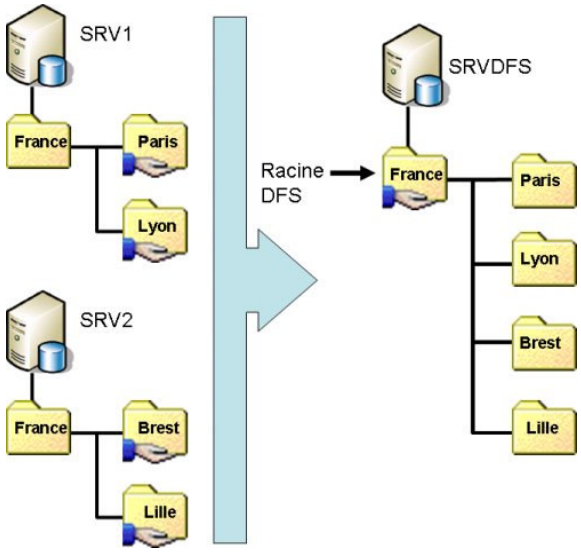
Distributed File System (DFS)

Concepts

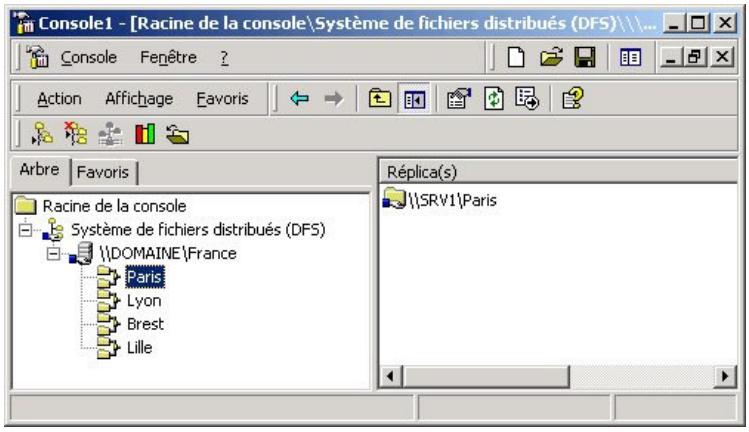
DFS est une solution de stockage distribuée offerte sous Windows 2003 et dans les versions « serveur » de Windows 2000. Cette solution permet de grouper logiquement des partages réseaux, éventuellement répartis sur plusieurs serveurs, au sein d'un espace de noms hiérarchique.

Avec DFS, les utilisateurs n'ont plus à connaître l'emplacement physique exact des partages (ie le nom du serveur où il se trouve). Contrairement à la notion de partage, la gestion du DFS n'est pas directement accessible dans les menus contextuels du système de fichier ; DFS se gère via le composant « Système de fichiers Distribué » de la MMC.

Une racine DFS est définie comme le point de départ d'une hiérarchie de liaisons DFS qui pointent vers des dossiers partagés. Cette architecture autorise la mise en commun de plusieurs partages existants, éventuellement sur plusieurs serveurs, sous une seule arborescence, gérée par un unique serveur.



L'opération de mise en place du DFS ne détruit pas l'architecture existante : le serveur DFS n'est qu'un serveur de présentation, redirigeant les requêtes vers le bon serveur, celui qui gère effectivement les partages.



Gestion de la réplication

En outre, le système DFS offre la possibilité de mettre en œuvre des mécanismes de réplications, permettant ainsi de résister à des indisponibilités des serveurs. Dans la capture d'écran qui précède, on voit une racine DFS nommée « France » et qui contient une liaison DFS nommé « Paris » et pointant sur le partage « Paris » du serveur SRV1. Deux problèmes de déni de service peuvent alors se présenter :

1. Panne du serveur SRV1 : la racine DFS sera toujours accessible, mais pas l'arborescence « Paris »

2. Panne du serveur DFS hébergeant la racine : toute l'arborescence DFS est alors inaccessible.

Avec DFS, les deux problèmes peuvent cependant être résolus par une stratégie de réplication :

- On peut répliquer le partage « Paris » sur un autre serveur (SRV2 par exemple) et déclarer ce partage répliqué comme un second « réplica » du lien DFS « Paris » : en cas de panne de l'un des deux serveurs, l'autre peut alors prendre la relève.

- On peut répliquer la racine DFS sur un second serveur : en cas de panne du serveur DFS, le second serveur sert alors les requêtes.

Sécurité de DFS

À part la création des autorisations d'administrateur nécessaires à la gestion des racines DFS, le service DFS ne met en place aucune autre mesure de sécurité supplémentaire, en dehors de ce qui est offert par le système Windows 2000. Les autorisations affectées à une racine ou à une liaison DFS déterminent qui est habilité à ajouter une nouvelle liaison DFS.

Les autorisations d'accès à un dossier partagé ne sont pas liées à la topologie DFS. Ainsi, le fait d'avoir des autorisations d'accès à une racine DFS permet de visualiser l'ensemble des liens DFS qui y sont proposés mais ne permet en aucun cas de garantir l'accès à ces dossiers ; **un tel accès est déterminé par les contrôles de sécurité standard de Windows 2000 sur les serveurs hébergeant les partages**.

En résumé, la sécurité est appliquée par le système de fichiers sous-jacent lorsque l'utilisateur tente d'accéder à un dossier partagé DFS ou à son contenu.

Ce mécanisme fonctionne donc exactement de la même manière que les autorisations sur un partage, qui ne permettent pas d'outrepasser les autorisations du système de fichier NTFS sous-jacent. Attention cependant à une confusion courante : les autorisation

positionnées sur la racine DFS ne servent pas à définir qui a le droit de faire quoi sur les fichiers et répertoires de cette racine, mais qui a le droit d'administrer la racine (création, suppression d'un lien…)

Nouveautés liées à Windows 2003

Avec Windows 2003, l'architecture DFS s'est enrichie de nouvelles fonctionnalités :

- Il est désormais possible de gérer plusieurs racines DFS sur un même serveur[1] (sous Windows 2000, un serveur ne pouvait en gérer qu'une seule).
- Les partages DFS peuvent être publiés en tant qu'objets « Volume » dans l'Active Directory (ils apparaissent isolément en tant que volume), et on peut en déléguer l'administration.
- Il est possible de définir une liaison DFS pointant vers une racine DFS (interlink).
- DFS intègre une fonctionnalité qui permet de sélectionner automatiquement le site de réplication le plus proche pour router un client vers le serveur de fichiers disponible le plus proche.

[1] Cette fonctionnalité n'est pas disponible pour Windows Server 2003 Standard Edition

Active Directory

« Il n'y a pas de forêts sans arbres tordus »

Proverbe bulgare

Concepts

L'une des principales évolutions du système d'exploitation Windows 2000 par rapport à son prédécesseur consiste dans la définition d'une architecture de système d'information, dans laquelle vont s'intégrer les postes Windows 2000, radicalement différente de celle que nous étions habitués à manipuler avec Windows NT 4.0.

Cependant, cette évolution s'est effectuée relativement en douceur dans la mesure où Microsoft a repris les grands concepts existants pour les améliorer et les compléter.

Le socle sur lequel repose cette évolution d'architecture s'appelle l'Active Directory, qui représente la même révolution que pouvait représenter l'arrivée de la base de registres avec Windows 9x et NT.

Gestion centralisée des ressources et services / Active Directory

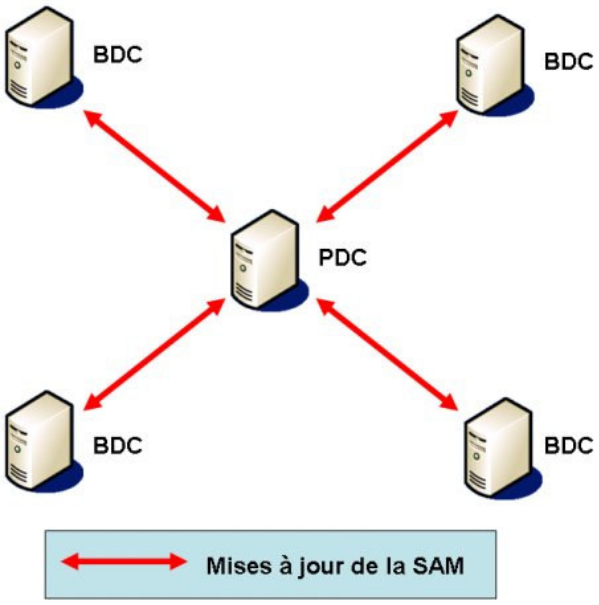
Sous Windows NT, il était possible de regrouper des ressources communes au sein d'une structure appelée "Domaine Windows NT". Le principe de base de ce regroupement était de centraliser sur des serveurs un certain nombre d'informations pour les rendre accessibles à l'ensemble des machines reliées à un domaine. Ainsi, la notion de domaine Windows NT impliquait la maintenance et la mise à jour d'une base centralisée des utilisateurs sur un ou plusieurs serveurs.

Ce principe a été retenu sous Windows 2000, mais la base s'est enrichie de nouveaux objets et permet en outre d'en gérer un plus grand nombre ; sous Windows NT la base SAM des utilisateurs était limitée à 40000 entrées, sous Windows 2000 la base d'annuaire autorise plusieurs millions d'entrées.

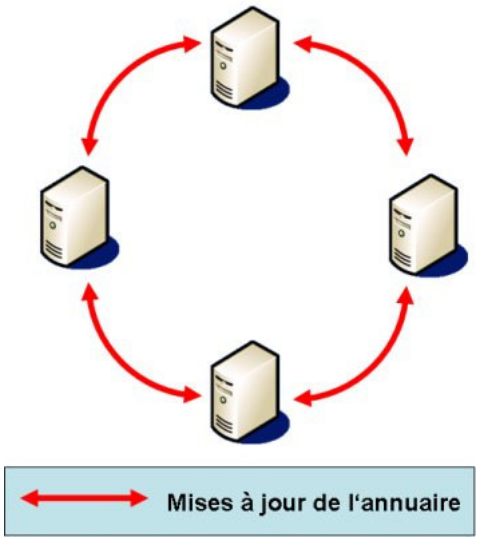
Du fait de l'extension du principe d'annuaire centralisé sous Windows 2000, la gestion de cet annuaire a été regroupée autour de "Active Directory", système de base d'annuaire hiérarchique centralisée sur des serveurs.

En fait la notion de "centralisation" sous Windows 2000 n'est pas tout à fait exacte stricto sensu. En effet, sous Windows NT la base était centralisée sur un serveur (le serveur principal de domaine ou PDC - Principal Domain Controler) éventuellement épaulé par d'autres serveurs (serveurs secondaires ou BDC -Backup Domain Controler), la relation entre PDC et BDC étant réalisé sur un mode maître-esclave ; chaque modification est

rapportée au PDC qui a la charge de répercuter ces modifications sur l'ensemble des BDC.



Sous Windows 2000, la base d'annuaire "Active Directory" est répartie sur l'ensemble des serveurs et chaque serveur possède un rôle équivalent aux autres serveurs. Alors que, sous Windows NT, l'organisation logique des serveurs était architecturée autour d'une organisation en étoile, l'organisation logique des serveurs Windows 2000 est architecturée autour d'une organisation en anneau ; chaque modification d'annuaire sur un serveur est répercutée auprès des serveurs contigus, de proche en proche.



Sous Windows 2000, donc, apparaît la notion d'Active Directory. Active Directory est un service d'annuaire, hiérarchique, et disponible uniquement dans la version serveur de Windows 2000. Son rôle essentiel est de fournir un service réseau qui identifie toutes les ressources d'un réseau et rend ces dernières accessibles aux utilisateurs et aux applications. Ces ressources sont définies dans des entités typées appelées "objets". Un objet peut consister en des données utilisateurs, des imprimantes, des serveurs, des stations de travail, des groupes etc.

Notons que Active Directory reste un annuaire propriétaire, même s'il peut s'interfacer avec des services standardisés (LDAP ou Novell NDS par exemple).

De plus, il est ici utile de préciser que Active Directory repose pour son fonctionnement sur le service de nommage DNS, ce qui peut apparaître curieux dans la mesure ou l'on a affaire à un service d'annuaire qui repose sur un autre service d'annuaire (dans son fonctionnement, DNS est un service d'annuaire).

Introduction à l'Active Directory

Active Directory (AD) reste la plus grande évolution qu'ait connue le système d'exploitation Microsoft. Cette évolution est comparable au passage sous Windows 3.11 des fichiers de paramétrage de type .INI à la notion de base de registre de Windows 9x et NT.

Active Directory reste avant tout une base d'annuaire dont le rôle est de regrouper tous les objets réseau au sein d'une structure hiérarchique. Dans son principe, Active Directory est une base répartie ; tous les serveurs d'un même domaine Windows 2000 disposent d'une réplique partielle de cette base.

L'AD permet alors :

- De centraliser la gestion de la forêt (y compris dans ses aspects sécurité),
- D'assurer l'authentification des utilisateurs (les données relatives à ceux-ci sont stockés dans l'AD, en particulier leurs mots de passe)
- D'autoriser la mise en place d'ACLs sur chaque objet stocké dans l'AD,
- De stocker toutes les informations de gestion d'une forêt (Stratégies de sécurité, utilisateurs, groupes, profils, certificats…)

Structure Logique de l'Active Directory

La structure logique de l'Active Directory est composée de forêts, d'arbres, de domaines, d'unités d'organisation et d'objets.

Objets

Un objet, dans Active Directory, peut être un compte d'utilisateur, une imprimante, un compte d'ordinateur, un dossier partagé publié, un groupe…

Chaque objet est composé d'un ensemble d'attributs, qui sont représentatifs de l'objet. Tous les attributs d'un objet sont définis dans ce qui est appelé le Schéma ; le schéma est modifiable pour permettre à certains objets de se voir affectés d'autres attributs.

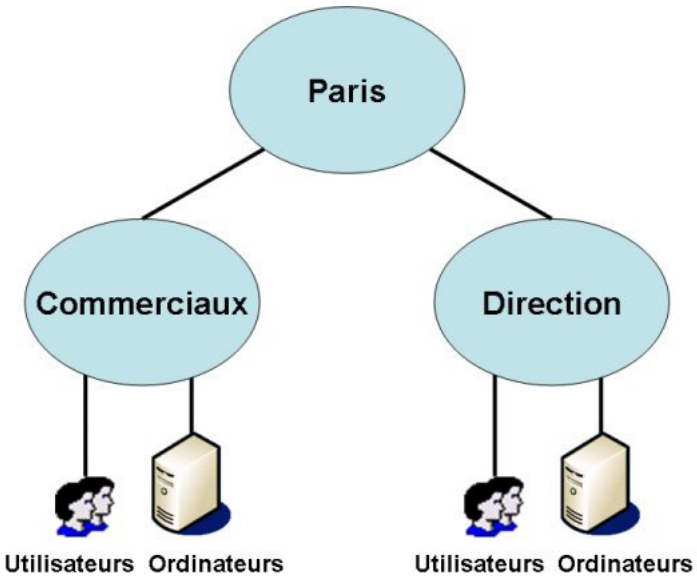
Un ensemble d'objets disposant des mêmes attributs est connu sous le nom de classe.

Unités d'organisation

Une Unité d'Organisation (OU) est un conteneur utilisé pour organiser les objets d'un domaine en groupe d'administration. Elle contient des objets tels que des comptes utilisateurs, des groupes, des ordinateurs, des imprimantes, des applications, des partages de fichiers etc.

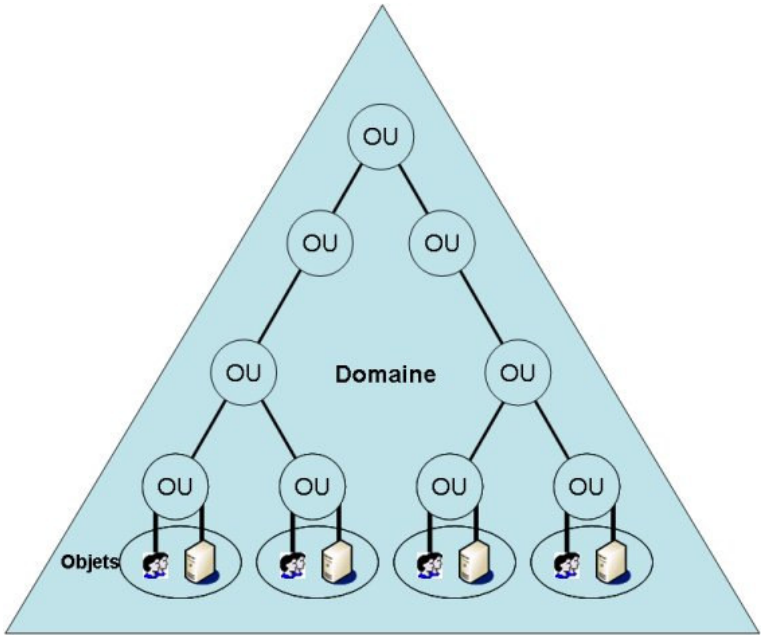
Les Unités d'organisation peuvent être structurées de façon hiérarchique.

L'exemple proposé ici précise une organisation d'OU dans laquelle on définit une OU "Paris" contenant deux OU "Commerciaux" et "Direction" :



Domaines

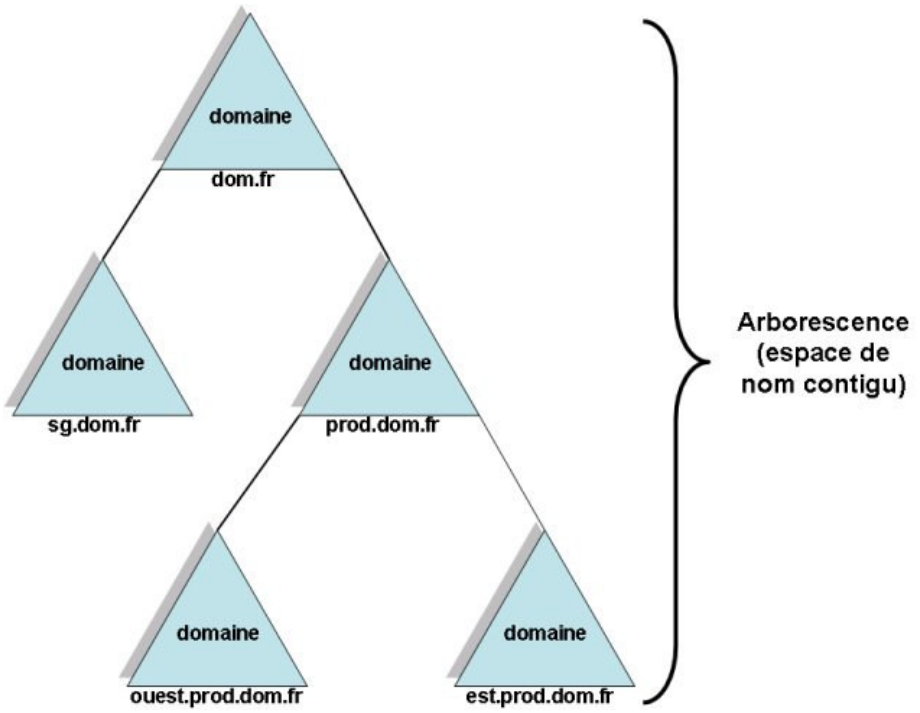
Un domaine est un regroupement logique de plusieurs OU organisés de façon hiérarchique et d'objets.



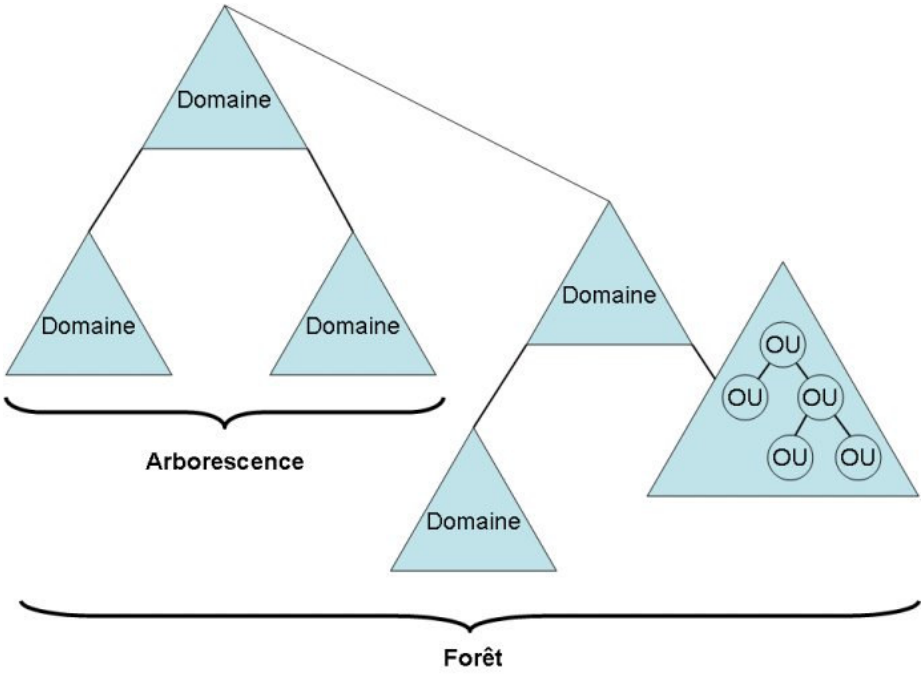
Un domaine est une limite de sécurité ; l'accès aux objets du domaine est géré par des listes à contrôles d'accès (ACLs). Toutes les stratégies et paramètres de sécurité (comme les autorisations administratives, les stratégies de sécurité et les listes à contrôle d'accès) ne passent pas d'un domaine à l'autre. L'administrateur du domaine possède des droits absolus pour définir les stratégies uniquement dans le domaine en question.

Arbres et Forêts

Une arborescence, ou arbre, est un regroupement ou une organisation hiérarchisée d'un ou de plusieurs domaines Windows 2000. Une arborescence est constituée de domaines partageant un espace de noms contigus, comme indiqué dans la figure suivante :



Une forêt est un regroupement ou une organisation hiérarchisée d'une ou de plusieurs arborescences.



L'arborescence et la forêt sont toutes deux des espaces de noms. Un espace de noms étant une zone limitée dans laquelle un nom peut être résolu. Il existe deux type d'espace de noms :

- Espace de noms contigus. Le nom de l'objet enfant dans une hiérarchie d'objet contient toujours le nom du domaine parent. Une arborescence est un espace de nom contigu.

- Espace de noms disjoints. Les noms d'un objet parent et d'un enfant de ce même objet parent ne sont pas directement liés entre eux. Une forêt est un espace de noms disjoint.

Le Catalogue Global

Active Directory est avant tout un service d'annuaire réparti : ce terme indique que les données participant à la constitution de la forêt ne sont pas toutes présentes sur chaque serveur (sauf, bien sûr si on ne définit qu'un seul contrôleur de domaine dans une architecture mono-domaine).

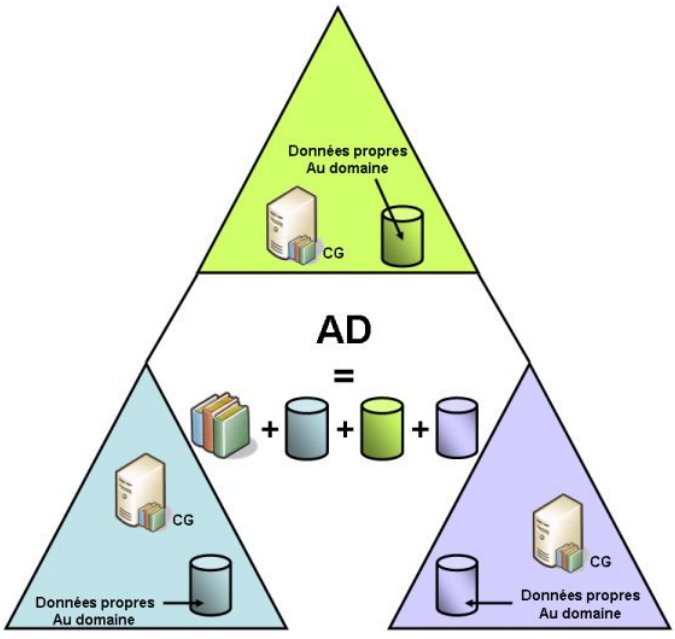
Chaque contrôleur de domaine ne dispose alors que d'une partie de l'annuaire appelée **Partition**. En particulier, les données utilisateurs ne sont présentes que sur les contrôleurs du domaine sur lesquels ils ont été définis. Rappelons que le domaine est une limite de sécurité et que, par conséquent, il est préférable que certaines informations de l'annuaire demeurent strictement dans leur domaine d'origine (clefs Kerberos des principaux, Stratégies de sécurité de domaine…).

Dès lors que les données ne sont pas complètement distribuées dans une forêt, la question de savoir comment retrouver à coup sûr un objet et ses attributs dans l'annuaire se pose de façon critique.

Cette question se trouve résolue par la notion de **Catalogue Global**. Le Catalogue Global est un listing des objets à l'intérieur d'Active Directory. Il contient assez d'information pour localiser une réplique d'une partition Active directory qui contient l'objet, sans que l'application ou l'utilisateur en faisant la demande ait besoin de savoir où se trouve physiquement l'objet dans la hiérarchie de l'Active directory.

Le Catalogue Global contient donc une réplique partielle de chaque contexte de nommage dans l'annuaire. Il est construit de façon automatique par le mécanisme de réplication de l'annuaire selon une topologie de réplication générée automatiquement mais modifiable par les administrateurs.

Dans un domaine, il existe au minimum un Contrôleur de Domaine hébergeant une réplique du Catalogue Global (généralement le premier qui a été créé).



Relations d'approbation

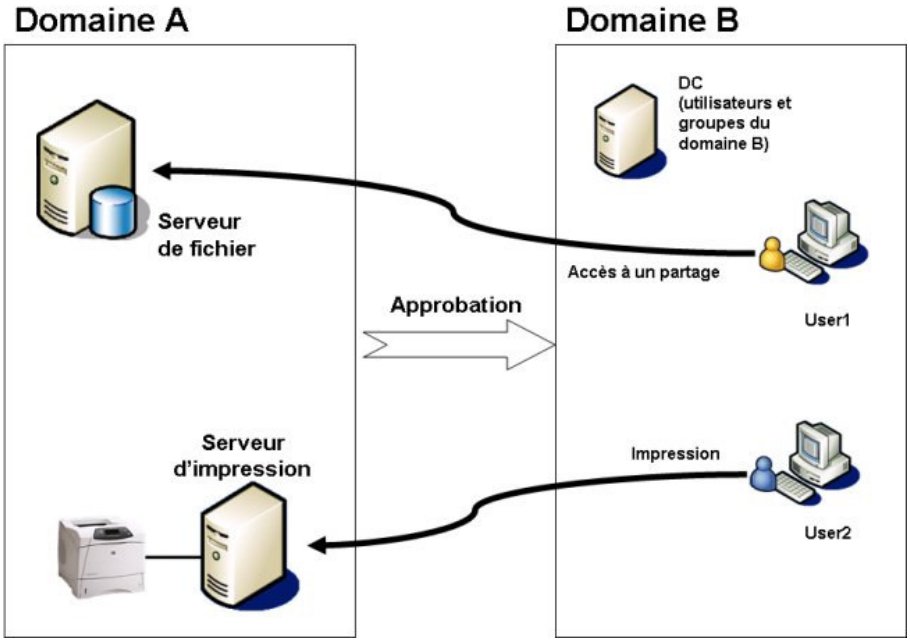
Une relation d'approbation est un lien entre deux domaines qui fait en sorte que le domaine d'approbation prend en compte les authentifications d'ouverture de session du domaine approuvé.

Ainsi, un utilisateur déclaré dans un domaine peut accéder à des ressources partagées dans un autre domaine.

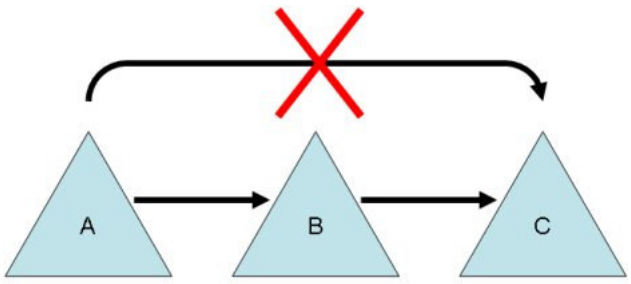
Pour établir une relation d'approbation entre deux domaines, il faut passer par le snap-in « Domaines et Approbation Active directory » de la MMC. Par nature, les relations d'approbation ne sont ni symétriques ni transitives, cependant il est possible de mettre en œuvre une relation d'approbation bidirectionnelle en faisant en sorte que deux domaines s'approuvent mutuellement. En outre, un domaine ne peut en approuver un autre que si l'administrateur du futur domaine approuvé autorise que d'autres domaines approuvent le sien.

Ainsi, si le domaine A approuve le domaine B :

- Les utilisateurs du domaine B pourront s'y connecter depuis n'importe quelle machine de A et de B,
- Les administrateurs de A pourront positionner des permissions sur les ressources du domaine A pour les utilisateurs du domaine B.



Sous Windows NT 4.0, la notion de relation d'approbation existait déjà mais ces relations étaient de type unidirectionnelles et non transitives. Ainsi, dans le cas de trois domaines A, B et C Windows NT 4.0, si A approuve B qui approuve C il est faux de dire que A approuve C (non transitivité des relation d'approbation) ou que B approuve A (approbation unidirectionnelle).

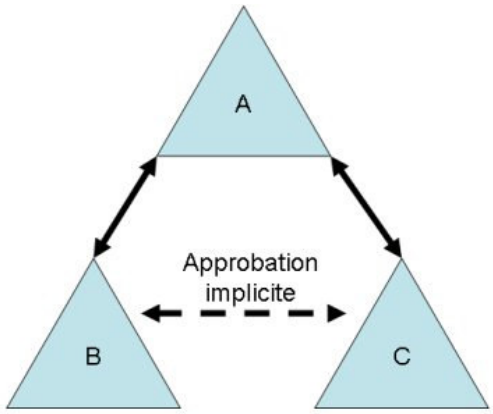


Approbations sous Windows NT 4.0

Windows 2000 est capable d'assurer la gestion de ce type de relations d'approbation si le système est dans un environnement Windows NT 4.0, mais le fonctionnement normal des

approbations entre serveurs Windows 2000 est celui de relations d'approbations transitives bidirectionnelles. Ainsi, dans le cas de trois domaines A, B et C Windows 2000, où les relations entre ces domaines sont de type Parent-Enfants, ou si ces domaines sont ceux de premier niveau d'une forêt, alors ;

- A approuve B, qui approuve A,
- B approuve C, qui approuve B,
- A approuve C, qui approuve A.



Approbations sous Windows 2000 - 2003

Les relations d'approbation existant entre les domaines d'une arborescence sont établies et gérées automatiquement. Lorsque l'on crée un domaine enfant, une relation d'approbation transitive bidirectionnelle est automatiquement établie avec le domaine parent, ce qui définit une relation d'approbation avec tous les autres domaines de l'arborescence.

Ce mécanisme d'approbation transitive bidirectionnelle est une conséquence directe de l'utilisation de Kerberos comme mécanisme natif d'authentification.

Notons enfin qu'il est possible de définir explicitement une relation d'approbation directe entre deux domaines éloignés d'une même forêt ; cette opération revient à partager une clef inter domaine Kerberos entre les contrôleurs de chaque domaine. Ce type de relation d'approbation explicite est généralement mis en place pour limiter le nombre d'échanges réseaux entre deux domaines éloignés mais échangeant souvent des données.

Relations d'approbations entre forêts

Avec les relations d'approbation implicites, transitives et bidirectionnelles, on a résolu un problème récurrent de Windows NT 4.0 : le partage des ressources entre domaines différents. Mais que se passe t-il si l'on a besoin de partager des ressources dans une forêt pour les utilisateurs d'une autres forêt ?

Hélas, sous Windows 2000, on retombe dans les travers de gestion de Windows NT 4.0. En effet, **une relation d'approbation entre deux domaines de deux forêts distinctes n'est valable que pour ces deux domaines, et cette approbation demeure monodirectionnelle et non transitive, exactement comme pour des domaines Windows NT 4.0**.

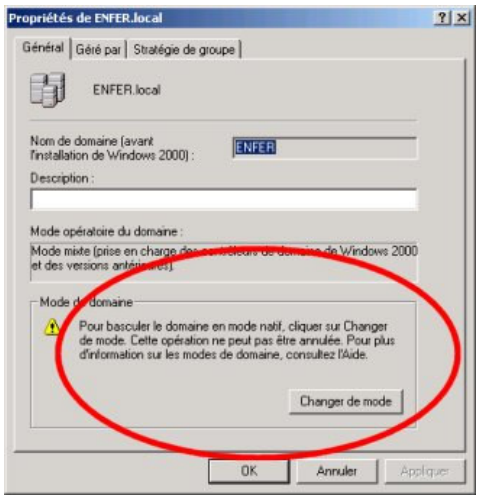
En clair, sous Windows 2000, il n'est pas possible de mettre en oeuvre une relation d'approbation entre forêts.

Précisons cependant que **cette limitation n'existe plus sous Windows 2003**.

Mode Mixte et Mode Natif

Lorsque l'on déploie un domaine Windows 2000, ce dernier peut être configuré pour fonctionner selon deux modes distincts : le mode **mixte**, qui est le mode par défaut, et le mode **natif**.

Le basculement du mode mixte au mode natif s'effectue via la console MMC « Utilisateurs et Ordinateurs Active Directory », en sélectionnant les propriétés du domaine.



Notez qu'il est impossible de revenir au mode mixte quand le mode natif a été activé. Le mode natif offre un certain nombre d'avantages dont :

- Des communications inter-serveurs utilisant systématiquement le schéma d'authentification Kerberos,
- Le support effectif du SID-History[1],
- La possibilité de gérer des groupes universels, et donc de pouvoir gérer des inclusions de groupes multiples.

En revanche, **le mode natif est résolument hostile à Windows NT 4.0** : même s'il est toujours possible, en mode natif, de conserver des postes clients Windows NT 4.0 cette option n'est pas vraiment recommandée. En effet, la bascule en mode natif impose que plus aucun serveur Windows NT 4.0 ne soit présent dans le domaine, ce qui signifie que les clients NT 4.0 Workstation devront s'authentifier sur l'unique machine du domaine disposant du rôle « Emulateur de PDC ». Enfin en cas de panne de cette machine, les clients NT 4.0 ne pourront plus s'authentifier sur le domaine puisque aucun BDC n'est alors possible dans le mode natif.

En conclusion, il est préférable de réserver le mode natif dans une configuration où tous les postes (clients et serveurs) ont migré sous Windows 2000 / 2003 / XP.

Niveaux fonctionnels de domaines et de forêts

Avec l'apparition de Windows 2003, les possibilités de gestion en domaine / forêt se sont étendues au point que Microsoft a introduit un la notion de **niveau fonctionnel,** étendant la simple notion de « mode natif / mixte ». Il est possible de définir un niveau fonctionnel au niveau d'un simple domaine, ou d'une forêt complète.

[1] Voir la description du mécanisme en annexe.

Il existe désormais **4 niveaux fonctionnels** accessibles dans un environnement Windows 2003, le changement de niveau fonctionnel s'effectuant de la même façon que le passage du mode mixte au mode natif :

- **Mode mixte** (identique au mode mixte de Windows 2000),
- **Mode natif** (identique au mode natif de Windows 2000),
- **Mode Windows 2003 Intérim** (ce mode particulier n'est à n'utiliser QUE lors d'une migration Windows NT 4.0 ⇨ Windows 2003, par ailleurs ce mode n'est disponible que dans certaines conditions),
- **Mode Windows 2003**.

Les tableaux suivants récapitulent les différentes fonctions et compatibilités des 4 niveaux fonctionnels.

Niveaux fonctionnels de domaine

Types de serveurs supportés selon le niveau fonctionnel :

Niveau fonctionnel du domaine	Types de serveurs supportés
Mode Mixte Windows 2000	Windows NT 4.0 Windows 2000 Windows 2003
Mode Natif Windows 2000	Windows 2000 Windows 2003
Mode Windows Serveur 2003 Intérim	Windows NT 4.0 Windows 2003
Mode Windows 2003	Windows 2003

Fonctions disponibles selon le niveau fonctionnel

Fonctions	Mixte	Natif	Windows 2003
Renommage des domaines. Permet de renommer un domaine.	Non	Non	**Oui**
Update Logon TimeStamp. Permet, grâce à l'ajout d'un nouvel attribut utilisateur répliqué dans le domaine, de déterminer le moment de sa dernière ouverture de session.	Non	Non	**Oui**
User Password on InetOrgPerson object. Autorise l'utilisation du mot de passe utilisateur sur l'objet InetOrgPerson (défini dans la RFC 2798, et utilisé dans certains types d'annuaires pour représenter un utilisateur au sein d'une organisation)	Non	Non	**Oui**
Groupes Universels. Permet de gérer des groupes universels (=disponibles dans toute la forêt).	**Oui** pour les groupes de distribution, Non pour les groupes de sécurité	**Oui**, pour les deux types de groupes	**Oui**, pour les deux types de groupes
Encapsulation de groupes. Permet de créer des groupes de groupes.	**Oui** pour les groupes de distribution, Non pour les groupes de sécurité, sauf pour les groupes de sécurité de domaine local qui peuvent contenir des groupes globaux	**Oui**	**Oui**

Fonctions	Mixte	Natif	Windows 2003
Conversion de groupes. Permet de convertir un groupe de diffusion en groupe de sécurité.	Non	**Oui**	**Oui**
SID History. Permet à un utilisateur de changer de domaine (voir en annexe pour le fonctionnement du mécanisme).	Non	**Oui**	**Oui**

Niveaux fonctionnels de forêt

Types de serveurs supportés selon le niveau fonctionnel :

Niveau fonctionnel de la forêt	Types de serveurs supportés
Mode Natif Windows 2000	Windows 2000 Windows 2003
Mode Windows Serveur 2003 Intérim	Windows NT 4.0 Windows 2003
Mode Windows 2003	Windows 2003

Fonctions disponibles selon le niveau fonctionnel

Fonctions	Windows 2000 Natif	Windows 2003
Améliorations de la réplication du catalogue global.	**Oui**, si les deux partenaires sont sous Windows 2003. Non dans le cas contraire	**Oui**
Désactivation de classes et d'attributs. Dans Active directory, la création d'une classe ou d'un attribut dans le schéma est définitive (pas de suppression), mais il est possible de les désactiver un objet si ceux-ci ne sont plus utilisés.	Non	**Oui**
Approbation de forêts. Permet de mettre en place des relations d'approbations entre forêts.	Non	**Oui**
Linked value replication. Permet de répliquer séparément une valeur d'un attribut multi valeurs. Par exemple, dans Windows 2000, lorsqu'un membre d'un groupe était modifié, le groupe dans son ensemble devait être répliqué. Avec ce nouveau mécanisme, seul le membre du groupe sera répliqué et pas le groupe entier.	Non	**Oui**
Renommage de domaines. Permet de renommer les domaines d'une forêt.	Non	**Oui**
Améliorations de l'algorithme de réplication de l'Active Directory.	Non	**Oui**
Dynamic Auxiliary classes. Permet de lier dynamiquement une classe à une instance d'objet (changement de clase d'un objet au cours de sa vie).	Non	**Oui**
InetOrgPerson objectClass change. Fonctionnalité présente dans la documentation Microsoft, mais non documentée…	Non	**Oui**

FSMO : des serveurs plus égaux que d'autres

On a vu que la topologie de l'Active Directory consistait en une série de serveurs n'ayant pas de relations hiérarchiques entre eux, suivant un mode d'échanges de type « égal à égal ». Certains serveurs ont cependant des rôles plus importants que d'autres, en ce sens qu'ils hébergent des services critiques : les **FSMO** (Flexible Single Master Operation) ou Maîtres d'Opérations.

En effet, certaines opérations ne peuvent être mises en œuvre de façon totalement répartie, essentiellement en raison de problèmes de synchronisation ; il est donc nécessaire d'avoir un chef d'orchestre pour ces opérations élémentaires.

Dans toute forêt Active Directory, **cinq rôles** de Maîtres d'Opérations peuvent être attribués à un ou plusieurs contrôleurs de domaine. Certains rôles sont uniques dans une forêt, d'autres sont uniques dans un domaine :

Rôle	Unicité	Description
Maître de schéma	Un par forêt	Seul le serveur qui dispose de ce rôle à la possibilité de modifier le schéma d'annuaire.
Maître d'attribution de noms de domaines	Un par forêt	Le serveur qui joue ce rôle contrôle l'ajout ou la suppression de domaines dans la forêt.
Maître RID	Un par domaine	Le maître RID alloue les séquences d'IDs dans chaque domaine. Lorsque l'on crée un utilisateur ou un groupe, c'est ce serveur qui attribue le SID de l'objet créé.
Emulateur de PDC	Un par domaine	Si il existe des systèmes Windows NT 4.0 dans le domaine, le serveur jouant le rôle d'émulateur de PDC prend en charge les authentifications en provenance de ces machines.
Maître d'infrastructure	Un par domaine	Le maître d'infrastructure contient et gère des références à des objets situés dans d'autres domaines (par exemple : un groupe extérieur auquel on a attribué une permission sur un objet). Le maître d'infrastructure est également responsable de la réplication du catalogue global.

Chaque rôle peut être transféré à un autre serveur du même domaine, soit via les outils graphiques des interfaces d'administration, soit à l'aide de l'utilitaire en ligne de commande ***ntdsutil***.

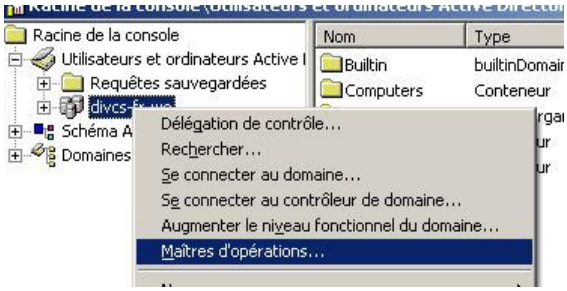
Il est important de noter que **les rôles uniques au sein d'une forêt ne sont pas transférables à un serveur appartenant à un autre domaine**, fût-il dans la même forêt.

En d'autres termes, les rôles **Maître de Schéma** et **Maître d'attribution de noms de domaine** sont toujours situés dans le premier domaine qui a été créé et **ces rôles ne peuvent être transférés dans un autre domaine**.

Afin de visualiser quel serveur dispose d'un rôle en particulier, l'interface graphique fournie avec Windows ne facilite pas le travail de l'administrateur.

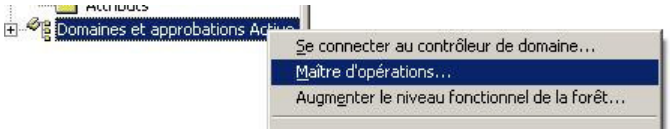
Pour visualiser (et changer) l'attribution des rôles :

- **Maître RID**, **Emulateur de PDC** et **Maître d'infrastructure** ; ouvrir le composant MMC « Utilisateurs et Ordinateurs Active Directory », effectuer un clic droit sur le domaine choisi et sélectionner l'option « Maîtres d'opérations… ».



- **Maître du schéma** ; ouvrir le composant MMC « Schéma Active Directory », effectuer un double clic sur le composant (pour ouvrir une session), effectuer un clic droit sur le composant et sélectionner l'option « Maîtres d'opérations… ».

- **Maître d'attribution de noms de domaines** : ouvrir le composant MMC « Domaines et approbations Active Directory », effectuer un clic droit sur le composant et sélectionner l'option « Maîtres d'opérations… ».



Note importante :

Le rôle de maître d'infrastructure est incompatible avec le fait de disposer d'une copie du catalogue global. En effet, le maître d'infrastructure gère des enregistrements d'annuaires particuliers appelés « phantoms » et qui sont des références à des objets d'autres domaines (des utilisateurs ou des groupes) ; si le maître d'infrastructure dispose du catalogue global, il dispose donc déjà des références des objets de toute la forêt et il n'a donc aucune raison de créer ou de mettre à jour de tels « phantoms ». Les deux seules exceptions à cette règle sont :

1) une architecture de forêt **mono-domaine** (il n'y a donc pas de références possibles à des objets d'autres domaines),

2) une architecture de forêt **multi-domaines** dans laquelle **tous** les contrôleurs d'un domaine possèdent une copie du catalogue global (le rôle de maître d'infrastructure est alors inutile dans ce domaine).

.NET

« Maintenant qu'on a liquidé le tout venant, on pourrait peut-être se risquer dans le bizarre ? »

Michel Audiard – « Les tontons flingueurs »

Le concept .NET

Lorsque l'on se pose la question de savoir ce qu'est .NET, un rapide détour sur la page officielle de Microsoft donne à peu près ceci :

Qu'est-ce que Microsoft .NET ?

Microsoft .NET est une plate-forme destinée à la création, à l'exécution et à l'utilisation de la future génération d'applications distribuées. Elle couvre les clients, les serveurs, les services et elle est constituée des éléments suivants :

- *un modèle de programmation qui permet aux développeurs de créer des applications et des Services Web (Extensible Markup Language) ;*
- *une série de Services Web tels que Microsoft .NET Mesy Services (précédemment désigné sous le nom de code " Hailstorm "), qui aident les développeurs à offrir aux utilisateurs une expérience simple et intégrée ;*
- *un ensemble de serveurs, y compris Windows® 2000, SQL Server™ et BizTalk™ Server, qui intègrent, exécutent, exploitent et gèrent des applications et des Services Web ;*
- *des logiciels clients, tels que Windows XP et Windows CE, grâce auxquels les développeurs peuvent offrir une expérience utilisateur complète et attrayante par le biais d'une gamme de périphériques ;*
- *des outils, tels que Visual Studio® .NET, qui permettent de développer des Services Web ainsi que des applications Web et Windows, pour une expérience utilisateur riche et attrayante.*

Qu'est-ce que .NET Framework ?

Il s'agit du modèle de programmation de la plate-forme .NET destiné à la création, au déploiement et à l'exécution d'applications et de Services Web. Ce modèle gère une grande partie des routines de base (désignées familièrement sous le terme de « plomberie »), de sorte que les développeurs peuvent se concentrer sur l'écriture du code de la logique métier de leurs applications. Le .NET Framework inclut le Common Language Runtime ainsi que des bibliothèques de classes.

Cette explication, parfois peu objective, a cependant du mal à faire comprendre ce que signifient les concepts cachés derrière ce nouveau concept. Tentons alors une explication plus pragmatique.

Contrairement aux idées reçues .NET n'est pas un langage ou un logiciel : .NET est en fait la nouvelle stratégie de Microsoft. .NET se présente donc comme une vision de la prochaine génération d'applications qui repose sur des standards tels que XML, HTTP, SOAP, WSDL…

Le Framework .NET est un environnement qui est distribuable gratuitement sur toutes les versions de Windows depuis Windows 95 : il s'agit très concrètement d'un ensemble de fichiers (exécutables, DLLs, fichiers de configuration) et de paramètres, téléchargeables gratuitement depuis le site de Microsoft et que l'on installe ensuite. Les applications .NET s'exécutent alors au sein de ce Framework.

Les .NET Servers sont la nouvelle génération des Serveurs Microsoft qui vont donc succéder aux Windows 2000 Servers ; Windows 2003 est ainsi le premier système d'exploitation .NET.

Pourquoi le Framework .NET ?

En réalisant le Framework .NET, Microsoft voulait tout d'abord sortir de l'enfer des technologies COM[1] .

En effet, toutes les versions COM devaient supporter les anciennes versions ce qui imposait certaines lourdeurs. De plus la communication pour accéder aux objets COM se faisait toujours sur le même port d'écoute TCP et qui plus est le port de communication réservé aux appels RPC Windows : cela ne posait pas de problème pour l'intranet de l'entreprise, mais lorsqu'une entreprise voulait utiliser un objet COM d'une autre entreprise, cela posait souvent des problèmes car la communication ne pouvait pas s'effectuer pour des raisons de sécurité (Firewall de l'entreprise).

Ce n'est pas pour autant que les technologies COM sont mortes : en effet il est tout à fait concevable de développer un objet COM puis ensuite de l'utiliser en .NET.

Pour résoudre le problème des appels RPC Windows avec les objets COM, le Framework propose une utilisation basée sur les « WebServices ». L'avantage des WebServices est que la communication entre le client et le serveur (ici le WebService) se fait via le protocole HTTP, c'est à dire via le port TCP 80 : il n'y a donc plus de problèmes de communication entre les entreprises. En outre les informations émises par le WebService sont sous format XML donc peuvent être traitées par la quasi-totalité des langages tels que le C, C++, Java, Perl, Python, PHP, Cobol… et bien sûr les langages .NET. Le fait de faire passer du XML via HTTP représente le protocole SOAP.

Ce qui fait donc la force de .NET c'est qu'il regroupe plusieurs technologies tels que COM, Applications Web (ASP.NET) , Applications Windows (Windows Form) et Applications Mobiles (Compact Framework)…

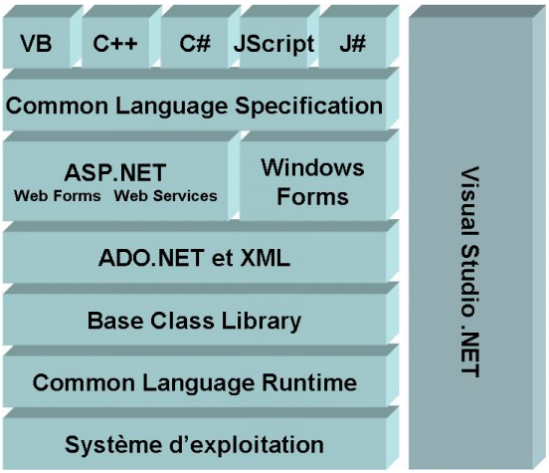
Autre avantage de taille, **la plateforme .NET est multi-langages** ce qui est une grande nouveauté par rapport à ses concurrents directs. Actuellement le Framework .NET supporte une vingtaine de langages dont le C#[2], VB.NET, J#, COBOL, Eiffel#, PERL.NET…

[1] Component Object Module
[2] Le dièse se prononce « sharp » : C-Sharp, J-Sharp, etc.

Enfin, avec le Framework .NET, Microsoft a voulu mettre fin aux nombreux problèmes causés par les DLL : il était impossible d'avoir plusieurs versions de DLL en même temps et cela posait donc de gros problème de compatibilité pour les applications : lorsque par exemple au sein d'une entreprise on changeait de version de DLL, certaines applications ne marchaient tout simplement plus car elles n'étaient pas compatibles avec la nouvelle version.

En .NET, lorsque l'on crée une application Windows ou Web, une assembly (également appelée « assemblée » dans sa – mauvaise – traduction francophone) est automatiquement créée. Une assembly est en fait le conteneur physique de classes d'un projet et donc ces classes peuvent être utilisées par plusieurs applications[1]. Et il est bien sûr possible d'avoir plusieurs versions de la même assembly.

Eléments du Framework .NET:



Contrairement aux API Windows, le Framework .NET est totalement objet : ce n'est plus en effet une simple liste de méthodes comme les API Windows. De plus les classes du Framework sont ordonnées hiérarchiquement. En effet, toutes les classes concernant la manipulation des fichiers XML se trouveront dans un namespace « System.XML » ; les classes permettant de faire de la manipulation de données seront dans le namespace « System.Data » etc.

Le Framework intègre de base des classes pour la connexion aux bases de données via ADO.NET, OLEDB, ODBC, ODBC.NET… Ces classes permettent donc de se connecter à toutes les bases de données existantes sur le marché telles que SqlServer, Oracle, Access, Sybase… Il existe aussi actuellement des drivers optimisés pour SqlServer et Oracle.

Il existe deux versions téléchargeables du Framework :

- « Framework Redistribuable» : Version qui contient les classes de bases ainsi que l'environnement d'exécution .NET.
- « Framework SDK » : Version identique à la version précédente à la différence près que la version SDK contient les compilateurs C# et VB.NET ainsi que des utilitaires. La version SDK va donc permettre de créer des applications .NET.

[1] Par analogie, on peut considérer qu'une assembly est la version .NET d'un fichier JAR sous Java.

.NET et Java

Le Framework est assez similaire à l'environnement de Java, le Java Runtime Environnement (JRE). En effet comme la JRE, le Framework dispose d'une machine virtuelle appelée la CLR (Common Language Runtime) ainsi que des classes de base mises à disposition du développeur.

Alors .NET est il un clone de Java ? Non, .NET n'est pas un clone de Java. Il reste malgré tout une riposte marketing et technique à l'approche Java/J2EE. Tout d'abord la Plateforme .NET est la seule pour l'instant à être multi langages. Cela a été rendu possible par la définition de format de type et de description standard. Cette définition s'appelle la CLS (Common Language Specification).

En terme d'architecture, la différence majeure réside dans le fait que les applications .NET ne sont pas interprétées comme pouvaient l'être les bytecodes java ; elles sont directement compilées, mais dans un langage d'assemblage « virtuel » et universel, créé de toute pièce pour les besoins du framework ; le MSIL (Microsoft Intermediate Language).

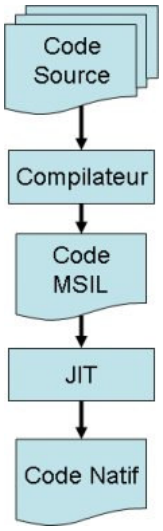
Le Common Language Runtime (CLR)

Principes

Le CLR est un des piliers du Framework .NET. Comme Sun avec Java, Microsoft avec .NET a choisi de se munir d'une machine virtuelle appelée la CLR (Common Language Runtime) et d'un code intermédiaire nommé MSIL (Microsoft Intermediate Language).

Le CLR se place juste au dessus du Système d'exploitation et c'est la CLR qui va exécuter les applications .NET puis gérer la gestion de mémoire et la sécurité de l'application.

Dans des langages traditionnels tels que le C ou le C++, le code est compilé directement dans un code natif, c'est à dire un code machine propre au processeur et à l'architecture de la machine. La différence entre ces langages et les langages .NET est qu'en .NET les langages ne sont pas compilés en langage machine, mais en code intermédiaire comme le montre le schéma ci-contre. Ce n'est qu'au moment de l'exécution de l'application que le CLR va interpréter le code intermédiaire (MSIL) en code machine via son compilateur JIT (Just In Time) : le code sera donc compilé à la volée, au moment où l'on aura besoin de l'utiliser.



Le schéma ci-contre représente de manière très simplifiée le processus de compilation et d'exécution d'une application .NET.

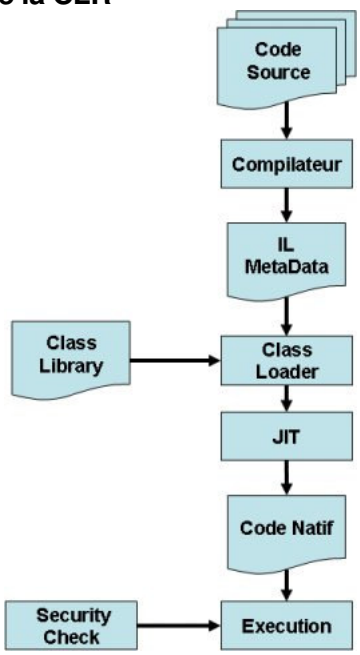
Chaque langage possède toujours son propre compilateur : pour C# le compilateur se comme « csc », pour VB.NET c'est le « vbc »… Le compilateur ne va donc pas compiler le programme en code machine mais en code intermédiaire appelé le MSIL : l'une des conséquences surprenante d'un tel fonctionnement est que deux programmes identiques, écrit dans deux langages différents, génèreront le même code MSIL (à de rares différences près)

L'autre conséquence est que tous les langages .NET ont les mêmes performances. Il n'y a donc pas un langage qui est plus performant qu'un autre.

Ce code intermédiaire donne aux langages .NET une grande portabilité, car le MSIL n'est pas propre à la plateforme ou au processeur et ce n'est que lors de l'exécution de l'application que le code intermédiaire est compilé à la volée en code machine. Microsoft nomme ce principe « Execute on Many Platforms » (ce que SUN désigne d'ailleurs sous les termes « Write Once, Run Anywhere » (WORA)).

Dans l'absolu on pourrait faire fonctionner les applications .NET sur n'importe quelle plateforme mais pour l'instant le Framework n'est implémenté que sur les versions de Windows à partir de Windows 98. Il existe des projets d'implémentation du Framework sur d'autres plateformes telles que Linux avec le projet « Mono », mais ce projet n'en est actuellement qu'à ses débuts et n'a donc pas encore implémenté toutes les classes du framework.

Fonctionnement détaillé de la CLR



Dans les faits, les compilateurs du framework ne compilent pas directement le code source en un simple fichier binaire contenant du code MSIL, mais en une « assembly ».

Une assembly est un conteneur, similaire dans son concept à une archive java (.jar), qui constitue une unité logique de déploiement d'application. Un compilateur .NET génère des assembly de deux types :

- Les assembly de type DLL
- Les assembly de type « exécutable »

Leur traitement au niveau du système est le même, la principale différence étant la présence, pour les types exécutables, d'un point d'entrée unique autorisant son exécution directe.

Une assembly se compose

- **D'un « manifest » ;**

  Le Manifest définit toutes les exigences de contrôle de version, l'auteur de l'assembly, les autorisations, et les dépendances avec les autres assembly (et pour chaque dépendance il y a le numéro de version de l'assembly, car il peut avoir plusieurs versions de la même assembly, contrairement aux DLL Windows).

- **D'un « metadata type » ;**

  Il s'agit de la définition complète de tous les types présents dans l'assembly (attributs, méthodes, paramètres, ressources…)

- **De code IL ;**

  Le langage intermédiaire. Tous les langages de programmation sont compilés en IL (également appelé ***code managé***).

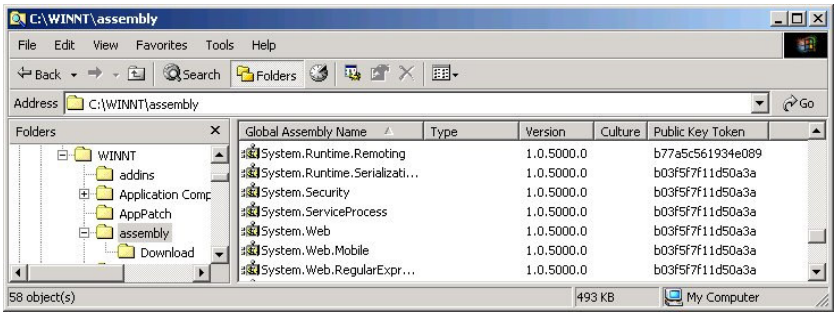
- **Et de ressources (.bmp, .jpg…)**

Lors de l'exécution de l'application, le CLR prend le relais en chargeant les types et les classes nécessaires puis fait appel au compilateur **Just In Time** (JIT) pour compiler le code MSIL en code natif. Durant l'exécution de l'application, le CLR gère la sécurité de celle-ci et décide si elle aura les droits nécessaires pour exécuter les routines voulues.

Les assembly générées par un compilateur .NET peuvent être publiques ou privées. Si elles sont privés, elles ne seront utilisées que par l'application pour laquelle elles ont été écrites, si elles sont publiques elles seront utilisables par d'autres applications.

Le GAC (Global Assembly Cache)

Les assembly publiques sont copiées dans une zone de stockage appelée le Global Assembly Cache, dans le sous répertoire « assembly » du répertoire système[1] : chaque assembly présente dans ce cache a un numéro unique et un numéro de version.



Les assembly publiques doivent nécessairement être signées. Cette opération est réalisée en phase de développement.

Le GAC demeure un mécanisme de cache, dont la création a pour but d'origine d'optimiser les accès aux classes les plus fréquemment usitées.

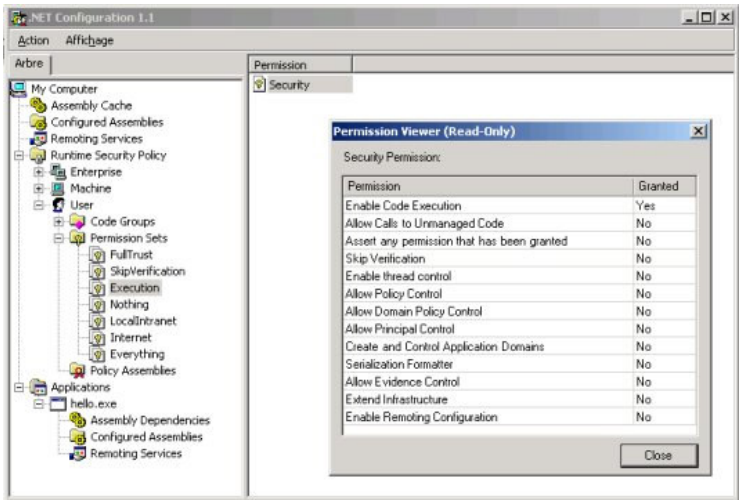
Sécurité du FrameWork

Environnement d'exécution

Nous avons vu que le CLR permettait, via son compilateur JIT, de compiler à la volée du langage intermédiaire en code natif. Mais le travail du CLR ne s'arrête pas là : le CLR gère aussi la sécurité de l'application tout au long de son exécution.

L'environnement d'exécution fournit par la CLR permet de contrôler comment les applications peuvent se comporter sur le système hôte : le développeur peut donc décider de donner des droits restreints à l'application. Par exemple on peut très bien interdire à l'application l'accès au disque dur de la machine.

[1] Chaque assembly peut être manuellement intégrée dans le GAC par simple « drag and drop », ou par la commande **gacutil**.



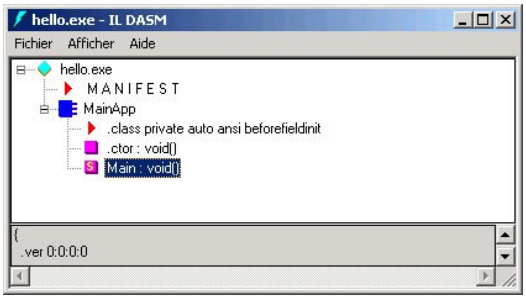
L'installation du framework .NET offre l'accès à la console d'administration du framework, dont une capture d'écran est présentée ci-dessus. Les assembly peuvent être regroupées et chaque groupe disposer de son propre jeu de permissions.

Selon l'emplacement de chargement de chaque assembly (local, Service Web…), il est également possible de définir des jeux de permissions différents.

Protection des droits d'auteur

En terme de protection du code source des applications, le framework .NET ne fait, hélas pas beaucoup mieux que l'approche « Java / Bytecode ». Tout comme il est relativement facile de faire un travail de rétro conception du bytecode Java, il est possible de remonter au code MSIL de toute application .NET ; la simplicité du langage MSIL rend par ailleurs le code beaucoup plus lisible qu'un désassemblage d'exécutable en code natif x86.

Cette opération est rendue d'autant plus facile qu'un désassembleur de MSIL (ildasm.exe) est livré avec le framework re-distribuable.



Pour illustrer la chose, les codes suivants correspondent au code source d'une application simple écrit en C# et à son désassemblage avec ildasm.exe :

```
using System;

class MainApp
{
   public static void Main()
   {
      Console.WriteLine("Bonjour !");
   }
}
```

```
.method public hidebysig static void  Main() cil managed
{
  .entrypoint
  // Code size       11 (0xb)
  .maxstack  1
  IL_0000:  ldstr      "Bonjour !"
  IL_0005:  call       void [mscorlib]System.Console::WriteLine(string)
  IL_000a:  ret
} // end of method MainApp::Main
```

Comme on peut le constater, le code MSIL est plus compréhensible par rapport à de l'assembleur x86.

Pour protéger les applications d'un possible désassemblage (c'est à dire qu'une personne puisse retrouver un code source à partir du fichier MSIL), il est possible d'utiliser des « Obfuscators » qui permettent de modifier le code intermédiaire en le rendant moins lisible.

Signature de Code

Autre mécanisme de sécurité intéressant, chaque assembly peut recevoir une signature cryptographique qui sera interprétée par le CLR lors de l'exécution de l'assembly.

La signature d'une assembly est basée sur un couple « clef publique/clef privée ». Une assembly signée ne peut être altérée sans devoir être re-signée à nouveau ; dans le cas contraire, la signature serait invalide.

Une modification d'une assembly partagée peut entraîner de graves dommages en termes de sécurité, car une assembly partagée s'exécute avec les droits du propriétaire du processus client qui l'utilise. En outre, deux sociétés différentes peuvent fournir deux assembly de même nom de fichier et de numéro de version identiques ; dans ce cas le système serait incapable de déterminer quelle assembly doit être chargée.

Ces deux problèmes sont résolus par le mécanisme de signature : également appelé « **nom fort**[1] », cette signature, générée avec l'utilitaire *sn.exe*, figure dans le manifeste de l'assembly du client. Le nom fort est calculé à partir du contenu de l'assembly. Le client connaissant le nom fort ne peut donc pas charger une assembly partagée autre que celle qu'il désire car il doit pouvoir réaliser une vérification de la signature (vérification du nom fort).

SignCode

Le mécanisme de noms forts n'est, hélas, pas associé à un niveau de confiance ; il ne sert qu'à assurer l'unicité de nom d'une assembly et à empêcher l'usurpation de nom.

Pour associer un niveau de confiance à une signature électronique de code, il est nécessaire d'employer un autre mécanisme au travers d'un nouvel utilitaire : *signcode.exe*.

Cet outil appose une signature cryptographique directement dans l'en-tête binaire (PE – Portable Executable) de l'assembly (et non pas dans son manifeste !) à partir d'un certificat d'éditeur délivré par une autorité de certification tierce.

Il est possible d'utiliser les deux techniques de signature indépendamment ou conjointement, tout en sachant que seul le SignCode permet d'associer à cette signature un niveau de sécurité.

[1] Strong Name

Gestion de la sécurité du système

« Les ordinateurs sont inutiles, ils ne savent que donner des réponses »

Pablo Picasso

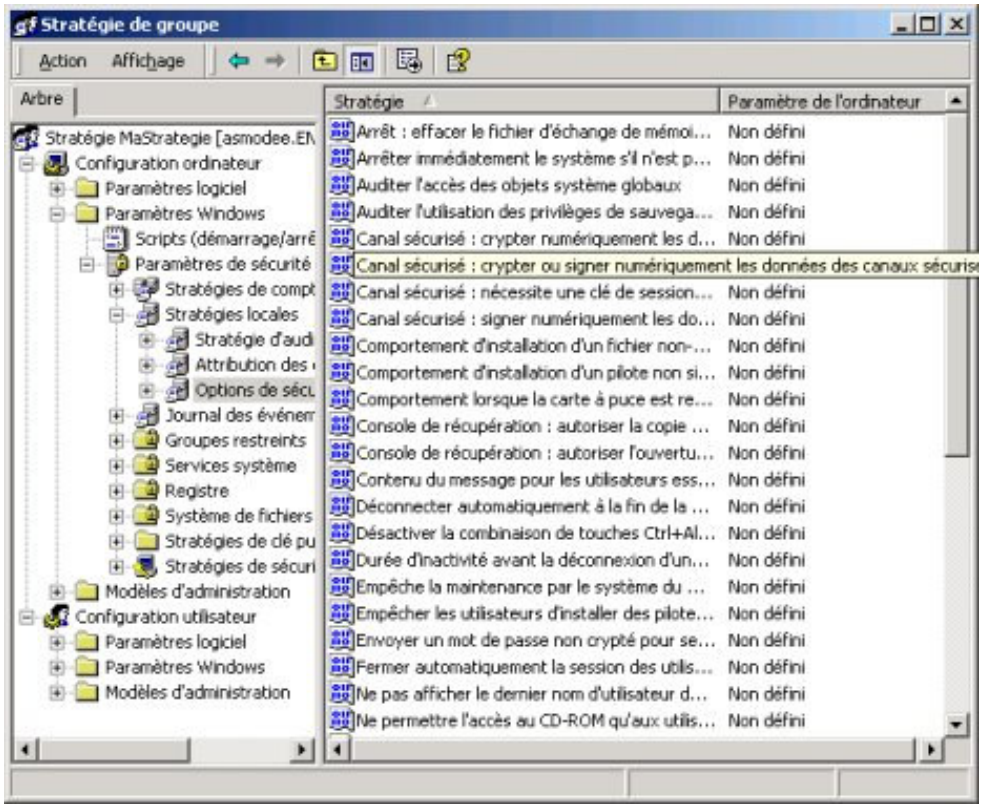
Les Stratégies de groupes (GPO)

Avec Windows 2000 est apparue la notion de « stratégies de groupes » (**GPO - Group Policy Object**) connue sous une autre forme sous Windows NT 4.0, avec l'éditeur de stratégies systèmes (Poledit.exe). Les « stratégies systèmes » permettaient d'associer un Ordinateur, un Groupe ou un Utilisateur, à un ensemble de valeurs de clefs dans la base des registres pour, entre autres, restreindre l'utilisation du système. Lors de l'ouverture de session, la « stratégie système » était chargée depuis un contrôleur de domaine vers la station hôte et appliquée tout le temps que durait la session utilisateur.

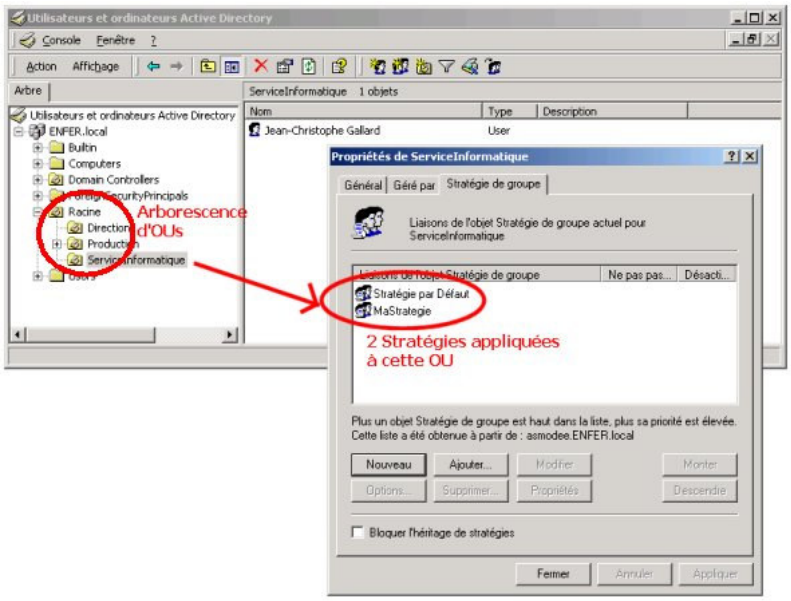
Au sein de Windows 2000 la « Stratégie de Groupe » représente alors une généralisation de la notion de « Stratégie Système » de Windows NT 4.0.

Le champ d'action des fonctionnalités de « Stratégie de Groupe » reste cependant plus vaste que ce qui se faisait sous Windows NT :

- Attribution des droits utilisateurs,
- ACLs sur les fichiers/répertoires et les entrées de registre,
- Stratégie de sécurité (options, mots de passe…),
- Etat des services,
- Installation/désinstallation automatique de logiciels (une seule contrainte : le setup d'installation doit impérativement être un package MSI – Microsoft Installer),
- Scripts d'ouverture ET de fermeture de session
- …



Sous Windows 2000, les objets GPO sont stockés dans Active Directory et ce n'est plus Poledit (l'éditeur de stratégies) qui est utilisé pour les gérer mais un "snap-in" de la MMC. De plus, alors que les « stratégies systèmes » étaient affectées à des groupes ou à des ordinateurs, les « Stratégies de Groupes » sont affectées à des conteneurs Active Directory particuliers[1] appelés Unités Organisationnelles (UO, ou Organisation Unit – OU). Enfin, sous Windows 2000, ces stratégies peuvent être modifiées « à chaud », puisqu'elles sont automatiquement rafraîchies toutes les 90 minutes[2].



[1] Voir le chapitre consacré à l'Active Directory.
[2] Sous Windows NT 4.0, les « stratégies systèmes » modifiées n'étaient prises en compte que lorsque les utilisateurs fermaient puis ré-ouvraient une session interactive.

Des Stratégies de Groupes peuvent être appliquées sur différents niveaux. L'ordre d'application des différentes stratégies est réalisé selon une méthode hiérarchique, comme suit :

- Les stratégies au niveau du site sont d'abord appliquées,
- Puis ensuite celles au niveau du domaine,
- Et enfin celles au niveau des OUs (Organisation Unit).

Il y a cumul des stratégies tant qu'aucun conflit n'intervient. S'il y a conflit, c'est alors la stratégie la plus prêt de l'utilisateur qui est alors appliquée (donc celle qui se trouve au niveau de la dernière OU d'appartenance de l'utilisateur). Dans le cas où plusieurs stratégies sont définies pour le même conteneur, les stratégies sont alors lues dans l'ordre de la liste, du bas vers le haut.

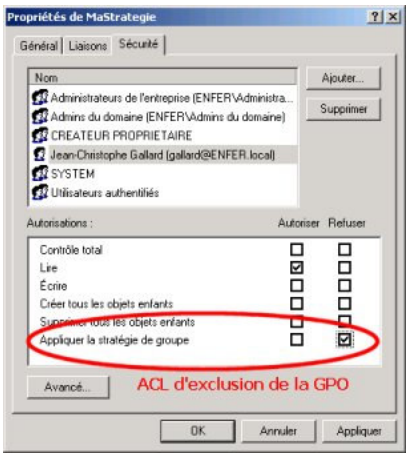
Une différence de taille apparaît lorsque l'on compare les GPO Windows 2000 avec les Stratégies système de NT 4.0 : alors que sous Poledit, les Stratégies étaient applicables à des utilisateurs et / ou des groupes, **les GPO ne s'appliquent qu'aux Unités Organisationnelles.**

Ce point qui semble compliquer la tâche d'administration ne fait dans les faits que la simplifier. En effet, lorsque l'on change un utilisateur ou un ordinateur d'unité organisationnelle, la stratégie de groupe de cette UO lui est directement appliquée sans qu'une modification des stratégies n'intervienne. En outre, il n'est plus nécessaire de créer spécifiquement des groupes pour les stratégies systèmes.

Astuce :

Si il devient malgré tout nécessaire d'appliquer une stratégie à une OU sauf à l'un de ses membres, il est cependant possible de contourner la difficulté autrement qu'en créant une OU spécifique ne contenant qu'un seul utilisateur ou en créant une hiérarchie complexe d'OUs. Pour cela, il faut rappeler que les OUs sont des objets de l'Active Directory et qu'en conséquence, ces objets sont par nature sécurisables, c'est-à-dire qu'ils ont une ACL.

Il devient donc possible de modifier l'ACL existante de façon à exclure l'utilisateur concerné de la stratégie de groupe.



Structure d'une stratégie de groupe

Les paramètres de configuration d'une stratégie de groupe sont stockés à deux endroits :

- les objets Stratégie de groupe, situés dans l'annuaire Active Directory ;
- les fichiers modèles de sécurité, situés dans l'arborescence de fichiers du système.

Les modèles de sécurité par défaut de Windows 2000 sont stockés sous forme de fichiers *.inf* dans le dossier `%SystemRoot%\Security\Templates`. Windows 2000 est livré avec quelques modèles de sécurité. Les modèles suivants sont applicables dans un environnement à faible sécurité.

- *Basicwk.inf* – pour Windows 2000 Professionnel.
- *Basicsv.inf* – pour Windows 2000 Serveur.
- *Basicdc.inf* – pour les contrôleurs de domaine Windows 2000.

Pour implémenter une sécurité plus élevée sur des ordinateurs Windows 2000, des modèles supplémentaires sont fournis. Ils apportent des paramètres de sécurité supplémentaires aux modèles de base :

- Securedc.inf et Hisecdc.inf – pour les contrôleurs de domaine.
- Securews.inf et Hisecws.inf – pour les serveurs et les stations de travail membres.

Ces modèles sont dits **incrémentiels** car avant de les ajouter, **il faut d'abord appliquer les modèles de base**.

Format des modèles de sécurité

Les fichiers modèles sont au format texte. Il est possible de modifier ces modèles soit directement au travers du composant logiciel enfichable MMC « *Modèles de sécurité* » ou à l'aide d'un éditeur de texte classique. Le tableau suivant montre la correspondance entre les sections de la stratégie et celles des fichiers modèles.

Section de la stratégie	Section du modèle
Stratégie de compte	[System Access]
Stratégie d'audit	[System Log] [Security Log] [Application Log]
Droits des utilisateurs	[Privilege Rights]
Options de sécurité	[Registry Values]
Journal des événements	[Event Audit]
Groupes restreints	[Group Membership]
Services système	[Service General Setting]
Registre	[Registry Keys]
Système de fichiers	[File Security]

Certaines sections du fichier modèle de sécurité, telles que [File Security] et [Registry Keys], contiennent des listes de contrôle d'accès spécifiques. Ces listes sont des chaînes de texte, définies par le langage SDDL (Security Descriptor Definition Language). Pour plus d'informations sur la modification des modèles de sécurité et sur le langage SDDL, consulter le site MSDN.

L'outil d'Analyse de la Sécurité

L'outil d'analyse de la sécurité prend ses origines dans le CD-Rom du Service Pack 4 de Windows NT 4.0 où il était déjà présent sous une forme embryonnaire. L'outil est disponible sur toute machine Windows 2000 et se présente sous la forme d'une extension de la MMC appelée « Configuration et analyse de la sécurité ».

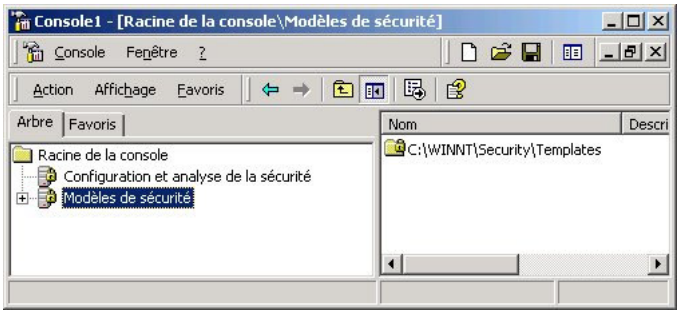
Cet outil répond à une double demande puisqu'il permet :

- D'analyser une configuration et de comparer le résultat avec un modèle de sécurité souhaité,
- D'appliquer un modèle de sécurité prédéfini ou personnalisé.

Nous présenterons ici la démarche générale nécessaire pour réaliser une analyse du système et pour appliquer un modèle personnalisé.

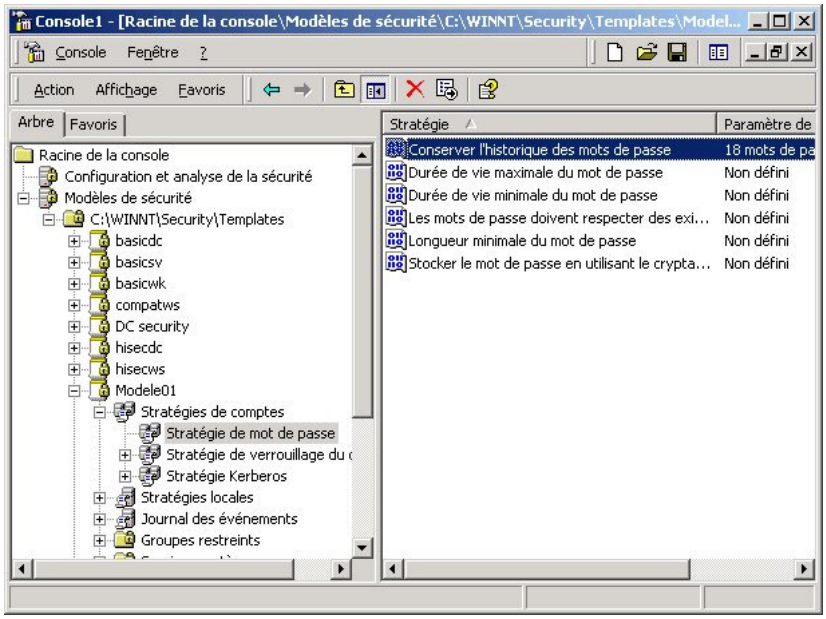
Création d'un modèle personnalisé

Lancer une MMC vierge (commande « mmc.exe »), et y insérer les deux composants « Configuration et analyse de la sécurité » et « Modèles de sécurité » : Menu « console / Ajout Supprimer une composant logiciel enfichable ».



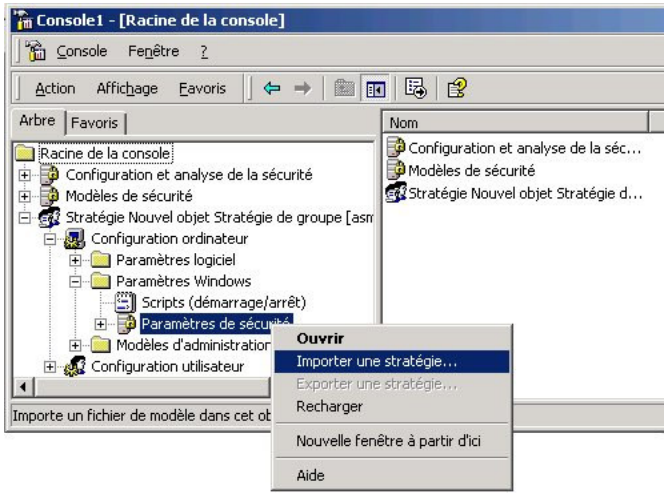
Dans le composant « Modèles de sécurité », sélectionner l'un des modèles proposés. Effectuer un clic bouton droit et sélectionner « enregistrer sous… » Choisir un nom de modèle, puis sélectionner ce modèle. Cette opération a pour but de copier un modèle existant.

Dérouler les options et positionner les paramètres souhaités.



Ne pas oublier d'enregistrer les modifications apportées au modèle (clic-droit + enregistrer).

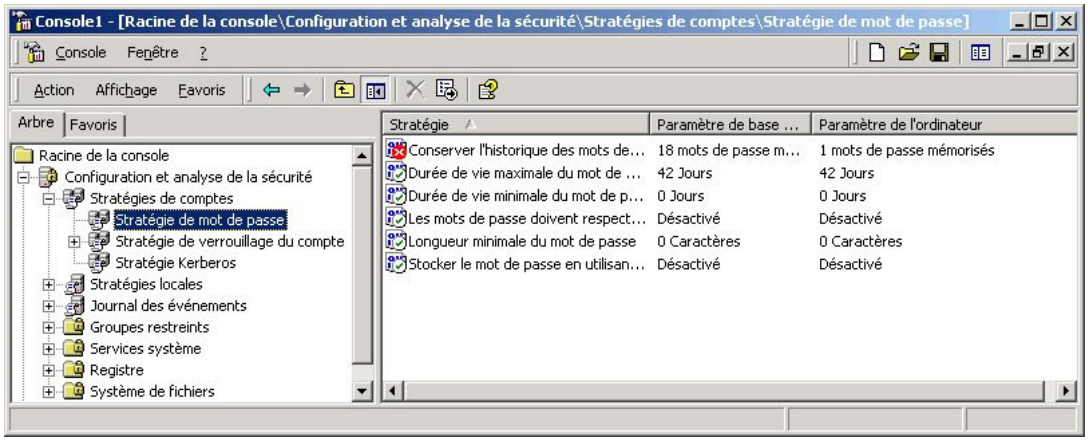
Nota : ces modèles de sécurités peuvent être par la suite importés dans une stratégie de groupe sous l'item « Paramètres de Sécurité / importer une stratégie… ».



Analyse et application de la sécurité

Dans le composant « Configuration et analyse de la sécurité », créer une nouvelle base de données (clic-droit + ouvrir base), et sélectionner le modèle préalablement créé.

Réaliser l'analyse de la sécurité (clic-droit + Analyser la configuration maintenant) et patienter le temps nécessaire à la comparaison entre l'état actuel du système et celui spécifié par le modèle : les écarts sont visualisables par une icône spécifique.



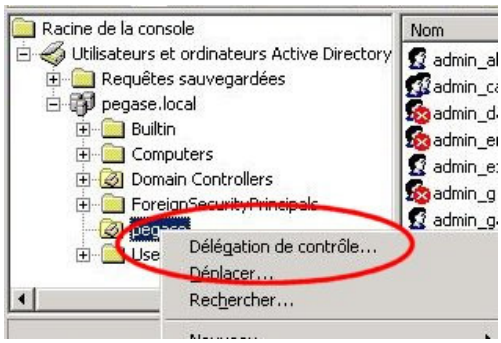
A ce stade, il est possible de lancer la configuration automatique de la machine, configuration spécifiée dans le modèle utilisé.

Suite à une analyse de la sécurité, il est également possible d'exporter la configuration actuelle sous la forme d'un modèle de sécurité réutilisable, ce qui permet de mettre en place une configuration de sécurité de référence et d'appliquer de façon automatique cette configuration sur un grand nombre de machines.

Délégation de l'Administration

Puisque avec Windows 2000, tous les paramètres de sécurité relatifs à l'administration du système sont désormais stockés dans l'Active Directory, et que les objets de l'Active Directory sont sécurisables, il est devenu possible de mettre en œuvre une forme de délégation de pouvoir en jouant sur les ACLs des objets de l'AD.

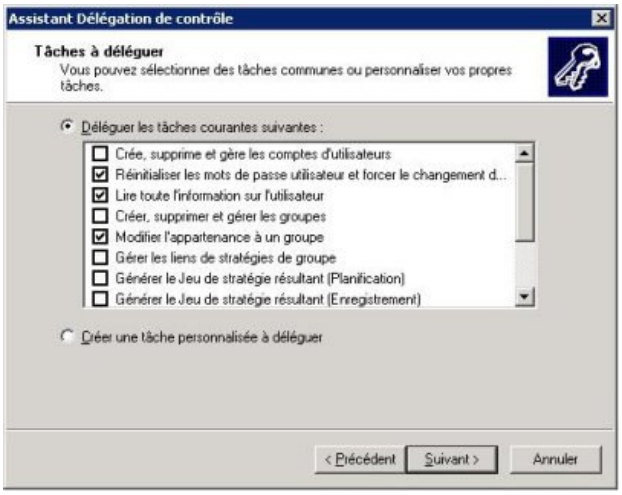
Il s'agit ici de ce que Microsoft appelle la « **Délégation d'Administration** » ou encore « **Délégation de contrôle** ». Ce mécanisme est accessible au travers du composant de la MMC « Utilisateurs et Ordinateurs Active Directory ». Sous ce composant, chaque objet de l'AD peut se voir affecter une délégation de contrôle (clic-droit, Délégation de contrôle).



Un « wizard » prend alors la main et guide l'administrateur dans la configuration du contrôle :

- Choix des utilisateurs et/ou des groupes à qui confier cette délégation,
- Types d'actions à déléguer (dépend du type d'objet sur lequel on agit).

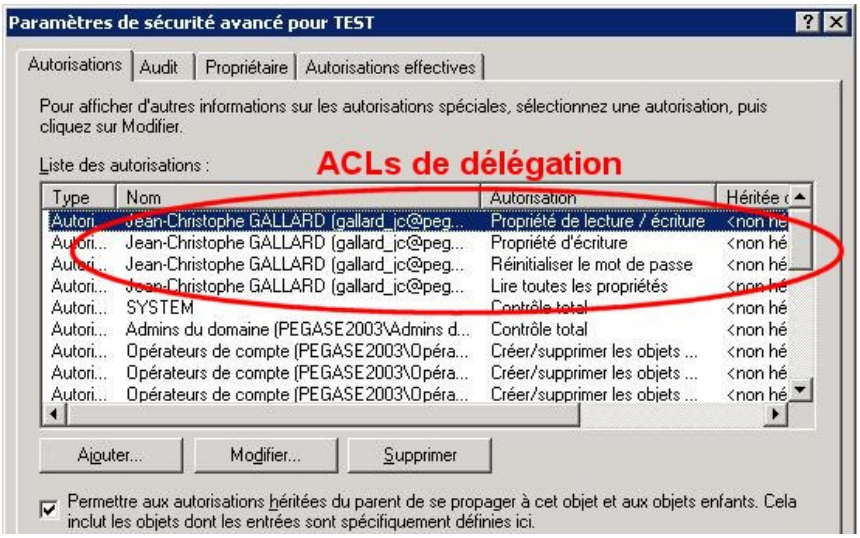
Il est ainsi possible de déléguer la création et la suppression des comptes utilisateurs à un autre utilisateur, voire de donner les pleins pouvoirs à un utilisateur mais uniquement pour l'ensemble des objets situées sous une Unité d'Organisation donnée, etc.



Concrètement, il ne s'agit ici que d'un habillage (plus ou moins réussi, et d'ailleurs plutôt moins que plus…) d'opérations de modifications d'ACLs élémentaires sur les objets de l'AD. Lorsque l'on crée une délégation, le système change les ACLs par défaut des objets de l'AD et de leurs attributs, afin de donner les privilèges nécessaires aux utilisateurs concernés.

Ce mécanisme peut s'avérer précieux dans le cas ou l'on souhaiterait une administration de foret ou de domaines décentralisée (typiquement ; un domaine géré par des Administrateurs tout puissants et des unités d'organisation au sein desquelles se trouveraient des administrateurs locaux ayant les privilèges pour gérer ces unités d'organisation).

Cependant, le fonctionnement de cette délégation souffre d'un inconvénient majeur : **une fois la délégation appliquée, aucun outil ne permet de visualiser cette délégation**. Seule une analyse approfondie des ACLs des objets de l'Active Directory pourra confirmer qu'une délégation de contrôle a bien été appliquée.



Un outil en ligne de commande, à la diffusion relativement confidentielle mais cependant téléchargeable depuis fin 2004 sur le site Internet de Microsoft, offre des possibilités d'analyse intéressantes : **DSREVOKE.EXE**.

L'outil offre la possibilité de lister des ACLs de délégation sur un objet de l'Active Directory, mais il permet surtout de révoquer une délégation préalablement accordée ; techniquement les fonctionnalités offertes sont très loin d'être parfaites, mais l'utilitaire à au moins l'avantage d'être disponible.

Sécurisation de la pile TCP/IP – IPSEC

Concepts

IPSEC est un standard de l'IETF qui définit une extension de sécurité pour le protocole IP afin de permettre la sécurisation des données échangées sur les réseaux basés sur ce protocole. Basé sur des mécanismes cryptographiques, IPSEC s'insère dans la pile protocolaire TCP / IP au niveau d'IP. Cela signifie qu'il agit sur chaque paquet émis ou reçu et peut soit le laisser passer sans traitement particulier, soit le rejeter, soit lui appliquer un mécanisme de sécurisation.

Les services de sécurité d'IPSEC sont fournis au travers de deux extensions du protocole IP appelées **AH** (Authentication Header) et **ESP** (Encapsulating Security Payload).

- **Authentication Header**

  AH est conçu pour assurer l'authenticité des paquets IP sans chiffrement des données. Le principe d'AH est d'adjoindre aux paquets IP un champ supplémentaire permettant à la réception de vérifier l'authenticité des données. Un numéro de séquence permet de détecter les tentatives de rejeu.

- **Encapsulating Security Payload**

  ESP a pour rôle premier d'assurer la confidentialité des données mais peut aussi être utilisé pour assurer l'authenticité de celles-ci. Le principe d'ESP consiste à encapsuler dans un nouveau paquet IP le paquet d'origine mais sous une forme chiffrée. L'authenticité des données peut être obtenue par l'ajout d'un bloc d'authentification et la protection contre le rejeu par celui d'un numéro de séquence.

Ces deux services peuvent être utilisés séparément ou conjointement afin d'obtenir les services de sécurité requis. Ces services ne sont pas restreints à un algorithme de chiffrement particulier ; en théorie, n'importe quel algorithme de chiffrement peut être employé, sous réserve que les équipements en communication disposent d'au moins un algorithme en commun. IPSEC comporte une liste d'algorithmes proposés pour être utilisés avec IPsec et dont l'utilisation est négociable en ligne par le biais du protocole IKE (CAST-128, BlowFish, RC5, DES, triple DES).

Pour garantir l'interopérabilité entre les équipements, le standard IPSEC rend certains de ces algorithmes obligatoires. Actuellement, DES-CBC et 3DES-CBC sont obligatoires pour le chiffrement ; pour l'authentification, HMAC-MD5 et HMAC-SHA-1 doivent être présents dans toute implémentation conforme d'IPSEC.

D'autre part, pour chacune des extensions IPSEC, deux modes de protection existent :

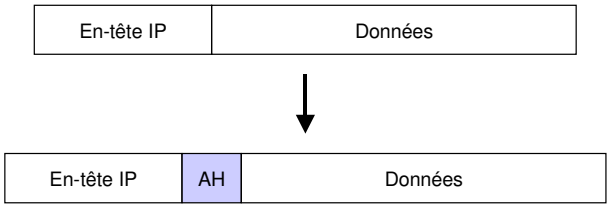
- Le **mode transport** protège uniquement le contenu du paquet IP sans toucher à l'en-tête ; ce mode n'est utilisable que sur les équipements terminaux (postes clients, serveurs).

- Le **mode tunnel** permet la création de tunnels par « encapsulation » de chaque paquet IP dans un nouveau paquet. Ainsi, la protection porte sur tous les champs des paquets IP arrivant à l'entrée d'un tunnel, y compris sur les champs des en-têtes (adresses source et destination par exemple). Ce mode est celui utilisé par les équipements réseau (routeurs, pare-feux...).

Authentication Header

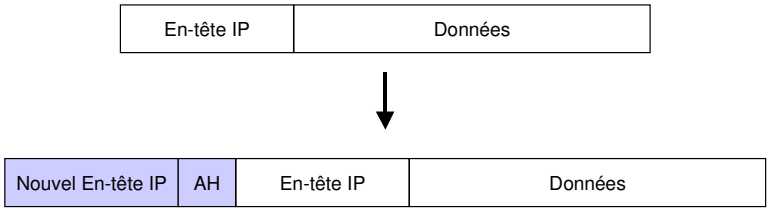
Mode transport :

Un en-tête AH est inséré entre l'en-tête IP et les données du paquet.



Mode tunnel :

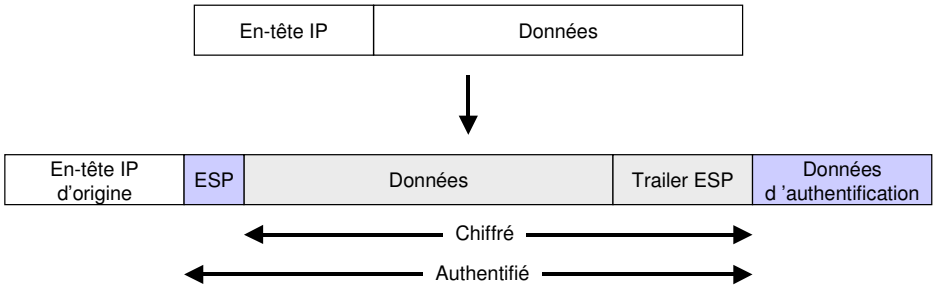
Le paquet d'origine est encapsulé dans le champ de DATA d'un nouveau paquet, possédant son propre en-tête, et auquel on adjoint un AH.



Encapsulating Security Payload

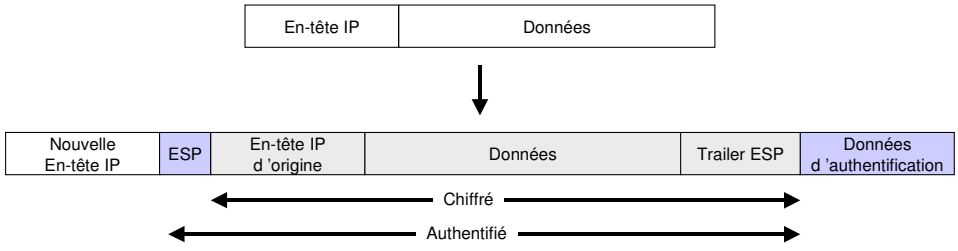
Mode transport :

On conserve l'en-tête IP d'origine, auquel on ajoute un en-tête ESP suivi du champ de DATA du paquet d'origine sous forme chiffrée et d'un trailer ESP[1], puis on complète le paquet avec les données d'authentification (ce champ n'est présent que si l'option d'authentification a été sélectionnée).



Mode tunnel :

On chiffre intégralement le paquet d'origine suivi d'un trailer ESP, puis on insère ce flux dans un nouveau paquet disposant de son propre en-tête, suivi d'un en-tête ESP et se terminant par des données d'authentification (ce champ n'est présent que si l'option d'authentification a été sélectionnée).



[1] Le « trailer ESP » contient éventuellement des octets de bourrage, la taille des octets de bourrages et un pointeur sur l'en-tête suivant

Implémentation sous Windows

Toute machine sous Windows 2000, 2003 ou XP, dispose d'une interface d'administration des stratégies IPSEC, sous forme d'un « snap-in » de la MMC, appelée « Gestion de la stratégie de sécurité du protocole IP » pour définir des stratégies IPSEC pour des ordinateurs.

Les stratégies IPSEC peuvent être appliquées à des ordinateurs, des sites, des domaines ou à toute unité d'organisation créée dans Active Directory. A travers l'utilisation d'actions de sécurité appelées ***règles***, une stratégie peut être appliquée à des groupes de sécurité hétérogènes d'ordinateurs ou d'unités d'organisation.

Il existe deux emplacements de stockage pour les stratégies IPSEC :

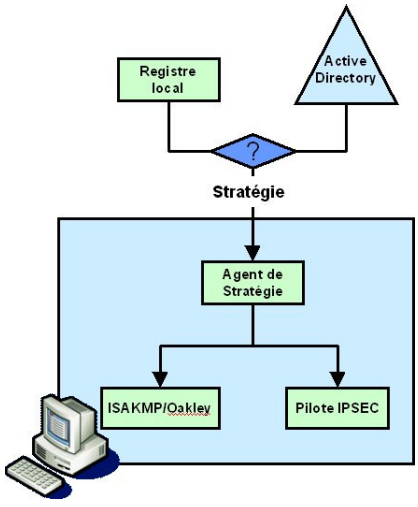
- **L'Active Directory** pour les ordinateurs en domaine,
- **Le Registre local** pour les ordinateurs autonomes et les ordinateurs qui ne sont pas rattachés au domaine (quand l'ordinateur n'est temporairement pas rattaché à un domaine Microsoft Windows 2000 approuvé, les informations de la stratégie sont mises en mémoire cache dans le Registre local).

Chaque stratégie peut être déployée au travers d'une ou plusieurs GPOs.

IPSEC se sert essentiellement de 2 composants pour son fonctionnement :

L'agent de stratégie IPSEC

L'agent de stratégie IPSEC est implanté sur chaque poste Windows 2000 sous la forme d'un service (sous Windows XP, le service prend le nom de « Services IPSEC »).



L'agent de stratégie effectue les tâches suivantes :

- Il extrait la stratégie IPSEC appropriée (si une stratégie a été attribuée) de l'Active Directory si l'ordinateur est un membre de domaine ou du Registre local si l'ordinateur n'est pas rattaché à un domaine,
- Il initialise la négociation des clefs et la mise en place des associations de sécurité entre un client et un serveur,
- Il envoie les informations de la stratégie IPSEC active au pilote IPSEC.

La recherche de la stratégie se fait au **démarrage** du système, à l'intervalle spécifié dans la stratégie IPSEC (si l'ordinateur est rattaché à un domaine), et à l'intervalle d'interrogation par défaut du Winlogon (s'il est rattaché à un domaine).

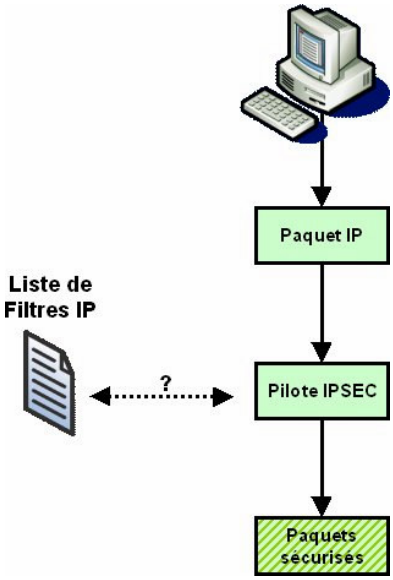
Le pilote IPSEC

Le cœur de l'implémentation Microsoft de IPSEC réside dans un pilote de périphérique, matérialisé par le fichier IPSEC.SYS.

Le pilote a pour but de contrôler les paquets IP entrants et sortant, et d'effectuer les vérifications avec les stratégies de sécurité locale et les filtres IP mis en place.

Le pilote IPSEC s'insère dans la pile IP et traite les paquets :

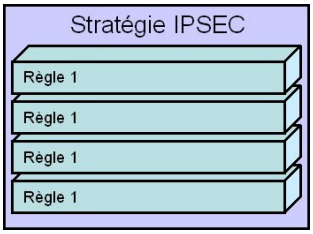
- entrants ; **avant** leur prise en charge par la pile IP
- sortants ; **après** leur formation par la pile IP.



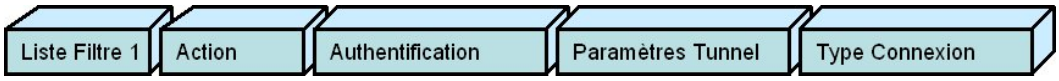
Structure d'une stratégie IPSEC

Pour une même machine, il est possible de définir autant de stratégies IPSEC que nécessaires, mais une et une seule stratégie est appliqué à un instant donné.

Une stratégie IPSEC est constituée d'un ensemble de règles.



Chaque règle contient une « liste de filtres IP » et un ensemble d'actions de sécurité qui sont réalisées selon des critères propres à chaque règle. A une liste de filtre, on fait alors correspondre une action, une méthode d'authentification, d'éventuels paramètres de tunnel, et les types de connexions (locale, distantes, les deux) à laquelle s'applique la règle.



Le champ « action » d'une règle peut être défini par l'administrateur selon 3 méthodes élémentaires :

- **Autoriser** : le flux décrit par le filtre est accepté.
- **Refuser** : le flux décrit par le filtre sera rejeté.
- **Négocier la sécurité** : le flux sera traité par le pilote IPSEC selon une politique à définir (chiffrement, intégrité, les deux, autorisation ou non des clients non IPSEC, type d'algorithmes cryptographiques utilisé…)

On voit donc ici que les fonctionnalités IPSEC de Windows ne se limitent pas à un simple support des technologies IPSEC : en jouant sur les valeurs « autoriser » et « refuser » il est alors possible de définir des règles de filtrage IP, comme le ferait un Firewall.

Exemple de Stratégie IPSEC

HTTP	Bloquer	(Kerberos)	Aucun	Toutes
ICMP	Autoriser	(Kerberos)	Aucun	Toutes
FTP	Demander	Clef Partagée	Aucun	Réseau Local
SMTP	Secu Requise	Clef Partagée	135.52.52.100	Réseau Local
POP	Secu Requise	Clef Partagée	Aucun	Accès Distant

La console d'administration IPSEC ne propose pas directement ces trois méthodes ; elle définit des actions types qui encapsulent ces méthodes. Par défaut, il existe trois méthodes type (et il est possible d'en définir de nouvelles : par exemple, il est recommandé de se définir une action « Blocage » correspondant à l'action élémentaire « Refuser ») :

- « **Autoriser** » : correspond à l'action élémentaire d'autorisation
- « **Demander la sécurité (optionnel)** » : correspond à une négociation de sécurité dans laquelle on accepte les connexions non sécurisées si la station distante ne gère pas l'IPSEC
- « **Sécurité requise** » : correspond à une négociation de sécurité imposant l'utilisation d'IPSEC (une station distante non IPSEC ne pourra donc pas réaliser de connexion)

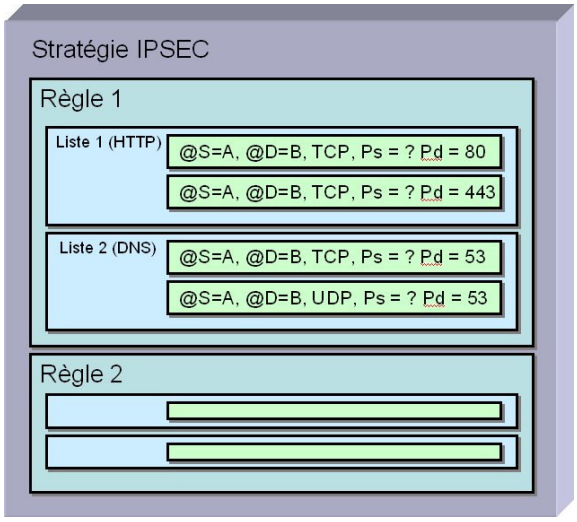
Une liste de filtre est constituée de un ou plusieurs filtres élémentaires. Chaque filtre est défini par un ensemble d'attributs définissant un flux élémentaire.

Un filtre contient les paramètres suivants :

- **L'adresse de la source et de la destination du paquet IP**. Celles-ci peuvent être configurées selon un niveau qui va de très détaillé, comme une adresse IP unique, à global, englobant alors la totalité d'un sous réseau ou d'un réseau.

- **Le protocole selon lequel le paquet va être transmis**. Celui-ci couvre par défaut tous les protocoles de la suite de protocoles TCP/IP. Le filtre peut être configuré avec un niveau de protocole individuel pour satisfaire des besoins particuliers, comme des numéros de protocoles personnalisés.

- **Le port du protocole de la source et de la destination pour TCP et UDP**. Ceci couvre également par défaut tous les ports, mais peut être configuré pour être appliqué seulement aux paquets envoyés ou reçus sur un port de protocole spécifique.



Contenu d'une stratégie IPSEC

L'ordre dans lequel les filtres s'appliquent n'est pas lié à l'ordre de leur affichage quand la stratégie IPSEC est visualisée. Tous les filtres sont extraits simultanément par l'agent de stratégie IPSEC lors du démarrage du système, et sont traités et **triés du plus spécifique au moins spécifique**. Il n'est pas garanti qu'un filtre spécifique sera appliqué avant un filtre général jusqu'à ce que tous les filtres aient été traités ; ceci peut affecter certains comportements des communications lors du démarrage du système.

Gestion des Correctifs

« Mais…c'est incorrect capitaine ! »

Mr Spock

Vocabulaire

Service packs

Les **Service packs** sont des mises à jours majeures d'un produit[1], ils permettent de corriger des problèmes connus, voire d'étendre les fonctionnalités d'un système Windows. Il s'agit donc d'un ensemble d'outils, de pilotes et de mises à jour et comprennent les améliorations développées après la publication du produit.

Les Service packs sont propres à un produit ; à chaque produit correspond donc une série distincte de Service packs. Toutefois, le même Service pack est généralement utilisé pour différentes familles / versions d'un même produit. Par exemple, le même Service pack sert à mettre à jour Windows 2000 Server et Windows 2000 Professionnel.

Les Service packs sont **cumulatifs** : tout nouveau Service pack contient les correctifs des précédents et des nouvelles modifications recommandées depuis. Le dernier Service pack peut donc être installé sans que les versions précédentes le soient.

Correctifs logiciel

Le **Quick Fix Engineering** (QFE) est un groupe de Microsoft qui produit des correctifs logiciels. Ces correctifs sont fournis aux clients qui rencontrent des problèmes bloquants auxquels il n'est pas possible de remédier par une astuce quelconque. Certains documents techniques utilisent l'abréviation QFE pour faire référence à des correctifs logiciels.

Ces correctifs ne sont pas soumis à des tests de régression intensifs et ils s'appliquent à des problèmes très spécifiques ; on ne doit utiliser un correctif logiciel que si l'on rencontre exactement le problème qu'il corrige.

Des groupes de correctifs logiciels sont incorporés périodiquement aux Service packs ; à cette occasion, ils subissent des tests plus rigoureux et sont mis à la disposition de tous les clients.

Correctifs de sécurité (hot fixes)

Les correctifs de sécurité, ou « hot fixes » selon la terminologie anglo-saxonne, sont conçus pour éliminer des vulnérabilités élémentaires. Correctifs de sécurité et correctifs

[1] Le terme « produit » est ici à prendre au sens le plus large ; « Windows XP Pro » est considéré comme un produit, au même titre que « Office 2000 » ou « Visual Studio .Net ».

logiciel sont similaires, mais les premiers sont obligatoires dans les circonstances auxquelles ils s'appliquent et doivent être déployés rapidement.

Installation manuelle des correctifs

La méthode de déploiement la plus courante dans la plupart des organisations est l'installation manuelle. Elle consiste à lancer le programme correspondant au correctif sur chaque machine. Cette méthode n'est pas la plus adaptée à un déploiement massif de correctifs sur de nombreuses machines

Le nom des correctifs est souvent riche en informations. Prenons l'exemple de Q292435_W2K_SP3_x86_en.EXE :

- Q292435 est le numéro de l'article de la Base de connaissances où l'on peut trouver des informations supplémentaires sur le correctif.
- W2K désigne le produit auquel le correctif est destiné (ici, Microsoft Windows 2000).
- SP3 est le numéro du Service pack auquel le correctif sera ajouté.
- x86 est l'architecture de processeur à laquelle le correctif est destiné.
- en désigne la langue (anglais).

Remarque : les correctifs dont le nom est de la forme QXXXXXX.exe, sans W2K_SP3_x86, sont généralement propres à une application du système, telle que Internet Explorer.

Les correctifs prennent en charge divers commutateurs de ligne de commande qui permettent de contrôler le comportement du processus d'installation.

Commutateur	Description
-y	Effectue une désinstallation
-f	Force la fermeture des applications lors de l'arrêt
-n	Ne crée pas de répertoire de désinstallation
-z	Ne redémarre pas à la fin de la mise à jour
-q	Mode silencieux : aucune interface utilisateur
-m	Mode sans assistance
-l	Dresse une liste des correctifs installés

Les correctifs propres à une application dont le nom de fichier est de la forme QXXXXXX.exe ne prennent généralement pas en charge tous les commutateurs ci-dessus.

Normalement, on doit redémarrer l'ordinateur après l'installation de chaque correctif. En effet, les fichiers qui sont verrouillés ou en cours d'utilisation ne peuvent pas être remplacés à chaud ; ils sont donc placés dans une file d'attente et remplacés lors du redémarrage du système. **QChain** est un outil qui permet d'enchaîner plusieurs correctifs et de redémarrer une seule fois.

Les correctifs se présentent sous la forme d'exécutables (fichier .exe) et non de paquettages d'installation (.msi) : la (fâcheuse) conséquence de ce mode de distribution est que **ces correctifs ne peuvent être déployés à l'aide d'une GPO**.

HfNetChk

Microsoft Network Security Hotfix Checker (**Hfnetchk**) est un utilitaire « ligne de commande » qui permet de vérifier si la configuration en cours des machines est à jour et dispose de tous les correctifs de sécurité pertinents. Cet outil télécharge directement du site de Microsoft un fichier XML à jour (*mssecure.xml*) qui contient la liste des derniers correctifs à appliquer pour rester dans un environnement sûr. Si l'on ne dispose d'aucune connexion Internet, Hfnetchk peut utiliser un fichier XML local.

Pour utiliser Hfnetchk, il est nécessaire de disposer de droits d'accès d'administrateur (administrateur local ou de domaine) à l'ordinateur dont on vérifie les correctifs.

Cet outil admet un certain nombre de commutateurs décrits dans le tableau suivant.

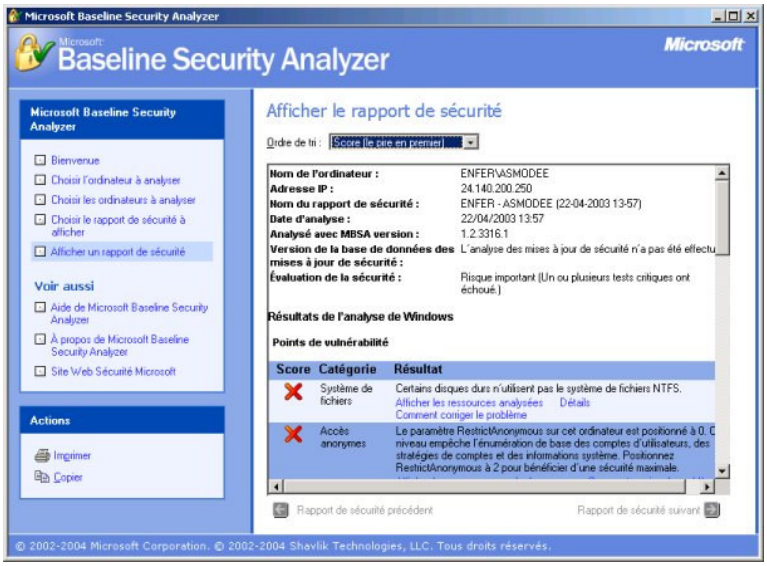
Commutateurs	Fonction
-about	À propos de hfnetchk.
-h <nom_hôte>	Spécifie le nom de l'ordinateur NetBIOS à analyser. Par défaut, il s'agit de l'ordinateur local.
-fh <fichier_hôtes>	Spécifie le nom d'un fichier contenant les noms des ordinateurs NetBIOS à analyser. Un seul nom par ligne, 256 lignes maximum par fichier.
-i <adresse_IP>	Spécifie l'adresse IP d'un ordinateur à analyser.
-fip <fichier_IP>	Spécifie le nom d'un fichier contenant les adresses à analyser. Une seule adresse IP par ligne, 256 lignes maximum par fichier.
-r <plage>	Spécifie la plage d'adresses IP à analyser (entre adresse_IP_1 et adresse_IP_2). Les bornes de l'intervalle sont comprises dans la plage.
-d <nom_domaine>	Spécifie le nom du domaine à analyser. Tous les ordinateurs appartenant à ce domaine seront analysés.
-n <réseau>	Tous les systèmes du réseau local seront analysés (tous les hôtes du Voisinage réseau).
-history <niveau>	Inutile pour une utilisation normale.
-t <threads>	Nombre de threads utilisés pour exécuter l'analyse. Les valeurs possibles vont de 1 à 128. La valeur par défaut est 64.
-o <sortie>	Spécifie le format de sortie souhaité. (tab) indique un format délimité par des tabulations. (wrap) définit un format de texte avec renvoi à la ligne automatique. La valeur par défaut est « wrap ».
-x <source_données>	Spécifie la source de données XML qui contient les informations concernant les correctifs logiciel. Il peut s'agir d'un nom de fichier XML, d'un fichier .cab XML compressé ou d'une URL. La valeur par défaut est « mssecure.cab » qui renvoie au site Web Microsoft.
-s <suppression>	Supprime les messages NOTE et WARNING. La valeur 1 supprime les messages NOTE uniquement ; la valeur 2 supprime les messages NOTE et WARNING. Par défaut, tous les messages sont affichés.
-z	Annule les vérifications de registre.
-nosum	Annule le calcul du total de contrôle des fichiers qui

Commutateurs	Fonction
	peut consommer une grande quantité de bande passante. Cette option permet d'accélérer l'analyse et d'utiliser moins de bande passante. Elle n'empêche pas les contrôles de version des fichiers.
-b	Affiche l'état des correctifs logiciel nécessaires au respect de normes de sécurité minimales.
-v	Affiche des informations concernant les messages Patch NOT Found, WARNING et NOTE. Cette option est activée par défaut en mode « tab ».
-f <fichier_sortie>	Spécifie le nom du fichier où enregistrer les résultats. Par défaut, les résultats sont simplement affichés à l'écran.
-u <nom_utilisateur>	Spécifie un nom d'utilisateur facultatif pour la connexion à un ordinateur distant.
-p <mot_de_passe>	Spécifie le mot de passe à utiliser avec le nom d'utilisateur.
-?	Affiche un menu d'aide.

La société Shavlik propose également un outil dénommé **HfnetchkPro**, basé sur la technologie Hfnetchk de Microsoft. Il offre une interface graphique évoluée et peut également automatiser le déploiement des Services Pack et des correctifs.

Utilisation de MBSA

Microsoft Baseline Security Analyser (MBSA) est un outil d'analyse de la sécurité d'un système Windows, mis gratuitement à la disposition des utilisateurs par Microsoft.

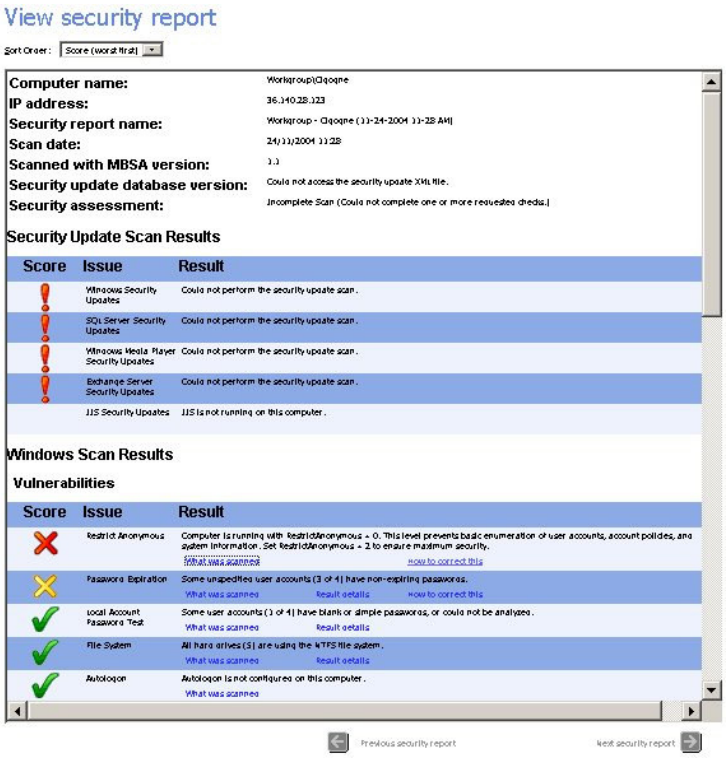


L'analyse peut se dérouler en local ou à distance, sous réserve d'avoir les privilèges nécessaires et suffisants, et se base sur un fichier de configuration au format XML, constamment remis à jour par Microsoft[1]. Ce fichier contient, entre autres, l'ensemble des références aux différents correctifs de sécurité édités par Microsoft.

L'utilitaire vérifie à la fois la configuration du système et le niveau de patch appliqué à la machine analysée.

[1] Il s'agit en fait du fichier *mssecure.xml*, précédemment rencontré pour hfnetchk.

A l'issue de l'analyse, un fichier rapport est généré, précisant les points à corriger et, surtout, les liens Internet permettant de récupérer les correctifs manquant.



Pour ceux que le site « Windows Update » de Microsoft rebute, pour les réseaux ne disposant pas d'une connexion directe à l'Internet, ou pour les administrateurs désireux de conserver les fichiers de correctifs, MBSA s'avère un outil indispensable.

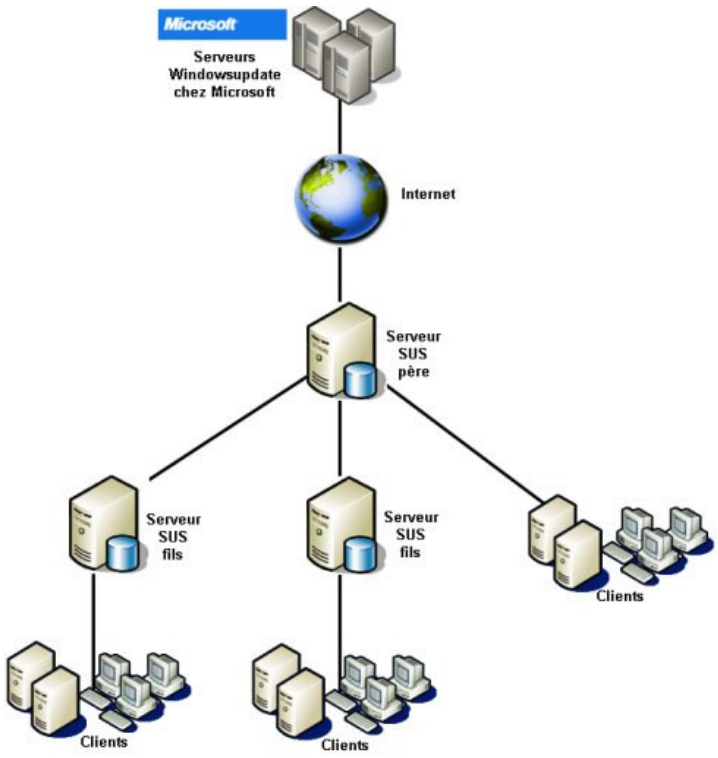
Il existe une version « ligne de commande » de MBSA, *mbsacli.exe*. Cette version à la capacité de vérifier les checksums des correctifs sur les machines cible, ce que ne permet pas la version graphique

Cette version « ligne de commande » peut être invoquée selon deux modes de fonctionnement : avec ou sans le commutateur /**hf**. L'option /**hf** réalise la seule analyse des mises à jour et génère un résultat au format XML (tout comme le fait hfnetchk), tandis que l'oubli de cette option provoque une analyse complète de la cible.

Software Update Service (SUS)

SUS (Software Update Service) constitue un ensemble d'outils, gratuits et développés par Microsoft, autorisant la mise à jour automatisée d'un parc de machines Windows. Son fonctionnement requiert un serveur spécialisé dans la fourniture des correctifs et un service d'accès sur chacune des machines à mettre à jour.

L'architecture SUS est décrite dans le schéma suivant, elle consiste en un serveur, devant pouvoir se connecter à un service de mise à jour Microsoft sur l'Internet, et de clients récupérant les mises à jour sur ce serveur d'entreprise.



Afin d'éviter de concentrer tout le mécanisme de mise à jour sur un seul serveur SUS, cette architecture peut se décliner de façon arborescente comme le précise ce schéma.

La procédure de récupération des correctifs nécessite, et **c'est là l'inconvénient majeur de ce système de mise à jour**, que le serveur SUS primaire puisse accéder en Web (port 80) aux serveurs WindowsUpdate de Microsoft.

Précisons que la distribution d'un correctif nécessite que ce dernier soit marqué comme « approuvé » par l'administrateur du serveur SUS de l'entreprise ; **seuls les correctifs approuvés seront répliqués sur les serveurs fils**. A ce titre, il est recommandé de créer un « serveur SUS fils » de tests auquel on raccordera des machines de tests représentatives de celles existant en production, et ce afin de vérifier le bon fonctionnement des machines après installation des correctifs.

Afin d'aider à cette approbation des correctifs, Microsoft signe numériquement chaque mise à jour émise sur l'Internet ; l'administrateur local peut ainsi vérifier l'origine et l'intégrité des correctifs, et éviter que des mises à jours « pirates » ne puissent venir polluer le processus.

Coté client, il existe trois procédures de mise à jour configurables :

- **Téléchargement et installation automatique** : tout correctif approuvé est automatiquement téléchargé et installé sur le client,

- **Téléchargement automatique et installation programmée** : tout correctif approuvé est automatiquement téléchargé mais l'installation est différée (installation de nuit par exemple).

- **Téléchargement et installation soumise à approbation** : l'utilisateur en cours de session est notifié de l'existence d'un correctif approuvé s'appliquant à sa machine. Il devra provoquer manuellement à la fois son téléchargement et son installation.

Sécurité d'Internet Information Service (IIS)

« Sur Internet, on peut écouter la radio tout en payant le téléphone ! »

Anne Roumanoff

Préambule

Microsoft IIS a longtemps été le premier « *fournisseur* » de vulnérabilités majeures sur la plate-forme Windows NT. Sorti trop vite des laboratoires de Microsoft, donc mal débugué, avec une architecture peu performante et, surtout, une configuration par défaut souvent trop laxiste, IIS s'est rapidement imposé dans le monde de l'informatique professionnelle comme un produit à fuir de toutes ses forces.

Cette réputation de serveur non sûr a par ailleurs connu son apogée en 2002, lorsque le réputé Gartner Group a encouragé les entreprises victimes des vers Code Red et Nimda à changer de serveur Web.

Il est vrai que IIS (jusqu'à sa version 5 incluse) demeure l'un des serveurs Web parmi les plus difficiles à sécuriser. Cette difficulté à rendre sûre ce service prend ses origines dans le fait que IIS n'est pas tout à fait un serveur comme les autres : son architecture et ses nombreuses possibilités en font d'ailleurs plus un serveur d'applications qu'un simple serveur Web.

Cependant, ce qu'omet de préciser le Gartner Group c'est que c'est la configuration « **par défaut** » d'IIS qui n'est pas sûre ; il est tout à fait possible d'obtenir, par une configuration adéquate et au prix de quelques efforts, un serveur solide et résistant aux agressions.

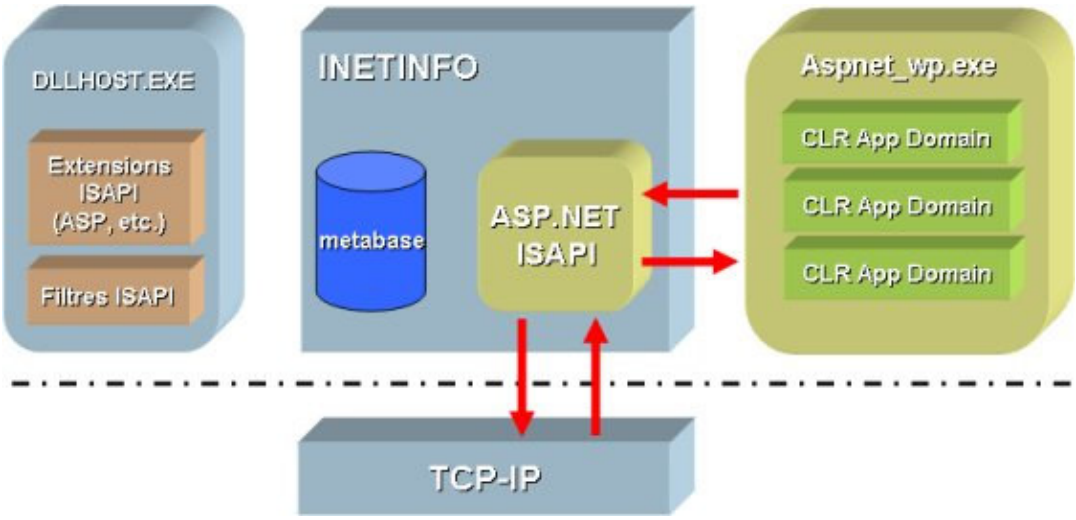
Dans les paragraphes qui suivent, nous nous attacherons à présenter l'architecture de sécurité et le fonctionnement d'IIS 5.0, tout en indiquant les mesures de protections nécessaires pour utiliser de façon sécurisée un tel serveur.

Le lecteur informé objectera que la dernière version à jour d'IIS est la version 6, livrée en standard avec Windows 2003 : pour autant, les lignes qui suivent se borneront à présenter la sécurisation d'IIS 5.0, la version 6 étant traitée en toute fin de chapitre.

Architecture d'IIS 5

IIS 5 est totalement construit autour du processus **INETINFO.EXE**, véritable chef d'orchestre du serveur, et qui dispose de privilèges systèmes. C'est ce même processus qui réalise à la fois l'interprétation des requêtes, leur traitement et le renvoi des données

au client. Construit directement au dessus de TCP-IP, en mode utilisateur, INETINFO est LE composant critique du serveur.



INETINFO repose sur un ensemble de paramètre de gestions et de configuration regroupés dans une « metabase ».

Le service INETINFO traite directement les requêtes sur des objets statiques tels que les pages HTML ou les images. Pour l'aspect « dynamique » du traitement des requêtes (traitement des scripts, des applications…) il existe deux façons distinctes de procéder :

- Un traitement « **externe** » au processus inetinfo.exe : typiquement, le traitement des extensions ISAPI comme les pages ASP est réalisé au sein d'un processus hôte **DLLHOST.EXE** (les applications .NET sont quant à elles traitées par le CLR au niveau d'un processus hôte différent ; **aspnet_wp.exe**).

- Un traitement « **interne** » au processus inetinfo.exe : dans ce cas, il est nécessaire d'intégrer un plug-in à IIS sous la forme d'une ou plusieurs DLLs.. Les fonctions de ces DLLs sont alors appelées directement par inetinfo.exe en tant que nouveau thread du service.

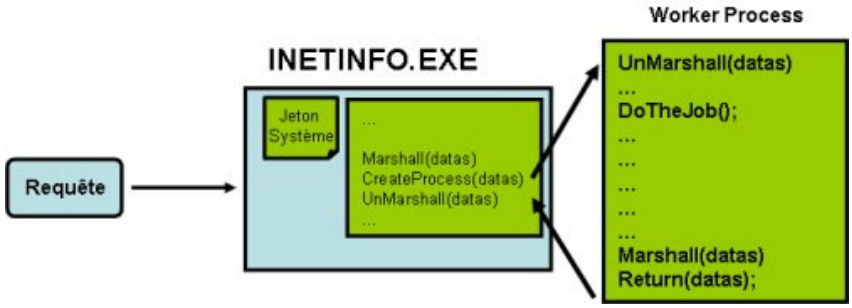
Selon les cas de figure, il est possible de choisir le type de traitement à appliquer[1]. Par exemple, des pages PHP peuvent être servies soit comme une extension ISAPI (donc au sein d'un processus hôte tiers) soit selon l'approche « DLL » (donc par le processus inetinfo.exe). Bien évidemment, chaque méthode a ses avantages et ses inconvénients.

Traitement externe :

Dans le cas d'un traitement externe, on préserve une certaine isolation entre le processus serveur et les traitements réalisés. Dans la mesure où les processus sont isolés entre eux, il s'agit donc de la méthode la plus sécuritaire ; une compromission de l'extension ne remet pas en cause la sécurité du serveur tout entier, de plus le traitement peut être réalisé dans un contexte de sécurité non privilégié.

En revanche, cette méthode n'est pas la plus performante : chaque requête devra être traitée par un processus différent et il sera nécessaire de gérer la communication des paramètres et des données entre le processus serveur (inetinfo.exe) et le processus de traitement (problème de « marshalling / unmarshalling » des données).

[1] Nota : ce choix est dépendant des capacités du composant et non de celles du serveur IIS.

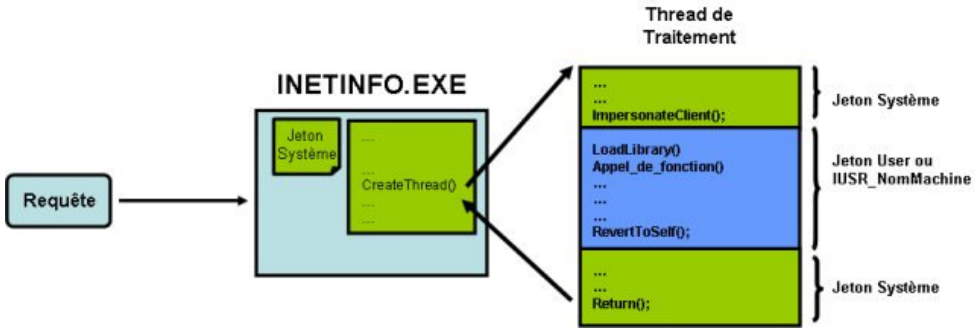


Traitement interne :

Fonctionnellement, il s'agit de la méthode la plus performante. Chaque requête est directement traitée par le processus hôte (inetinfo.exe) au sein d'un nouveau thread. On bénéficie alors d'un double avantage : les données sont transmises directement, sans marshalling, entre le processus serveur et le thread de traitement et l'on profite de l'environnement d'exécution du processus serveur d'où un gain de temps CPU (commutations de contextes réduites).

En revanche, il s'agit de la méthode la moins sécuritaire puisque les traitements sont réalisés dans le contexte de sécurité du processus inetinfo.exe, qui est lancé avec des privilèges systèmes.

Dans ce mode de fonctionnement, le thread de traitement de la requête n'est pas exécuté dans un contexte de sécurité privilégié ; inetinfo.exe réalise une impersonation pour ce thread.



Si l'accès est anonyme, c'est le compte « IUSR_NomMachine » qui est utilisé pour le jeton d'impersonation. En revanche, si l'accès est authentifié, inetinfo.exe impersone l'utilisateur appelant, ce qui permet donc d'exécuter la requête dans le contexte de sécurité de l'utilisateur à l'origine du traitement. Ce fonctionnement peut paraître sûr, puisqu'il empêche les traitements dans un contexte de sécurité trop privilégié, mais le système d'impersonation demeure fragile (voir en page 28 la description de la sécurité de ce mécanisme).

Sécurisation du serveur

Installation

Avant toute manipulation ultérieure, il convient de s'assurer que seule la toute dernière version disponible pour sa plate-forme est bien installée sur son système et non une version antérieure. Quel que soit le cas de figure, la version minimale recommandée demeure la version 5.0 d'IIS : toute autre version plus ancienne est à proscrire.

Si une telle ancienne version pré-existe sur le serveur, le plus simple consiste à désinstaller complètement le service et à **installer la version la plus à jour**. L'expérience

montre en effet que les mises à jours de services IIS laissent trop souvent en place des composants et paramètres anciens préjudiciables à la sécurité.

Un serveur IIS ne devrait JAMAIS être installé sur un contrôleur de domaine. En effet un contrôleur de domaine contient une copie des authentifiants des utilisateurs de ce domaine et la compromission du serveur entraînerait *de facto* la compromission du domaine. Si malgré tout il est nécessaire que le serveur Web soit inclus dans un domaine (par exemple dans le cas ou l'on souhaiterait bénéficier de l'authentification Microsoft sur le serveur), il conviendra d'installer le serveur en tant que « serveur membre » du domaine, caractérisé par le fait qu'il ne dispose pas de copie des authentifiants utilisateurs.

Le serveur hôte doit être installé avec le minimum de fonctionnalités et doit disposer de l'ensemble des volumes formatés en NTFS. Installer IIS sans les fonctionnalités FTP et SMTP si le besoin n'est pas avéré, et surtout sans les extensions FrontPage si celles-ci ne sont pas indispensables.

Il convient également de **ne jamais installer d'arborescences de publications, statiques ou dynamiques, sur le disque Système** : par défaut toutes les versions d'IIS, jusqu'à la version 5 incluse, proposent de créer la racine de l'arborescence principale sous `C:\Inetpub`. Choisir, en lieu et place, un disque dédié et un nom d'arborescence moins prévisible (`D:\toultoutim` par exemple). Cette recommandation bloque à elle seule 90% des attaques connues sur le service IIS

Positionnement des Listes à Contrôle d'Accès

Dans un premier temps, il est nécessaire de séparer les différents types de fichiers dans des arborescences distinctes et d'appliquer à ces arborescences des ACLs plus restrictives. Le tableau suivant donne des recommandations classiques en ce sens :

Type de fichiers	ACLs
Fichiers CGI (*.exe, *.dll, *.cmd, *.pl…)	Tout le monde = X Administrateurs = Contrôle total Système = Contrôle Total
Fichiers Scripts (ASP)	Tout le monde = X Administrateurs = Contrôle total Système = Contrôle Total
Fichiers d'inclusions (*.inc, *.shtm, *.shtml…)	Tout le monde = X Administrateurs = Contrôle total Système = Contrôle Total
Contenu statique (*.txt, *.gif, *.html…)	Tout le monde = R Administrateurs = Contrôle total Système = Contrôle Total

Plutôt que de définir des ACLs pour chaque fichier, il est préférable de créer des répertoires spécifiques pour ces types de fichiers, d'appliquer des ACLs sur ces répertoires et de permettre l'héritage des ACLs.

Les répertoires « mailroot » et « ftproot » créés par défaut sous IIS 5.0 disposent d'une ACL trop peu restrictive (Tout le monde = Contrôle Total) qui devrait être remplacée par quelque chose de plus strict.

Dans le même état d'esprit, veillez à ce que les ACL correspondant aux fichiers journaux générés par IIS (`%systemroot%\system32\LogFiles`) soient les suivantes :

Administrateurs = Contrôle total
Système = Contrôle total
Tout le monde = RWC (lecture, écriture, modification)

Cela permet d'empêcher d'éventuels utilisateurs malveillants de supprimer les fichiers pour faire disparaître les traces de leur passage.

Journalisation

L'activation des options de journalisation du serveur IIS est primordiale si l'on souhaite détecter d'éventuelles agressions sur son système. On peut pour cela faire appel au format de fichier journal étendu du W3C ; ce format est activable dans les propriétés du « site Web » de l'interface d'administration MMC de IIS. Il faut alors « activer l'enregistrement dans le journal », puis sélectionner le « Format de fichier journal étendu du W3C » dans la liste déroulante du « Format de journal actif ».

Ce format de journal autorise l'enregistrement de paramètres personnalisés (propriétés/ propriétés étendues) ; les propriétés suivantes sont celles recommandés par Microsoft :

- Adresse IP du client
- Nom d'utilisateur
- Methode
- Ressource URI
- etat HTTP
- etat Win32
- Agent utilisateur
- Adresse IP du serveur
- Port du serveur

Les deux dernières propriétés ne sont utiles que si IIS héberge plusieurs serveurs Web sur un seul ordinateur.

La propriété « état Win32 » est utile en cas de debogage ; lorsque l'on examine le fichier journal, chercher l'erreur 5, qui signifie un refus d'accès. On trouvera la signification d'autres erreurs Win32 en saisissant « `net helpmsg XXX` » sur la ligne de commande, le XXX étant à remplacer par le numéro d'erreur qui nous intéresse.

Suppression des composants inutiles

Afin de réduire la surface d'attaque de son serveur, la recommandation de base qui s'impose consiste à ne laisser en place que ce est strictement indispensable au fonctionnement du serveur.

A ce titre, on pensera à supprimer tous les exemples d'applications (les arborescences ET les répertoires virtuels), même si ces répertoires ne sont censés n'être accessibles que sur le réseau de loopback (127.0.0.1) :

- Exemples IIS sous C:\Inetpub\iissamples (Répertoire virtuel /IISSAMPLES),
- Documentation IIS sous C:\Winnt\help\iishelp (Répertoire virtuel /IISHelp)
- Accès aux données sous c:\program files\fichiers communs\system\msadc (Répertoire virtuel /MSADC).

De même il est fortement recommandé de supprimer les mappages de scripts non utilisés. En effet, IIS 5.0 est conçu pour prendre en charge nativement certaines extensions de noms de fichiers courantes telles que *.asp et *.shtm.

Ainsi, lorsqu'IIS reçoit une requête pour un fichier appartenant à l'un de ces types, l'appel est géré par une DLL. Si l'on n'utilise pas certaines de ces extensions ou fonctionnalités, il est possible (et recommandé) de les supprimer comme suit

- Ouvrir le Gestionnaire des services Internet;
- Cliquer avec le bouton droit sur le serveur Web et choisir « Propriétés » dans le menu contextuel;
- Dans les « Propriétés principales », sélectionner « Service WWW / Modifier / Répertoire de base / Configuration » et commencer la suppression des mappings inutiles.

Idéalement, un serveur IIS 5 ne servant que des pages statiques en html ne devrait disposer d'aucun mapping de fichiers.

Le tableau suivant présente les recommandations de Microsoft sur le sujet :

Si vous n'utilisez pas	**Supprimez l'entrée suivante**
La redéfinition du mot de passe pour le Web	.htr
Connexion de base de données Internet (les sites Web IIS 5 doivent s'appuyer sur la technologie ADO ou une technologie semblable)	.idc
Inclusions côté serveur	.stm, .shtm et .shtml
Impression Internet	.printer
Index Server	.htw, .ida et .idq

Désactivation des chemins d'accès relatifs au répertoire parent

L'option de chemins d'accès relatifs au répertoire parent permet d'utiliser des guillemets doubles ".." dans les appels aux fonctions telles que MapPath. Cette option est activée par défaut et il vous faut la désactiver. Pour ce faire, procédez comme suit :

- Cliquez avec le bouton droit sur la racine du site Web, puis choisissez Propriétés dans le menu contextuel;
- Cliquez sur l'onglet Répertoire de base;
- Cliquez sur Configuration;
- Cliquez sur l'onglet Options de l'application;
- Désactivez la case Activer les chemins d'accès relatifs au répertoire parent.

Utilisation d'outils de sécurisation

Pour parfaire la sécurisation de son serveur des outils complémentaires peuvent éventuellement être employés. Sans être exhaustif les produits phares du domaine son listés ci-après :

- IIS LockDown Tool ; outil de sécurisation d'IIS, disponible sur le site de Microsoft)
- URLScan ; filtre ISAPI qui va appliquer des règles aux URLs reçues, avant même que ces dernières ne soient traitées par le moteur d'IIS (disponible gratuitement sur le site de Microsoft)
- SecuredIIS ; script VB de sécurisation d'IIS, disponible sur le site de la ntbugtraq (www.ntbugtraq.com), écrit par Patrick Chambet et Russ Copper.

IIS 6.0

Architecture

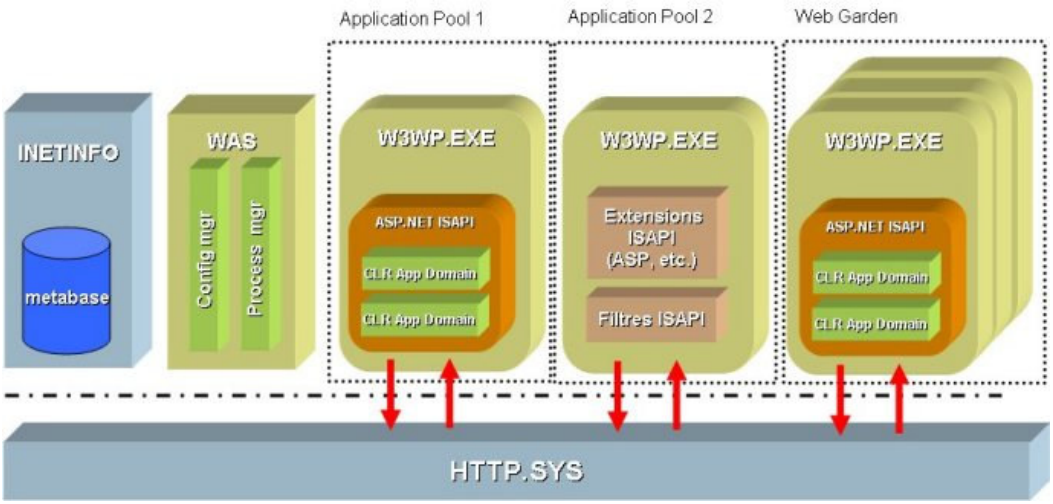
La version 6 du serveur Internet de Microsoft est souvent présentée comme une évolution majeure du dit service. A y regarder de plus près, on s'aperçoit que IIS 6.0 est bien plus que cela : si l'on considère l'architecture du serveur et sa philosophie d'emploi et de configuration, **IIS 6 est ni plus ni moins qu'un nouveau serveur Web** n'ayant de commun avec IIS 5 que son nom.

En termes d'architecture, d'abord, c'est un bouleversement plus que majeur qui a frappé cette nouvelle mouture de serveur.

Le premier changement majeur concerne le traitement du protocole HTTP : la gestion de base des requêtes au serveur et l'émission des résultats sont désormais traités au niveau **noyau** par un nouveau pilote HTTP.SYS, déchargeant ainsi de cette tâche le processus inetinfo.exe. Ce dernier conserve à sa charge le maintien en condition des services, la gestion de la metabase et le service de fourniture des pages statiques ; de fait, il n'a plus besoin de se lancer avec des privilèges SYSTEM, inetinfo.exe tourne avec les privilèges moindres du compte virtuel « service réseau ».

En termes de performance ce type d'architecture est bien meilleur que ce qui se faisait avec IIS 5, le passage de données entre les applications et le driver étant directement effectués grâce à des appels systèmes. En outre, le nombre de changement d'états UserMode / KernelMode diminue considérablement du fait de la bascule du traitement des requêtes HTTP vers le mode noyau[1].

Le traitement du contenu dynamique est réalisé maintenant de façon **systématique** au sein d'un processus de travail (Worker Process) hôte W3WP.EXE constituant un « pool d'application ». Chaque pool d'application peut utiliser son propre compte utilisateur, ce qui limite considérablement l'impact d'une compromission d'un pool par l'isolation renforcée des processus qui en découle.



Afin de garantir de meilleures performances, IIS 6 supporte également un mécanisme de WebGarden : un WebGarden est un pool d'application qui héberge plus d'un processus de travail simultanément ; en cas d'engorgement, IIS génère alors autant de processus de

[1] Ce qui éloigne d'ailleurs encore plus le noyau de Windows de sa philosophie « micronoyau » d'origine…

travails que nécessaires pour traiter les requêtes, parallélisant ainsi le traitement au lieu de le sérialiser dans une file d'attente.

Enfin, et pour permettre une meilleure disponibilité, un processus WAS (Web Awareness Service) a la charge de surveiller en permanence les pools d'application pour les relancer en cas de plantage.

Sécurité d'IIS 6

Du fait de son architecture mieux pensée, IIS 6 bénéficie d'emblée d'une bien meilleure sécurité que ses prédécesseurs.

En outre, la philosophie générale du serveur a été complètement repensée afin de coller au mieux à celle de Windows 2003.

En effet, IIS 6 est **par défaut** un serveur sécurisé :

- IIS **n'est pas** installé par défaut sur un Windows Serveur 2003 (contrairement à Windows Serveur 2000),
- **Aucune** extension n'est activée (un serveur IIS 6 brut d'installation ne saura servir que des pages statiques en HTML),
- Il n'y a plus par défaut de répertoires virtuels pouvant contenir des exécutables ; les répertoires /MSADC et autres /SCRIPTS devront désormais être explicitement créés,
- URLScan est intégré à IIS 6,
- Les utilitaires en ligne de commande sont inaccessibles aux processus IIS,
- Le contenu des arborescences de publication est par défaut protégé en écriture au moment de leur création,
- La gestion des erreurs est améliorée (journaux d'évènement plus explicites pour les administrateurs, erreurs émises vers l'utilisateur donnant moins d'informations sur la configuration du système, enregistrement dans les logs des sous-états HTTP…)
- Des ACLs plus restrictives sont positionnées sur les objets d'IIS (journaux, cache…),
- L'accès au chemin parent est maintenant désactivé (..)…

Pour résumer, et à l'inverse de ses ancêtres, IIS est installé avec les options minimales tout justes nécessaires pour servir des pages statiques. Toute extension de fonctionnalité doit alors être explicitement déclarée et mise en œuvre par l'administrateur du système.

Dans ces conditions, la sécurisation d'un serveur IIS 6 ne consiste plus qu'à sélectionner les bons composants lors de l'installation, et à administrer proprement la configuration de son serveur.

A propos de ce support…

« La conclusion résulte souvent du moment où vous en avez eu marre.»

Anonyme

Le contenu de ce support de cours a été rédigé au fil de l'eau depuis 1998 et a subi de nombreuses évolutions et refontes depuis cette date.

La durée normale de la formation liée à ce support est de 28 heures, ponctuée de nombreux travaux pratiques et exercices, sur un réseau local composé de serveurs sous Windows. Il est habituellement accompagné d'un CD-Rom contenant l'ensemble des outils nécessaires aux exercices et de nombreux documents ayant trait à la sécurité sous Windows (la plupart de ces documents proviennent de chez Microsoft).

Afin de faire évoluer ce cours, l'auteur est preneur de toute remarque et proposition d'amélioration et vous en remercie par avance. Il est joignable à l'adresse email jgallard@free.fr.

Jean-Christophe GALLARD – Rennes, 2005

ANNEXES

Liste des SIDs Réservés

S-1-0
Nom: Null Authority
Description: Une identifier authority.

S-1-0-0
Nom: Nobody
Description: Pas de security principal.

S-1-1
Nom: World Authority
Description: Une identifier authority.

S-1-1-0
Nom: Tout le Monde
Description: Un groupe qui inclut tous les utilisateurs, y compris les utilisateurs anonymes et les invités. L'appartenance à ce groupe est gérée par le système d'exploitation.

S-1-2
Nom: Autorité Locale
Description: Une identifier authority.

S-1-3
Nom: Createur Authority
Description: Une identifier authority.

S-1-3-0
Nom: CREATEUR PROPRIETAIRE
Description: un conteneur virtuel pour SID hérité. Lorsqu'une ACE est héritée, le système remplace ce SID par celui du créateur de l'objet.

S-1-3-1
Nom: GROUPE CREATEUR
Description: un conteneur virtuel pour SID hérité. Lorsqu'une ACE est héritée, le système remplace ce SID par celui du groupe primaire du créateur de l'objet. La notion de groupe primaire n'est utilisée que par le sous-système POSIX.

S-1-3-2
Nom: SERVEUR CREATEUR PROPRIETAIRE
Description: Ce SID n'est pas utilisé dans Windows 2000.

S-1-3-3
Nom: SERVEUR GROUPE CREATEUR
Description: Ce SID n'est pas utilisé dans Windows 2000.

S-1-4
Nom: Non-unique Authority
Description: Une identifier authority.

S-1-5
Nom: NT Authority
Description: Une identifier authority.

S-1-5-1
Nom: LIGNE
Description: Un groupe qui inclut tous les utilisateurs s'étant connecté au système par l'intermédiaire d'une ligne téléphonique. L'appartenance à ce groupe est gérée par le système d'exploitation.

S-1-5-2
Nom: RESEAU
Description: Un groupe qui inclut tous les utilisateurs qui se sont connectés depuis une session réseau. L'appartenance à ce groupe est gérée par le système d'exploitation.

S-1-5-3
Nom: TACHE (Batch)
Description: Un groupe qui inclut tous les utilisateurs qui se sont connectés en tant que tâche. L'appartenance à ce groupe est gérée par le système d'exploitation.

S-1-5-4
Nom: INTERACTIF
Description: Un groupe qui inclut tous les utilisateurs qui se sont connectés au travers d'une session interactive. L'appartenance à ce group est gérée par le système d'exploitation.

S-1-5-5-X-Y
Nom: Session de Logon.
Description: Une session de logon. Les valeurs X et Y sont différentes pour chaque session.

S-1-5-6
Nom: SERVICE
Description: Un groupe qui inclut tous les principaux qui se sont connectés en tant que service. L'appartenance à ce group est gérée par le système d'exploitation.

S-1-5-7
Nom: ANONYMOUS LOGON
Description: Un groupe qui inclut tous les utilisateurs qui se sont connectés anonymement. L'appartenance à ce

groupe est gérée par le système d'exploitation.

S-1-5-8
Nom: PROXY
Description: Ce SID n'est pas utilisé dans Windows 2000.

S-1-5-9
Nom: ENTERPRISE DOMAIN CONTROLLERS
Description: Un groupe qui inclut tous les contrôleurs de domaine dans une forêt qui utilise le service Active Directory. L'appartenance à ce groupe est gérée par le système d'exploitation.

S-1-5-10
Nom: Principal Self
Description: un conteneur virtuel pour SID hérité. Lorsqu'une ACE est héritée, le système remplace ce SID par celui du SID primaire du créateur de l'objet..

S-1-5-11
Nom: Utilisateurs Authentifiés
Description: Un groupe qui inclut tous les utilisateurs qui se sont authentifiés sur le système. L'appartenance à ce groupe est gérée par le système d'exploitation.

S-1-5-12
Nom: RESTRICTED
Description: Ce SID n'est pas utilisé dans Windows 2000.

S-1-5-13
Nom: UTILISATEUR TERMINAL SERVER
Description: Un groupe qui inclut tous les utilisateur s'éant connecté au travers d'une session Terminal Server. L'appartenance à ce groupe est gérée par le système d'exploitation.

S-1-5-18
Nom: SYSTEM
Description: Compte spécial utilisé par le système.

S-1-5-19
Nom: NT Authority
Description: Service Local

S-1-5-20
Nom: NT Authority
Description: Service Réseau

S-1-5-domaine-500
Nom: Administrateur
Description: Le compte administrateur par défaut (également appelé BUILTIN Administrateur).

S-1-5-domaine-501
Nom: Invité
Description: Le compte « Invité » créé par défaut. Compte désactivé par défaut

S-1-5-domaine-502
Nom: KRBTGT
Description: Un compte de service utilisé par le service KDC.

S-1-5-domaine-512
Nom: Administrateurs du Domaine
Description: Un groupe global dont les membres peuvent administrer le domaine.

S-1-5-domaine-513
Nom: Utilisa. du Domaine
Description: Un groupe global incluant tous les utilisateur du domaine.

S-1-5-domaine-514
Nom: Invités du Domaine
Description: Un groupe global qui, par défaut, ne contient qu'une seul membre, le compte Invité par défaut.

S-1-5-domaine-515
Nom: Ordinateurs du Domaine
Description: Un groupe global qui inclut tous les clients et les serveurs qui on rejoint le domaine.

S-1-5-domaine-516
Nom: Contrôleurs de Domaine
Description: Un groupe global qui inclut tout contrôleur de domaine dans le domaine.

S-1-5-domaine-517
Nom: Editeurs de certificats
Description: Un groupe global qui inclut tous les ordinateurs hébergeant une PKI Les éditeurs de certificats peuvent publier des certificats dans l'Active Directory.

S-1-5-domaine racine-518
Nom: Administrateurs du Schema
Description: Un groupe Universel, présent uniquement en mode natif. Les membres de ce groupe sont autorisés à effectuer des modifications du schéma de l'Active Directory. Le seul membre de ce groupe est le compte Administrateur du domaine racine.

S-1-5-domaine racine-519
Nom: Administrateurs de l'Entreprise

Description: Un groupe Universel en mode natif, sinon un groupe global en mode mixte. Les membres de ce groupe sont autorisés à effectuer des modifications de la forêt, comme l'ajout d'un domaine fils. Par défaut, le seul membre de ce groupe est le compte est le compte Administrateur du domaine racine.

S-1-5-domaine-520
Nom: Propriétaires Créateurs de la Stratégie de Groupe
Description: Un groupe global autorisé à créer de nouveaux objets GPO. Par défaut, le seul membre de ce groupe est l'Administrateur.

S-1-5-domaine-533
Nom: Serveurs RAS et IAS
Description: Un groupe local de domaine. Par défaut ce groupe ne contient aucun membre.

S-1-5-32-544
Nom: Administrateurs
Description: Un groupe "built-in" créé systématiquement lors de chaque installation de système.

S-1-5-32-545
Nom: Utilisateurs
Description: Un groupe "built-in" créé systématiquement lors de chaque installation de système.

S-1-5-32-546
Nom: Invités
Description: Un groupe "built-in" créé systématiquement lors de chaque installation de système.

S-1-5-32-547
Nom: Utilisateurs avec Pouvoirs
Description: Un groupe "built-in". Les membres de ce groupe peuvent créer des utilisateurs et des groupes locaux, modifier et effacer les comptes qu'ils ont créés, et retirer des utilisateurs des groupes « Utilisateurs avec Pouvoir », « Utilisateurs » et « Invités ». Ils peuvent également installer des logiciels, créer, gérer et supprimer des imprimantes locales, ainsi que créer et supprimer des partages de fichiers.

S-1-5-32-548
Nom: Opérateurs de Compte
Description: Un groupe "built-in" qui n'existe que sur les contrôleurs de domaines. Par défaut, ce groupe ne contient aucun membre. Les opérateurs de comptes peuvent créer, modifier et effacer des comptes utilisateurs, des groupes et des comptes de machines dans l'Actvie Directorysauf dans le conteneur « built in » et dans l'OU Contrôleurs de domaines. Ils ne peuvent modifier les groupes Administrateurs et Administrateurs du domaine, pas plus qu'il ne peuvent gérer les comptes utilisateurs appartenant à ces groupes.

S-1-5-32-549
Nom: Opérateurs de Serveur
Description: Un groupe "built-in" qui n'existe que sur les contrôleurs de domaines. Par défaut, ce groupe ne contient aucun membre. Les opérateurs de serveurs peuvent ouvrir une session interactive sur les serveurs, créer et effacer des partages réseau, démarrer et arrêter les services, sauvegarder et restaurer les données, formater les disques et éteindre les machines..

S-1-5-32-550
Nom: Opérateurs d'Impression
Description: Un groupe "built-in" qui n'existe que sur les contrôleurs de domaines. Par défaut, ce groupe contient le groupe Utilisateurs du Domaine. Les Opérateurs d'impression peuvent gérer les imprimantes et les files **d'impression.**

S-1-5-32-551
Nom: Opérateurs de sauvegarde
Description: Un groupe "built-in" qui n'existe que sur les contrôleurs de domaines. Par défaut, ce groupe ne contient aucun membre. Les opérateurs de sauvegarde peuvent sauvegarder et restaurer les fichiers, quels que soient les ACLs positionnées. Ils peuvent en outre ouvrir une session interactive sur les serveurs et les éteindre.

S-1-5-32-552
Nom: Duplicateurs
Description: Un groupe "built-in" utilise par le Service de Réplication sur les contrôleurs de domaines. Par défaut, ce groupe ne contient aucun membre : ne **jamais** ajouter d'utilisateurs dans ce groupe.

Nouveaux Groupes "built in" créés lorsqu'un contrôleur de domaine Windows 2003 est ajouté dans un domaine :

<NOTE : version anglaise uniquement, à vérifier sur la traduction en français>

S-1-5-32-554
Nom: BUILTIN\Accès compatible pré-Windows 2000
Description: An alias added by Windows 2000. A backward compatibility group which allows read access on all users and groups in the domain.

S-1-5-32-555
Nom: BUILTIN\Remote Desktop Users
Description: An alias. Members in this group are granted the right to logon remotely.

S-1-5-32-556
Nom: BUILTIN\Network Configuration Operators
Description: An alias. Members in this group can have some administrative privileges to manage configuration of networking features.

S-1-5-32-557
Nom: BUILTIN\Incoming Forest Trust Builders
Description: An alias. Members of this group can create incoming, one-way trusts to this forest.

S-1-5-32-557
Nom: BUILTIN\Incoming Forest Trust Builders
Description: An alias. Members of this group can create incoming, one-way trusts to this forest.

S-1-5-32-558
Nom: BUILTIN\Performance Monitor Users
Description: An alias. Members of this group have remote access to monitor this computer.

S-1-5-32-559
Nom: BUILTIN\Performance Log Users
Description: An alias. Members of this group have remote access to schedule logging of performance counters on this computer.

S-1-5-32-560
Nom: BUILTIN\Windows Authorization Access Group
Description: An alias. Members of this group have access to the computed tokenGroupsGlobalAndUniversal attribute on User objects.

S-1-5-32-561
Nom: BUILTIN\Terminal Server License Servers
Description: An alias. A group for Terminal Server License Servers.

L'injection de DLL

L'injection de DLL est une technique de programmation destinée à faire exécuter, par un processus existant, du code que l'on injecte directement dans son espace mémoire. En un sens cette technique s'apparente à un mécanisme de *parasitisme* de processus par un autre.

Il existe plusieurs méthodes pour réaliser une telle injection de code, mais nous nous attacherons ici à ne décrire que la technique la plus efficace et, surtout, la plus utilisée.

Théorie de l'injection

Toute la mécanique d'injection de code repose sur le droit **SeDebugPrivilege**, également connu sous les termes « **Déboguer les programmes** ». Contrairement à ce que l'intitulé de ce privilège peut laisser penser, ce droit n'est pas nécessaire à l'utilisation d'un débuggeur fourni avec un environnement de programmation tel que Visual C++.

Ce privilège particulier permet de verrouiller ou d'autoriser l'accès à certaines APIs du système. Nous nous intéresserons plus particulièrement à trois de ces APIs : *VirtualAllocEx()*, *WriteProcessMemory*() et *CreateRemoteThread*() :

- *VirtualAllocEx()* permet de réaliser une opération de réservation de mémoire. Selon les paramètres fournis, cette fonction autorise à demander à un processus tiers une opération d'allocation mémoire.
- *WriteProcessMemory()* permet d'écrire directement dans l'espace mémoire d'un processus tiers.
- *CreateRemoteThread()* autorise à demander à un processus tiers de lancer un nouveau thread, implémentant une fonction f(). Cette fonction doit préexister dans l'espace mémoire du processus tiers pour pouvoir être exécutée.

Pour ces trois fonctions du système d'exploitation, l'appel ne peut réussir qu'à la condition que le descripteur de sécurité du processus cible autorise une telle action (l'ACL doit autoriser les opérations d'écriture et d'exécution pour l'utilisateur à l'origine de l'appel). En d'autres termes, **ces trois fonctions peuvent être utilisées par n'importe quel utilisateur sous condition que le processus cible leur appartienne**.

Dans le cas où l'ACL du processus interdit une telle manipulation (par exemple quand un utilisateur souhaite utiliser ces fonctions sur un processus système comme LSASS.EXE), l'appel à ces fonctions ne peut alors réussir qu'à l'unique condition de disposer du droit *SeDebugPrivilege*.

En conclusion, un utilisateur peut appeler ces fonctions sur les processus qui lui appartiennent, mais doit posséder le droit SeDebugPrivilege pour les appeler sur un processus ne lui appartenant pas.

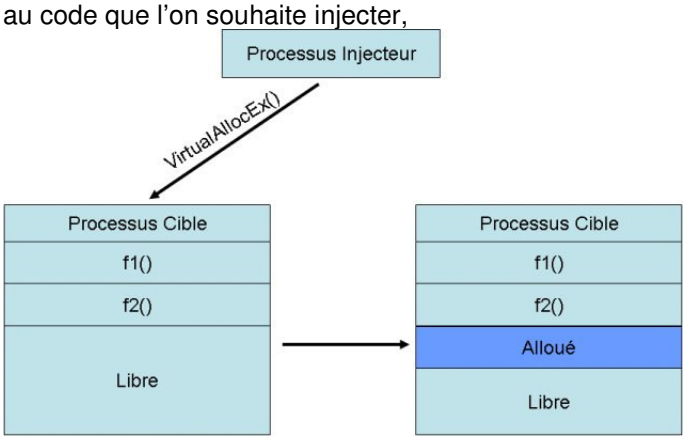
Passage à la pratique

Pour injecter du code dans un autre processus, il devient alors nécessaire de réaliser les opérations suivantes :

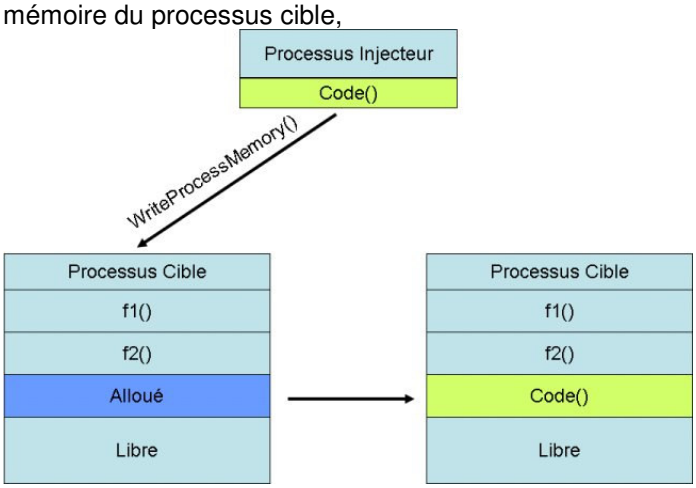
1. Activation du droit SeDebugPrivilege[1],

[1] Pour certains privilèges Windows, comme le SeDebugPrivilege, il ne suffit pas de disposer de ce droit pour l'utiliser ; il est nécessaire de l'activer explicitement avant de mener l'opération nécessitant ce droit.

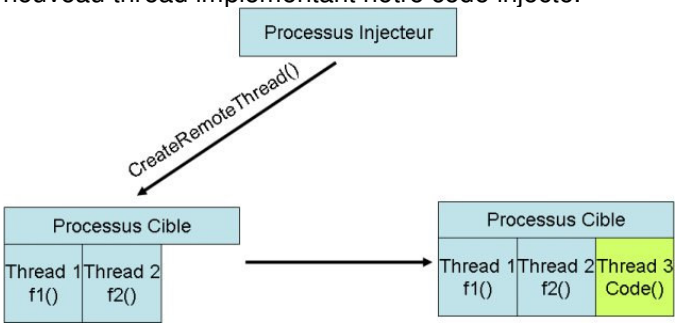
2. Appel à ***VirtualAllocEx()*** pour que le processus cible alloue l'espace nécessaire au code que l'on souhaite injecter,



3. Appel à ***WriteProcessMemory()*** afin de copier notre code dans l'espace mémoire du processus cible,



4. Appel à ***CreateRemoteThread()*** pour demander au processus cible de lancer un nouveau thread implémentant notre code injecté.



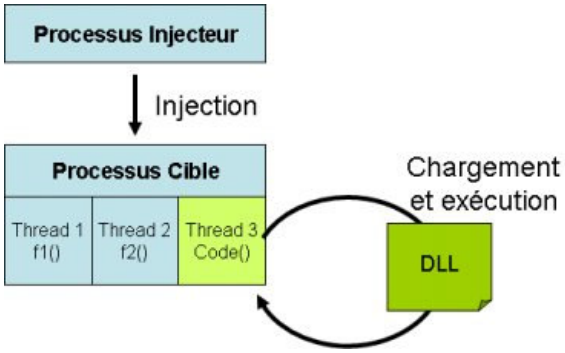
Pourquoi appelle t-on cette technique « l'injection de DLL » ? Parce qu'il n'est pas aisé de coder une grosse application destinée à être injectée telle quelle, et ce pour deux raisons principales ;

1. Ne disposant pas de références à l'environnement d'exécution du processus cible, les appels systèmes doivent être réalisé « à la main » en chargeant

explicitement les fonctions exportées par les DLLs systèmes via des appels à LoadLibrary() et GetProcAdress()[1].

2. Plus un segment de code est important, plus le risque que l'ensemble du code soit stocké dans des zones mémoires non contiguës augmente, ce qui gênerait considérablement la procédure de copie du code dans le processus cible.

Pour remédier à ces deux problèmes, on a recours à une astuce qui consiste à écrire le code que l'on souhaite faire exécuter par le processus cible dans une DLL exportant une la fonction en question.



Le code que l'on va alors injecter ne constitue alors qu'un simple code d'amorce qui va charger la DLL et exécuter la fonction qu'elle exporte. Cette fonction étant chargé par le processus cible, elle bénéficie de l'ensemble de l'environnement d'exécution et n'est donc plus gênée dans ses appels externes.

Intérêt de cette technique

Pour un agresseur, cette technique demeure précieuse parce qu'elle constitue une violation d'un principe de sécurité de base, à savoir l'isolation mémoire des processus entre eux.

Elle permet, par exemple, de faire tourner un programme en toute furtivité :

- un programme injecté dans un autre processus n'apparaîtra pas en tant que tel dans le gestionnaire des programmes (ce qui le rend donc « invisible »),
- le seul moyen d'être sûr d'arrêter un tel programme consiste à tuer le processus hôte, ce qui peut s'avérer gênant,
- un tel programme parasite pourrait parfaitement se répliquer de cette manière, infectant l'ensemble des processus utilisateurs ce qui le rendrait virtuellement immortel et indétectable.

Cette technique est également utilisée par de nombreux outils, dont le programme **pwdump3**, qui utilise cette technique d'injection dans le LSASS.EXE pour récupérer les cryptogrammes des mots de passe des utilisateurs. Plus inquiétant, certains chevaux de Troyes utilisent d'ailleurs déjà cette technique de reproduction.

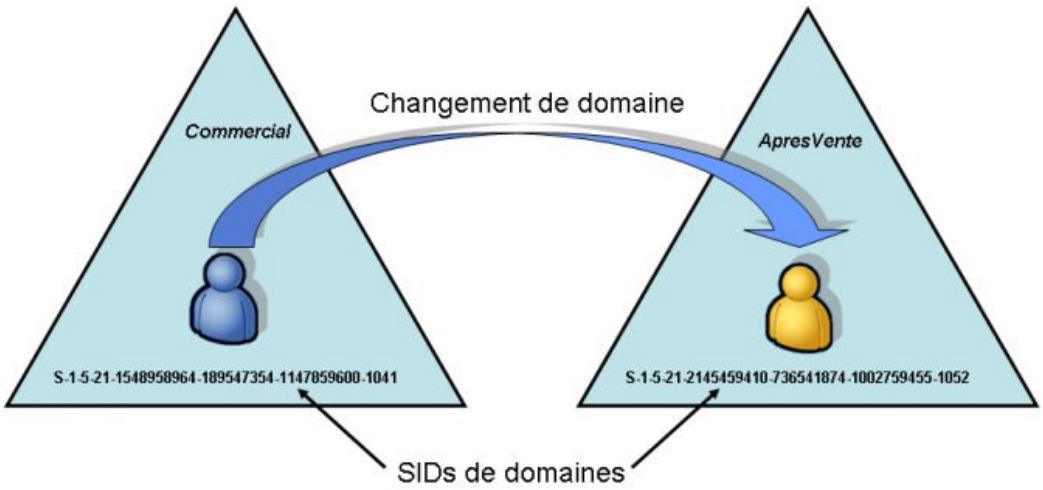
[1] Ce qui implique d'ailleurs que les adresses de ces deux fonctions doivent être transportées avec le code dans le processus injecté…

Le SID-History

Lorsqu'un groupe ou un utilisateur est créé, le SID de cet objet se retrouve stocké dans l'Active Directory en tant que propriété *Object-SID* de l'objet Utilisateur ou Groupe. L'Active Directory assigne également à chaque nouvel objet un GUID (globally unique identifier), représentant un nombre de 128 bits unique non seulement dans la forêt mais également dans le monde entier. Un GUID est assigné à tout objet de l'Active Directory et stocké dans sa propriété *Object-GUID*.

L'Active Directory utilise les GUID en interne afin d'identifier les objets. Ainsi le GUID est l'une des propriétés des objets qui est publiée dans le catalogue global de la forêt. Alors que les autres propriétés d'un objet peuvent évoluer dans le temps, un objet conserve son GUID pour la vie.

Dans certains cas, le SID d'un utilisateur peut changer[1]. En effet, il est possible de déplacer un utilisateur d'un domaine vers un autre domaine. Supposons ainsi que l'utilisateur Martin, appartenant à l'entreprise BIDON change d'affectation tout en restant dans la même entreprise. Alors qu'il appartenait au domaine BIDON\Commercial, il est affecté dans le domaine BIDON\ApresVente. Dans ce cas de figure, cet utilisateur a besoin d'un nouvel SID puisque, dans son SID, l'identificateur du domaine *Commercial* est différent de celui du domaine *ApresVente.*



Un nouvel SID pour son nouveau domaine est alors créé et vient remplacer le contenu de l'attribut *Object-SID* de son compte utilisateur. Avant d'écraser le contenu de cet attribut, l'ancienne valeur est copiée dans l'attribut *sIDHistory*. Cet attribut peut contenir plusieurs valeurs[2].

Lors d'un accès à une ressource, le sous-système de sécurité ne se contente pas de vérifier le SID de l'utilisateur pour le comparer au Security Descriptor de l'objet ; il teste également l'accès en fonction du contenu de l'attribut *sIDHistory*.

Ainsi, une ACL positionnée sur une ressource du domaine *Commercial* et autorisant l'accès à l'utilisateur Martin permettra toujours à Martin, après son déplacement dans un autre domaine, d'accéder à cette ressource et ce grâce au SID-History.

[1] En revanche, le SID d'un groupe ne change par contre jamais : un groupe reste confiné dans le domaine depuis lequel il a été créé.
[2] Cet attribut stocke donc tous les SIDs qu'un utilisateur a pu être amené à posséder.

Cette intéressante fonctionnalité du système constitue cependant une vulnérabilité majeure de conception du système :

L'existence même de ce mécanisme autorise un administrateur d'un domaine à violer la mécanique de cloisonnement inter domaine au sein d'une forêt. Il suffit à un administrateur de domaine de récupérer le SID d'un administrateur d'un autre domaine[1] et d'intégrer ce SID dans son propre sIDHistory pour obtenir immédiatement les privilèges d'administrateur sur l'autre domaine.

[1] L'opération, triviale et ne nécessitant aucun privilèges particuliers, s'effectue par un appel à l'API *LookUpAccountName()*.

Description des droits utilisateurs

Le système Windows 2000 définit deux types de droits d'utilisateur : les privilèges et les droits d'ouverture de session. Ce qui suit constitue une rapide description des différents droits positionnables.

Les Privilèges

Agir en tant que partie du système d'exploitation

Ce privilège permet à un processus de s'authentifier en tant que tout utilisateur et, par conséquent, d'avoir accès aux mêmes ressources que tout utilisateur. Seuls les services d'authentification de bas niveau ont besoin de ce privilège.

L'accès potentiel n'est pas limité à ce qui est associé à l'utilisateur par défaut, le processus appelant pouvant demander l'ajout dans le jeton d'accès de plusieurs autres accès arbitraires. De plus, le processus appelant peut créer un jeton anonyme fournissant un accès quelconque et tous les accès. Ce jeton ne fournit pas d'identité permettant de suivre les événements dans le journal d'audit.

Les processus nécessitant ce privilège doivent utiliser un compte l'incluant déjà, c'est-à-dire un compte LocalSystem, plutôt qu'un compte d'utilisateur séparé auquel le privilège a été affecté spécialement.

Ajouter des stations de travail au domaine

Permet à l'utilisateur d'ajouter un ordinateur à un domaine spécifique.

Sauvegarder des fichiers et des répertoires

Permet à l'utilisateur de contourner les autorisations sur les fichiers et les répertoires pour sauvegarder le système.

Outrepasser le contrôle de parcours

Ce droit permet à un utilisateur de traverser des répertoires d'une arborescence NTFS sur lesquels il n'a pas la permission de passage, en vue d'accéder à un fichier sur lequel il dispose de permissions suffisantes. Est utilisé par IIS pour autoriser l'accès sécurisé à une arborescence Web d'un serveur sans que le disque tout entier soit accessible à l'utilisateur. Ce privilège n'est pas auditable.

Modifier l'heure système

Permet à l'utilisateur de définir l'heure de l'horloge interne de l'ordinateur.

Créer un objet-jeton

Permet à un processus de créer un jeton et de s'en servir pour accéder aux ressources locales lorsqu'il utilise NtCreateToken() ou une autre API de création de jeton.

Les processus nécessitant ce privilège doivent utiliser un compte l'incluant déjà, c'est-à-dire un compte LocalSystem, plutôt qu'un compte d'utilisateur séparé auquel le privilège aurait été affecté spécialement. Ce privilège n'est pas auditable.

Créer des objets partagés permanents

Permet à un processus de créer un objet répertoire dans le gestionnaire d'objets de Windows 2000. Ce privilège est utile aux composants Mode noyau qui prévoient d'étendre l'espace des noms d'objets Windows 2000. Comme les composants exécutés en mode noyau disposent déjà de ce privilège, il est inutile de le leur affecter spécifiquement.

Créer un fichier paginé

Permet à l'utilisateur de créer un fichier paginé et d'en modifier la taille. Il doit spécifier la taille des fichiers paginés d'un lecteur donné dans les Options de performances des propriétés système. Nota : ce droit **n'est pas** utilisé sous Windows NT 4.0, bien que présent.

Déboguer des programmes

Ce droit permet à un utilisateur de déboguer les applications et les services, ainsi que la possibilité de modifier le comportement des threads en temps réel. Ce privilège n'est pas auditable. Donner ce droit à un utilisateur revient à lui donner les privilèges d'administration puisqu'il lui est désormais possible de modifier le comportement de processus systèmes (comme le LSASS.EXE) par une technique d'injection de DLL.

Autoriser que l'on fasse confiance aux comptes ordinateur et utilisateur pour la délégation

Permet à l'utilisateur de définir le paramètre Approuvé pour la délégation sur un objet utilisateur ou ordinateur. L'utilisateur ou l'objet auquel le privilège est accordé doit avoir le droit d'écrire dans les indicateurs de contrôle de compte de l'objet utilisateur ou ordinateur. Un processus serveur exécuté sur un ordinateur ou un utilisateur approuvé pour délégation peut accéder à des ressources d'un autre ordinateur. Il utilise les informations d'identification déléguées d'un client, pourvu que l'indicateur. Le compte est sensible et ne peut pas être délégué n'ait pas été défini sur le compte de ce client. Si ce privilège ou les paramètres Approuvé pour délégation sont mal utilisés, ils fragilisent le réseau s'il est agressé par des programmes de type cheval de Troie, qui empruntent l'identité de clients entrants et l'utilisent pour accéder aux ressources du réseau.

Forcer l'arrêt à partir d'un système distant

Permet à un utilisateur d'arrêter un ordinateur à partir d'un emplacement distant sur le réseau. Voir aussi le privilège « *Arrêter le système* ».

Générer des audits de sécurité

Permet à un processus d'inscrire des entrées dans le journal de sécurité en vue d'un audit des accès à l'objet. Ce processus peut également générer d'autres audits de sécurité. Le journal de sécurité sert à suivre les accès non autorisés au système. Voir aussi le privilège Gérer le journal d'audit et de sécurité.

Augmenter les quotas

Permet à un processus autorisé à accéder à un autre processus avec le droit Écrire la propriété d'augmenter le quota de processeurs affectés à cet autre processus. Ce privilège est utile pour le réglage du système mais risque d'être utilisé abusivement, par exemple dans une agression contre la protection du service.

Augmenter la priorité de planification

Ce droit permet à un utilisateur d'augmenter la priorité d'exécution d'un processus. Il n'est pas recommandé de laisser ce droit à un utilisateur dans la mesure il est serra

alors capable de donner une priorité de type "Temps réel" à un processus, ce qui occupera toutes les ressources CPU tant que le processus ne sera pas achevé.

Charger et décharger les pilotes de périphérique

Permet à un utilisateur d'installer et de désinstaller des pilotes de périphériques Plug-and-Play. Les pilotes de périphériques autres que Plug-and-Play ne sont pas affectés par ce privilège et ne peuvent être installés que par des administrateurs. Comme les pilotes de périphériques sont exécutés en tant que programmes approuvés (hautement privilégiés), ce privilège risque d'être détourné au profit de programmes malveillants qui pourraient accéder aux ressources et les détruire.

Verrouiller des pages mémoire

Ce droit permet à un utilisateur de verrouiller des pages mémoire de sorte que celles-ci ne seront plus atteintes par le mécanisme de "Write Back" utilisé par le fichier de swap (ces pages resteront en mémoire vive).

Gérer le journal d'audit et de sécurité

Permet à un utilisateur de spécifier les options d'audit de l'accès à des objets tels que des fichiers, des objets Active Directory et des clés de Registre, c'est-à-dire des ressources individuelles. L'audit de l'accès aux objets n'a réellement lieu que si vous l'avez activé dans les paramètres de stratégie d'audit au niveau de l'ordinateur ou dans Active Directory, sous Stratégie de groupe ; ce privilège ne donne pas accès à la stratégie d'audit décidée au niveau de l'ordinateur en entier.

Un utilisateur disposant de ce privilège peut également consulter et supprimer le journal de sécurité avec l'Observateur d'événements.

Modifier les valeurs d'env. de microprogrammation

Permet à un utilisateur, par les Propriétés système, ou à un processus, de modifier les variables d'environnement système.

Optimiser un processus

Permet de suivre les performances des processus non-système avec les outils d'analyse des performances de Windows NT et de Windows 2000.

Régler les performances système

Permet de suivre les performances des processus système avec les outils d'analyse des performances de Windows NT et de Windows 2000.

Remplacer un jeton niveau de processus

Permet à un processus de remplacer le jeton associé par défaut à un sous-processus déjà lancé.

Restaurer des fichiers et des répertoires

Permet à un utilisateur de contourner les autorisations sur les fichiers et les répertoires lorsqu'il restaure ces derniers et de définir un principal de sécurité quelconque comme propriétaire d'un objet. Voir aussi le privilège Sauvegarder des fichiers et des répertoires.

Arrêter le système

Permet à un utilisateur d'arrêter l'ordinateur local.

Prendre possession des fichiers ou d'autres objets

Permet à un utilisateur de prendre possession de tout objet du système nécessitant une sécurité, dont les objets Active Directory, les fichiers et les dossiers, les imprimantes, les clés de Registre, les processus et les threads.

Retirer l'ordinateur de la station d'accueil

Permet de déconnecter un ordinateur portable de sa station d'accueil, avec l'interface utilisateur Windows 2000.

Droits d'ouverture de session

Accéder à cet ordinateur depuis un réseau

Permet à un utilisateur de se connecter à l'ordinateur via le réseau. Par défaut, ce privilège est accordé aux personnes disposant des droits des niveaux suivants : Administrateurs, Tout le monde et Utilisateurs avec pouvoir.

Ouvrir une session en tant que tâche

Permet à utilisateur d'ouvrir une session avec une fonction de traitement différé. Par défaut, ce privilège est accordé aux administrateurs.

Ouvrir une session en tant que service

Permet à un principal de sécurité d'ouvrir une session en tant que service, ce qui est une façon d'établir un contexte de sécurité. Le compte LocalSystem conserve toujours le droit d'ouvrir une session en tant que service. Tout service exécuté sous un compte séparé doit disposer de ce droit. Par défaut, ce droit est accordé aux administrateurs.

Ouvrir une session localement

Permet à un utilisateur d'ouvrir une session sur le clavier de l'ordinateur. Par défaut, sur un serveur, ce droit est accordé aux personnes suivantes : Administrateurs, Opérateurs de compte, Opérateurs de sauvegarde, Opérateurs d'impression et Opérateurs de serveur. Sur les stations Windows 2000 Pro, ce droit est accordé au groupe « Tout le monde ».

Liste des services Windows courants

Note : *Le processus svchost est un processus « hôte » pouvant héberger des programmes Windows écrits sous la forme de DLLs ; il permet de développer un service sans se soucier de son interfaçage avec le système de gestion des services. Pour lister les services hébergés par un processus svchost, saisir dans une invite de commande MSDOS la commande «* `tasklist /SVC` *»*

Accès à distance au Registre
Nom de l'exécutable : svchost.exe
Nom interne : remote registry
Description : Permet aux utilisateurs à distance de modifier les paramètres du Registre sur cet ordinateur.
Commentaire : Pour des raisons évidentes de sécurité, ce service doit être « Désactivé ».

Acquisition d'image Windows (WIA)
Nom de l'exécutable : svchost.exe
Nom interne : sticvc
Description : Fournit des services d'acquisition d'images pour les scanners et les appareils photo.
Commentaire : Ce service peut être mis sur « manuel » compte tenu que les applications et matériels installés sur votre PC suffisent à assurer leur propre fonctionnalité.

Affichage des messages
Nom de l'exécutable : svchost.exe
Nom interne : Messenger
Description : Envoie et reçoit les messages des services d'alertes entre les clients et les serveurs. Ce service n'est pas lié à Windows Messenger.
Commentaire : Ce service doit être « désactivé » si vous ne souhaitez pas être submergé de publicités sur Internet.

Aide et support
Nom de l'exécutable : svchost.exe
Nom interne : helpsvc
Description : Permet à l'application « Aide et support » de fonctionner sur cet ordinateur.
Commentaire : Ce service peut être « désactivé » si vous connaissez bien votre système, sinon vous pouvez le laisser en « manuel » pour l'utiliser.

Appel de procédure distante (RPC)
Nom de l'exécutable : svchost
Nom interne : RpcSs
Description : Fournit le mappeur du point de sortie et divers services RPC.
Commentaire : **ce service doit être impérativement en « Automatique » pour assurer le bon fonctionnement de tous les services**, si vous le « désactivez », vous ne pouvez plus revenir en arrière, et vous êtes obligés de ré-installer votre OS.

Application système COM+
Nom de l'exécutable : dllhost.exe
Nom interne : COMSysApp

Description : Gère la configuration et le suivi des composants de base COM+ (Component Object Model).
Commentaire : Si le service est « arrêté », la plupart des composants de base COM+ ne fonctionneront pas correctement. Vous pouvez laisser ce service sur « automatique » pour assurer un fonctionnement optimal de votre PC. Pour une sécurité renforcée, mettez le en « manuel ».

Assistance TCP/IP NetBIOS
Nom de l'exécutable : svchost.exe
Nom interne : LmHosts
Description : Permet la prise en charge pour NetBIOS sur un service TCP/IP(NetBT) et la résolution des noms NetBIOS.
Commentaire : Si vous n'êtes pas en réseau et pour des raisons de sécurité, vous pouvez « désactiver » ce service, si vous êtes en réseau, laissez ce service en « automatique ».

Audio Windows
Nom de l'exécutable : svchost.exe
Nom interne : AudioSrv
Description : Gère les périphériques audio pour les programmes basés sur Windows.
Commentaire : Ce service doit être sur « automatique » sinon vous ne pourrez pas obtenir de son. Si vous souhaitez désactiver le son sur un PC, mettez ce service sur « manuel » ou « désactivé ».

Avertissement
Nom de l'exécutable : svchost.exe
Nom interne : Alerter
Description : Informe les utilisateurs et les ordinateurs sélectionnés des alertes administratives.
Commentaire : Vous pouvez mettre ce service en « désactivé » si vous ne souhaitez pas recevoir d'alerte.

Carte à puce
Nom de l'exécutable : SCardSvr.exe
Nom interne : SCardSvr
Description : Gère l'accès aux cartes à puce lues par cet ordinateur. Si ce service est arrêté, cet ordinateur ne pourra plus lire de cartes à puces.
Commentaire : Vous pouvez laisser ce service en « manuel ».

Carte de performance WMI
Nom de l'exécutable : wmiapsrv.exe
Nom interne : WmiApSrv
Description : Fournit des informations concernant la bibliothèque de performance à partir des fournisseurs HiPerf WMI.
Commentaire : Vous pouvez laisser ce service en « manuel ».

Cliché instantané de volume
Nom de l'exécutable : vssvc.exe
Nom interne : VSS
Description : Gère et implémente les clichés instantanés de volumes pour les sauvegardes et autres utilisations. Si ce service est arrêté, les clichés instantanés ne seront pas disponibles pour la sauvegarde et celle-ci échouera.
Commentaire : Vous pouvez laisser ce service en « manuel ».

Client de suivi de lien distribué
Nom de l'exécutable : svchost.exe
Nom interne : TrkWks
Description : Maintient les liens entre les fichiers NTFS au sein d'un ordinateur ou de plusieurs ordinateurs dans un domaine de réseau.
Commentaire : Vous pouvez laisser ce service en « désactivé ».

Client DHCP
Nom de l'exécutable : svchost.exe
Nom interne : Dhcp
Description : Gère la configuration réseau en inscrivant et en mettant à jour les adresses IP et les noms DNS.
Commentaire : Si vous êtes en réseau et que vous avez configuré vos adresses IP en manuel, vous pouvez laisser ce service en « désactivé ». Ce service n'a (curieusement) aucune influence pour surfer sur l'Internet.

Client DNS
Nom de l'exécutable : svchost.exe
Nom interne : Dnscache
Description : Résout et met en cache les noms DNS pour cet ordinateur.
Commentaire : Vous pouvez laisser ce service en mode « désactivé », ce service n'a (curieusement) aucune influence pour surfer sur l'Internet.

Compatibilité avec le Changement rapide d'utilisateur
Nom de l'exécutable : svchost.exe
Nom interne : FastUserSwitchingCompatibity
Description : Fournit un système de gestion à des applications qui nécessitent de l'Assistance dans un environnement d'utilisateurs multiples.
Commentaire : Si vous êtes seul à utiliser votre PC, vous pouvez « désactiver » ce service, sinon vous pouvez mettre ce service en « manuel » si vous avez configuré cette fonctionnalité dans le « Panneau de configuration / comptes d'utilisateurs ».

Configuration automatique sans fil
Nom de l'exécutable : svchost.exe
Nom interne : WZCSVC
Description : Fournit la configuration automatique des cartes 802.11.
Commentaire : Vous pouvez « désactiver » ce service si vous n'utilisez pas de connexion sans fil.

Connexion secondaire
Nom de l'exécutable : svchost.exe
Nom interne : seclogon
Description : Permet le démarrage des processus sous d'autres informations d'identification.
Commentaire : Ce service est utilisé par la commande RunAs. Il permet d'exécuter des applications avec les privilèges d'un autre utilisateur, par exemple en faisant un clic droit sur le raccourci de l'application puis « Exécuter en tant que ».

Connexions réseau
Nom de l'exécutable : svchost.exe
Nom interne : Netman
Description : Prend en charge les objets dans le dossier Connexions réseau et accès à distance, dans lequel vous pouvez afficher à la fois les connexions du réseau local et les connexions à distance.
Commentaire : Vous pouvez laisser ce service en « manuel ».

DDE réseau
Nom de l'exécutable : netdde.exe
Nom interne : NetDDE
Description : Fournit le transport en réseau et la sécurité pour l'échange dynamique de données pour les programmes exécutés sur un même ordinateur ou des ordinateurs différents.
Commentaire : Vous pouvez laisser ce service en mode « manuel ».

Détection matériel noyau
Nom de l'exécutable : svchost.exe
Nom interne : ShellHWDetection
Description : Fournit des notifications à des évènements matériels de lecture automatique
Commentaire : Vous devez laisser ce service en mode « automatique ».

Distributed Transaction Coordinator
Nom de l'exécutable : msdtc.exe
Nom interne : MSDTC
Description : Coordonne les transactions qui comportent plusieurs gestionnaires de ressources, tels que des bases de données, des files d'attente de messages et des systèmes de fichiers.
Commentaire : Si vous ne partagez pas de base de données ou que votre PC ne fais pas office de serveur, vous pouvez mettre ce service en mode « manuel ».

DSDM DDE réseau
Nom de l'exécutable : netdde.exe
Nom interne : NetDDEdsdm
Description : Gère l'échange dynamique de données partagées de réseau. Si ce service est arrêté, l'échange dynamique de données partagées de réseau ne sera plus disponible.
Commentaire : Si vous n'avez pas de réseau, vous pouvez mettre ce service en mode « désactivé » sinon, si vous vous partagez des ressources, vous pouvez le mettre en mode « manuel ».

Emplacement protégé
Nom de l'exécutable : lsass.exe
Nom interne : ProtectedStorage
Description : Fournit un stockage protégé pour les données sensibles, telles que les clés privées, afin d'empêcher l'accès par des services, des processus ou des utilisateurs non autorisés.
Commentaire : Si votre système de fichier est en NTFS et que vous chiffrez vos données, il est préférable de laisser ce service en mode « automatique », sinon, vous pouvez le laisser en mode « manuel ».

Explorateur d'ordinateur
Nom de l'exécutable : svchost.exe
Nom interne : Browser
Description : Tient à jour une liste des ordinateurs présents sur le réseau et fournit cette liste aux ordinateurs désignés comme navigateurs. Si ce service est arrêté, la liste ne sera pas mise ou tenue à jour.
Commentaire : Si vous n'avez pas de réseau domestique, vous pouvez laisser ce service en mode « manuel », sinon, laissez le en mode « automatique ».

Extensions du pilote WMI
Nom de l'exécutable : svchost.exe
Nom interne : Wmi

Description : Fournit des informations de gestion du système vers et à partir des pilotes.
Commentaire : Vous pouvez mettre ce service en mode « manuel ».

Fournisseur de la prise en charge de sécurité LM NT
Nom de l'exécutable : lsass.exe
Nom interne : NtLmSsp
Description : Assure la sécurité des programmes RPC (appels de procédure distante) qui utilisent des transports autres que des canaux nommés.
Commentaire : Vous pouvez mettre ce service en mode « manuel ».

Gestion d'applications
Nom de l'exécutable : svchost.exe
Nom interne : AppMgmt
Description : Fournit des services d'installation de logiciels tels que Attribuer, Publier et Supprimer.
Commentaire : Vous pouvez mettre ce service en mode « manuel ».

Gestionnaire de comptes de sécurité
Nom de l'exécutable : lsass.exe
Nom interne : SamSs
Description : Stocke les informations de sécurité pour les comptes d'utilisateurs locaux.
Commentaire : Si vous avez modifié des paramètres de sécurité avec « gpedit.msc », laissez ce service en mode « automatique », sinon, vous pouvez le laisser en mode « manuel ».

Gestionnaire de connexion automatique d'accès distant
Nom de l'exécutable : svchost.exe
Nom interne : RasAuto
Description : Crée une connexion vers un réseau distant à chaque fois qu'un programme référence un nom ou une adresse DNS ou NetBIOS distant.
Commentaire : Si vous utilisez une connexion Internet par modem, vous pouvez laisser ce service en mode « automatique », sinon, vous pouvez le laisser en mode « manuel ».

Gestionnaire de connexions d'accès distant
Nom de l'exécutable : svchost.exe
Nom interne : RasMan
Description : Crée une connexion réseau.
Commentaire : Si vous utilisez une connexion réseau local ou Internet, laissez ce service en mode « automatique », sinon, vous pouvez le mettre en « manuel ».

Gestionnaire de disque logique
Nom de l'exécutable : svchost.exe
Nom interne : dmserver
Description : Détecte et analyse de nouveaux lecteurs de disque durs et envoie les informations de volume de disque au service gestionnaire administratif de disque logique pour la configuration. Si ce service est arrêté, l'état des disques dynamiques et les informations de configuration peuvent devenir obsolètes.
Commentaire : Vous devez laisser ce service en mode « automatique ».

Gestionnaire de l'Album
Nom de l'exécutable : clipsrv.exe

Nom interne : ClipSrv
Description : Active le Gestionnaire de l'Album afin de stocker les informations et les partager avec des ordinateurs à distance. Si le service est arrêté, le Gestionnaire de l'Album ne pourra pas partager les informations avec des ordinateurs à distance.
Commentaire : Vous pouvez laisser ce service en mode « manuel », pour des raisons de sécurité et si vous ne partagez rien avec d'autres ordinateurs à distance, vous pouvez « désactiver » ce service.

Gestionnaire de session d'aide sur le Bureau à distance
Nom de l'exécutable : sessmgr.exe
Nom interne : RDSessMgr
Description : Gère et contrôle l'assistance à distance. Si ce service est arrêté, l'assistance à distance n'est pas disponible.
Commentaire : Pour des raisons de sécurité vous pouvez « désactiver » ce service, sinon, laissez le en mode « manuel ».

Gestionnaire de téléchargement
Nom de l'exécutable : svchost.exe
Nom interne : uploadmgr
Description : Gère les transferts de fichiers synchrones et asynchrones entre les clients et les serveurs sur le réseau. Si ce service est arrêté, les transferts de fichiers synchrones et asynchrones entre les clients et les serveurs ne seront pas possibles.
Commentaire : Vous pouvez mettre ce service en mode « manuel ».

Horloge Windows
Nom de l'exécutable : svchost.exe
Nom interne : W32Time
Description : Conserve la synchronisation de la date et de l'heure sur tous les clients et serveurs sur le réseau. Si ce service est arrêté, la synchronisation de la date et de l'heure sera indisponible.
Commentaire : A moins de toujours vouloir être à l'heure d'Internet, vous pouvez « désactiver » ce service.

Hôte de périphérique universel Plug-and-Play
Nom de l'exécutable : svchost.exe
Nom interne : upnphost
Description : Offre la prise en charge des périphériques hôtes universels Plug-and-Play.
Commentaire : Vous pouvez laisser ce service en mode « manuel ».

Infrastructure de gestion Windows
Nom de l'exécutable : svchost.exe
Nom interne : winmgmt
Description : Fournit une interface commune et un modèle objet pour accéder aux informations de gestion du système d'exploitation, des périphériques, des applications et des services. **Si ce service est arrêté, la plupart des logiciels sur base Windows ne fonctionneront pas correctement**.
Commentaire : Vous devez laisser ce service en mode « automatique ».

Journal des événements
Nom de l'exécutable : services.exe
Nom interne : Eventlog
Description : Active les messages d'événements émis par les programmes fonctionnant sous Windows et les composants devant être affichés

dans l'observateur d'événements. Ce service peut être « désactivé », mais s'il est en fonctionnement il ne peut être « arrêté ».
Commentaire : Il est préférable de laisser ce service en mode « automatique ».

Journaux et alertes de performance
Nom de l'exécutable : smlogsvc.exe
Nom interne : SysmonLog
Description : Collecte les données de performances des ordinateurs locaux ou distants basés sur des paramètres planifiés pré configurés, puis écrit les données dans un journal ou déclenche une alerte.
Commentaire : Ce service n'a aucune dépendance ; pour des raisons de sécurité, mettez ce service en mode « désactivé », sinon vous pouvez le laisser en mode « manuel ».

Localisateur d'appels de procédure distante (RPC)
Nom de l'exécutable : locator.exe
Nom interne : RpcLocator
Description : Gère la base de données du service de nom RPC.
Commentaire : Vous pouvez laisser ce service en mode « manuel ».

Machine Debug Manager
Nom de l'exécutable : mdm.exe
Nom interne : MDM
Description : Débuggage de programme avec Visual Studio Debuggers
Commentaire : Vous pouvez « désactiver » ce service.

Mises à jour automatiques
Nom de l'exécutable : svchost.exe
Nom interne : wuauserv
Description : Active le téléchargement et l'installation de mises à jour Windows critiques. Depuis Septembre 2004, le service doit être démarré pour utiliser le service Web « Microsoft Windows Update ».
Commentaire : Vous pouvez « désactiver » ce service, à condition de le réactiver lorsque vous souhaitez faire des mises à jours avec Windows Update.

MS Software Shadow Copy Provider
Nom de l'exécutable : dllhost.exe
Nom interne : SwPrv
Description : Gère les copies logicielles de clichés instantanés de volumes créés par le service de cliché instantané de volumes. Si ce service est arrêté, les copies logicielles de clichés instantanés ne peuvent pas être gérées.
Commentaire : Ce service est gourmand en espace disque et son utilité demeure douteuse ; à « désactiver ».

NLA (Network Location Awareness)
Nom de l'exécutable : svchost.exe
Nom interne : Nla
Description : Recueille et stocke les informations de configuration et d'emplacement réseau, et notifie les applications quand ces informations changent.
Commentaire : Laissez ce service en mode « manuel ».

Notification d'événement système
Nom de l'exécutable : svchost.exe
Nom interne : SENS
Description : Scrute les événements systèmes tels que les ouvertures de session Windows et les événements concernant le réseau et

l'alimentation. Avertit les abonnés du système d'événements COM+ de ces événements.
Commentaire : Vous pouvez « désactiver » le service.

Numéro de série du média portable
Nom de l'exécutable : svchost.exe
Nom interne : WmdmPmSp
Description : Lit le numéro de série du baladeur numérique connecté à votre ordinateur.
Commentaire : Vous pouvez laisser ce service en mode « manuel ».

Onduleur
Nom de l'exécutable : ups.exe
Nom interne : UPS
Description : Gère un onduleur connecté à l'ordinateur.
Commentaire : Si vous ne possédez pas d'onduleur, vous pouvez « désactiver » ce service.

Ouverture de session réseau
Nom de l'exécutable : lsass.exe
Nom interne : Netlogon
Description : Prend en charge l'authentification directe des événements d'ouverture de session du compte pour les ordinateurs dans un domaine.
Commentaire : Vous pouvez laisser ce service en mode « manuel » si vous êtes en réseau local, sauf si vous êtes sur un domaine ou vous devez le laisser en « automatique », si vous n'avez pas de réseau, vous pouvez « désactiver » le service.

Pare-feu de connexion Internet (ICF) / Partage de connexion Internet (ICS)
Nom de l'exécutable : svchost.exe
Nom interne : SharedAccess
Description : Assure la traduction d'adresses de réseau, l'adressage, les services de résolution de noms et/ou les services de prévention d'intrusion pour un réseau de petite entreprise ou un réseau domestique.
Commentaire : Le Firewall intégré à Windows XP a au moins le mérite d'exister ☺ ; de préférence, « désactiver » le Firewall et en installer d'un éditeur tiers.

Partage de Bureau à distance NetMeeting
Nom de l'exécutable : mnmsrvc.exe
Nom interne : mnmsrvc
Description : Permet aux personnes autorisées d'accéder à votre Bureau Windows en utilisant NetMeeting.
Commentaire : Pour des raisons de sécurité, vous pouvez « désactiver » ce service si vous n'utilisez pas ce logiciel.

Planificateur de tâches
Nom de l'exécutable : svchost.exe
Nom interne : Schedule
Description : Permet à un utilisateur de configurer et de planifier des tâches automatisées sur cet ordinateur. Si ce service est arrêté, ces tâches ne seront pas exécutées.
Commentaire : Si vous avez l'habitude de gérer vos taches manuellement (nettoyage, défragmentation, etc..) vous pouvez « désactiver » ce service, sinon, laissez-le sur « automatique ».

Plug-and-Play
Nom de l'exécutable : services.exe
Nom interne : PlugPlay

Description : Permet à l'ordinateur de reconnaître et d'adapter les modifications matérielles avec peu ou pas du tout d'intervention de l'utilisateur. **Arrêter ou désactiver ce service provoque une instabilité du système**.
Commentaire : Il est préférable de laisser ce service en mode « automatique » car beaucoup d'autres services en dépendent.

Prise en charge des cartes à puces
Nom de l'exécutable : SCardSvr.exe
Nom interne : SCardDrv
Description : Permet la prise en charge des lecteurs de cartes à puce non Plug-and-Play hérités utilisés par cet ordinateur. Si ce service est arrêté, cet ordinateur ne supportera pas de lecteur hérité.
Commentaire : Vous pouvez laisser ce service en mode « manuel ».

QoS RSVP
Nom de l'exécutable : rsvp.exe
Nom interne : RSVP
Description : Fournit la signalisation de réseau et la fonctionnalité d'installation du contrôle de trafic local pour les programmes reconnaissant QoS et les applets de contrôle.
Commentaire : Vous pouvez laisser ce service en mode « manuel ».

Routage et accès distant
Nom de l'exécutable : svchost.exe
Nom interne : RemoteAccess
Description : Offre des services de routage dans les environnements de réseau local ou étendu.
Commentaire : Pour des questions de sécurité, laissez ce service « désactivé », sauf si vous utilisez le partage de connexion ICS, laissez ce service sur « automatique ».

Serveur
Nom de l'exécutable : svchost.exe
Nom interne : lanmanserver
Description : Prend en charge le partage de fichiers Windows, d'impression et des canaux nommés via le réseau pour cet ordinateur.
Commentaire : Sur un poste client connecté à l'Internet vous devez « désactiver » ce service. Sur un réseau local d'entreprise seuls les serveurs devraient exécuter ce service en mode « automatique ».

Service COM de gravage de CD IMAPI
Nom de l'exécutable : imapi.exe
Nom interne : ImapiService
Description : Gère la gravure des CD via l'interface série IMAPI (Image Mastering Applications Programming Interface). Si ce service est arrêté, cet ordinateur ne pourra plus enregistrer de CD depuis les outils Windows (Windows Media Player par exemple).
Commentaire : Si vous utilisez un autre logiciel de gravure (Nero, Easy CD Creator, etc.), vous pouvez « désactiver » ce service.

Service d'administration du Gestionnaire de disque logique
Nom de l'exécutable : dmadmin.exe
Nom interne : dmadmin
Description : Configure les lecteurs de disque durs et les volumes. Le service ne s'exécute que pour les processus de configuration puis s'arrête.
Commentaire : Vous devez laisser ce service en mode « manuel ».

Service de découvertes SSDP
Nom de l'exécutable : svchost.exe
Nom interne : SSDPSRV

Description : Active la découverte de périphériques Plug and Play universels sur votre réseau domestique.
Commentaire : Si votre PC est connecté à des périphériques UPnP, laissez ce service en « automatique », sinon, laissez le en mode « manuel ». Le service « Hôte de périphérique universel Plug-and-Play » est un autre nom de ce service.

Service de la passerelle de la couche Application
Nom de l'exécutable : alg.exe
Nom interne : ALG
Description : Fournit la prise en charge des plugins de protocoles tiers pour le partage de connexion Internet et le pare-feu Internet.
Commentaire : Ce service fournit les mécanismes de base au Firewall intégré de Windows XP. Si vous utilisez un autre Firewall, ce dernier a de forte chances de ne pas utiliser ce service : vous pouvez dès lors le « désactiver ».

Service de rapport d'erreurs
Nom de l'exécutable : svchost.exe
Nom interne : ERSvc
Description : Active le rapport d'erreurs pour les services et les applications s'exécutant sur des environnements non standard.
Commentaire : Vous pouvez « désactiver » ce service, sauf si vous souhaitez envoyer des rapports d'erreur à Microsoft.

Service de restauration système
Nom de l'exécutable : svchost.exe
Nom interne : srservice
Description : Effectue des opérations de restauration du système. Pour arrêter ce service, désactivez Restauration du système dans l'onglet Restauration du système des propriétés du Poste de travail.
Commentaire : Si vous ne souhaitez pas faire de restauration système, vous pouvez laisser ce service en mode « manuel », sinon, si vous utilisez ce service, laissez-le en mode « automatique ». Pour un usage domestique, le service de restauration permet de revenir « en arrière » en cas d'installation d'un logiciel défectueux ou perturbant le fonctionnement du système.

Service de transfert intelligent en arrière-plan
Nom de l'exécutable : svchost.exe
Nom interne : BITS
Description : Utilise la bande passante réseau inactive pour transférer des données.
Commentaire : Cette fonctionnalité est peu utilisée et nécessite que les applications soient développées pour tenir compte de ce service ; vous pouvez laisser ce service en mode « manuel » voire le « désactiver ». Si vous faites beaucoup de transferts de données depuis Internet Explorer, il est préférable de laisser ce service en mode « automatique ».

Service d'indexation
Nom de l'exécutable : cisvc.exe
Nom interne : cisvc
Description : Construit un index des contenus et des propriétés des fichiers sur les ordinateurs locaux et distants ; fournit un accès rapide aux fichiers par le biais d'un langage d'interrogation flexible.
Commentaire : Si vous ne souhaitez pas faire un catalogue de vos données, vous pouvez « désactiver » ce service, sinon, vous pouvez construire votre catalogue dans « Outils d'administration => gestion de l'ordinateur =>service et application => service d'indexation ». Dans ce cas, vous devez mettre le service en mode

« automatique ». Dans la majorité des cas ce service est totalement inutile.

Services de cryptographie
Nom de l'exécutable : svchost.exe
Nom interne : CryptSvc
Description : Fournit trois services de gestion : le service de base de données de catalogue, qui confirme la signature des fichiers Windows ; le service de racine protégée, qui ajoute et supprime des certificats d'autorité de certification de racine approuvés et le service Clé, qui fournit une aide dans l'inscription de cet ordinateur pour les certificats.
Commentaire : Pour plus de sécurité vous devez laisser ce service en mode « automatique ».

Services IPSEC
Nom de l'exécutable : lsass.exe
Nom interne : PolicyAgent
Description : Gère la stratégie de sécurité IP et démarre les pilotes de gestion de sécurité IP et ISAKMP/Oakley (IKE).
Commentaire : Vous pouvez mettre ce service en mode « manuel », voire le « désactiver » si vous n'utilisez jamais les fonctionnalités IPSEC.

Services Terminal Server
Nom de l'exécutable : svchost.exe
Nom interne : TermService
Description : Permet à plusieurs utilisateurs de se connecter en même temps à un ordinateur, tout en affichant les bureaux et les applications sur les ordinateurs distants. Contient les fonctions sous-jacentes de Bureau à distance (y compris le Bureau à distance pour les administrateurs), le Changement rapide d'utilisateur, l'Assistance à distance et le service Terminal Server.
Commentaire : Si vous utilisez le changement rapide d'utilisateur, vous devez laisser ce service en mode « automatique », sinon, pour plus de sécurité, vous pouvez « désactiver » ce service.

Spouleur d'impression
Nom de l'exécutable : spoolsv.exe
Nom interne : Spooler
Description : Charge des fichiers en mémoire pour une impression ultérieure.
Commentaire : Si vous utilisez régulièrement une imprimante, laissez ce service en mode « automatique », si vous utilisez très peu l'imprimante ou voire pas du tout, mettez-le en mode « manuel », dans ce cas, n'oubliez pas de redémarrer le service avant si vous souhaitez imprimer.

Station de travail
Nom de l'exécutable : svchost.exe
Nom interne : lanmanworkstation
Description : Crée et maintient des connexions de réseau client à des serveurs distants. Si ce service est arrêté, ces connexions ne seront pas disponibles.
Commentaire : Laisser ce service en mode « automatique ».

Stockage amovible
Nom de l'exécutable : svchost.exe
Nom interne : NtmsSvc
Description : Gère les médias amovibles, les lecteurs et les bibliothèques.
Commentaire : Si vous utilisez un lecteur Ioméga par exemple, laissez ce service en mode « automatique », sinon, laissez le en mode « manuel ».

Système d'événements de COM+
Nom de l'exécutable : svchost.exe
Nom interne : EventSystem
Description : Prend en charge le service de notification d'événements système (SENS, System Event Notification Service), qui fournit une distribution automatique d'événements aux composants COM (Component Object Model) abonnés. Si le service est arrêté, SENS sera fermé et ne pourra fournir des informations d'ouverture et de fermeture de session.
Commentaire : Vous pouvez mettre ce service en mode « manuel » si vous êtes le seul à utiliser le PC, sinon, laissez ce service en mode « automatique ».

Téléphonie
Nom de l'exécutable : svchost.exe
Nom interne : TapiSrv
Description : Fournit la prise en charge des API de téléphonie (TAPI) pour les programmes contrôlant les périphériques de téléphonie, les connexions vocales basées sur le protocole IP, sur l'ordinateur local, via le réseau local, sur le serveur où ce service fonctionne également.
Commentaire : Laissez ce service en mode « automatique ».

Telnet
Nom de l'exécutable : tlntsvr.exe
Nom interne : TlntSvr
Description : Permet à un utilisateur distant de se connecter au système et d'exécuter des programmes et prend en charge divers clients Telnet TCP/IP dont les ordinateurs sous UNIX et sous Windows.
Commentaire : Pour des raisons de sécurité, il est préférable de « désactiver » ce service.

Thèmes
Nom de l'exécutable : svchost.exe
Nom interne : Thèmes
Description : Fournit un système de gestion de thèmes graphiques.
Commentaire : Laissez ce service en mode « automatique » si vous utilisez les thèmes de Windows. Le fait de « désactiver » ce service remet par défaut le thème en 2D de Windows 2000, moins consommateur de ressources.

WebClient
Nom de l'exécutable : svchost.exe
Nom interne : WebClient
Description : Permet à un programme fonctionnant sous Windows de créer, modifier et accéder à des fichiers Internet. Si ce service est arrêté, Ces fonctions ne seront pas disponibles.
Commentaire : Pour des raisons de sécurité, il est préférable de « désactiver » ce service.

Windows Installer
Nom de l'exécutable : msiexec.exe
Nom interne : MSIServer
Description : Installe, répare et supprime des logiciels selon les instructions contenues dans les fichiers .MSI.
Commentaire : Laissez ce service en mode « manuel »

Création d'une stratégie IPSec

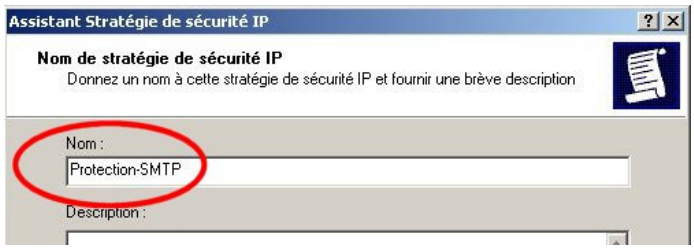
Microsoft Windows 2000 comprend trois stratégies de sécurité pré-installées de base, dont les niveaux vont de « *non sécurisé* » à « *fortement sécurisé* ». Cependant, les administrateurs souhaiteront probablement créer leur propre stratégie pour les faire correspondre à leurs besoins et leurs contraintes particulières.

Dans l'exemple qui suit, on supposera que l'on souhaite sécuriser par IPSEC les échanges SMTP entre deux serveurs de messagerie « dbruz01.celar » et « dbruz03.celar », tous les deux membres d'une même forêt Active Directory.

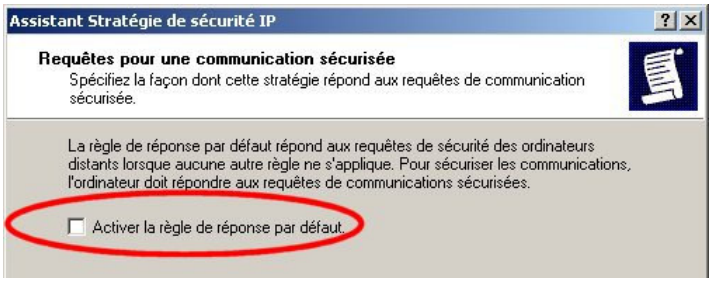
La création d'une stratégie IPSec sur un contrôleur de domaine s'effectue comme suit :

1. Ouvrir l'Explorateur Windows ; cliquer sur *Démarrer*, pointer sur *Programmes*, pointer sur *Outils d'administration*, puis cliquer sur *Stratégie de sécurité du contrôleur de domaine*.

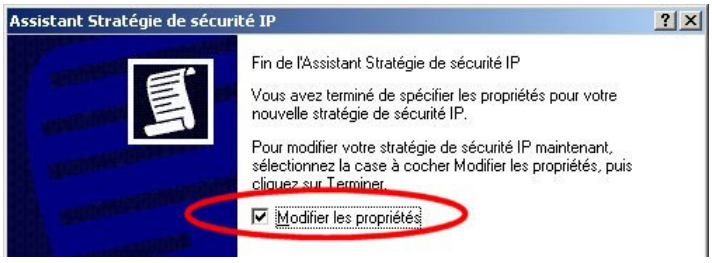
2. Développer *Paramètres de sécurité*, cliquer avec le bouton droit *sur Stratégies de sécurité IP sur Active Directory*, puis cliquer sur *Créer une stratégie de sécurité IP*. L'Assistant Stratégie de sécurité IP apparaît. Choisir un nom pour cette stratégie.



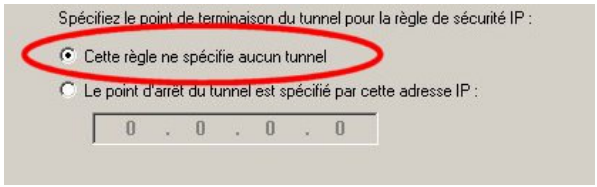
3. Cliquer sur *Suivant* et décocher la case Activer la règle de réponse par défaut.



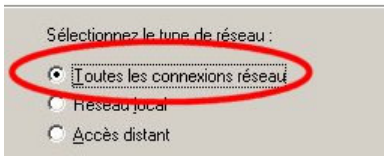
4. Cliquer sur *Suivant*, vérifier que la case *Modifier les propriétés* est bien cochée (c'est le cas par défaut), puis valider.



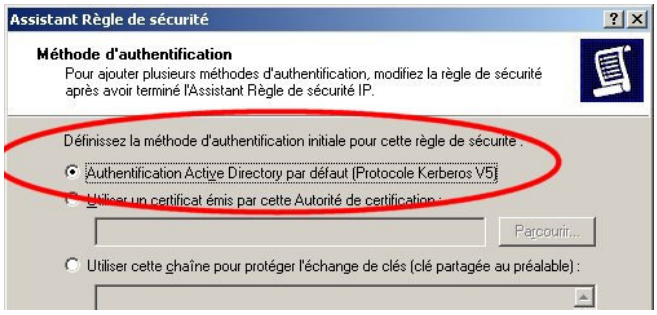
5. Cliquer sur *Ajouter* puis sur *Suivant* : dans la boîte de dialogue qui s'affiche alors, cocher la case *Cette règle ne spécifie aucun tunnel*.



6. Valider la règle pour tout type de connexion.

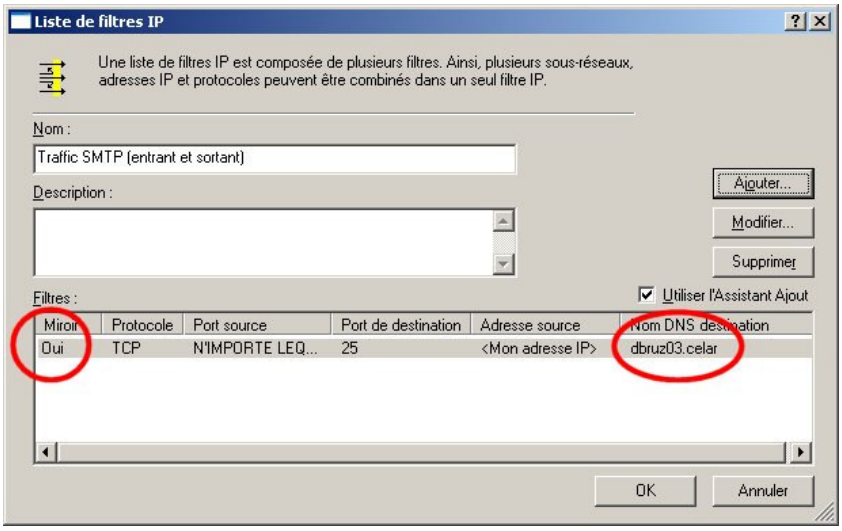


7. Sélectionner l'authentification Kerberos par défaut.



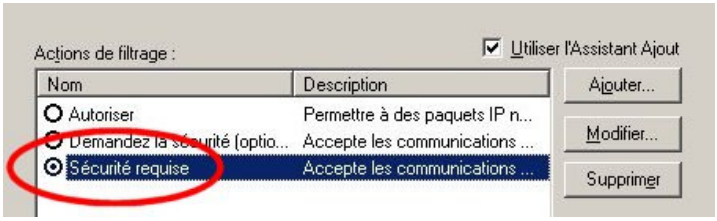
8. Dans la boite de dialogue qui suit, cliquer sur *Ajouter* afin de créer un filtre spécifique pour le protocole SMTP.

9. Configurer le filtre pour autoriser les connexions TCP vers le port 25, depuis sa machine et vers le serveur distant ; il s'agit donc d'une connexion sortant vers le serveur de messagerie. L'option Miroir permet la création implicite d'une règle inverse (connexion entrante sur le port 25 et depuis le serveur de messagerie). Sans cette option, il aurait fallu créer une règle en ce sens.

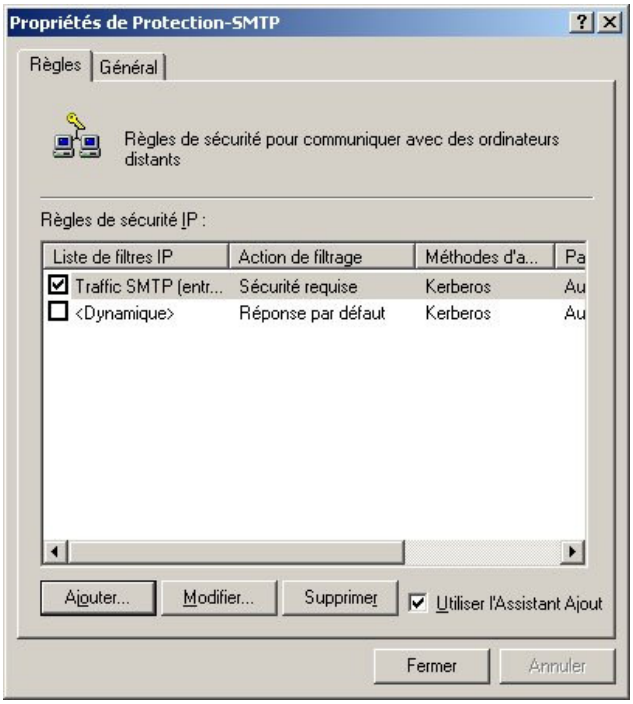


10. Valider, vérifier que le nouveau filtre est sélectionné et cliquer sur Suivant.

11. Sélectionner l'option *Sécurité Requise,* afin d'imposer le chiffrement des données, puis valider.



12. Vérifier que la case correspondant au filtre nouvellement créé est bien cochée et valider.



13. Il ne reste plus qu'à créer la même règle (inversée) sur le second serveur et à appliquer les stratégies sur les machines

Outils de sécurité

La liste suivante est très largement inspirée de la page « Top 75 security tools », disponible sur le site Internet de Fyodor, auteur de l'outil « nmap » (http://www.insecure.org/tools.html).

Pour chaque outil, on précise les points suivants :

Outil commercial, payant, parfois disponible en version limitée.

Outil disponible pour plate-forme Linux.

Outil disponible pour plate-formes FreeBSD/NetBSD/OpenBSD et/ou UNIX propriétaire (Solaris, HP-UX, IRIX, etc.).

Outil disponible pour plate-forme Windows.

OS	Description
	**Achilles** http://achilles.mavensecurity.com/ Un outil de type « web Proxy », permettant de réaliser des attaques de type « man in the middle » sur le protocole HTTP.
	**AirSnort** http://airsnort.shmoo.com/ AirSnort est un outil pour réseaux sans fils, permettant de retrouver les clefs de chiffrement de ces réseaux.
	**Cain & Abel** http://www.oxid.it/cain.html Le « l0phtcrack du pauvre ». Un outil permettant de retrouver les mots de passe dans les environnements Windows (essentiellement Windows 95 et 98).
	**Crack / Cracklib** http://www.users.dircon.co.uk/~crypto/ Le premier cracker de mots de passe Unix du genre, par Alec Muffec.
	**Dsniff** http://naughty.monkey.org/~dugsong/dsniff/ Un ensemble d'outils pour l'audit réseau et la pénétration de systèmes, partiellement portés et maintenus sous Windows.
	**Ethereal** http://www.ethereal.com/ Un sniffer réseau performant et surtout gratuit !
	**Ettercap** http://ettercap.sourceforge.net/ Ettercap est un outil en ligne de commande pour sniffer / intercepter / enregistrer les communications sur un réseau.

OS	Description
	
	**Firewalk** http://www.packetfactory.net/projects/firewalk/ Firewalk est un « traceroute » amélioré, permettant à la fois la découverte de nœuds intermédiaires mais également les options de filtrages positionnées.
	**Fport** http://www.foundstone.com/index.htm?subnav=resources/navigation.htm&subcontent=/resources/proddesc/fport.htm Fport est un « netstat » amélioré, permettant de montrer localement quels sont les ports de services ouverts et quelles applications écoutent sur ces ports. Ne tourne que sous Windows (sous Linux, la commande « *netstat –pan* » a le même effet).
	**Fragroute** http://www.monkey.org/~dugsong/fragroute/ Outil de test de routeurs filtrants et de Firewalls.
	**GFI Languard** http://www.gfi.com/lannetscan/ Un outil réseau d'énumération pour machines Windows, agrémenté d'une GUI. Fonctionne selon les mêmes principes que Winfingerprint.
	**Hping2** http://www.hping.org/ Un générateur de paquets réseau capable d'analyser les réponses aux requêtes émises. Le complément indispensable à *nmap*.
	**Hunt** http://lin.fsid.cvut.cz/~kra/index.html#HUNT Un outil de captures réseau, permettant d'implémenter des attaques de type TCP Hijacking.
	**ISS Internet Scanner** http://www.iss.net/products_services/enterprise_protection/vulnerability_assessment/scanner_internet.php Le produit phare commercial pour la détection de vulnérabilités sur un réseau. Très complet et puissant mais, hélas, hors de prix.
	**John the ripper** http://www.openwall.com/john/ Un cracker de mots de passe multi plateformes. Permet de casser les mots de passe Unix et NT, par brute force et dictionnaires avec ou sans règles de compositions
	**Kismet** http://www.kismetwireless.net/ Un très puissant sniffer pour réseaux sans fils (802.11b).
	**L0phtcrack** http://www.atstake.com/research/lc/ Un cracker de mots de passe pour Windows NT. Particulièrement rapide en attaque exhaustive.

OS	Description
	**NBTScan** http://www.inetcat.org/software/nbtscan.html Un outil réseau d'énumération pour machines Windows. Fonctionne selon les mêmes principes que Winfingerprint.
	**Nessus** http://www.nessus.org/ **LE** Scanner de vulnérabilités Open Source
	**Nmap** http://www.insecure.org/nmap/ Le scanner de port de service le plus complet à l'heure actuelle, par Fyodor. Inclut un module de détection de systèmes d'exploitation. La version Windows est généralement en retard par rapport à la version Unix.
	**Netcat** http://www.atstake.com/research/tools/network_utilities/ L'incontournable « couteau suisse » du réseau. Outil en ligne de commande.
	**Network Stumbler** http://www.stumbler.net/ Un autre sniffer pour réseaux sans fil 802.11, très employé pour le *wardriving*.
	**Ngrep** http://www.packetfactory.net/projects/ngrep/ Le « grep » du réseau. Permet de réaliser des captures réseaux.
	**Nikto** http://www.cirt.net/code/nikto.shtml Un scanner de vulnérabilités HTTP très complet.
	**N-Stealth** http://www.nstalker.com/nstealth/ Un autre scanner de vulnérabilités, commercial cette fois et plus souvent mis à jour que d'autres scanners du même type.
	**Ntop** http://www.ntop.org/ Le « top » (utilitaire unix d'occupation du CPU) du réseau. Permet de monitorer le trafic sur un réseau.
	**Pwdump3** http://www.polivec.com/pwdump3.html Récupère la base des comptes Windows NT (base SAM) et la formate sous la forme d'un fichier récupérable par John The Ripper et L0phtcrack.
	**Retina** http://www.eeye.com/html/Products/Retina/index.html Un autre scanner de vulnérabilités généraliste, tout comme Nessus et ISS.

OS	Description
	**SAINT** http://www.saintcorporation.com/saint/ Security Administrator's Integrated Network Tool. Un scanner de vulnérabilités réseaux, issu des travaux de Dan Farmer sur SATAN.
	**Sam Spade** http://www.samspade.org/ssw/ Un outil réseau assez généraliste avec une belle interface graphique et permettant de nombreuses requêtes (ping, nslookup, traceroute, dns queries, finger, raw HTTP web browser, SMTP relay checks…).
	**SARA** http://www-arc.com/sara/ Security Auditor's Research Assistant. Un autre outil d'analyse de vulnérabilités générique.
	**Snort** http://www.snort.org/ Le système de détection d'intrusion réseau Open Source, basé sur des signatures d'attaques.
	**SolarWinds ToolSets** http://www.solarwinds.net/ Une impressionnante collection d'outils d'investigation réseau.
	**SuperScan** http://www.foundstone.com/index.htm?subnav=resources/navigation.htm&subcontent=/resources/proddesc/superscan.htm Un scanner de ports de services graphique.
	**TCPDump / Windump** http://www.tcpdump.org/ et http://windump.polito.it/ Le très classique outil de capture réseau.
	**Whisker / LibWhisker** http://www.wiretrip.net/rfp/p/doc.asp?id=21&iface=2 Un scanner de vulnérabilités HTTP écrit en Perl.
	**Winfingerprint** http://winfingerprint.sourceforge.net/ Un outil réseau d'analyse de machines Windows pour énumérer les utilisateurs, les groupes, les partages…
	**Xprobe2** http://www.sys-security.com/html/projects/X.html Un outil réseau spécifiquement développé par Ofir Atkin et Fyodor pour la détection de systèmes d'exploitations. Fonctionne par un algorithme à logique floue.

Glossaire

ACL : Access Control List. Liste à contrôle d'accès, utilisée pour positionner des permissions sur des objets.
AD : Active Directory.
Adresse IP : Adresse réseau de niveau 3 pour le protocole IP. Une adresse IP est constituée de 4 octets.
Adresse MAC :Adresse réseau de niveau 2.
API : Application Program Interface. Interface de programmation documentée permettant d'accéder à un service.
Client : Un client est un logiciel demandant des services à un programme situé en local ou à distance qui gère l'information, le serveur. Par extension, la machine sur laquelle fonctionne le logiciel client.
CPU : Control Process Unit. Synonyme de Microprocesseur.
DNS : Domain Name Server. (Cf. *serveur de noms*, *adresse IP*)
Firewall : Elément actif de réseau, de niveau 3 et supérieur, assurant des mécanismes de filtrage.
Framework : Nom générique donné à un environnement d'exécution.
FTP : File Transfert Protocol. Protocole de transfert de fichier.
Hash : Motif binaire, résultat d'un calcul (généralement cryptographique), appelé Fonction de Hachage, sur un flux d'octets. Permet d'obtenir une empreinte, ou signature, d'un flux d'octets.
HPFS : High Performance File System. Système de fichiers développé à l'origine pour les besoins d'OS/2.
HTTP : HyperText Transport Protocol. Protocole utilisé pour le transfert des pages web.
ICMP : Internet Control Message Protocole. Protocole de contrôle de flux de niveau 3 et 4.
IETF : Internet Engineering Task Force. Groupe de normalisation des technologies utilisées sur l'Internet.
IHM : Interface Homme Machine.
IP : Internet Protocol. Protocole réseau de niveau 3 utilisé sur l'Internet.
IPSEC : IP Secure. Protocole permettant d'assurant une protection en confidentialité et en intégrité sur les paquets IP transitant sur un réseau.
Kerberos : Protocole d'authentification, développé dans le cadre du projet Athena du Massachusetts Institute of Technology.
LAN : Local Area Network. Réseau local.
LDAP: Lightweight Directory Access Protocol. Protocole d'accès à un annuaire, version simplifiée du protocole DAP.
MAC : Medium Access Control. Sous couche réseau de niveau 2 permettant de résoudre les problèmes d'accès concurrent à un médium.
Mail / **E-mail** : Messages électroniques.
NNTP : NT File System. Système de fichier utilisé dans Windows NT.
OS/2 : Système d'exploitation développé par IBM dans les années 80-90.
Proxy : Relais applicatif.
Récursif : Voir *Récursif*.
RFC : Request For Comment. Nom donné aux standards proposés par l'IETF.
Routeur : Elément actif de réseau permettant le passage d'un paquet vers un autre réseau.
RTC : Réseau Téléphonique Commuté.
Serveur : Logiciel traitant les requêtes des clients et, par extension, le poste sur lequel tourne le logiciel.
Serveur de noms : Un serveur de correspondance entre adresse IP et nom de machine. Voir *DNS*.
SMTP : Simple Mail Transfert Protocol. Protocole de messagerie.

SNMP : Simple Network Management Protocol. Protocole d'administration de réseau.
SQL : Structured Query Language : Langage de requête structuré.
SSL : Secure Socket Layer. Protocole sécurisé de niveau 4.
TCP : Transmission Control Protocol. Protocole réseau de niveau 4 offrant une liaison fiable entre deux postes.
TCP/IP : Transmission Control Protocol / Internet Protocol. Ensemble de 2 protocoles qui définit les notions d'adresse de machine, de numéro de port (utilisable pour différencier les protocoles de haut niveau comme *ftp*, *http*, etc.), de *datagramme* (paquet de données). Utilisé sur l'Internet pour établir et maintenir des connexions de bout en bout.
UDP : User Datagram Protocol. Protocole réseau de niveau 4, en mode non-connecté.
VPN : Virtual Private Network. Réseau privé virtuel, généralement constitué par des tunnels chiffrés.
WAN : Wide Area Network. Réseau étendu.
WWW : World Wide Web. Système d'information distribué et multimédia basé sur le protocole *HTTP*.
X509 : Format utilisé pour les certificats cryptographique.