L'Implémentation des Spécifications du TCG au sein de la plateforme Windows : un aperçu de BitLocker^{TM}

Bernard Ourghanlian

Microsoft France,
Directeur Technique et Sécurité
http://www.microsoft.com/france

Abstract. BitLocker^{TM} est la première mise en œuvre des spécifications de la version 1.2 du TCG (*Trusted Computing Group*) au sein de la plateforme Windows et permet d'effectuer le chiffrement complet d'un volume disque au niveau secteur afin d'offrir une protection contre les attaques logicielles réalisées en mode déconnecté. Cette fonctionnalité repose sur la mise en œuvre d'algorithmes de chiffrement AES complémentés par un algorithme de diffusion spécifique appelé *Elephant* [6].

1 Introduction

En juin 2002, Microsoft annonçait les premiers éléments du projet « Palladium ». En janvier 2003 Microsoft annonçait sa volonté de changer le nom de « Palladium » en *Next-Generation Secure Computing Base* (NGSCB). Windows ® BitLocker^{TM} est la première mise en œuvre de cette génération de technologies qui vise à ancrer la confiance du logiciel dans le hardware.

Windows ® BitLocker^{TM} Drive Encryption (appelé dans la suite BitLocker BDE ou tout simplement BDE et connu précédemment sous le nom de *Secure Startup – Full Volume Encryption* et sous le nom de projet « Cornerstone » [3], [4], [5]) est une fonctionnalité disponible avec les éditions *Enterprise* et *Ultimate* de Windows Vista. BitLocker BDE effectue le chiffrement complet d'un volume disque au niveau secteur et offre donc une protection contre des attaques logicielles en mode *offline*, attaques par lesquelles l'attaquant peut démarrer un autre système d'exploitation et accéder ainsi au disque directement pour en extraire les informations qu'il contient. BDE est conçu pour offrir une protection (confidentialité) contre le vol ou la perte d'ordinateurs portables ou d'ordinateurs dont la sécurité physique est insuffisante.

Note importante : Les informations fournies dans le présent document se fondent sur une version préliminaire de Windows Vista (pré-beta 2) ; celles-ci peuvent donc être soumises à changement et ce, jusqu'à la mise à disposition du logiciel dans sa version finale.

2 BitLocker^{TM}, TPM et TCG

BitLocker^{TM} Drive Encryption est une fonctionnalité utilisant de manière optionnelle un TPM (*Trusted Platform Module*) tel que spécifié par le TCG[1] afin de protéger les données de l'utilisateur et pour s'assurer que le contenu du disque dur d'un ordinateur exécutant Windows Vista n'a pas été altéré alors que le système était *offline*. BDE fournit aux utilisateurs mobiles un bon niveau de protection de données en cas de perte ou de vol de système.

BDE améliore la securite des données en mettant en œuvre deux fonctionnalités majeures : le chiffrement de volume (CV) et le Contrôle d'Intégrité des premiers composants d'Amorçage (CIA).

- CV protège les données en interdisant aux utilisateurs non autorisés de casser la protection Windows des fichiers ou du système sur des ordinateurs perdus ou volés. Cette protection est obtenue en chiffrant la totalité du volume Windows. Avec une telle fonctionnalité, tous les fichiers utilisateur et tous les fichiers système sans exception sont chiffrés.
- CIA assure que le déchiffrement des données n'est effectué que si les composants initiaux de l'amorçage ont passé un contrôle d'intégrité.

BDE offre aussi des capacités d'extension et de management en exposant une interface WMI et en offrant des capacités d'écriture de scripts.

BDE est conçu principalement pour utiliser la protection matérielle offerte par le TPM (*Trusted Platform Module*) version 1.2. Un TPM est une puce matérielle inviolable (dans les limites offertes par l'implémentation du matériel considéré) disponible sur la plateforme hardware et qui fournit le niveau d'attestation machine et le niveau de confiance nécessaire pour assurer le stockage matériel des secrets et interdire un accès inapproprié à des informations confidentielles et sensibles. Les TPM reposent sur des technologies ouvertes et standardisées qui assurent l'interopérabilité de différents produits de sécurité dans un environnement hétérogène.

En utilisant le TPM comme solution matérielle pour protéger les clés, la partition Windows toute entière peut être chiffrée, y compris les fichiers d'hibernation et de pagination, les fichiers temporaires et les fichiers de *dump*.

2.1 BitLocker^{TM} et l'amorçage du système

BDE utilise les fonctionnalités du TPM, et plus spécifiquement le mécanisme de *Static Root of Trust Measurement* (SRTM) du *firmware* d'amorçage (défini par le TCG), par lequel la confiance peut être établie à partir de « mesures » effectuées au sein du processus de démarrage afin de construire une empreinte du système au sein des *Platform Configuration Registers*, ou PCR du TPM. La figure 1 illustre le *Static Root of Trust Measurement* des premiers composants d'amorçage. Dans le modèle SRTM, la confiance est établie en prenant des « mesures » du système quand celui est supposé être sûr. Les mécanismes de SRTM sont définis dans le document *TCG v1.2 PC Client Implementation Specification*

[1] TPM 1.2 - `https://www.trustedcomputinggroup.org/groups/tpm/`.

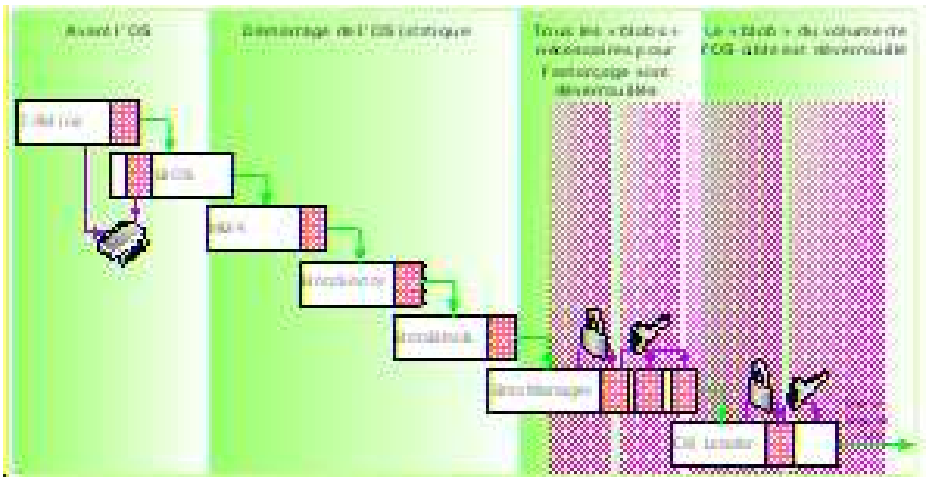


Fig. 1. *Static Root of Trust Measurement* des premiers composants d'amorçage

for Conventional BIOS, disponible en `https://www.trustedcomputinggroup.org/specs/PCClient/`.

La séquence d'amorçage initiale dans le cas d'un BIOS conventionnel fonctionne de la façon suivante :

- Après un *reset* du système, les PCR 0 à 15 du TPM sont réinitialisés et l'on transfère l'exécution à une portion digne de confiance du *firmware*. Cette portion digne de confiance du *firmware* en combinaison avec le *hardware* est connue sous le nom de *Core Root of Trust Measurement* (CRTM). Cette portion du *firmware* peut uniquement être « flashée » à nouveau dans un environnement contrôlé et approuvé par le fabriquant du système.
- Le code du CRTM « mesure » la prochaine étape du *firmware* dans PCR [0] avant de l'exécuter. C'est typiquement la portion du *firmware* utilisé pour tester et configurer des éléments critiques du hardware comme la mémoire. Au fur et à mesure de l'exécution du processus d'amorçage, davantage de code est « mesuré » et enregistré dans PCR [0]. Le principe fondamental sous-jacent est que le code est toujours « mesuré » avant d'être exécuté. Après qu'une « mesure » ait été enregistrée dans un PCR, cette valeur est considérée comme permanente et ne peut être altérée. La nouvelle valeur du PCR est le *hash* (SHA-1) résultant de la concaténation des contenus précédents et de la nouvelle « mesure » (SHA-1 or PCR[x] || data). Tant que chaque portion de code s'assure que tout nouveau code est « mesuré » avant son exécution, on peut maintenir la chaîne de confiance en encrant cette chaîne dans le hardware.
- Le *firmware* « mesure » le code optionnel en lecture seule (*read only code*) dans PCR [2]. Les ROM optionnelles peuvent également « mesurer » d'autre code en PCR [2]. Finalement, le *firmware* « mesure » la portion de code du

Master Boot Record (MBR) en PCR [4] et la table de partitions en PCR [5].
- Le MBR prend alors le contrôle du processus d'amorçage en déterminant la partition active, en chargeant le premier secteur de la partition d'amorçage en mémoire et en « mesurant » les 512 premiers octets de ce secteur en PCR [8]. Le MBR transfère alors l'exécution à ce secteur d'amorçage.
- Le secteur d'amorçage de la partition active charge et « mesure » le restant du code d'amorçage dans PCR [9] avant de lui transférer l'exécution.
- Après que le code d'amorçage ait trouvé et chargé BOOTMGR, il « mesure » BOOTMGR dans PCR [10] avant de lui transférer l'exécution. On peut noter que dans le cas d'un futur code d'amorçage basé sur EFI (*Extensible Firmware Interface*), BOOTMGR sera plutôt « mesuré » directement dans PCR [4] et PCR [8], PCR [9] et PCR[10] n'étant pas utilisés.
- BOOTMGR contrôle l'intégrité de toute application d'amorçage pour ce qui concerne les volumes protégés par BDE et s'assure du maintien de l'intégrité lors de l'accès au volume protégé par BDE après que BOOTMGR ait gagné l'accès à la partition. Si le *Root of Trust Measurement* avant ce point est invalide, alors aucun des volumes protégés par BDE n'est accessible. Après avoir descellé le *Volume Master Key*, PCR [11] (qui au moment du descellement contient zéro) est rempli avec une valeur aléatoire afin que soit préservée la restriction d'accès aux secrets de BDE.

BOOTMGR transfère le contrôle au *loader* du système d'exploitation pour le volume spécifié et ce *loader* contrôle l'intégrité de tous les composants Windows avant de transférer le contrôle au système d'exploitation. Le système d'exploitation contrôle alors l'intégrité de tous les exécutables qui seront chargés, y compris ceux qui sont concernés par la réalisation d'un *logon* authentifié.

2.2 2.1 Mesures, scellement, descellement et clés

Lors de la mise sous tension, tous les PCR sont mis à zéro. Les PCR ne sont modifiés uniquement que par la fonction « *extend* » qui positionne effectivement le contenu d'un PCR au *hash* de son ancienne valeur et d'une chaîne de caractère qui lui est fournie comme donnée. On peut donc voir le contenu d'un PCR comme correspondant à un *hash* effectué sur toutes les chaînes de données qui ont été fournies lors des appels à la fonction *extend* pour ce PCR. Il n'y a aucun mécanisme permettant de positionner la valeur d'un PCR à un contenu donné, de telle façon que si un PCR a une valeur x après une série d'*extends*, la seule façon d'obtenir à nouveau la valeur x est d'effectuer exactement la même séquence d'*extends* après une mise sous tension.

La même technologie fournit le moyen de sécuriser par chiffrement le fichier d'hibernation (ou n'importe quel fichier de dump ou de pagination). Le réveil depuis l'hibernation correspond donc exactement à ce qui se passerait depuis un amorçage traditionnel.

La fonction de chiffrement de volume de BDE chiffre la partition système Windows et scelle la clé symétrique dans le TPM. L'opération de scellement

séquestre effectivement la clé au sein du TPM, de telle façon que lors de prochains amorçages, le TPM puisse n'autoriser le descellement de la clé symétrique que lors de l'exécution d'un code dument spécifié sur l'ordinateur considéré. Le code spécifié en question (qui a la possibilité de desceller la clé symétrique) est déterminé par les valeurs contenues dans les *Platform Configuration Registers* (PCR). Le chiffrement de volume est effectué sur la base d'une granularité par secteur. Un chiffrement par bloc est utilisé afin que la modification d'un seul octet d'un bloc affecte la totalité du bloc en question.

Afin d'en apprendre davantage sur la façon dont le logiciel utilise les PCR contenus au sein d'un TPM, il est possible de consulter la documentation du TCG citée en référence. En général, les logiciels tels que BDE « étendent les mesures » des composants d'amorçage initial dans les PCR.

Les nouvelles « mesures » effectuées lors de l'amorçage doivent correspondre aux « mesures » qui étaient contenues dans les PCR au moment où BDE savait que l'ordinateur était dans un état sûr, ou bien alors la clé ne sera pas descellée. De plus, lors du scellement, l'appelant peut spécifier quels PCR spécifiques doivent être comparés avec les nouvelles valeurs de « mesure » lors du descellement. De cette façon, l'utilisateur peut être assuré que le code introduit par quelqu'un en train d'attaquer l'ordinateur ne pourra pas s'exécuter et qu'il ne pourra pas accéder à des secrets chiffrés sur le disque.

Les fonctions *seal/unseal* du TPM permettent un accès sélectif aux clés de chiffrement en fonction des valeurs des registres PCR. La fonction *seal* est utilisée pour chiffrer une clé dans un « blob » que seul le TPM peut déchiffrer. De plus le TPM ne déchiffrera le « blob » en question que si et seulement si les registres PCR sélectionnés ont la même valeur qu'elles avaient au moment de l'opération *seal*. En d'autres termes, on peut stocker une clé dans un « blob » de telle façon qu'elle ne puisse être accédée que lorsque les PCR sélectionnés ont une valeur particulière.

Le chiffrement de BDE est fondé sur une *Volume Master Key* (VMK) qui est créée (générée aléatoirement) quand BDE est mis en service et configuré pour la première fois. La VMK est protégée (scellée) en utilisant le TPM ainsi que décrit précédemment et stockée sur le disque. Pendant la séquence d'amorçage, le système d'exploitation utilise le TPM pour desceller la VMK (en se basant sur le fait que les PCR contiennent les bonnes valeurs pour authentifier le code d'amorçage) et accède aux clés suivantes qui sont nécessaires pour effectuer le déchiffrement et le chiffrement lors des lectures et des écritures sur le disque. La protection fondée sur le TPM de la VMK peut être améliorée en utilisant un code PIN. Les données dérivées du code PIN fourni par l'utilisateur sont présentées au TPM comme des données d'autorisation afin de desceller la VMK, ce qui permet la mise en place d'un mécanisme de sécurité à deux facteurs.

La VMK peut aussi être chiffrée à l'aide d'une clé externe qui peut être stockée sur un périphérique externe comme une clé USB. Même si BDE a été principalement conçu pour utiliser les services hardware du TPM, BDE peut être paramétré pour être utilisé sur une machine ne renfermant pas de TPM à travers l'utilisation d'une clé externe. Toutefois, quand BDE est paramétré et

configuré sur une machine dont le TPM est en service, BDE requerra et mettra en vigueur la protection de la VMK à travers le TPM.

La clé externe joue aussi le rôle d'une clé de récupération, ce qui permet de déplacer un volume chiffré depuis une machine où le TPM est en service vers une autre machine. Après avoir déplacé un volume chiffré vers une autre machine, la clé externe ou clé de récupération permet l'accès à la VMK afin de déchiffrer le volume.

BDE offre également une protection fondée sur un mot de passe pour des fonctions de récupération. Lors du paramétrage initial, quand l'utilisateur met BDE en fonction, un mot de passe de récupération est généré automatiquement et ce dernier peut être affiché, imprimé ou stocké dans un fichier sur un périphérique externe afin de permettre une conservation sécurisée (dans un coffre) et autoriser une récupération ultérieure. Si la machine rejoint un domaine, le mot de passe peut être généré en fonction d'une politique de groupe (qui peut rendre la création d'un mot de passe de récupération obligatoire) et stocké dans l'Active Directory de façon transparente pour l'utilisateur. Quand une récupération est nécessaire, l'administrateur peut fournir le mot de passe approprié (en provenance de l'Active Directory) à l'utilisateur afin de permettre le déchiffrement de la VMK sur la machine de l'utilisateur.

BDE fournit également un mode de protection multi-niveau par lequel la VMK est scellée à la fois sur le TPM et sur la clé externe, les deux étant nécessaires pour démarrer la machine et amorcer le système d'exploitation. Dans cette configuration, la clé externe devient une « clé partielle » requise pour déverrouiller la VMK en plus du TPM.

Dans certaines circonstances, la VMK peut être temporairement scellée avec une clé externe, appelée « clé en clair » qui est stockée en clair sur le disque dur de telle façon que le système puisse démarrer sans requérir le TPM et/ou le périphérique de stockage de la clé externe. Ce mode temporaire sans protection est appelé « *disabled mode* ». Ce mode est requis quand l'administrateur est en train de mettre BDE en fonction ou de le mettre temporairement hors service (afin, par exemple, de mettre à jour le BIOS ou d'autres composants qui sont « mesurés » lors du processus d'amorçage). Une fois la mise à jour effectuée et que BDE est placé à nouveau en « *enabled mode* », la « clé en clair » est supprimée du disque dur (effacée plusieurs fois).

BDE peut être dans l'un des trois états suivants (tableau 1).

BDE requiert un minimum de deux partitions ou volumes disque en plus du MBR (*Master Boot Record*). La première partition, appelée Partition d'Amorçage ou Système (ou volume actif) est le premier volume accédé quand l'ordinateur démarre. Ce volume contient des fichiers spécifiques au hardware qui sont nécessaires pour charger Windows et comprend le *boot manager* de l'ordinateur (pour charger plusieurs systèmes d'exploitation) et des utilitaires d'amorçage. La taille de la Partition Système sera typiquement aux environs de 350 MO. Bien que la Partition Système puisse être sur le même volume que la Partition du Système d'Exploitation (Volume du Système d'Exploitation), BDE requiert que ces deux partitions soient séparées et distinctes.

Turned off – pas de chiffrement	Dans cet état, BDE n'a pas été paramétré ou configuré ou n'a pas été mis en service. Tous les secteurs du volume sont non chiffrés.
Enabled mode – chiffré	Dans cet état, tous les secteurs du volume sont chiffrés. La *Volume Master Key* est scellée dans l'un ou plusieurs des éléments suivants (dépendant de la configuration) : le TPM (avec ou sans code PIN), une clé externe, une clé dérivée d'un mot de passe, une combinaison de TPM et de clé externe.
Disabled mode – chiffré, clé en clair	Dans cet état, tous les secteurs du volume sont chiffrés. La *Volume Master Key* est scellée dans une « clé en clair », c'est à dire une clé qui est disponible en clair sur le disque dur.

Tab. 1. Trois états de BDE

La Partition d'Amorçage est non chiffrée et sera « mesurée » et comparée par rapport au TPM pour déverrouiller la VMK. Le Partition d'Amorçage contient le secteur d'amorçage, du code d'amorçage et le *boot manager* (anciennement NTLDR) et contient le support nécessaire pour un périphérique de stockage USB et la possibilité de réaliser des saisies sur le clavier.

La Partition du Système d'Exploitation (et les partitions données subséquentes, s'il y en a) est le volume chiffré qui convient le système d'exploitation, les applications et les données.

3 Architecture

BDE est une application qui utilise l'architecture des services du TPM de Windows Vista. Dans la figure 2, les composants *Services de base du TPM* et *Driver du TPM* sont au cœur des services du TPM. Le driver du TPM (tpm.sys) est un driver en mode noyau qui est conçu pour toutes les puces TPM qui sont conformes aux spécifications du TCG en version 1.2. Cette implémentation, conforme aux spécifications du TCG en version 1.2, évite le besoin d'avoir à fournir des drivers spécifiques à chacune des implémentations de TPM et procure donc une meilleure stabilité et une meilleure sécurité de la plateforme. BDE communique avec le TPM dans les phases initiales de l'amorçage du système d'exploitation à travers un BIOS compatible TPM et à travers le driver du TPM. TBS (*TPM Base Service*) est le service qui permet les communications avec le TPM. Il a un accès exclusif au driver du TPM : une fois le service démarré, aucun autre service ne peut parler au TPM.

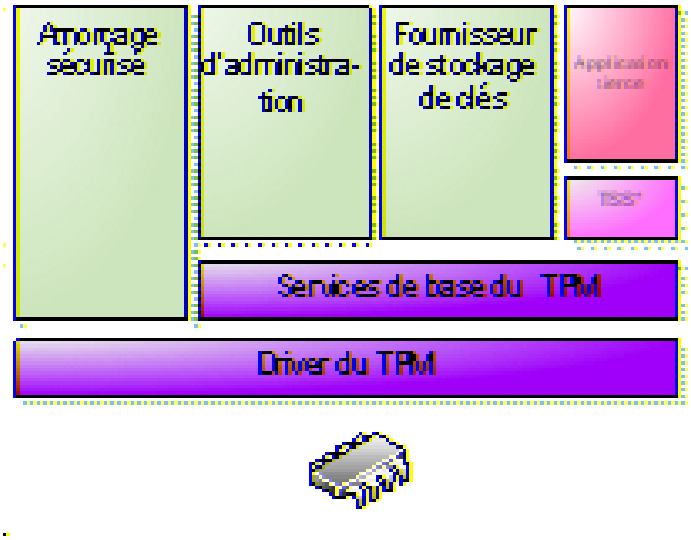


Fig. 2. L'architecture simplifiée du TPM de Windows Vista

Le chiffrement et le déchiffrement de la partition NTFS (dans le cours du chemin de données normal) est effectué en mode noyau au sein du driver filtre fvevol.sys en utilisant AES avec une couche de diffusion additionnelle (une méthode appelée *Elephant*) ainsi que précisé plus loin. Comme tout driver filtre, fvevol.sys agit comme un filtre de bas niveau entre le driver de bus, volmgr.sys et le driver de fonction volsnap.sys. A cet emplacement dans la chaîne, il peut consommer un volume logique chiffré BDE (qui peut être composé de une ou plusieurs partitions physiques) et produire un volume logique déchiffré NTFS qui sera consommé à son tour par le service de *volume snapshot*. Le déchiffrement et le chiffrement est effectué en réponse à des requêtes de lecture/écriture disque qui sont bornées par des frontières de secteur disque.

La figure 3 illustre l'architecture du système d'exploitation et du noyau dont BDE fait partie. De plus, le chiffrement et le déchiffrement en utilisant la méthode *Elephant* sont également effectués :

- Lors du démarrage au sein du *boot manager* (bootmgr) qui contient le code d'*Elephant* grâce à un mode d'édition de lien statique ;
- Pendant la conversion de volume (depuis un volume en clair vers un volume chiffré ou vice versa) qui est lancée au moyen de l'interface homme – machine appropriée ou à travers des appels WMI à l'API privée (fveapi.dll) ;
- Pendant la fonction de retour depuis l'hibernation dans les drivers filtres : rsmhbr32.exe, rsmhbr64.exe (en environnement 64 bits) et le filtre de fichiers dump : dumpfve.sys.

Les API de BDE (fveapi.dll) fournissent un ensemble d'API privées et non exposées qui sont réservées à un usage interne pour l'interface home – machine

(pour convertir un volume, mettre BDE hors service, etc.) ou pour le fournisseur WMI (*Windows Management Instrumentation*).

Les API de BDE servent essentiellement d'accès direct au driver BDE. En dehors, de la génération de clé (utilisant la CryptoAPI), fveapi.dll appelle directement fvevol.sys et les fonctions de chiffrement (telles que la conversion de volume) sont actuellement implémentées et exécutées dans le driver fvevol.sys. BDE communique avec le driver BDE (fvevol.sys) à travers un ensemble d'appels IOCTL tels que IOCTL_GET_DATASET, IOCTL_SET_DATASET, IOCTL_GET_FVEK, IOCTL_GET_VMK, IOCTL_PROVIDE_VMK, etc. Le driver fvevol.sys récupère la VMK déchiffrée depuis le bootmgr et la maintient en cache en mémoire non paginée. BDE est fourni avec un fournisseur *Windows Management Instrumen-*

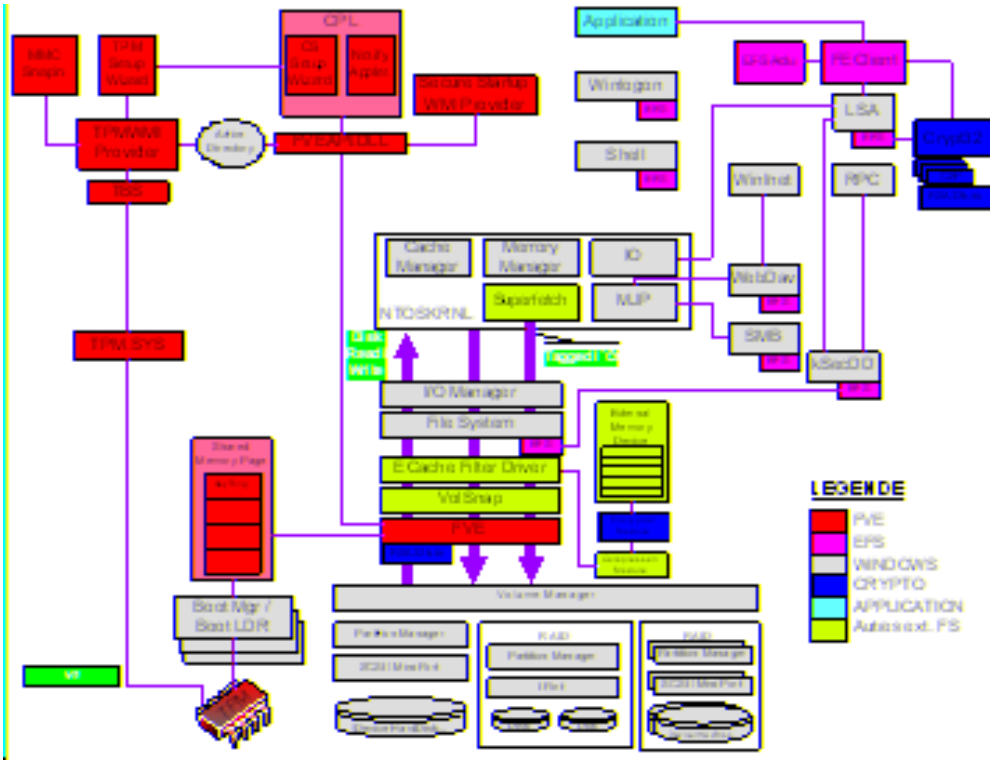


Fig. 3. Architecture du système d'exploitation et du noyau

tation (WMI) appelé Win32_EncryptableVolume qui expose des méthodes (documentées publiquement) permettant l'administration et la configuration. Les administrateurs du système peuvent configurer BDE localement et à distance à travers les interfaces exposées par Win32_EncryptableVolume. Les interfaces de ce fournisseur WMI comprennent des fonctionnalités de management pour com-

mencer, suspendre, revenir en arrière sur la configuration de chiffrement d'un volume disque et pour configurer la façon dont la clé de chiffrement du volume disque peut être protégée.

Un autre fournisseur WMI, Win32_TPM, permet aux administrateurs de configurer le hardware de sécurité TPM qui peut être utilisé pour protéger de façon transparente la clé de chiffrement du volume BDE.

Les méthodes exposées par la classe WMI Win32_EncryptableVolume comprennent :

```
Decrypt, Encrypt, ProtectKeyWithTPM, ProtectKeyWithExternalKey,
ProtectKeyWithNumericalPassword, ProtectKeyWithTPMAndPIN,
ProtectKeyWithTPMAndStartupKey, etc{\ldots}
```

Un outil disponible en ligne de commande (manage-bde) est disponible afin de fournir aux administrateurs de système un moyen simple pour vérifier l'état des disques et pour effectuer des tâches d'administration classiques. Cet outil est écrit comme un script fondé sur les fournisseurs WMI disponibles et peut être facilement modifié pour aider à la construction de solutions custom permettant de répondre à d'autres besoins d'administration. En voici un exemple :

manage-bde[.wsf] -secure
[{**-RecoveryPassword**|**-rp**} {*NumericalPassword*|}]
[{**-RecoveryKey**|**-rk**} *PathToExternalKeyDirectory*]
[{**-StartupKey**|**-sk**} *PathToExternalKeyDirectory*]
[{**-TPM_PIN**|**-tp**} *PIN*]
[{**-TPM_StartupKey**|**-tsk**} *PathToExternalKeyDirectory*]

L'interface home – machine de BDE honore et met en vigueur les politiques de groupe telles qu'elles sont définies quand la machine fait partie d'un domaine Windows. Les politiques de groupe permettent le contrôle et la configuration de fonctionnalités de BDE telles que :

- Obliger de faire la sauvegarde des informations de récupération dans l'Active Directory
- Contrôler quels mécanismes de récupération sont disponibles
- Contrôler la méthode de chiffrement utilisée pour le volume
- Requérir une authentification à deux facteurs et contrôler d'autres mécanismes de protection dans l'assistant de configuration de BDE
- Etc...

4 Gestion des clés

Nous présentons ci-dessous un résumé synthétique de la gestion des clés effectuée par BDE.

4.1 Volume Master Key

La *Volume Master Key* (VMK) est une clé de 256 bits générée de façon aléatoire quand BDE est paramétré pour la première fois. Il n'y a pas de mécanisme

pour changer ou renouveler la VMK, excepté en mettant hors service BDE (ce qui provoque le déchiffrement du volume entier) et en le paramétrant à nouveau (ce qui provoque la génération d'une nouvelle VMK et le chiffrement du volume).

La VMK est protégée par des « *key protectors* » qui peuvent exister simultanément. La table 2 synthétise les différents mécanismes de protection de la *Volume Master Key* (VMK) pour BDE :

Le chiffrement des clés en utilisant AES 256 est effectué en utilisant le mode CCM (*Counter with CBC-MAC*) du RFC 4309 de l'IETF. Le code en question utilise l'implémentation Microsoft d'AES qui est linké statiquement avec la librairie interne rsa32.lib.

Nous présentons ensuite les différentes façons de protéger la VMK en fonction de la configuration, ce qui produira les *blobs* VMK correspondants (ou *key protectors*) sur le volume.

Sur une machine TPM

Mot de passe de récupération Sur une machine qui ne fait pas partie d'un domaine, le paramétrage d'un mot de passe de récupération est optionnel. Le mot de passe est généré de façon aléatoire par BDE sous la forme d'une entité 128 bits qui est affichée sous la forme d'une représentation de 48 digits (8 groupes of 6) à l'utilisateur

- La clé 128 bits est éclatée en 8 blocs de valeurs de 16 bits.
- Chaque valeur de 16 bits génère un bloc de 6 digits en multipliant le nombre par 11.
- Le mot de passe est formaté pour l'affichage sous la forme 111111-222222-333333-444444-555555-666666-777777-888888. Chaque bloc de 6 digits peut être entré individuellement et vérifié ; il n'est est valide que si :

$$(\text{nombre} \ \% \ 11)==0 \text{ and } (\text{nombre}/11)<2\hat{}16).$$

L'utilisateur peut imprimer le mot de passe ou le sauvegarder dans un fichier.

Sur une machine faisant partie d'un domaine, la politique de groupe dicte si un mot de passe de récupération est créé ou non. Si la politique de groupe requiert un mot de passe de récupération, le mot de passe est généré aléatoirement et stocké dans l'Active Directory. Le mot de passe en question n'est pas mis à la disposition de l'utilisateur pendant le paramétrage de BDE.

Un processus de dérivation itérative de clé qui combine le mot de passe sur 128 bits avec un salage sur 128 bits permet de générer une clé AES sur 256 bits (désignée « R » ci-dessous) qui est utilisée pour chiffrer la VMK comme suit :

```
P = SHA-256(Password data)

S = 16 bytes of salt

R = 0 // 256-bit chaining variable
```

Protection de la VMK	**Algorithme et longueur de la clé utilisée pour chiffrer la VMK**	**Clé utilisée pour chiffrer la VMK**
TPM seul	RSA 2048	SRK (*Storage Root Key*) du TPM : RSA 2048, clé publique
TPM avec le code PIN de l'utilisateur	RSA 2048 Le code PIN est sur 4 à 20 digits Les premiers 160 bits de SHA256 (PIN) sont utilisés comme donnée d'autorisation dans le TPM	SRK (*Storage Root Key*) du TPM : RSA 2048, clé publique
Plusieurs couches – TPM avec une clé externe utilisée comme clé partielle	AES 256	SHA256 (IK2, ExK) IK2 est une clé intermédiaire générée aléatoirement, stockée et protégée par le TPM sur le disque. ExK est la clé externe.
Clé externe *Machine sans TPM seulement*	AES 256	ExK (clé externe)
Clé de récupération	AES 256	RK (Clé de récupération)
Mot de passe de récupération	AES 256	DF (Salage, mot de passe) Le mode de passe est une valeur sur 48 digits générée aléatoirement. « Salage » est une valeur sur 128 bits générée aléatoirement et stockée en clair sur le disque. DF est une fonction de dérivation itérative basée sur SHA-256.
Clé en clair	AES 256	CC (Clé en clair)

Tab. 2. Les différents mécanismes de protection de la *Volume Master Key* (VML)

```
For ( i=0; i$<$2\^{}20; i++ )

R = SHA-256( R $\vert $ P $\vert $ S $\vert $ i ) \newline
 // i is encoded as an 8-byte integer

Return R
```

Quand on effectue la récupération, les digits du mot de passe de récupération sont entrés en utilisant les clés de fonction du clavier (F1-F10), les clés numériques 0-9, ou les flèches.

Clé de récupération La clé de récupération est optionnelle. Pendant le paramétrage initial, l'utilisateur se voit présenter le choix de créer une clé de récupération et de la sauvegarder dans un fichier. La clé de récupération est une clé sur 256 bits et générée aléatoirement. Sur une machine faisant partie d'un domaine, la politique de groupe dicte le fait qu'une clé de récupération soit créée ou non.

L'interface homme – machine permet à l'utilisateur de sauvegarder sa clé dans un fichier situé sur une clé USB amovible ou sur un partage réseau.

TPM avec ou sans code PIN L'utilisation d'un code PIN est optionnelle. Pendant le paramétrage initial, l'utilisateur peut choisir de rentrer un code PIN qui sera utilisé comme donnée d'autorisation par le TPM. Le code PIN peut avoir de 4 à 20 digits. Sur une machine faisant partie d'un domaine, l'utilisation d'un code PIN peut être mandatée par une politique de groupe.

TPM avec clé externe (protection à deux niveaux) L'utilisation d'une clé externe est optionnelle. Pendant le paramétrage initial, l'utilisateur peut choisir d'avoir une clé externe générée aléatoirement et stockée sur un périphérique externe (clé USB). La clé externe sera utilisée comme clé partielle afin de permettre la mise en œuvre d'une protection à deux niveaux. Sur une machine faisant partie d'un domaine, la politique de groupe dicte le fait qu'une clé externe sout créée ou non.

Clé en clair La VMK est chiffrée en utilisant une clé de 256 bits générée aléatoirement et qui est stockée sur le disque au sein du *blob* VMK. Cette situation est une situation temporaire qui correspond au fait d'avoir BDE en *Disabled Mode* (voir plus haut).

Sur une machine sans TPM Sur une machine sans TPM, la protection fondée sur le TPM est remplacée par le mécanisme de protection par clé externe selon lequel la VMK est chiffrée par un algorithme AES 256 en utilisant la clé externe. L'*External Key VMK blob* est alors le *blob* principal utilisé pour déchiffrer le volume.

De plus le *Recovery Password VMK blob* et le *Recovery Key VMK blob* peuvent aussi être présents sur une machine sans TPM.

Le *Disabled Mode* associé au *Clear Key VMK blob* fonctionne également sur une machine sans TPM.

Les *key protectors* (ou les *VMK key blobs*) sont stockés comme métadonnées sur le volume du système d'exploitation (celui qui est chiffré par BDE) à trois emplacements différents. Ces métadonnées représentent une très faible portion du volume du système d'exploitation qui n'est pas chiffré par BDE. Malgré que ces zones du volume du système d'exploitation soient en clair dans le sens où elles ne sont pas chiffrées par la FVEK (*Full Volume Encryption Key*), ces métadonnées contiennent typiquement des données qui sont chiffrées par la VMK (*Volume Master Key*).

Chaque structure de métadonnées contient des informations en état à la fois chiffré et non chiffré. Plusieurs *VMK blobs* ou « *key protectors* » peuvent résider sur les sections métadonnées du volume. Ce sont :

- Le mot de passe de récupération du *VMK blob*.
- La clé de récupération du *VMK blob*.
- Le *TPM+PIN VMK blob*.
- Le *Two-Layer VMK blob*.
- Le *start-up key VMK blob* (pour des scénarios de clé de démarrage seule).
- Le *clear key VMK blob*.

Les *TPM+PIN VMK blobs* et *Two-Layer VMK blobs* peuvent être présents simultanément sur le volume, afin de permettre à l'utilisateur de démarrer avec soit un code PIN ou une clé externe (deux niveaux).

Toutefois, si l'un quelconque des TPM+PIN VMK blobs ou *Two-Layer VMK blob* est présent, alors, il n'y a pas de « *TPM Only* » *blob* (car sa présence neutraliserait la sécurité plus forte que permet le code PIN ou la clé externe). Quand le *TPM only blob* est présent, les seuls autres *blobs* qui peuvent être présents sur le volume sont les *Recovery Password* et *Recovery Key blobs*.

En *disabled mode*, un *key blob* chiffrant la VMK (la clé en clair) est stocké en clair dans la structure de métadonnées sur le volume, en plus des autres *blobs* existants.

4.2 4.2 Récupération depuis l'Active Directory

Les informations de récupération relatives à la fois à la propriété du TPM et à la récupération de la CMK sont stockées dans l'Active Directory quand la machine fait partie d'un domaine. Les entrées en question sont protégées par des ACL et transmises à travers des canaux LDAP sécurisés par Kerberos. On utilise les attributs et les sous-attributs de l'objet Computer.

Fonctionnalité TPM Puisqu'il n'y a qu'un mot de passe de propriétaire du TPM par ordinateur, le *hash* du mot de passe de propriétaire du TPM est stocké sous la forme d'un attribut de l'objet Computer.

Fonctionnalité BDE Les informations de récupération de BDE sont stockées dans un sous-objet de l'objet Computer. L'objet Computer est donc le container de l'objet de récupération BDE.

Il peut exister plus d'un objet de récupération BDE pour chaque objet Computer puisqu'il peut y avoir plus d'un mot de passe de récupération associé à un volume BDE.

Chaque objet de récupération BDE a un nom unique et contient un GUID et un mot de passe de récupération. Le GUID est un identificateur unique pour le mot de passe de récupération d'un volume FVE.

Le nom d'un objet de récupération BDE est limité à 64 caractères en raison des contraintes de l'Active Directory. Ce nom incorpore le GUID du mot de passe de récupération ainsi que des informations sur la date et l'heure sur une longueur fixe de 63 caractères. Il est de la forme :

<Object Creation Date and Time><Recovery Password GUID>

Par exemple :

2005-09-30T17 :08 :23-08 :00{063EA4E1-220C-4293-BA01-4754620A96E7}

Le *common name* (cn) Active Directory pour l'objet de récupération BDE est `ms-FVE-RecoveryInformation` et a, en plus des attributs obligatoires, deux attributs, « *musthave* » nommés ms-FVE-RecoveryPassword et ms-FVE-RecoveryGuid.

Le mot de passe de récupération BDE est stocké sous la forme d'une chaîne Unicode en utilisant le *common name* ms-FVE-RecoveryPassword. Le GUID BDE est stocké sous la forme d'une chaîne d'octets en utilisant le *common name* ms-FVE-RecoveryGuid.

4.3 Full Volume Encryption Key

La VMK est utilisée pour protéger la FVEK (*Full Volume Encryption Key*) qui sera utilisée comme clé pour la méthode de chiffrement *Elephant* qui sera décrite ultérieurement. Le fait d'utiliser un niveau d'indirection pour la protection du disque à travers la VMK permet au système de pouvoir facilement utiliser une autre clé (*re-keying*) quand l'une ou l'autre des clés situées plus haut dans la chaîne de confiance est perdue ou compromise, ce qui est particulièrement important quand le déchiffrement et le re-chiffrement d'un volume est coûteux.

La FVEK est une clé de chiffrement générée aléatoirement de 128, 256, ou 512 bits en fonction de la méthode de chiffrement qui est configurée (par une politique de groupe sur une machine appartenant à un domaine). La FVEK

Méthode de chiffrement utilisée	Taille de la FVEK
Elephant 128 (défaut)	512
Elephant 256	512
AES 128	128
AES 256	256

est chiffrée en utilisant un algorithme AES 256 en mode CCM tel que spécifié par le RFC 4309 de l'IETF. Le code utilise correspond à l'implémentation par Microsoft d'AES linké en statique avec la librairie interne rsa32.lib. La VMK est

utilisée comme clé AES. Le *blog* FVEK est stocké comme métadonnées (structure FVE_DATUM_KEY) avec les *VMK key protectors* sur le volume du système d'exploitation.

5 Chiffrement des blocs disque

BDE chiffre les données d'une partition disque au niveau secteur en utilisant la méthode de chiffrement *Elephant*. *Elephant* utilise AES en mode CBC (*Cypher Block Chaining*) pour le chiffrement de base en l'améliorant à l'aide d'un « diffuseur » pour lutter contre les attaques par manipulation de bits. La sécurité du chiffrement de base est fournie par la couche AES qui a été largement revue par la communauté scientifique et est généralement acceptée par l'industrie. La couche de diffusion ajoute certaines propriétés de sécurité supplémentaires qui sont souhaitables pour chiffrer un disque mais ne sont pas fournies par des méthodes de chiffrement traditionnelles telles qu'AES.

La méthode *Elephant* a les caractéristiques suivantes :

- Elle chiffre et déchiffre les secteurs disque de 512, 1024, 2048, 4096 ou 8192 octets.
- Elle prend le numéro de secteur comme un paramètre supplémentaire (« la distorsion ») et implémente des algorithmes de chiffrement/déchiffrement différents pour chacun des secteurs.
- Elle protégé la confidentialité du contenu en clair.
- Un attaquant ne peut contrôler ou prédire aucun des aspects des changements du contenu en clair résultant d'une modification ou d'un remplacement du contenu un secteur chiffré.
- Elle est suffisamment rapide pour que le ralentissement des performances d'un portable induit par le chiffrement soit acceptable pour la plupart des utilisateurs.
- Elle a été validée à travers un examen public et est généralement considérée comme sûre.

La plateforme n'expose pas d'API permettant à des développeurs d'effectuer des opérations de chiffrement fondées sur *Elephant*. Au sein de BDE, le code effectuant le chiffrement/déchiffrement en utilisant *Elephant* s'exécute en mode noyau.

Rentrons maintenant dans quelques détails sur la méthode *Elephant*. Dans la droite ligne de Bear, Lion, et Beast[2], la méthode de chiffrement de BDE a été appelée *Elephant*. *Elephant* consiste en l'utilisation d'AES-CBC avec un diffuseur additionnel qui permet de stopper les attaques en modification de bits.

Il y a quatre opérations séparées pour chaque chiffrement. Le contenu en clair est composé à travers un XOR avec une clé de secteur, puis passe à travers deux diffuseurs (sans clé), et est finalement chiffré en AES en mode CBC.

[2] Bear and Lion [1], [2] sont deux méthodes de type *block cipher* utilisant des grands blocs proposées par Ross Andersen et Eli Biham.

Malgré le fait que les méthodes Bear, Lion et Beast auraient pu être retenus pour satisfaire les exigences de BDE, il n'en a rien été pour des raisons de performance.

Le composant clé de secteur et le composant AES-CBC utilisent des clés indépendantes, ce qui permet d'administrer une preuve facile du fait que la méthode *Elephant* soit au moins aussi sûre qu'AES-CBC. Chacun de ces deux composants utilise une clé de 256 bits, ce qui revient à dire que la clé *Elephant* complète est une clé de 512 bits qui n'est autre que la clé FVEK que l'on a décrit précédemment. Les premiers 256 bits de la FVEK sont utilisés par le composant clé de secteur, les derniers 256 bits de la FVEK constituant la clé de l'algorithme AES.

En fonction de la version choisie, les deux composants peuvent n'utiliser que 128 bits parmi les 256 bits fournis, auquel cas, seuls les derniers 128 bits sont retenus et utilisés.

Quand la méthode *Elephant* est utilisée, la clé *Elephant* est toujours de 512 bits afin de pouvoir supporter des clés plus longues sans pour autant changer le système de gestion de clés. La méthode par laquelle les deux composants utilisent des clés 128 bits est appelée *Elephant-128*, qui est la méthode de chiffrement par défaut. La version où les deux composants utilisent des clés 256 bits est appelée *Elephant-256*.

La taille de bloc d'*Elephant* est variable : elle peut être de n'importe quelle puissance de 2 entre 512 et 8192 octets (4096 – 65536 bits).

AES-CBC Le composant AES-CBC est simple. La clé AES K_{AES} est soit de 128 bits soit de 256 bits, en fonction de la méthode qui a été choisit lors de la configuration (*Elephant-128*, le défaut, ou *Elephant-256*). La taille du bloc est toujours un multiple de 16 octets, de telle façon qu'aucun remplissage ne soit nécessaire. Le vecteur d'initialisation pour le secteur s est calculé comme suit :

$$IV_s := E(K_{\text{AES}}, e(s)), \tag{1}$$

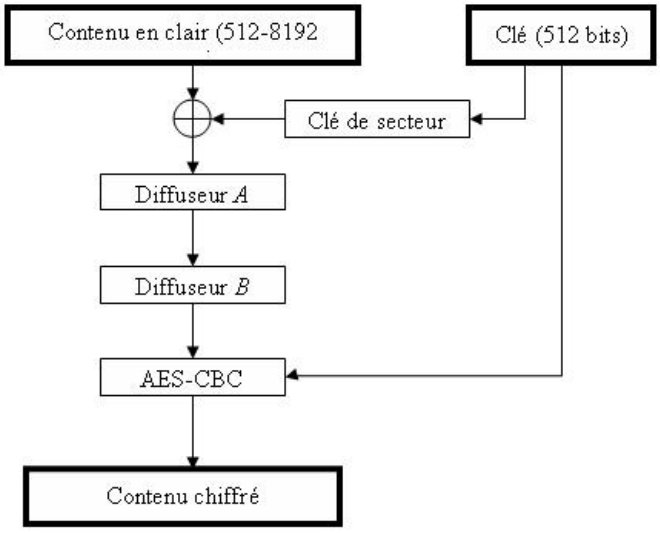
où $E()$ est la fonction de chiffrement AES, et $e()$est une fonction d'encodage qui fait correspondre à chaque numéro de secteur s une valeur unique de 16 octets. Il est important de noter qu'IV_s dépend de la clé et du numéro de secteur mais pas des données.

Le contenu en clair est chiffré en utilisant AES-CBC et le vecteur d'initialisation IV du secteur. Le déchiffrement utilise la fonction inverse.

Le résultat de $e()$est la valeur de distorsion qui est utilisée dans cette partie de l'algorithme de chiffrement. Le choix de $e()$ n'a pas d'implications en termes de sécurité (tant que cela reste une injection) et pourra varier avec l'application. Dans le cas de l'application BDE la fonction d'encodage e est la plus simple possible pour l'implémentation. Les premiers 8 octets du résultat correspondent au déplacement en octets pour atteindre le secteur sur le volume (le décalage en octets depuis le début du volume). Cet entier est encodé en utilisant une logique de type *little endian* (*least-significant-byte first encoding*). Les derniers 8 octets du résultat sont toujours à zéro.

Clé de secteur La clé de secteur de 256 bits est calculée comme suit :

$$K_s := E(K_{\text{sec}}, e(s))||E(K_{\text{sec}}, e'(s)). \tag{2}$$


Contenu en clair (512-8192
Clé (512 bits)
Clé de secteur
Diffuseur A
Diffuseur B
AES-CBC
Contenu chiffré

où $E()$est la fonction de chiffrement AES, K_{sec} est la clé 128 ou 256 bits (en fonction de la méthode : *Elephant-128*, le défaut, ou *Elephant-256*) pour ce composant, $e(s)$est la fonction d'encodage utilisée dans la couche AES-CBC et $e'(s)$est la même fonction que $e(s)$excepté le fait que le dernier octet du résultat est 1.

La clé de secteur K_s est répétée autant de fois que nécessaire pour obtenir une clé de la taille du bloc, et le résultat se voit appliqué un XOR avec le contenu en clair.

Diffuseurs La diffusion est la propriété d'un algorithme de chiffrement qui assure que le changement de quelques bits en entrée conduit au changement de nombreux bits en sortie.

Les diffuseurs A et B sont très semblables mais ils fonctionnent dans des directions opposées. La conception de base de notre diffuseur a de bonnes propriétés de diffusion dans une direction mais de mauvaises propriétés de diffusion dans l'autre direction ; en utilisant deux diffuseurs, cela permet d'avoir de bonnes caractéristiques de diffusion dans les deux directions.

Les diffuseurs ont été conçus pour favoriser le déchiffrement dans la mesure où c'est l'opération la plus commune. Nous décrirons d'abord ces algorithmes dans le sens du déchiffrement et ensuite la fonction de chiffrement correspondante.

Chaque diffuseur interprète les données du secteur dans un tableau de mots de 32 bits, chaque mot étant encodé suivant une logique *little endian*. Soit n le nombre de mots dans le secteur et d_i le $i$ème mot du secteur où i est considéré modulo n pour permettre de simplifier la notation. La fonction de chiffrement du diffuseur A est donné par :

$$\text{for} i = 0, 1, 2, \ldots, n.A_{\text{cycles}} - 1 \qquad d_i \leftarrow d_i + (d_{i-2} \text{XOR} (d_{i-5} <<< R^{(a)}_{i mod 4})). \tag{3}$$

La valeur i est un compteur de boucle qui parcourt le tableau de données A_{cycles} fois. (Il faut se souvenir que tous les indices sont modulo n). L'addition est effectuée modulo 2^{32}, $<<<$ est l'opérateur de rotation à gauche et $R^{(a)} := [9,0,13,0]$ est un tableau de 4 constantes qui spécifient les quantités de rotation à effectuer.

La fonction de chiffrement correspondante pour le diffuseur A est facile à dériver :

$$\text{for} i = n.A_{\text{cycles}} - 1, \ldots, 2, 1, 0 \qquad d_i \leftarrow d_i - (d_{i-2} \text{XOR} (d_{i-5} <<< R^{(a)}_{i mod 4})). \tag{4}$$

Les propriétés asymétriques de diffusion sont faciles à percevoir. Si l'on regarde le diffuseur A en mode déchiffrement, le résultat d'une itération est utilisé 2 et 5 itérations plus tard, ce qui propage rapidement les changements au reste du secteur. Dans la direction du chiffrement, la sortir d'une itération est utilisée n-5 et n-2 itérations plus tard, ce qui fournit une diffusion beaucoup plus lente.

Le diffuseur B est très semblable. Il a une bonne propriété de diffusion dans le sens du chiffrement. La fonction de déchiffrement du diffuseur B est définie

par :

$$\text{for} i = 0, 1, 2, \ldots, n.B_{\text{cycles}} - 1 \qquad d_i \leftarrow d_i + (d_{i+2} \text{XOR} (d_{i+5} <<< R^{(a)}_{i mod 4})). \tag{5}$$

où $R^{(b)} := [0, 10, 0, 25]$. La fonction de chiffrement du diffuseur B est donnée par :

$$\text{for} i = n.B_{\text{cycles}} - 1, \ldots, 2, 1, 0 \qquad d_i \leftarrow d_i - (d_{i+2} \text{XOR} (d_{i+5} <<< R^{(a)}_{i mod 4})). \tag{6}$$

Les constantesA_{cycles} etB_{cycles} définissent combien de fois chacun des diffuseurs bouclent sur le secteur. Elles sont définies comme $A_{cycles} := 5$ and $B_{cycles} := 3$.

Pour choisir le nombre de rotations, nous avons mené des expérimentations autour des propriétés de diffusion de nos diffuseurs (dans la direction de bonne diffusion). Nous nous sommes concentrés sur la vitesse à laquelle une différence d'un simple bit se diffusait à travers un mot de 32 bits. Nos résultats ont montré que si l'on changeait un seul bit dans d_i, chacun des bits de d_{i+43}avait une chance d'au moins ? de changer à leur tour en un seul cycle avant d'un diffuseur, les 43 mots correspondant à environ ? d'un secteur de 512 octets[3].

Références

1. Two practical and provable secure block ciphers : BEAR and LION. In Dieter Gollmann, editor, Fast Software Encryption : Third International Workshop (FSE'96), LNCS 1039, pages 113–120. Springer Verlag, 1996.
2. BEAST : A fast block cipher for arbitrary block sizes. In Patrick Horster, editor, Communications and Multimedia Security II, Proceedings of the IFIP TC6/TC11 International Conference on Communications and Multimedia Security, IFIP Conference Proceedings 70, pages 144–153. Chapman & Hall, 1996.
3. Trusted Platform Module Services in Windows Longhorn - WinHEC 2005 Version - April 25, 2005
4. Secure Startup – Full Volume Encryption : Executive Overview - WinHEC 2005 Version - April 22, 2005
5. Secure Startup – Full Volume Encryption : Technical Overview - WinHEC 2005 Version - April 22, 2005
6. Elephant : a disk-block encryption method, Niels Ferguson, Microsoft. September 2005 (still unpublished).

[3] Ceci est vrai pour le diffuseur avant B. Pour le diffuseur arrière A, un bit en d_i modifie les bits en d_{i-43}avec une probabilité ?.