MISC

Multi-System & Internet Security Cookbook

100 % SÉCURITÉ INFORMATIQUE

L 19018 - 55 - F: 8,00 € - RD

N° 55 MAI/JUIN 2011

France Métro : 8 € DOM : 8,80 € TOM Surface : 990 XPF TOM Avion : 1300 XPF
CH : 15,50 CHF BEL, LUX, PORT.CONT : 9 Eur CAN : 15 $CAD

SCIENCE & TECHNOLOGIE *AES*

Structure algébrique de l'AES ou l'AES comme vous ne l'avez jamais vu

p. 72

ARCHITECTURE *ALCASAR*

Contrôler l'usage de vos réseaux ouverts avec un portail captif libre et sécurisé

p. 50

ARCHITECTURE *FLUX WEB*

Étude pratique du déploiement d'une solution clés en main de filtrage de flux web pour protéger les écoles primaires

p. 56

SOCIÉTÉ *CORÉE*

« Cyberguerres » entre les deux Corées : multiplication des incidents et stratégies de développement numérique

p. 62

DOSSIER

AU CŒUR DES TECHNOLOGIES SÉCURITÉ DE MICROSOFT

1- Une brève histoire de Windows

2- Windows vu du réseau

3- Contournement des sécurités applicatives sous Windows 7

4- Code integrity

EXPLOIT CORNER

swapgs;swapgs; ~ nop : analyse d'une élévation de privilèges sous FreeBSD

p. 04

MALWARE CORNER

Le retour de Gpcode. Analyse de la nouvelle version du ransomware

p. 09

PENTEST CORNER

Transformez votre session Terminal Service en proxy et explorez les réseaux internes !

p. 12

PETIT TRAITÉ DE SÉCURITÉ

À L'USAGE DES HONNÊTES GENS ...
... MAIS PAS SEULEMENT (Malheureusement !)

MISC HS 3
Actuellement en kiosque !

DANS MISC HORS-SÉRIE N°3
CHEZ VOTRE MARCHAND DE JOURNAUX !

DISPONIBLE ÉGALEMENT SUR : www.ed-diamond.com

ÉDITO Ancrés Pour Toujours

Encore un numéro de *MISC*, encore un édito à écrire, et encore une fois, je suis à l'arrache. Et pourtant, je suis à la « retraite », puisque c'est Benjamin Caillat qui a concocté ce numéro. C'est que ce n'est pas facile d'être à la retraite : sport, cuisine, sieste, je suis débooooordé.

Du coup, ça me laisse le temps d'aller jouer aux devinettes. Quel est le point commun entre Comodo, RSA, Bercy, la commission européenne, MySQL pour n'en citer que quelques-uns dans l'année, Aurora ou Stuxnet l'an passé ? Réponse : APT.

Il ne s'agit pas de la commande historique de gestion des packages des utilisateurs Debian, mais d'un acronyme nouvellement à la mode. Il signifie *Advanced Persistent Threat*. En gros, ce sont des super-méchants qui mettent au point des armes super-dangereuses pour infiltrer nos réseaux et nos communications, et nous, pauvres gentils, on se fait trouer parce que les super-méchants, ils sont super forts et super-méchants.

J'avoue, j'avoue, ce résumé m'a plus que fortement été inspiré par... presque tous les responsables qui ont parlé suite aux différentes affaires évoquées en préambule. Enfin, je devrais écrire que c'est par ceux qu'on a majoritairement entendu et lu partout. D'ailleurs, j'ai trouvé particulièrement truculente l'omniprésence des éditeurs antivirus, à chaque fois. Serait-ce parce que leurs produits, supposés contrer toutes les attaques connues ou inconnues, débiles ou évoluées, ont totalement échoué ? Du coup, pour sauver la baraque, chaque vendeur explique que c'est tellement super-méchant que l'antivirus entier a pété malgré eux ! Sous-entendu, ce n'est pas de leur faute...

Et pourtant, quand on lit ce qui a été communiqué, ces attaques reposent sur les mêmes techniques que celles qu'on voit partout : parfois, il y a un 0 day, parfois une injection SQL.

Un autre truc qui m'a fait sourire est comment le patron de Comodo - oui, oui, le Comodo qui délivre certains des certificats de votre *browser* favori, grâce auquel vous pouvez surfer en confiance - a tout de suite expliqué comment il était la cible d'une attaque orchestrée par l'Iran... jusqu'à ce qu'un petit malin, prétendument iranien certes, révèle les détails de son « opération » et la clé privée d'un certificat qu'il avait validé. Même s'il ne manquait pas de sang-froid à débiter ses explications, la défense du garant se lézarde.

Au moment où la guerre asymétrique est à la mode, on constatera également une asymétrie amusante dans les APT : les victimes sont occidentales, l'attaquant chinois (et accidentellement iranien ;)). Quoi qu'il en soit, maintenant, toutes les attaques qui se déroulent sur Internet sont d'origine chinoise. Mais je n'y crois pas trop à cette hypothèse du tout chinois, je suis un filtre athée.

Au final, quand on enlève le bruit autour, pas grand chose de neuf dans ces APT, si ce n'est peut-être une sérieuse prise de conscience que les attaques informatiques peuvent être évoluées, complexes et irréversibles. Ça ne serait déjà pas si mal.

Bonne lecture,

Fred Raynal

Rendez-vous au 24 juin 2011 pour le n°56 ! **www.miscmag.com**

MISC est édité par **Les Éditions Diamond**
B.P. 20142 / 67603 Sélestat Cedex
Tél. : 03 67 10 00 20 - **Fax :** 03 67 10 00 21
E-mail : cial@ed-diamond.com
Service commercial : abo@ed-diamond.com
Sites : www.miscmag.com
www.ed-diamond.com
IMPRIMÉ en Allemagne - PRINTED in Germany
Dépôt légal : A parution
N° ISSN : 1631-9036
Commission Paritaire : K 81190
Périodicité : Bimestrielle
Prix de vente : 8 Euros

LES ÉDITIONS DIAMOND

Directeur de publication : Arnaud Metzler
Chef des rédactions : Denis Bodor
Rédacteur en chef : Frédéric Raynal
Secrétaire de rédaction : Véronique Wilhelm
Conception graphique : Kathrin Troeger
Responsable publicité : Tél. : 03 67 10 00 27
Service abonnement : Tél. : 03 67 10 00 20
Impression : VPM Druck Rastatt / Allemagne
Distribution France : (uniquement pour les dépositaires de presse)
MLP Réassort :
Plate-forme de Saint-Barthélemy-d'Anjou. Tél. : 02 41 27 53 12
Plate-forme de Saint-Quentin-Fallavier. Tél. : 04 74 82 63 04
Service des ventes : Distri-médias : Tél. : 05 34 52 34 01

Membre April
www.april.org

Charte de MISC

MISC est un magazine consacré à la sécurité informatique sous tous ses aspects (comme le système, le réseau ou encore la programmation) et où les perspectives techniques et scientifiques occupent une place prépondérante. Toutefois, les questions connexes (modalités juridiques, menaces informationnelles) sont également considérées, ce qui fait de MISC une revue capable d'appréhender la complexité croissante des systèmes d'information, et les problèmes de sécurité qui l'accompagnent. MISC vise un large public de personnes souhaitant élargir ses connaissances en se tenant informées des dernières techniques et des outils utilisés afin de mettre en place une défense adéquate.
MISC propose des articles complets et pédagogiques afin d'anticiper au mieux les risques liés au piratage et les solutions pour y remédier, présentant pour cela des techniques offensives autant que défensives, leurs avantages et leurs limites, des facettes indissociables pour considérer tous les enjeux de la sécurité informatique.

SOMMAIRE

FREEBSD - ÉLÉVATION DE PRIVILÈGES SUR AMD64

Gabriel Campana – Sogeti/ESEC

mots-clés : CVE-2008-3890 / FREEBSD / X86-64

Le 4 septembre 2008, FreeBSD a publié un bulletin de sécurité détaillé [1] (CVE-2008-3890) pour corriger une vulnérabilité trouvée par Nate Eldredge dans la partie amd64 du noyau. Nous verrons dans quelle mesure un attaquant peut l'exploiter afin d'élever ses privilèges.

1 Analyse du patch

Comme indiqué par le bulletin de sécurité, le registre de segment **GS** du CPU est aussi bien utilisé par le noyau que par des processus utilisateurs pour stocker des données. Les processus utilisateurs l'utilisent pour gérer les informations sur les threads, et le noyau pour les informations spécifiques aux processeurs. Lorsqu'un processus entre ou sort du noyau, l'instruction **swapgs** est utilisée pour échanger le contenu des deux MSR (*Machine state register*) **IA32_GSBASE** et **IA32_KERNEL_GSBASE** ; et ainsi switcher entre la valeur du registre **GS** noyau et celle de l'utilisateur.

La description du problème est claire : « si une GPF (General Protection Fault) a lieu sur un système FreeBSD/amd64 pendant le retour d'une interruption, d'une exception ou d'un appel système, l'instruction CPU **swapgs** peut être appelée une fois de trop, entraînant l'inversion des états userland et kernel land ».

Le correctif ne fait que quelques lignes et supprime simplement l'exécution de l'instruction **swapgs** dans le cas où l'instruction **iret** échoue :

```
--- sys/amd64/amd64/exception.S 24 May 2008 06:32:26 -0000      1.132
+++ sys/amd64/amd64/exception.S 18 Aug 2008 08:47:27 -0000      1.133
@@ -636,13 +636,10 @@
        .globl  doreti_iret_fault
 doreti_iret_fault:
        subq    $TF_RIP,%rsp            /* space including tf_err, tf_trapno */
-       testb   $SEL_RPL_MASK,TF_CS(%rsp) /* Did we come from kernel? */
-       jz      1f                      /* already running with kernel GS.base */
-       swapgs
-1:     testl   $PSL_I,TF_RFLAGS(%rsp)
-       jz      2f
+       testl   $PSL_I,TF_RFLAGS(%rsp)
+       jz      1f
        sti
-2:     movq    %rdi,TF_RDI(%rsp)
+1:     movq    %rdi,TF_RDI(%rsp)
        movq    %rsi,TF_RSI(%rsp)
        movq    %rdx,TF_RDX(%rsp)
        movq    %rcx,TF_RCX(%rsp)
```

2 Gestion des interruptions

Lors d'une interruption, d'une exception ou d'un appel système, la routine correspondante est exécutée par le noyau. Dans le cas de l'interruption 3 (correspondant à un *breakpoint*), la routine **Xbpt** (**amd64/amd64/exception.S**, les sources correspondent à la version 6.3 de FreeBSD) est appelée. La macro **TRAP_NOEN** enregistre les informations relatives à la trap (l'adresse de l'interruption, son numéro, etc.) sur la pile tout en laissant les interruptions désactivées :

```
#define TRAP_NOEN(a)    \
            subq $TF_RIP,%rsp; \
            movq $(a),TF_TRAPNO(%rsp) ; \
            movq $0,TF_ADDR(%rsp) ; \
            movq $0,TF_ERR(%rsp) ; \
            jmp alltraps_noen
IDTVEC(bpt)
        TRAP_NOEN(T_BPTFLT)
```

L'instruction **swapgs** est ensuite exécutée pour que le registre **GS** utilisé soit celui du noyau puisque la trap a été générée en espace utilisateur :

```
alltraps_noen:
    testb   $SEL_RPL_MASK,TF_CS(%rsp) /* Did we come from kernel? */
    jz      alltraps_pushregs         /* already running with kernel GS.base */
    swapgs
    jmp     alltraps_pushregs
```

Finalement, les registres sont sauvegardés sur la pile et la fonction **trap()** (**amd64/amd64/trap.c**) est appelée :

```
alltraps_pushregs:
        movq    %rdi,TF_RDI(%rsp)
alltraps_pushregs_no_rdi:
        movq    %rsi,TF_RSI(%rsp)
        movq    %rdx,TF_RDX(%rsp)
        movq    %rcx,TF_RCX(%rsp)
        movq    %r8,TF_R8(%rsp)
        movq    %r9,TF_R9(%rsp)
        movq    %rax,TF_RAX(%rsp)
        movq    %rbx,TF_RBX(%rsp)
        movq    %rbp,TF_RBP(%rsp)
        movq    %r10,TF_R10(%rsp)
        movq    %r11,TF_R11(%rsp)
        movq    %r12,TF_R12(%rsp)
        movq    %r13,TF_R13(%rsp)
        movq    %r14,TF_R14(%rsp)
        movq    %r15,TF_R15(%rsp)
        FAKE_MCOUNT(TF_RIP(%rsp))
.globl  calltrap
.type   calltrap,@function
calltrap:
        movq    %rsp, %rdi
        call    trap
        MEXITCOUNT
        jmp     doreti                  /* Handle any pending ASTs */
```

Cette fonction sauvegarde la *stack frame* puis la restaure après avoir effectué l'exception. Après l'appel à **trap()**, le noyau saute sur **doreti** puis **doreti_exit**, qui restaurent les registres sauvegardés et exécutent l'instruction **iret** pour retourner en espace utilisateur (notons que le registre **GS** est lui aussi restauré) :

```
doreti_exit:
     MEXITCOUNT
     movq    TF_RDI(%rsp),%rdi
     movq    TF_RSI(%rsp),%rsi
     movq    TF_RDX(%rsp),%rdx
     movq    TF_RCX(%rsp),%rcx
     movq    TF_R8(%rsp),%r8
     movq    TF_R9(%rsp),%r9
     movq    TF_RAX(%rsp),%rax
     movq    TF_RBX(%rsp),%rbx
     movq    TF_RBP(%rsp),%rbp
     movq    TF_R10(%rsp),%r10
     movq    TF_R11(%rsp),%r11
     movq    TF_R12(%rsp),%r12
     movq    TF_R13(%rsp),%r13
     movq    TF_R14(%rsp),%r14
     movq    TF_R15(%rsp),%r15
     testb   $SEL_RPL_MASK,TF_CS(%rsp) /* Did we come from kernel? */
     jz      1f                        /* keep running with kernel GS.base */
     cli
     swapgs
1:   addq    $TF_RIP,%rsp              /* skip over tf_err, tf_trapno */
     .globl  doreti_iret

doreti_iret:
        iretq
```

3 Déclenchement de la vulnérabilité

Sur l'architecture amd64, une adresse canonique est une adresse de 64 bits pour laquelle les 16 bits de poids fort sont égaux au bit 47. Toute adresse de forme différente n'est pas canonique. Si le registre **RIP** n'est pas canonique lors de l'exécution de l'instruction **iret**, **#GP(0)** est lancée **[2]**.

La vulnérabilité est donc exceptionnellement simple à déclencher : il suffit de placer un breakpoint sur la dernière adresse userland du processus (**0x800000000000-1**) :

```
void (*f)();
mmap((void *)0x0007ffffffff000, PAGE_SIZE,
     PROT_READ|PROT_WRITE|PROT_EXEC,
     MAP_PRIVATE|MAP_ANON|MAP_FIXED, -1, 0);
*(long *)&f  = 0x0008000000000000 - 1;
*(char *)(f) = 0xcc;
f();
```

Lors de l'exécution de l'instruction **iret** dans le code **doreti_iret**, la faute a lieu et déclenche l'exécution du code vulnérable **doreti_iret_fault**.

Notons que l'exploit doit ignorer le signal **SIGTRAP** pour que le processus ne soit pas terminé avant le déclenchement de la vulnérabilité.

4 Analyse du crash

Nous avons vu qu'une **#GP** est lancée si le registre **RIP** restauré est invalide lors de l'exécution **iret**. Le handler **Xprot** associé est exécuté et va de nouveau faire appel à **trap()**. La trap **T_PROTFLT** est cette fois-ci générée par le noyau :

```
case T_PROTFLT:         /* general protection fault */
case T_SEGNPFLT:        /* segment not present fault */
     if (td->td_intr_nesting_level != 0)
             break;

     /*
      * Invalid segment selectors and out of bounds
      * %rip's and %rsp's can be set up in user mode.
      * This causes a fault in kernel mode when the
      * kernel tries to return to user mode.  We want
      * to get this fault so that we can fix the
      * problem here and not have to check all the
      * selectors and pointers when the user changes
      * them.
      */
     if (frame->tf_rip == (long)doreti_iret) {
             frame->tf_rip = (long)doreti_iret_fault;
             goto out;
     }
     if (PCPU_GET(curpcb)->pcb_onfault != NULL) {
             frame->tf_rip =
                     (long)PCPU_GET(curpcb)->pcb_onfault;
             goto out;
     }
     break;
```

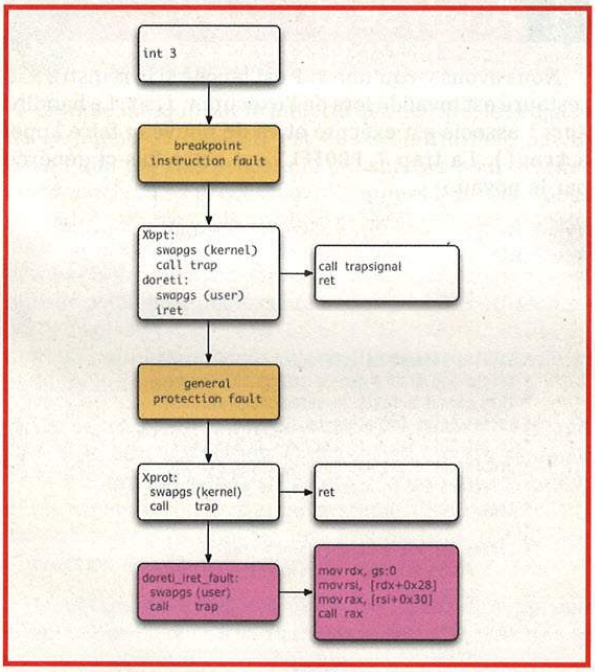
La fonction détermine que la fonction **doreti_iret** a déclenché la trap. Le **RIP** de la frame est modifié pour retourner sur **doreti_iret_fault**, dans lequel se situe le bug :

```
     /*
      * doreti_iret_fault.  Alternative return code for
      * the case where we get a fault in the doreti_exit code
      * above.  trap() (amd64/amd64/trap.c) catches this specific
```

```
     * case, sends the process a signal and continues in the
     * corresponding place in the code below.
     */
    ALIGN_TEXT
    .globl  doreti_iret_fault
doreti_iret_fault:
    subq    $TF_RIP,%rsp            /* space including tf_err, tf_trapno */
    testb   $SEL_RPL_MASK,TF_CS(%rsp) /* Did we come from kernel? */
    jz      1f                      /* already running with kernel GS.base */
    swapgs
1:  testl   $PSL_I,TF_RFLAGS(%rsp)
    jz      2f
    sti
2:  movq    %rdi,TF_RDI(%rsp)
    ...
```

Le code exécuté fait appel à la fonction **swapgs** et se retrouve alors avec le **GS** de l'userland. **doreti_iret_fault** va une dernière fois faire appel à la fonction **trap()** afin d'envoyer le signal approprié au processus. Comme le noyau assimile **curthread** au segment **GS**, qui est contrôlé, un pointeur de fonction peut être détourné. Finalement, **trap()** retourne sur **doreti**.

La figure suivante montre les différentes étapes déclenchant la vulnérabilité :



Flot d'exécution, de l'interruption à la vulnérabilité

5 Out of jail

Aucune erreur ne peut être commise par l'attaquant car le noyau est en train d'exécuter une procédure de traitement d'interruption. Une double faute sera levée si jamais une nouvelle exception se produit, la corruption du registre **GS** entraînant le *freeze* ou le redémarrage de la machine.

Par chance, un pointeur de fonction dépendant de la valeur contenue dans **GS:0** est appelé dans la fonction **trap()** juste après la préparation de la stack frame :

```
void trap(struct trapframe *frame) {
        struct thread *td = curthread;
        struct proc *p = td->td_proc;
        ...
        /* Translate fault for emulators (e.g. Linux) */
        if (*p->p_sysent->sv_transtrap)
                i = (*p->p_sysent->sv_transtrap)(i, type);
```

Il n'existe aucune protection pour le noyau sur FreeBSD. Des adresses userland peuvent donc être directement déréférencées par le noyau et contenir le payload qui devra effectuer les actions suivantes :

- restaurer le registre de segment GS du noyau à l'aide de l'instruction **swapgs** ;
- modifier la structure **curthread->td_proc->p_ucred** afin d'élever les privilèges de l'exploit ;
- modifier l'adresse (0x800000000000) qui se trouve dans la pile, pour retourner en userland et ne pas déclencher la faute à l'infini ;
- restaurer les registres modifiés par la mauvaise valeur de GS:0.

Le développement d'un exploit totalement stable (seule la connaissance des offsets de la structure **proc** est nécessaire, et celle-ci ne change qu'à chaque version majeure du noyau) est possible, permettant l'exécution de code en noyau, et donc l'élévation de privilèges et/ou la sortie d'une jail.

Conclusion

Cette vulnérabilité – passée largement inaperçue – est relativement ancienne mais montre la fragilité du code responsable des interruptions dans un système d'exploitation. Toujours d'actualité, des vulnérabilités semblables ont été trouvées sur NetBSD, par exemple **[3]**, ainsi que sur des logiciels de virtualisation comme VMware **[4]**. Notons que dans ce dernier cas, l'exploitation semble être exactement la même. ■

■ REMERCIEMENTS

Je tiens à remercier Ivanlef0u pour sa relecture et ses suggestions.

■ RÉFÉRENCES

[1] http://security.freebsd.org/advisories/FreeBSD-SA-08:07.amd64.asc

[2] http://www.intel.com/products/processor/manuals/

[3] http://blog.cr0.org/2009/09/cve-2009-2793-iret-gp-on-pre-commit.html

[4] http://dl.packetstormsecurity.net/0811-advisories/vmware-guestescalate.txt

Abonnez-vous !

Profitez de nos offres d'abonnement spéciales disponibles au verso !

Téléphonez au 03 67 10 00 20 ou commandez par le Web

Économisez plus de 20%*

* Sur le prix de vente unitaire France Métropolitaine

6 Numéros de MISC

Les 3 bonnes raisons de vous abonner :

- Ne manquez plus aucun numéro.
- Recevez MISC dès sa parution chez vous ou dans votre entreprise.
- Économisez 10,00 €/an !

4 façons de commander facilement :

- par courrier postal en nous renvoyant le bon ci-dessous
- par le Web, sur www.ed-diamond.com
- par téléphone, entre 9h-12h et 14h-18h au 03 67 10 00 20
- par fax au 03 67 10 00 21

par ABONNEMENT : **38€***

au lieu de 48,00 €* en kiosque

Économie : 10,00 €*

*OFFRE VALABLE UNIQUEMENT EN FRANCE MÉTROPOLITAINE
Pour les tarifs hors France Métropolitaine, consultez notre site : www.ed-diamond.com

Bon d'abonnement à découper et à renvoyer à l'adresse ci-dessous

Tournez SVP pour découvrir toutes les offres d'abonnement >>

MISC
Multi-System & Internet | Security | Cookbook

Édité par Les Éditions Diamond
Service des Abonnements
B.P. 20142 - 67603 Sélestat Cedex
Tél. : + 33 (0) 3 67 10 00 20
Fax : + 33 (0) 3 67 10 00 21

Voici mes coordonnées postales :	
Société :	
Nom :	
Prénom :	
Adresse :	
Code Postal :	
Ville :	
Pays :	

Vos remarques :

En envoyant ce bon de commande, je reconnais avoir pris connaissance des conditions générales de vente des Éditions Diamond à l'adresse internet suivante : **www.ed-diamond.com/cgv** et reconnais que ces conditions de vente me sont opposables.

Tournez SVP pour découvrir toutes les offres d'abonnement

Profitez de nos offres d'abonnement spéciales !

Vous pouvez également vous abonner sur : www.ed-diamond.com ou par Tél. : 03 67 10 00 20 / Fax : 03 67 10 00 21

- Europe 1 : Allemagne, Belgique, Danemark, Italie, Luxembourg, Norvège, Pays-Bas, Portugal, Suède
- Europe 2 : Autriche, Espagne, Finlande, Grande Bretagne, Grèce, Islande, Suisse, Irlande
- Zone Reste du Monde : Autre Amérique, Asie, Océanie
- Zone Afrique : Europe de l'Est, Proche et Moyen-Orient

(Nos tarifs s'entendent TTC et en euros)	F France Métro	D DOM	T TOM	E1 Europe 1	E2 Europe 2	EUC Etats-Unis Canada	A Afrique	RM Reste du Monde
1 Abonnement MISC	38 €	40 €	44 €	45 €	44 €	46 €	45 €	49 €
2 LPE + LP	57 €	62 €	69 €	71 €	69 €	73 €	71 €	79 €
3 GLMF + LP	78 €	85 €	96 €	99 €	95 €	101 €	98 €	111 €
4 GLMF + GLMF HS	83 €	89 €	101 €	104 €	100 €	105 €	103 €	116 €
5 GLMF + MISC	84 €	90 €	102 €	105 €	101 €	107 €	104 €	117 €
6 GLMF + GLMF HS + Linux Pratique	110 €	119 €	134 €	138€	133 €	140 €	137 €	154 €
7 GLMF + GLMF HS + MISC	116 €	124 €	140 €	144 €	139 €	146 €	143 €	160 €
8 GLMF + GLMF HS + MISC + LP	143 €	154 €	173 €	178 €	172 €	181 €	177 €	198 €
9 GLMF + GLMF HS + MISC + LP + LPE	173 €	186 €	209 €	215 €	208 €	219 €	214 €	239 €
10 MISC + MISC HS	44 €	47 €	53 €	55 €	52 €	56 €	54 €	60 €
11 LP + LP HS	42 €	46 €	52 €	54 €	51 €	55 €	53 €	60 €
12 GLMF + GLMF HS + MISC + MISC HS + LP + LP HS + LPE	199 €	214 €	243 €	250 €	239 €	254 €	247 €	279 €
13 Open Silicium Magazine	27 €	29 €	31 €	32 €	31 €	33 €	32 €	36 €

* Toutes les offres d'abonnement : en exemple, les tarifs ci-dessus correspondant à la zone France Métro (F) ** Base tarifs kiosque zone France Métro (F)

offre 1 Misc (6 nos)

par ABO : **38€***

au lieu de **48,00€**** en kiosque

Economie : 10,00 €

offre 10 MISC (6 nos) + MISC Hors-Série (2 nos)

par ABO : **44€***

au lieu de **64,00€**** en kiosque

Economie : 20,00 €

offre 2 Linux Pratique Essentiel (6 nos) + Linux Pratique (6 nos)

par ABO : **57€***

au lieu de **74,70€**** en kiosque

Economie : 17,70 €

offre 3 GNU/Linux Magazine (11 nos) + Linux Pratique (6 nos)

par ABO : **78€***

au lieu de **107,20€**** en kiosque

Economie : 29,20 €

offre 4 GNU/Linux Magazine (11 nos) + GNU/Linux Magazine HS (6 nos)

par ABO : **83€***

au lieu de **110,50€**** en kiosque

Economie : 27,50 €

offre 5 + GNU/Linux Magazine (11 nos) + Misc (6 nos)

par ABO : **84€***

au lieu de **119,50€**** en kiosque

Economie : 35,50 €

offre 6 + GNU/Linux Magazine (11 nos) + GNU/Linux Magazine HS (6 nos) + Linux Pratique (6 nos)

par ABO : **110€***

au lieu de **146,20€**** en kiosque

Economie : 36,20 €

offre 7 + GNU/Linux Magazine (11 nos) + GNU/Linux Magazine HS (6 nos) + Misc (6 nos)

par ABO : **116€***

au lieu de **158,50€**** en kiosque

Economie : 42,50 €

offre 8 + GNU/Linux Magazine (11 nos) + GNU/Linux Magazine HS (6 nos) + Linux Pratique (6 nos) + Misc (6 nos)

par ABO : **143€***

au lieu de **194,20€**** en kiosque

Economie : 51,20 €

offre 9 Linux Pratique Essentiel (6 nos) + GNU/Linux Magazine (11 nos) + GNU/Linux Magazine HS (6 nos) + Linux Pratique (6 nos) + Misc (6 nos)

par ABO : **173€***

au lieu de **233,20€**** en kiosque

Economie : 60,20 €

offre 11 Linux Pratique (6 nos) + Linux Pratique HS (3 nos)

par ABO : **42€***

au lieu de **55,20€**** en kiosque

Economie : 13,20 €

offre 12 Linux Pratique Essentiel (6 nos) + GNU/Linux Magazine (11 nos) + GNU/Linux Magazine HS (6 nos) + Linux Pratique (6 nos) + Linux Pratique HS (3 nos) + Misc (6 nos) + MISC Hors-Série (2 nos)

par ABO : **199€***

au lieu de **268,70€**** en kiosque

Economie : 69,70 €

Bon d'abonnement à découper et à renvoyer

Je fais mon choix de l'offre de mon (mes) abonnement(s) :

Mon 1er choix	Je sélectionne le N° (1 à 13) de l'offre choisie :	
Mon 2ème choix	Je sélectionne le N° (1 à 13) de l'offre choisie :	
	Je sélectionne ma zone géographique (F à RM) :	
	J'indique la somme due : (Total)	€

Exemple : je souhaite m'abonner à l'offre GNU/Linux Magazine + GNU/Linux Magazine Hors-série + MISC (offre 7) et je vis en Belgique (E1), ma référence est donc 7E1 et le montant de l'abonnement est de 144 euros.

Je choisis de régler par :

- ❑ Chèque bancaire ou postal à l'ordre des Éditions Diamond
- ❑ Carte bancaire n° ⌊_⌊_⌊_⌊_⌋ ⌊_⌊_⌊_⌊_⌋ ⌊_⌊_⌊_⌊_⌋ ⌊_⌊_⌊_⌊_⌋

Expire le : ⌊_⌊_⌊_⌊_⌋

Cryptogramme visuel : ⌊_⌊_⌊_⌋

Date et signature obligatoire

Découvrez notre nouveau magazine !

OPEN SILICIUM

LE MAGAZINE DE L'OPEN SOURCE POUR L'ÉLECTRONIQUE & L'EMBARQUÉ

sur : www.opensilicium.com

➡ Abonnez-vous

offre 13 Open Silicium Magazine (4 nos)

par ABO : **27€***

au lieu de **36,00€**** en kiosque

Economie : 9,00 €

En kiosque à partir du 24 décembre 2010 !

LE RETOUR DE GPCODE

Nicolas Brulez – nicolas.brulez@kaspersky.fr
Senior Malware Researcher – Global Research and Analysis Team
Kaspersky Lab

mots-clés : CODES MALICIEUX / RANSOMWARE / RSA / ANALYSE DE CODE / AES

25 mars 2011, alors que vous surfiez tranquillement sur Internet, le fond d'écran de votre Windows change et Notepad s'ouvre soudainement pour vous informer que tous vos fichiers personnels ont été chiffrés à l'aide de cryptographie forte (RSA 1024) et qu'il est impossible de les récupérer sans payer 125 dollars en carte pré-payée. Vous ne rêvez pas, vous êtes la victime du nouveau Gpcode.

Figure 1 : Demande de rançon par Gpcode

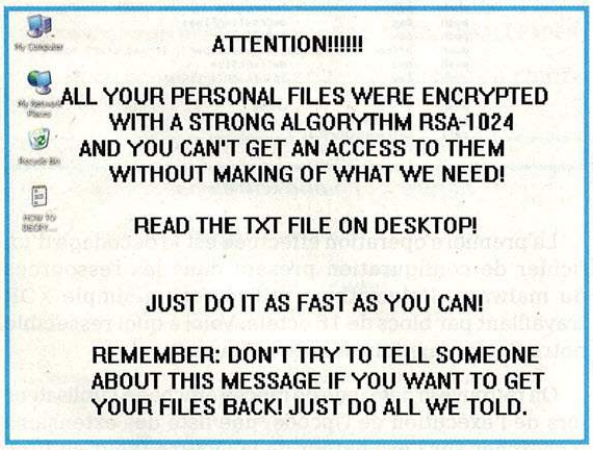


Figure 2 : Fond d'écran modifié par Gpcode

Tout ceci est dû à la visite d'un site malveillant (*drive by downloads*) par une machine non mise à jour. Une fois exécuté, Gpcode génère une clé AES 256 aléatoire puis la chiffre à l'aide de la clé publique des criminels, qui n'est autre que du RSA 1024. Nous allons maintenant voir les détails dans une analyse du code.

1 Obfuscation

La dernière version de Gpcode apparue en novembre 2010 était simplement compressée à l'aide d'UPX.

Une fois décompressée (**upx -d**), nous obtenions un binaire prêt à être analysé. Bien qu'UPX soit toujours présent dans la nouvelle version, le binaire obtenu après décompression est encore crypté. En effet, nous sommes en présence d'un packer « custom » utilisé pour rendre l'analyse du code plus compliquée :

```
.text:00412187                  push    ebp
.text:00412188                  mov     ebp, esp
.text:0041218A                  add     esp, 0FFFFFFD4h
.text:0041218D                  add     edi, ebx
.text:0041218F                  call    IsUPX_removed
.text:00412194                  mov     eax, edx
.text:00412196                  neg     edi
.text:00412198                  dec     edx
.text:00412199                  inc     edx
.text:0041219A                  not     edi
.text:0041219C                  jmp     loc_4012AC
.text:0041219C start            endp ; sp-analysis failed
.text:0041219C
```

Figure 3 : Point d'entrée du second packer

L'analyse de ce packer dans son intégralité remplirait à lui seul le *Malware Corner*, je vous invite à lire mon article dans le numéro 51 de *MISC*, qui présente des techniques similaires à celles employées ici.

On notera cependant une routine intéressante utilisée pour détecter l'unpacking de la première couche UPX :

```
.text:00401000 IsUPX_removed   proc near               ; CODE XREF: start+8↓p
.text:00401000                 push    ebp
.text:00401001                 dec     ecx
.text:00401002                 not     esi
.text:00401004                 mov     ebp, esp
.text:00401006                 neg     esi
.text:00401008                 mov     edx, edi
.text:0040100A                 add     esi, edx
.text:0040100C                 add     esp, 0FFFFFFD0h
.text:0040100F                 dec     esi
.text:00401010                 mov     edi, edx
.text:00401012                 mov     ecx, esp
.text:00401014                 inc     ebx
.text:00401015                 mov     edx, ebx
.text:00401017                 not     edx
.text:00401019                 add     ecx, 0FFFFFFE8h
.text:0040101C                 dec     esi
.text:0040101D                 cmp     eax, ecx        ; if EAX = ECX then UPX still present
.text:0040101F                 jz      short not_unpacked
.text:00401021                 inc     eax
.text:00401022                 dec     ecx
.text:00401023                 leave
.text:00401024                 mov     ecx, edx
.text:00401026                 leave
.text:00401027                 add     edx, esi
.text:00401029                 inc     ecx
.text:0040102A                 retn                    ; RET leads to ExitThread
```

Figure 4 : Fonction de détection de l'unpacking d'UPX

Le fonctionnement de la routine est très simple. Elle utilise le fait qu'UPX place une adresse de la pile dans le registre EAX juste avant d'exécuter l'application décompressée. Lors de l'appel de la routine de vérification, EAX doit contenir cette adresse pour passer le « CMP EAX, ECX ».

Le packer effectue des opérations simples pour obtenir une adresse sur la pile qui sera la même que celle placée dans EAX par UPX. En cas d'unpacking préalable (de la première couche UPX), le registre EAX ne sera pas correctement initialisé et l'unpacking sera détecté. L'exécution se terminera par l'appel de la fonction **ExitThread**.

Pour analyser le packer, il suffit de ne pas retirer la couche UPX, ou de patcher le « JZ » en « JMP » à l'adresse « 0x40101F ».

Déboguer l'intégralité du packer prend pas mal de temps. Il existe cependant une petite astuce pour y arriver en moins de 10 secondes. Si vous avez lu l'article dans le numéro 51, vous vous souviendrez de l'allocation de mémoire pour décrypter le fichier original sur le heap. La majorité des packers employés pour protéger des programmes malveillants utilisent ces techniques.

Il semblerait que les développeurs, dans un souci de propreté du code, utilise la fonction **VirtualFree** pour désallouer la mémoire contenant une copie du fichier unpacké dans son intégralité.

Il suffit donc de patcher l'anti unpacking du point d'entrée, de placer un point d'arrêt sur **VirtualFree** et d'exécuter notre ransomware pour arriver à l'appel :

```
0012F3BC  003D18B8  CALL to VirtualFree from 003D18B6
0012F3C0  00910000  Address = 00910000
0012F3C4  00002400  Size = 2400 (9216.)
0012F3C8  00004000  FreeType = MEM_DECOMMIT
```

Figure 5 : Appel à la fonction VirtualFree

Le paramètre de **VirtualFree** est l'adresse à désallouer, ici 0x910000. Avant d'effectuer le nettoyage, il est possible de récupérer un fichier décrypté. Pour se faire, il suffit d'utiliser la fonction **follow in dump** d'ollydbg, par exemple. Un rapide coup d'œil à cette adresse nous donne ceci :

```
Address   Hex dump                                          ASCII
00910000  4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00  MZ..........ÿÿ..
00910010  B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  ........@.......
00910020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00910030  00 00 00 00 00 00 00 00 00 00 00 00 80 00 00 00  ................
00910040  0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68  ..º..´.Í!¸.LÍ!Th
00910050  69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F  is program canno
00910060  74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20  t be run in DOS
00910070  6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00  mode....$.......
00910080  50 45 00 00 4C 01 03 00 52 36 0F 4C 00 00 00 00  PE..L...R6.L....
```

Figure 6 : Dump de la mémoire allouée

Nous avons ici l'exécutable malicieux totalement décrypté. Il ne nous reste plus qu'à le dumper, comme décrit dans l'article précédemment cité (clic droit sur le dump hexadécimal, ***Backup*** et ***Save data to File***).

À notre grande surprise, cet exécutable est une fois de plus compressé par UPX. Un rapide **upx -d** nous donne enfin l'exécutable final. Temps total d'unpacking : moins de 30 secondes.

2 Analyse du Ransomware

Voici à quoi ressemble le point d'entrée de Gpcode une fois celui-ci totalement décompressé et décrypté :

```
                public start
start           proc near
                call    Decrypt_config_file
                test    eax, eax
                jz      loc_4019DD
                push    offset aIlold   ; "ilold"
                push    0               ; bInheritHandle
                push    MUTEX_ALL_ACCESS ; dwDesiredAccess
                call    OpenMutexA
                test    eax, eax
                jnz     short loc_4019DD
                push    offset aIlold   ; "ilold"
                push    0               ; bInitialOwner
                push    0               ; lpMutexAttributes
                call    CreateMutexA
                call    Generate_random_AES_256_key
                test    eax, eax
                jz      short loc_4019DD
                call    Encrypt_AES_key_with_RSA1024
                xor     eax, eax
                push    eax             ; lpThreadId
                push    eax             ; dwCreationFlags
                push    eax             ; lpParameter
                push    offset drop_ransom_txt_wallpaper ; Threat start address
                push    eax             ; dwStackSize
                push    eax             ; lpThreadAttributes
                call    CreateThread
                push    1               ; uMode
                call    SetErrorMode
                call    GetLogicalDrives
```

Figure 7 : Point d'entrée unpacké

La première opération effectuée est le décodage d'un fichier de configuration présent dans les ressources du malware. L'algorithme utilisé est un simple XOR travaillant par blocs de 16 octets. Voici à quoi ressemble notre fichier une fois décodé : Figure 8.

On retrouve la demande de rançon affichée à l'utilisateur lors de l'exécution de Gpcode, une liste des extensions à chercher sur l'ordinateur de la victime (pour chiffrer les fichiers), et un blob RSA qui contient les paramètres « e » et « n », soit la clé publique des criminels qui sera utilisée pour chiffrer la clé AES 256 employée lors du chiffrement des données. Il contient aussi des informations telles que le pourcentage du fichier à chiffrer.

La seconde opération de Gpcode consiste à générer une clé AES de 256 bits : Figure 9

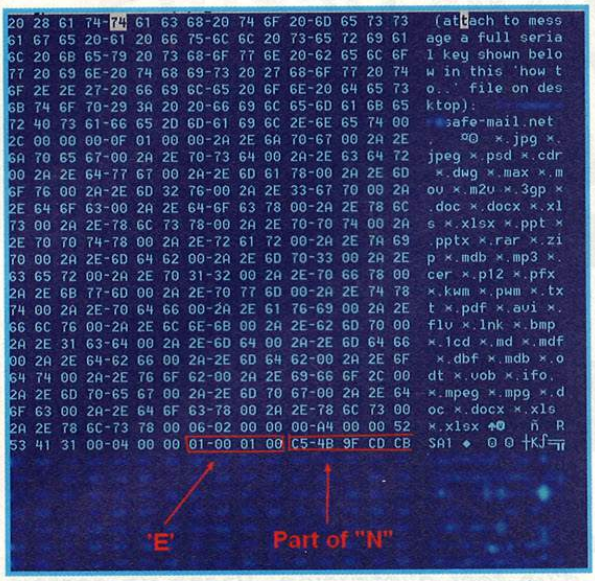


Figure 8 : Fichier de configuration décodé

```
AES_Key_Generation:                     ; CODE XREF: Generate_random_AES
                push    offset phKey    ; phKey
                push    1 or 1000000h   ; upper 16 bits = 0100h = 256
                push    CALG_AES_256    ; Algid
                push    phProv          ; hProv
                call    CryptGenKey
                push    offset pdwDataLen ; pdwDataLen
                push    0               ; pbData
                push    0               ; dwFlags
                push    PLAINTEXTKEYBLOB ; dwBlobType
                push    0               ; hExpKey
                push    phKey           ; hKey
                call    CryptExportKey
```

Figure 9 : Génération de la clé AES 256

La 3ème opération consiste à la chiffrer à l'aide de la clé publique (RSA 1024) des criminels. Un thread est créé et sert à la création du fichier TXT sur le bureau de la victime ainsi qu'au changement du fond d'écran (**SystemParametersInfoA** avec **SPI_SETDESKWALLPAPER**).

Gpcode commence à chercher les fichiers à chiffrer sur le disque :

```
SearchFile      proc near               ; CODE XREF: SearchFile+94↓p
                                        ; start+83↓p

FindFileData    = _WIN32_FIND_DATAA ptr -144h

                push    ebp
                mov     ebp, esp
                sub     esp, 144h
                lea     eax, [ebp+FindFileData]
                push    eax             ; lpFindFileData
                push    offset unk_403570 ; lpFileName
                call    FindFirstFileA
                inc     eax
                jz      locret_40194E
                dec     eax
                mov     dword ptr [ebp+FindFileData._padding], eax

loc_4017EB:                             ; CODE XREF: SearchFile+17A↓j
                mov     eax, [ebp+FindFileData.dwFileAttributes]
                and     eax, 10h
                jz      short loc_40186F
                lea     ebx, [ebp+FindFileData.cFileName]
                push    ebx             ; lpString2
                push    offset a_       ; "."
                call    lstrcmpA
                test    eax, eax
                jz      FindNextFile
                push    ebx             ; lpString2
                push    offset a__      ; ".."
                call    lstrcmpA
```

Figure 10 : Recherche de fichiers à chiffrer

Une fois un fichier à chiffrer découvert, celui-ci sera ouvert et chiffré directement :

```
push    0               ; lpOverlapped
lea     eax, [ebp+nNumberOfBytesToWrite]
push    eax             ; lpNumberOfBytesW
push    [ebp+nNumberOfBytesToRead] ; nNumb
push    ds:file_content ; lpBuffer
push    [ebp+hObject]   ; hFile
call    ReadFile
or      eax, eax
jnz     short loc_4011C0
jmp     loc_401260

                        ; CODE XREF: Encr
push    65536           ; dwBufLen
lea     eax, [ebp+nNumberOfBytesToWrite]
push    eax             ; pdwDataLen
push    ds:file_content ; pbData
push    0               ; dwFlags
push    0               ; Final
push    0               ; hHash
push    ds:phKey        ; hKey
call    CryptEncrypt
```

Figure 11 : Routine de chiffrement de Gpcode

Il est important de noter que le fichier n'est pas chiffré dans son intégralité. En effet, le fichier de configuration contient le pourcentage du fichier à chiffrer.

Une fois le fichier chiffré, celui-ci est renommé en **fichier_original.extention.ENCODED**.

Les premières versions de Gpcode faisaient une copie du fichier à chiffrer et effaçaient l'original avec un effacement classique. Il était possible de récupérer le fichier original à l'aide d'outils de récupération de données. Cette vulnérabilité n'est plus présente dans les dernières versions.

Gpcode passe aux fichiers suivants jusqu'à avoir parcouru tous les disques. Une fois les fichiers chiffrés, la clé AES 256 est détruite à l'aide de la fonction **CryptDestroyKey** :

```
Kill_AES_Key    proc near               ; CODE XREF
                push    phKey           ; hKey
                call    CryptDestroyKey
                push    0               ; dwFlags
                push    phProv          ; hProv
                call    CryptReleaseContext
                retn
Kill_AES_Key    endp
```

Figure 12 : Destruction de la clé AES

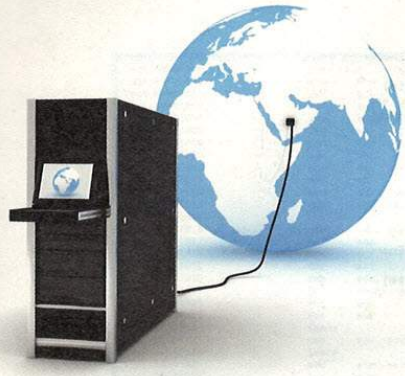
Il est alors impossible de retrouver la clé AES, même en dumpant la mémoire physique d'une machine infectée. Pour pouvoir récupérer les données, il faudrait freezer Gpcode et dumper la mémoire avant que la clé soit détruite.

Pour terminer, Gpcode génère un fichier **.BAT** avant d'appeler la fonction **ExitProcess**. Ce fichier tente d'effacer l'exécutable de Gpcode et y parviendra une fois celui-ci terminé.

Ainsi s'achève l'analyse de Gpcode. Il n'existe à ma connaissance aucune attaque permettant de récupérer les fichiers chiffrés ou la clé AES générée sur une machine affectée dans les conditions réelles. L'utilisation de sauvegardes reste la seule solution pour récupérer les fichiers pris en otage.

Il est aussi intéressant de noter que les criminels ont commencé à utiliser les cartes pré-payées (Ukash) plutôt que les transferts d'argent pour récupérer la rançon. En novembre 2010, il fallait toujours effectuer un versement d'argent pour payer celle-ci.

Quelques jours avant la découverte de Gpcode, un autre ransomware beaucoup moins dangereux se faisant passer pour une alerte de la police fédérale allemande demandait aussi le paiement par cartes pré-payées. ■



INTRUSION DEPUIS UN ENVIRONNEMENT TERMINAL SERVER AVEC RDP2TCP

Nicolas Collignon

mots-clés : RDP / TERMINAL SERVER / TUNNELS

Il était une fois un pentester qui découvrait le mot de passe pour se connecter sur un service Terminal Server lors d'un test d'intrusion. Passé un certain temps, il réussit (ou échoua) à compromettre le système localement avec une élévation de privilèges. Ennuyé, il commença à s'intéresser à la configuration réseau du Terminal Server. Soudain, il constata que ce serveur était connecté à un réseau invisible depuis son ordinateur !

Le pentester décida donc d'attaquer le nouveau réseau depuis le Terminal Server. Malheureusement, il se lassa rapidement car il ne disposait pas des outils nécessaires sur le serveur pour attaquer. Aucun moyen de transférer un fichier, pas même un accès à Internet.

Sans nouvelle découverte intéressante, le pentester finit par oublier ce serveur...

1 Introduction

Le déport d'affichage pour les applications est une fonctionnalité utilisée depuis longtemps avec des protocoles comme X11, VNC et RDP. Par exemple, un client se connecte à un serveur Citrix et se retrouve dans un environnement graphique restreint, généralement dédié à une application bien particulière. Un aspect de ces environnements peu abordé concerne les attaques par rebonds possibles sur ces systèmes. Lorsque l'on se retrouve projeté dans un environnement restreint (ex : Terminal Server, Application Citrix), il est difficile et parfois impossible d'attaquer les services réseau visibles seulement depuis cet environnement.

Cet article propose une solution pour réaliser des relais TCP via une session Terminal Server afin de pouvoir attaquer le réseau uniquement visible depuis ce serveur. Le scénario est donc qu'il existe des services réseau sensibles et visibles uniquement à partir du Terminal Server. Le filtrage mis en place dans l'architecture réseau nous autorise seulement les connexions depuis et vers le port 3389/TCP de ce serveur.

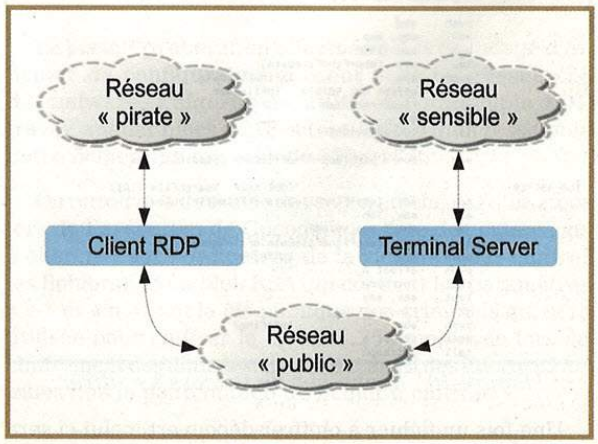


Figure 1

2 Le protocole RDP

Lorsque l'on est connecté à un Terminal Server qui n'est pas filtré en sortie, la technique la plus simple permettant de relayer les connexions TCP vers l'extérieur consiste à déposer un outil comme socat (http://www.dest-unreach.org/socat/) ou Fpipe (http://www.mcafee.com/us/downloads/free-tools/fpipe.aspx) et d'établir manuellement les tunnels. Cependant, dans le cas où le filtrage réseau limite les flux en sortie, les choses se compliquent. Si le Terminal Server est uniquement autorisé à émettre des paquets à des clients Terminal Server (3389/TCP), il va falloir trouver une méthode permettant de relayer les attaques depuis notre poste à travers la session Terminal Server.

Terminal Server utilise plusieurs protocoles regroupés autour de ce qu'on appelle « *Remote Desktop Protocol : RDP* ». Au même titre que le protocole SSH, RDP permet de multiplexer plusieurs canaux de données dans une seule connexion TCP. Le principal canal de données est celui associé au déport d'affichage : affichage des fenêtres, gestion de la souris, gestion du clavier, etc. Les autres canaux de données généralement utilisés par Terminal Server sont :

- CLIPRDR – gestion du copier/coller ;
- SOUND – déport du son sur le client ;
- RDPDR – redirection des I/O sur les fichiers.

Le point intéressant est qu'une application exécutée dans une session RDP, sur le Terminal Server, peut ouvrir des nouveaux canaux de données à la demande. La création d'un canal ne nécessite aucun privilège particulier.

Afin de relayer les connexions TCP de notre station Linux vers le réseau accessible depuis le Terminal Server, nous allons donc exécuter une application qui gérera son propre canal de communication dans RDP.

3 rdesktop

Côté client, il existe a priori un seul projet open source fiable permettant de se connecter à un Terminal Server : rdesktop (http://www.rdesktop.org/). Cet outil intègre déjà le support des canaux RDP (options `-r disk`, `-r sound`, etc.). Mais il sera nécessaire d'appliquer un *patch* sur le code source de rdesktop pour ajouter la possibilité de gérer des canaux arbitraires avec un système de *plugins* externes.

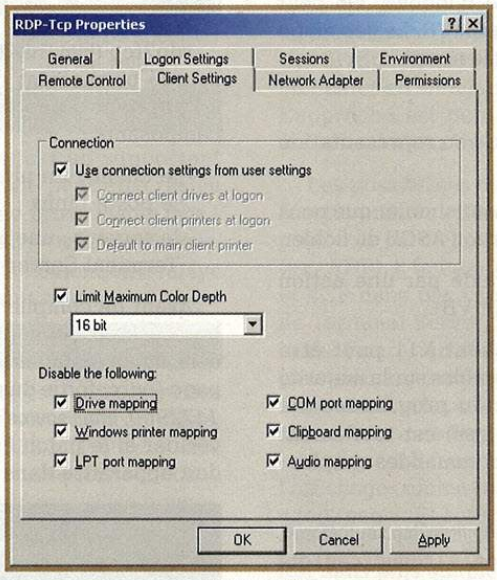


Figure 3

Une fois le *patch* installé, l'option `-r addin` a été ajoutée à rdesktop. Elle permet de spécifier un programme qui sera exécuté pour traiter le nouveau canal RDP. Les données extraites par rdesktop sont envoyées par un tube au plugin. Il est possible d'ajouter plusieurs canaux en spécifiant des noms différents.

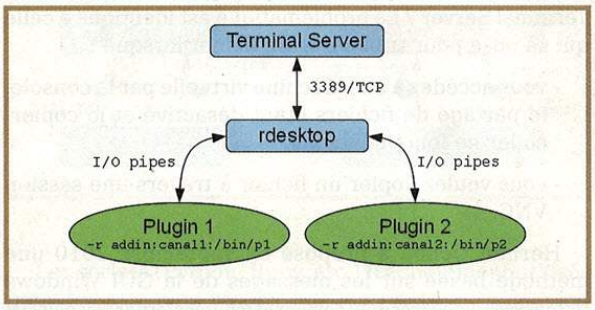


Figure 2

Passons à la pratique !

4 Transfert de fichiers via RDP

Pour relayer du TCP à travers le protocole RDP, il nous faut donc un plugin côté rdesktop et une application côté serveur qui utilisera le même canal RDP. Cependant, l'application va devoir être transférée sur le Terminal Server.

La manière la plus simple de transférer des fichiers depuis et vers un Terminal Server est d'utiliser le chemin `\\tsclient` dans l'explorateur de fichiers. Ce chemin est intercepté par le noyau Windows afin de rediriger les entrées/sorties (les *I/O Requests Packets* plus précisément) à l'intérieur de la session RDP. Le client, rdesktop dans notre cas, récupère les requêtes et manipule le système de fichiers du poste Linux.

Pour que notre client rdesktop active le partage du répertoire `/tmp`, il faut par exemple utiliser la ligne de commandes suivante :

```
$ rdesktop -r disk:mon_partage:/tmp
192.168.0.1
```

Évidemment, si cette méthode fonctionnait partout, cela serait trop simple. Il est possible de régler la politique de sécurité de Terminal Server afin de restreindre certaines fonctionnalités, comme le fameux partage de fichiers « *Drive mapping* ».

Supposons maintenant que le partage de fichiers soit désactivé. Les services de partage de fichiers génériques (ex : RPC/SMB, FTP, etc.) constituent la première roue de secours.

Si seul le port 3389/TCP est ouvert et si le partage de fichiers est bloqué, comment copier un fichier sur le Terminal Server ? La problématique est identique à celle qui se pose pour transférer un fichier lorsque :

- vous accédez à une machine virtuelle par la console, le partage de fichiers étant désactivé et le copier/coller ne fonctionne pas.
- vous voulez copier un fichier à travers une session VNC.

Hernan Ochoa a proposé en septembre 2010 une méthode basée sur les messages de la GUI Windows (http://www.ampliasecurity.com/research/transferringfilesonisolatedRDenvironments-ampliasecurity.pdf). L'approche que nous allons étudier est un peu plus basique.

Le problème est que nous ne pouvons pas recopier bêtement au clavier le fichier à transférer si celui-ci contient des données non ASCII. Il faut encoder le fichier dans une représentation qui peut être tapée au clavier. Et côté serveur, le fichier doit être décodé par un moyen automatique. Évidemment, le concept n'est pas de taper 100000 fois sur le clavier pour un transfert :). Un client X11 peut être automatisé pour nous éviter cette lourde tâche manuelle.

Concernant l'encodage, les choix sont nombreux. Une technique utilisée fréquemment sur les injections SQL sous Windows est d'encoder les données dans un format qui sera lu par le programme `debug.com`. Pour simplifier, l'encodage fait une conversion binaire → hexadécimal. Cette technique est efficace mais nécessite la possibilité d'exécuter le programme `debug.com`. Celui-ci n'est plus livré avec les versions récentes de Windows. On peut aussi convertir le fichier dans un script VB. Le décodage sera limité à un clic pour exécuter le script.

Pour résumer :

1. Le fichier non ASCII est encodé en représentation ASCII (ex : `debug.com`, VB).
2. Le client X11 est automatisé pour simuler que nous tapons au clavier la représentation ASCII du fichier.
3. Le fichier non ASCII est décodé par une action manuelle (ex : clic sur le script VB).

La partie automatisation du client X11 peut être effectuée avec plusieurs outils disponibles sur la majorité des distributions Linux. Le choix du programme xte (http://hoopajoo.net/projects/xautomation.html) est totalement arbitraire. xte prend en entrée des commandes simples : `key K`, `keydown K`, `keyup K`, `str SSS`.

Les sources de l'outil rdp2tcp (http://rdp2tcp.sourceforge.net/) contiennent un script Python (`tools/rdpupload`) qui permet d'automatiser toute la partie encodage et transfert.

L'option `-x` du script permet de convertir un fichier en commandes xte. L'extrait de code VB (voir code 1) produira une sortie similaire à (voir code 2). Le script est suffisamment générique pour être utilisé sur d'autres clients que rdesktop (ex : console d'une machine virtuelle).

Code 1 :

```
With CreateObject("ADODB.Stream")
.Type=2
.Open
.WriteText chr(49)
```

Code 2 :

```
str With
key space
str CreateO
str bject("
str ADODB.S
str tream")
key Return
str .Type=2
key Return
str .Open
key Return
str .WriteT
str ext
key space
str chr(49)
```

Dans la pratique, le plus simple est de lancer un `notepad.exe` sur le Terminal Server, d'utiliser rdpupload avec l'option `-x` pour transférer le fichier en prenant garde de ne pas oublier de placer le *focus* X11 dans la fenêtre du notepad et de sauvegarder le fichier généré sur le Terminal Server. Il reste ensuite à décoder le fichier en utilisant une méthode spécifique à l'encodage : exécuter le programme `debug.com` ou le script VB.

5 L'outil rdp2tcp

rdp2tcp (http://rdp2tcp.sourceforge.net/) est un outil open source permettant de multiplexer plusieurs flux TCP à l'intérieur d'une session RDP. L'outil est séparé en 2 composants :

- le client : un plugin rdesktop qui sera exécuté sur le poste Linux ;
- le serveur : une application qui sera exécutée sur le Terminal Server dans une session RDP.

Avant de compiler rdp2tcp, il est nécessaire d'avoir un rdesktop qui supporte le système de *plugins*. Pour cela, il faut installer le *patch* `oop.patch` disponible sur la page sourceforge du projet rdesktop (*Tracker > Patch ID 1472969*, http://sourceforge.net/tracker/?group_id=24366). Pour vérifier si le patch est bien intégré, l'option `-r addin` doit apparaître dans l'aide de rdesktop.

```
$ ./rdesktop 2>&1 | grep addin
        '-r addin:<channelname>:</path/to/executable>[:arg1[:arg2:]...]':
        enable third party virtual channel add-in.
```

La compilation du client (client/rdp2tcp) se résume à un **make**. Pour compiler le composant serveur, il est nécessaire d'avoir un compilateur Windows à disposition. La suite mingw32 (http://www.mingw.org/) permet cependant de s'en sortir sans avoir de Windows. En fonction des distributions, il sera peut-être nécessaire de changer le chemin vers le compilateur (CC).

```
$ head -2 server/Makefile.mingw32
BIN=rdp2tcp.exe
CC=i586-mingw32msvc-gcc
```

Le client et le serveur sont compilés, les choses sérieuses commencent...

6 Gestion des tunnels TCP

Le serveur **rdp2tcp.exe** a été transféré sur le Terminal Server, tout est prêt. En supposant que le client rdesktop a été correctement compilé pour supporter l'option **-r addin**, rdesktop est démarré avec le plugin rdp2tcp.

```
$ rdesktop -r addin:rdp2tcp:/usr/bin/rdp2tcp
```

La session s'ouvre, il faut taper le mot de passe, cette partie n'a pas changé :). Ensuite, on démarre le programme **rdp2tcp.exe** dans la fenêtre rdesktop. Si tout se passe bien, le client et le serveur rdp2tcp indiquent que le canal de communication est correctement établi.

Le script **rdp2tcp.py** permet enfin de contrôler l'ajout et la suppression de relais TCP. Par exemple, pour relayer le port 8080/TCP du client vers le port 80/TCP du serveur 192.168.0.2 :

```
$ rdp2tcp.py add forward 127.0.0.1 8080 192.168.0.2 80
```

Cependant, ajouter manuellement des redirections de ports peut vite devenir fastidieux. Il est plus simple de mettre un serveur SOCKS en écoute avec rdp2tcp et de spécifier aux outils d'attaques réseau de passer par ce relais :

```
$ rdp2tcp.py add socks5 127.0.0.1 7878
```

Même si tous les outils de *pentest* ne supportent pas le protocole SOCKS, le problème peut être résolu avec l'aide d'un programme comme proxychains (http://proxychains.sourceforge.net/) ou socksify (http://www.inet.no/dante/). Un *scan* de ports TCP peut facilement être relayé à travers RDP avec la commande suivante :

```
$ proxychains nmap -sT -PN 192.168.0.2
```

Une autre possibilité intéressante est l'utilisation des relais TCP inverses. Supposons que suite au scan nmap, vous trouvez un service sur le port 3333/TCP qui doit être exploité avec un *shellcode* de type « reverse connect TCP ». Pour simplifier l'exploitation, rdp2tcp est utilisé pour :

- Relayer l'attaque vers le port 3333/TCP de la victime.

```
$ rdp2tcp.py add forward 127.0.0.1 3333 192.168.0.2 3333
```

- Relayer la connexion TCP issue du shellcode et reçue par le Terminal Server.

```
$ rdp2tcp.py add reverse 127.0.0.1 5555 192.168.0.2 4444
```

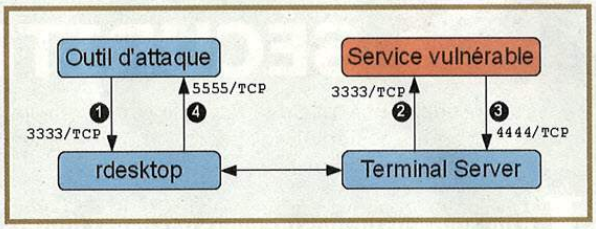


Figure 4

Les scripts (Python, Ruby, etc.) fréquemment utilisés pour les attaques réseau par les *pentesters* peuvent être facilement utilisés depuis le poste Linux sans être déployés sur le Terminal Server.

Conclusion

Nous venons de voir comment relayer des attaques réseau à travers le programme rdesktop et une connexion à un Terminal Server. La possibilité d'utiliser rdesktop comme serveur SOCKS avec rdp2tcp permet d'attaquer le réseau visible depuis le Terminal Server de façon transparente. La politique de sécurité Windows ne permet pas d'autoriser uniquement une liste restreinte de canaux RDP. Il semble donc impossible d'empêcher un client de relayer des connexions TCP à partir du moment où il peut exécuter un programme arbitraire. Par ailleurs, les flux TCP relayés bénéficient de la couche de chiffrement du protocole RDP.

L'hypothèse initiale est qu'il est possible de transférer un programme qui servira de point de relais sur le serveur. L'outil rdpupload montre qu'il est possible de transférer un fichier en simulant une séquence de touches clavier. L'approche est générique et peut être appliquée aux autres solutions du marché.

Les possibilités de faire du relais TCP sont liées aux fonctionnalités offertes par les protocoles réseau. Si le protocole permet nativement de multiplexer des canaux (ex : RDP), il faut implémenter un serveur qui encapsule le TCP dans des canaux. Il faut noter que dans le cas de Terminal Server, il n'est pas nécessaire de modifier la configuration du système ou d'être administrateur pour démarrer une application qui crée un canal RDP. Cependant, si le protocole est limité au déport d'affichage, clavier et souris (ex : VNC), la tâche sera nettement plus compliquée. Il faudra développer un serveur qui utilisera l'affichage comme canal de communication, une sorte de « tcp2png ». Et côté client, il faudra analyser l'affichage pour reconvertir en TCP :). Une autre approche, moins fun mais plus simple, serait d'utiliser le presse-papiers comme vecteur de transfert. ■

DOSSIER

AU CŒUR DES TECHNOLOGIES SÉCURITÉ DE MICROSOFT

Je me souviens de mes premiers pas dans l'univers de Microsoft. 1994, à l'époque, Windows était pour moi une interface DOS dans laquelle il s'agissait de taper le plus vite possible « cd jeux\doom2 », « doom2 ». Une relation superficielle et purement ludique. Mais avec Windows 95, tout a pris une autre tournure, je découvrais la richesse du système en lui-même et commençais à en explorer la complexité. Et ce fut le début d'une belle relation qui dure depuis maintenant plus de 15 ans. Il y a bien eu des hauts et des bas, quelques écarts avec une pingouinette de passage que je dois confesser, mais après 15 ans, nous sommes toujours là.

Pourtant, j'avoue parfois perdre pied. Il y a déjà une complexité inhérente à sa manie parfois agaçante de garder un attachement obstiné à de vieilles technologies utilisées dans des obscurs recoins du monde industriel (entre autres sur des chaînes de production critiques sur lesquelles la notion de mise à jour frise le blasphème). Et puis il y a toutes ces nouvelles lubies. Elle qui rangeait soigneusement code, données et structures toujours à la même place en mémoire, voilà qu'elle décide maintenant de les mettre à des adresses aléatoires ! Et que dire des questions existentielles qu'elle me pose avant de faire des opérations toutes simples qui ne lui posaient aucun problème au préalable. Mais le plus dur pour moi a probablement été sa prise de distance, refusant brusquement de me laisser accéder à son noyau, moi qui aimais tant patcher sa SSDT. Il était donc temps de sortir du brouillard et de faire un point sur tout cela.

Première phase : comprendre l'historique. Pour cela, j'ai fait appel à un vieux routard de Windows qui, grâce à sa culture forgée au fil d'années de *happy hacking*, nous donne un aperçu des principales technologies développées par Microsoft.

« La communication est la base de toute relation », dit-on. La deuxième partie poursuit cette exploration par une présentation non moins experte des technologies Microsoft orientées réseau.

Enfin, les deux derniers volets s'attachent à présenter des mécanismes fondamentaux en termes de sécurité, qui ont été introduits récemment. Le premier analyse les différentes protections logicielles mises en œuvre pour prévenir l'exploitation des célèbres débordements de tampons. /GS, SafeSEH, SEHOP, ASLR, DEP, EMET, Sandboxing, la liste des technologies s'allonge chaque année un peu plus. Mais qu'en est-il de leur efficacité ? Peut-on enfin enlever le point d'interrogation dans la présentation de Nicolas Ruff en 2004 : « La fin des buffers overflows (?) » ? Enfin, le dernier article présente en profondeur les mécanismes sous-jacents de la signature de code.

Afin de garder une taille raisonnable dans ce numéro (et d'éviter les sarcasmes faciles sur l'utilisation abusive de ressources par Windows), ces deux derniers articles ont été scindes en deux parties, la seconde sera publiée dans le prochain numéro de *MISC*.

Vouloir expliquer Windows dans son ensemble est une tâche impossible. De nombreuses autres technologies mériteraient tout autant d'être présentées dans ce dossier. Faute de place, il a fallu faire des choix. Ces articles vous apportent cependant des éléments clés pour comprendre l'historique et les nouvelles orientations en termes de sécurité de Windows. Je souhaite surtout qu'ils vous donnent la curiosité de poursuivre l'exploration de cet univers, parfois trop critiqué par ces détracteurs linuxiens et pourtant ô combien passionnant.

Bonne lecture,

Benjamin Caillat

UNE BRÈVE HISTOIRE DE WINDOWS

Nicolas Ruff – EADS Innovation Works – nicolas.ruff@eads.net

mots-clés : MICROSOFT / WINDOWS / NETBIOS / LM / NTLM / MS/RPC / OLE / (D)COM / (NET)DDE / ACTIVEX / .NET / WMI

L'histoire de la société Microsoft ne se résume pas à Windows. Cette histoire a commencé en 1975 avec un interpréteur BASIC [1] dans lequel la légende raconte que Bill Gates a lui-même placé un Easter Egg. Microsoft a ensuite réalisé l'affaire du siècle en vendant le système MS-DOS [2] à IBM, au nez et à la barbe du système CP/M. Mais en 1979, Microsoft développait également le système XENIX [3] (compatible UNIX), qui devint par la suite SCO UNIX.

En ce qui concerne Windows [4], la saga débute en 1985 par Windows 1.0, mais le premier système réellement connu du grand public sera Windows 3.1 en 1992 - ainsi que la variante Windows for Workgroups, qui ajoute le support des protocoles réseau NetBEUI et IPX (réseaux Novell) pour le transport du protocole SMB.

1 Genèse technologique

1.1 Noyau

Les branches Windows 3.x et Windows 9x (ainsi que toutes les versions antérieures) sont essentiellement des applications MS-DOS. Il est d'ailleurs possible de « quitter » ces versions de Windows pour revenir à la ligne de commandes MS-DOS.

Windows 3.x est un système d'exploitation 16 bits. Windows 3.0 peut s'exécuter en mode réel ou en mode protégé (selon la ligne de commandes utilisée pour lancer `WIN.COM`), tandis que Windows 3.1 exige la disponibilité du mode protégé sur les processeurs Intel - mode qui est apparu avec le 80286. Il faut noter que le mode protégé du 80286 souffrait d'importantes limitations, par exemple l'absence de pagination ou l'impossibilité de revenir en mode réel sans réinitialiser le processeur. Cette limite avait été astucieusement contournée à l'époque : à chaque fois qu'un passage en mode réel était nécessaire, il suffisait de provoquer une triple faute du processeur, ce qui engendrait un redémarrage contrôlable (*soft reboot*) en mode réel.

Les fichiers exécutables étaient alors au format NE (*New Executable*) ou LE (*Linear Executable*), qui ont depuis été remplacés par le format PE (*Portable Executable*), devenu un standard de fait - y compris pour des technologies indépendantes comme l'UEFI.

Il est possible d'exécuter des applications 32 bits dans Windows 3.x à l'aide de la couche de compatibilité « Win32s » (optionnelle). À partir de Windows 95, une partie du noyau a été réécrite en 32 bits, assurant de fait une compatibilité native avec le 32 bits.

Notons que MS-DOS a supporté le 32 bits avant Windows, grâce au standard DPMI (*DOS Protected Mode Interface*) - dont l'implémentation commerciale la plus connue est DOS/4GW (utilisée entre autres par le jeu Doom). Ce standard repose sur l'interruption 0x31. Le standard DPMI 0.9 est toujours supporté par Windows XP **[5]** - ce système en mode protégé offre donc aux programmes émulés en mode réel une émulation du mode protégé. Bel exemple de contorsion technologique au nom de l'interopérabilité.

L'héritage MS-DOS/Windows 3.x a duré fort longtemps, car sur un Windows XP SP3 dernier cri, on trouve encore une application `WIN.COM` et le gestionnaire de fenêtres `PROGMAN.EXE` dans le répertoire `%WinDir%\System32`, ainsi que différents utilitaires MS-DOS parfaitement fonctionnels tels que `COMMAND.COM`, `EDLIN.EXE`, `EDIT.COM` ou `DEBUG.EXE`.

```
C:\WINDOWS\system32> dir *.com

14/04/2008  13:00             7 680 chcp.com
14/04/2008  13:00            50 620 command.com
14/04/2008  13:00             9 216 diskcomp.com
14/04/2008  13:00             7 168 diskcopy.com
14/04/2008  13:00            69 886 edit.com
14/04/2008  13:00            29 696 format.com
14/04/2008  13:00            26 112 graftabl.com
14/04/2008  13:00            19 694 graphics.com
14/04/2008  13:00            14 710 kb16.com
14/04/2008  13:00             1 131 loadfix.com
14/04/2008  13:00            19 456 mode.com
14/04/2008  13:00            16 896 more.com
14/04/2008  13:00            12 800 tree.com
14/04/2008  13:00            18 432 win.com
              14 fichier(s)         303 497 octets
```

Il faut toutefois préciser que la branche Windows NT (comprenant Windows NT3.1, NT3.5, NT4, 2000, XP et toutes les versions ultérieures) représente un système d'exploitation à part entière, conçu à partir de la feuille blanche par des anciens ingénieurs du système VMS. Cette branche n'a aucun lien de parenté avec MS-DOS ou toutes les versions de Windows qui l'ont précédée.

Windows XP a marqué la fin de la branche Windows 9x avec l'unification des systèmes d'exploitation « professionnels » et « grand public ». Les différences qui subsistent entre SKU (*Stock Keeping Units*) - c'est-à-dire les dénominations « Familiale », « Professionnelle », « Intégrale », etc. - sont de nature purement commerciale et non technique.

Afin d'assurer une transition « en douceur », toutes les versions de Windows depuis XP embarquent une base de compatibilité appelée *Windows Compatibility Database* (fichiers « .SDB »), qui peut être éditée avec l'outil ACT (*Application Compatibility Toolkit*). Lorsqu'une application connue pour être incompatible est lancée, le *loader* Windows va injecter **SHIMENG.DLL** (*Shim Engine*) dans cette application pour modifier le comportement des API.

Dans la capture d'écran ci-dessous, on voit par exemple que le comportement de l'API **GetCommandLine()** est modifié dynamiquement pour assurer le bon fonctionnement du jeu Dungeon Keeper sur Windows XP.

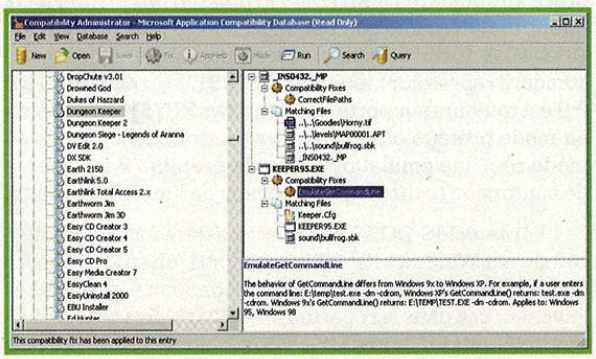


Figure 1

1.2 Réseau

1.2.1 Couches basses

Divers protocoles réseau aujourd'hui oubliés existaient bien avant le succès commercial de Windows : NetBEUI, IPX, DECnet ou même Banyan VINES. Ces protocoles étaient largement déployés « dans la vraie vie » et Windows se devait de les supporter. On peut lire sur Internet que Banyan VINES était le protocole réseau officiel de l'état-major américain pendant la guerre en Irak de 1991.

Même au niveau des couches basses, plusieurs protocoles se sont concurrencés jusque dans les années 2000, comme Ethernet, Token Ring ou LocalTalk (Apple) - sans parler d'ARCnet, qui a eu son petit succès commercial.

Il faut signaler d'ailleurs que la plupart des protocoles réseau disposaient de pilotes MS-DOS bien avant l'apparition de Windows (souvenez-vous : **LSL.COM**, **IPXODI.COM**, ...). Ces protocoles étaient mis en œuvre par des applications MS-DOS telles que Doom ou Duke Nukem, ..., ou encore très récemment par Norton Ghost.

Windows 3.x ne parle pas nativement TCP/IP - il est nécessaire d'installer des applications tierce partie pour supporter ce protocole. L'ensemble des éditeurs tiers de piles TCP/IP pour Windows se sont mis d'accord à l'époque pour standardiser une API qui reste à la base du support réseau dans Windows aujourd'hui : l'API WinSock. L'implémentation la plus connue à l'époque est Trumpet Winsock, disponible pour Windows 3.x, 9x et NT. Notons au passage que Trumpet Winsock 5.0 (pour Windows 95) supporte déjà IPv6 - car même à l'époque, IPv6 c'était pour demain.

Quant au protocole NetBIOS, il peut être nativement intégré au BIOS (comme son nom l'indique) ou rendu disponible sous MS-DOS grâce à une application TSR (*Terminate and Stay Resident*). Il est alors accessible par n'importe quelle application en mode réel grâce à des interruptions (classiquement 0x5C).

Windows définit une API - **NetBIOSCall()** - pour que le système d'exploitation lui-même puisse appeler le code (en mode réel) chargé du trafic NetBIOS. Bien que ce mécanisme puisse sembler aberrant en termes de performance et de stabilité (implémenter la pile réseau du système d'exploitation dans le BIOS), il faut savoir que les systèmes d'exploitation modernes continuent d'appeler des API du BIOS en mode réel - du moins avant le standard UEFI, qui redistribue les cartes. Windows XP va par exemple faire appel à l'interruption 10h (liée à la carte graphique) au travers de l'API noyau **Ke386CallBios()** **[6]**.

Microsoft n'a donc inventé aucun protocole réseau : Windows doit vivre avec les forces et les faiblesses des protocoles existants à l'époque de sa conception.

1.2.2 Couches hautes

Même le « fameux » protocole SMB n'a pas été inventé par Microsoft, mais par IBM, en même temps que le système d'exploitation OS/2, même s'il est vrai que l'implémentation de référence du protocole est désormais celle de Microsoft.

L'indépendance entre les différentes couches protocolaires explique de nombreux problèmes de sécurité. En effet, chaque couche peut gérer un mécanisme de nommage (et donc potentiellement un cache), un mécanisme de fragmentation, un mécanisme de session, etc.

Ceci implique entre autres :

- Qu'il est possible de contourner les IDS/IPS en jouant sur la complexité des protocoles et les mécanismes de fragmentation à chaque couche.
- Qu'il existe des attaques en empoisonnement de cache à tous les niveaux également (ex. : ARP, NetBIOS, etc.).
- Que la plupart des protocoles ayant été conçus pour des réseaux fermés en des temps archaïques ne spécifient aucun mécanisme de sécurité, ou délèguent la sécurité à une autre couche protocolaire.
- Que la compatibilité avec l'existant ne permet pas d'activer les options de sécurité les plus robustes par défaut.

1.3 Authentification

1.3.1 Windows 3.1 / 9x

Ne parlons pas de la fenêtre d'authentification Windows 95, qui pouvait être contournée à l'aide de la touche ESC, ou des fichiers « .PWL » utilisés par Windows 95 pour stocker les mots de passe de l'utilisateur (chiffrés en RC4).

Le système Windows 9x n'ayant pas de notion « d'utilisateur », les ressources réseau partagées sont protégées par un simple mot de passe - qui était malheureusement vulnérable à une attaque distante permettant de deviner les caractères du mot de passe un par un, car le paramètre de taille du `strncmp()` est spécifié par le client **[7]** (faille corrigée par MS00-072).

Samba continue à supporter ce mécanisme d'authentification (*Share-Level security*).

1.3.2 LM/NTLM

Le plus ancien protocole digne d'étude aujourd'hui est donc le « fameux » protocole d'authentification *Lan Manager* (LM), qui a lui aussi été développé par IBM dans le cadre du système OS/2.

Ce protocole présente quasiment toutes les failles cryptographiques et d'implémentation possibles, c'est pourquoi il est intéressant de s'y attarder.

Les vocables « LM » et « NTLM » englobent indistinctement une méthode de *hashage* du mot de passe utilisateur et le protocole d'authentification réseau associé. Il est important de dissocier les deux concepts.

1.3.2.1 Hash LM et NTLM

Au niveau de la transformation du mot de passe en *hash*, l'algorithme est le suivant :

- Le mot de passe est tronqué en deux blocs de 7 caractères, qui sont utilisés comme clés DES de manière indépendante, pour chiffrer une chaîne fixe : `KGS!@#$%`. La concaténation des deux valeurs de sortie est appelée « hash LM ».
- Avant cette opération, le mot de passe est converti en majuscules et la plupart des caractères non supportés sont convertis dans un alphabet réduit, ce qui réduit l'espace des clés à environ 2^{43} possibilités.
- Aucun diversifiant (sel) n'est ajouté dans l'opération. Un mot de passe donné produit toujours le même hash.
- La génération du hash est très peu coûteuse car elle ne demande (au pire) que deux itérations de DES.

Etant donné un hash LM, il est donc trivial de remonter au mot de passe sur du matériel moderne, car l'intégralité de l'espace des clés peut être exploré en quelques heures. En l'absence de sel, il est également possible de générer des tables pré-calculées très performantes (*Rainbow Tables*), qui réduisent la recherche à quelques dizaines de minutes.

Le hash NTLM est généré en prenant l'empreinte MD4 du mot de passe complet au format Unicode, il est donc plus robuste que le hash LM - mais ne corrige pas l'absence de diversifiant et la simplicité calculatoire.

1.3.2.2 Protocoles LM et NTLM

Au niveau de l'authentification réseau, les protocoles LM et NTLM ne transmettent pas le hash directement, mais reposent sur un mécanisme de défi/réponse.

Le serveur envoie un aléa (défi) au client qui le chiffre avec le hash LM et/ou NTLM de l'utilisateur, en fonction de l'algorithme négocié par les deux parties. Le serveur effectue la même opération de son côté (sur la base du hash connu) et compare avec la réponse du client.

Ce protocole souffre des défauts suivants :

- L'aléa doit être unique et imprévisible (ce sont deux propriétés distinctes). Les premières versions de Windows NT utilisaient le même aléa pendant 15 minutes, ce qui ouvre la voie à des attaques par rejeu. En 2010, Hernan Ochoa a découvert que l'état interne du générateur d'aléa était également prédictible à distance **[8]** (faille corrigée par MS10-012).

- Il n'y a pas d'authentification mutuelle entre le client et le serveur (cette propriété n'est disponible que dans les protocoles NTLMv2 ou Kerberos). C'est la cause de l'attaque « SMB Relay » publiée en 2001 par Sir Dystic, qui a été corrigée en 2008 suite à sa redécouverte (et son intégration dans l'outil Metasploit).
- Enfin, la connaissance du mot de passe n'est pas nécessaire pour s'authentifier, la simple connaissance du hash est suffisante. Cette attaque s'appelle « Pass The Hash » **[9]**. Comme la précédente, elle avait été publiée pour la première fois en... 1997 **[10]**.

Il faut savoir que toutes les versions de Windows NT (depuis la 3.1) supportent le protocole NTLM. La génération du hash LM et l'utilisation du protocole LM sont donc des aberrations, sauf pour les utilisateurs qui exploitent encore du Windows 95 sur leur réseau...

Le protocole NTLMv2 assure l'authentification mutuelle entre le client et le serveur, il est supporté depuis Windows NT4 SP4, ne pas l'utiliser est donc inexcusable.

Enfin, depuis Windows Seven et Windows 2008R2, on commence à entrevoir la disparition complète des protocoles LM et NTLM, au profit exclusif de Kerberos. Cette possibilité reste « expérimentale » aux dires de Microsoft **[11]**.

1.3.2.3 Kerberos

Le protocole Kerberos v5 est devenu un composant essentiel de l'authentification Windows avec Windows 2000. Il ne sera toutefois pas traité ici.

1.4 Architectures distribuées

1.4.1 MS-RPC

L'explosion des réseaux informatiques dans les années 1990 s'est accompagnée d'une autre révolution : le *Cloud Computing*. Enfin, à l'époque, on ne parlait pas de Cloud, mais d'architectures distribuées. C'est la grande époque où furent conçus des monuments comme CORBA, ONC-RPC (Sun) et DCE-RPC (OSF), ce dernier servant de base au protocole Microsoft RPC (qui s'en inspire librement mais y ajoute de nombreuses fonctionnalités comme le support des chaînes Unicode).

Le protocole MS-RPC est horriblement complexe **[12]**, en partie à cause de son abstraction complète de la couche de transport (à l'époque, il existait encore plusieurs protocoles réseau utilisés en entreprise). MS-RPC gère donc des mécanismes comme la fragmentation, qui est pourtant fournie « gratuitement » par TCP/IP.

C'est sans compter sur les bogues d'implémentation avec lesquels Microsoft a dû vivre. Ainsi, toutes les implémentations antérieures à Windows NT4 SP4 calculent de manière incorrecte la valeur du champ « Authentication Length » - et la compatibilité avec ce bogue a été maintenue dans les versions ultérieures de Windows **[13]**.

Les principes de fonctionnement du protocole MS-RPC seront décrits plus en détail dans l'article suivant du dossier.

1.4.2 COM

Difficile de parler de Windows sans parler de COM (*Component Object Model*). Difficile de définir COM également, tant ce terme englobe des concepts et des technologies changeantes. L'utilisation « officielle » de ce terme par Microsoft a d'ailleurs évolué au cours du temps : COM désigne au final plusieurs technologies relativement différentes.

Le principe général des technologies de « composants » (il en existe tant et plus - comme XPCOM dans les produits Mozilla, ou UNO dans OpenOffice) est de standardiser une interface (ABI - *Application Binary Interface*), afin de permettre une interaction dynamique entre des composants logiciels autonomes.

- « Dynamique » signifie que les consommateurs de ces composants n'ont pas nécessairement besoin de connaître leurs interfaces à la compilation pour pouvoir les utiliser. Pour cela, il faut néanmoins que le composant soit auto-documenté (c'est-à-dire qu'il implémente l'interface IDispatch dans le cas Microsoft - c'est alors un composant dit « ActiveX »), sinon la bibliothèque de types (TLB - *Type Library*) devra être fournie avec le composant.
- « Autonome » signifie dans ce contexte qu'il n'est pas nécessaire non plus de se préoccuper des détails d'implémentation (langage de programmation utilisé, *endianness* de la plateforme d'exécution, etc.).

Bien sûr, cela pose des problèmes d'implémentation incommensurables, tels que :

- Le *marshalling* des données (ex. : comment un VBScript va-t-il passer une chaîne de caractères à un composant en Delphi ?).
- La gestion des références (ex. : comment le composant en Delphi va-t-il savoir qu'il peut libérer la mémoire allouée pour la chaîne de caractères retournée ?).
- La gestion « côte à côte » des différentes versions du même composant (ex. : MSXML 2.6 et MSXML 6.0).
- Etc.

Les deux premiers points ont été résolus en introduisant des types de données spécifiques au modèle COM : ce sont `BSTR`, `VT_*`, etc.

Le modèle COM est central à la notion de *plugin* dans Windows, que ce soit les extensions d'Internet Explorer (BHO - Browser Helper Object) ou d'Explorer lui-même (ex. : menu contextuel lié au bouton droit de la souris).

Une bonne introduction au fonctionnement de COM (Microsoft) et XPCOM (Mozilla) a été publiée en 2009 par Mark Dowd et al. sous le titre « Attacking Interoperability » **[14]**. Cette étude inclut également les problèmes de sécurité du modèle, ce qui ne gâche rien.

Un composant COM doit au minimum exposer l'interface IUnknown, qui contient les méthodes `AddRef()`, `Release()` et `QueryInterface()`.

Il n'est pas rare qu'un composant COM exporte également `DllRegisterServer()` et `DllUnregisterServer()` pour s'enregistrer dans la base de registre (opération réalisée par la commande `REGSVR32 <COMPOSANT.DLL>`), bien que Windows XP ait introduit la notion de composant COM « Registration Free » **[15]**.

Un composant COM peut s'exécuter dans le processus qui l'invoque, ou dans un processus distinct : le fameux `DLLHOST.EXE` (la documentation parle alors de serveur *out-of-process*). Il est possible de mutualiser plusieurs instances d'objets dans des « appartements » (*apartments* dans la documentation Microsoft). Enfin, un composant COM peut s'exécuter sous une identité différente de celle du *thread* qui l'invoque - tous ces paramètres étant réglables localement par l'administrateur système à l'aide de l'outil DCOMCNFG présenté plus loin.

COM fournit de nombreuses primitives (changement d'identité, hébergement mutualisé de composants, gestion du *multi-threading*, etc.) qui répondent à la quasi-totalité des situations auxquelles un développeur peut être confronté. C'est une technologie puissante et mature. Le prix à payer est une très forte complexité dans les développements...

1.4.3 COM+ / DCOM

« COM+ » est une évolution de la technologie COM apparue avec Windows 2000, qui supporte les transactions distribuées, la notion d'événements, les queues, le *pooling* (les composants ne sont pas déchargés après chaque utilisation), etc. Ces fonctions sont aujourd'hui indissociables de COM, et le nom « COM+ » n'est plus tellement utilisé.

Des technologies de support sont nécessaires au bon fonctionnement de cet ensemble, parmi lesquelles MSMQ (*Microsoft Message Queuing*), MSDTC (*Microsoft Distributed Transaction Coordinator*) et DCOM.

DCOM est le nom donné par Microsoft à la couche de communication réseau qui permet effectivement à des composants COM situés sur des machines différentes de communiquer entre eux. Ce protocole repose massivement sur le protocole MS-RPC. Il a été popularisé en 2003 avec le ver Blaster, qui exploitait une faille dans l'interface MS-RPC IremoteActivation **[16]** (liée à DCOM).

L'ensemble des composants COM et leur configuration sont accessibles à l'aide de l'outil `DCOMCNFG.EXE`. Un simple coup d'œil à cet outil permet de réaliser la complexité de cette technologie.

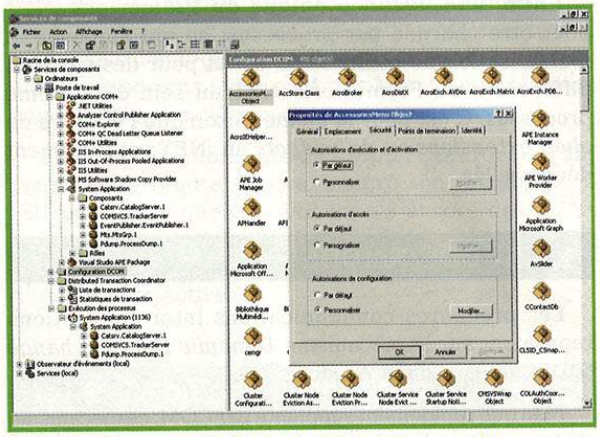


Figure 2 : DCOMCNFG.EXE en action

1.4.4 .NET Remoting / WCF

Avec l'introduction du *framework* .NET, de nouvelles technologies concurrentes à COM/DCOM sont apparues. Il s'agit de *.NET Remoting*, puis WCF (*Windows Communication Framework*) à partir de la version 3.0 du framework .NET.

Ces technologies sont clairement conçues pour être transportées sur le protocole HTTP, bien qu'un protocole natif soit également disponible.

Le protocole MS-RPC (et donc DCOM) souffre en effet de limitations importantes pour le passage à l'échelle d'Internet, comme l'utilisation de ports dynamiques et la difficulté à inspecter/proxy-fier le trafic, ce qui oblige essentiellement à ouvrir tous les ports entre 1024 et 65535 pour qu'une application distribuée fonctionne.

Il existe quelques bémols à cette assertion, comme la possibilité de limiter la plage des ports utilisés **[17]** (à partir de Windows Vista, seuls les ports supérieurs à 49152 sont utilisés par défaut **[18]**), ou le transport RPC sur HTTP.

Cependant, il est irréaliste d'envisager que COM disparaisse à court terme, compte tenu de son importance au sein de l'écosystème Microsoft.

D'autre part, il faut noter que les dernières versions de la technologie .NET Remoting ne rivalisent toujours pas avec les capacités de la technologie DCOM. DCOM est nativement « sûr » grâce aux mécanismes de sécurité disponibles dans le protocole MS-RPC, qui dérivent de SSPI (*Security Support Provider Interface*). A contrario, .NET Remoting est transporté par défaut sur HTTP et le composant doit implémenter lui-même la prise en charge

de la sécurité **[19]** (qui s'avère grandement facilitée si le composant est hébergé au sein du serveur web IIS grâce au support des mécanismes d'authentification sur HTTP(S)).

Enfin, une instance unique du Framework .NET peut être chargée par processus, ce qui empêche la mutualisation de composants écrits pour des versions différentes du Framework .NET au sein d'un même processus. C'est pour la même raison que l'écriture de *shell extensions* ou de *widgets* en .NET est fortement déconseillée.

1.4.5 DDE/NetDDE

Les premières communications inter-applications reposaient sur le mécanisme *Dynamic Data Exchange* (DDE), apparu dans Windows 2.0.

Bien que rapidement supplanté par COM, ce mécanisme reste en usage dans les versions modernes de Windows, puisqu'il est utilisé, par exemple, pour implémenter les associations de fichiers par extension, ou la fonction copier/coller (cf. application **CLIPBRD.EXE** sous Windows XP). Il faut dire que l'utilisation de DDE reste plus simple et plus « générique » que celle de COM.

Microsoft a acquis auprès d'une société tierce une extension - appelée NetDDE - permettant l'échange de données DDE à travers le réseau. Le service NetDDE peut encore être démarré sur Windows XP (**%windir%\system32\netdde.exe**), même s'il est désactivé par défaut. L'application Microsoft la plus connue reposant sur le protocole NetDDE est probablement le jeu « Dame de Pique » en réseau.

1.4.6 OLE

OLE (*Object Linking and Embedding*) est une évolution naturelle de DDE. Au lieu d'échanger des blocs de données binaires non structurées, les applications peuvent désormais échanger des objets complets, qui continuent à « vivre » dans leur application hôte (*Embedding*). Ces objets peuvent être édités in situ si l'application source est installée (*Linking*). Cette fonction est visible dans n'importe quelle application de la suite Microsoft Office, via le menu ***Insérer > Objet***.

Il existe deux versions du standard OLE. La version 2.0 a été entièrement reconstruite autour du modèle COM, étendue sous le nom de « OLE Automation », puis renommée en technologie « ActiveX ».

Un conteneur OLE se définit comme un objet COM qui expose une interface IOleObject.

L'application **PACKAGER.EXE** (livrée avec le système jusqu'à Windows XP inclus) permet de créer un conteneur OLE autour de n'importe quel objet.

1.4.7 ActiveX

La technologie ActiveX (anciennement « OLE Automation ») est l'aboutissement ultime de toutes les technologies vues précédemment.

Un composant ActiveX est un composant COM qui expose l'interface standard IDispatch, qui dérive de IUnknown et y ajoute les méthodes suivantes : **GetTypeInfoCount**, **GetTypeInfo**, **GetIDsOfNames**, **Invoke**.

L'apport principal de cette technologie est la possibilité d'invoquer des méthodes du composant dont le nom et les paramètres sont déterminés dynamiquement à l'exécution. Ceci permet d'intégrer très facilement les composants ActiveX dans des langages de script, par exemple. Ce fut d'ailleurs l'un des objectifs de conception, l'utilisation de composants COM en C/C++ restant relativement... rébarbative (c'est un euphémisme). Sans parler du navigateur Internet Explorer, grand consommateur de composants avec lesquels les pages web n'interagissent que par HTML et JavaScript.

Prenons un exemple bien connu : celui de l'AJAX. Pour faire de l'AJAX avec Internet Explorer 6, le code JavaScript suivant (simplifié) doit être intégré à la page web :

```
var o = new ActiveXObject("MSXML2.XMLHTTP");
o.open("GET", "http://test.com/test.xml", true);
[…]
```

Ceci va créer une instance du composant MSXML2 préalablement installé sur le poste client (ce composant est distribué avec Windows - et a par ailleurs été source de nombreuses failles de sécurité). Il est tout à fait possible d'utiliser le même composant dans un fichier « .VBS » (VBScript) avec la syntaxe suivante (cet exemple est 100 % fonctionnel si exécuté avec l'interpréteur **CSCRIPT.EXE**) :

```
Dim objXML

Function objXML_onreadystatechange()
    If (objXML.readyState = 4) Then
        If (objXML.status = 200) Then
            MsgBox objXML.responseText, 0, objXML.statusText
        End If
    End If
End Function

Set objXML = CreateObject("MSXML2.XMLHTTP")

objXML.Open "GET", "http://test.com/test.xml", False
objXML.OnReadyStateChange = GetRef("objXML_onreadystatechange")
objXML.Send
```

Les objectifs de réutilisation du composant et d'abstraction du langage ont donc été atteints.

Avec Internet Explorer, les choses peuvent être plus complexes. En effet, Internet Explorer peut charger des composants ActiveX. Mais il se comporte lui-même comme un composant. Ainsi, l'exemple de code VBScript suivant va créer une instance visible du navigateur directement ouverte sur le site **http://test.com/**.

```
Dim IE
Set IE = CreateObject("InternetExplorer.Application")
IE.Visible = 1
IE.Navigate "http://test.com/"
```

Cette technique a connu son heure de gloire en tant que méthode de contournement des pare-feu personnels - mais elle est détectée par la plupart des produits de sécurité récents.

1.4.8 En pratique ...

Tout auditeur en sécurité peut se retrouver un jour confronté à un audit de contrôle ActiveX sans disposer des sources.

Malgré l'aide de quelques scripts et plugins (dont le support a été abandonné pour la plupart), l'approche naïve consistant à ouvrir le contrôle dans le logiciel IDA n'est pas la plus simple. En effet, les technologies sous-jacentes conduisent à un empilement important de code.

Il existe plusieurs outils permettant d'énumérer efficacement les méthodes d'un contrôle ActiveX et leurs paramètres. L'un de mes favoris est TypeLib Browser **[20]**. Visual Studio permet également d'énumérer les contrôles installés et les méthodes qu'ils exposent, mais installer Visual Studio est une solution parfois surdimensionnée.

Petit coup de pouce également à l'outil injustement méconnu WinAPIOverride32 **[21]**. Cet outil de *hooking* open source français remplace avantageusement des outils commerciaux comme AutoDebug. Mais surtout, il est capable d'enregistrer et d'interpréter tous les appels à des composants COM ou ActiveX lors de l'exécution d'une application Windows.

Enfin, le vénérable outil de *fuzzing* COMRaider **[22]** peut encore rendre service, malgré son grand âge. Il est regrettable que cet outil trouve encore des failles aujourd'hui, alors qu'il devrait être intégré à tout processus d'assurance qualité lors du développement d'un contrôle ActiveX.

1.4.9 Là où ça devient rigolo

Les technologies COM et DCOM sont à la source de nombreux mécanismes Windows, et donc de nombreuses failles au fil du temps.

On peut penser au ver Blaster, qui exploite une faille d'implémentation dans le processus serveur DCOM.

COM est également la technologie sous-jacente aux ActiveX, qui furent la cause de tant de failles dans Internet Explorer, jusqu'à ce que Microsoft ajoute un certain nombre de mécanismes de sécurité (*Kill Bits*, alerte au premier chargement d'un contrôle sur une page inconnue, etc.).

Mais enfin, les technologies OLE/COM/DCOM sont également à la base d'un autre protocole : *OLE for Process Control* (OPC), qui est utilisé dans de nombreux systèmes SCADA. La fondation OPC cherche aujourd'hui à s'abstraire au maximum de l'implémentation Microsoft de ces protocoles, et à rendre le protocole OPC indépendant des choix technologiques sous-jacents.

2 Mais encore

2.1 .NET

Pour utiliser un raccourci certes rapide mais pas totalement faux, « .NET » est le Java de Microsoft. En effet, Microsoft a conclu en 2001 un procès avec Sun (détenteur des droits sur Java), au terme duquel toute référence à Java devait être bannie des produits Microsoft - au point que Microsoft ait dû republier un Service Pack 1 pour Windows XP, nommé SP1a, qui ne contenait plus la machine virtuelle `MSJAVA.EXE`.

« .NET » est un mot-valise qui englobe plusieurs concepts et implémentations :

- La spécification d'un *bytecode* : CLI - *Common Language Infrastructure*.
- Une machine virtuelle permettant l'exécution de ce bytecode : CLR - *Common Language Runtime* (ex. : Mono sous Linux).
- Des bibliothèques de base que toute implémentation doit offrir : BCL - *Base Class Libraries*.
- Des bibliothèques additionnelles : FCL - *Framework Class Librairies*.

Plusieurs langages de programmation sont actuellement utilisables pour générer du bytecode .NET : C#, VB.NET, ASP.NET, J#, F#, IronPython, IronRuby, COBOL.NET... et même C++ (on parle alors de Managed C++).

La plupart des composants de l'univers .NET sont standardisés ECMA et/ou ISO, par exemple :

- ECMA-334 pour le langage C# (le langage de référence) ;
- ECMA-335 et ISO/IEC 23271 pour CLI et BCL ;
- ECMA-372 pour le langage Managed C++.

L'ensemble « CLR + BCL + FCL » est couramment appelé « framework .NET ». Il en existe plusieurs versions en circulation (la compatibilité ascendante étant assurée dans la plupart des cas) :

- 1.0, utilisable par Visual Studio 2002 ;
- 1.1, utilisable par Visual Studio 2003 ;
- 2.0, utilisable par Visual Studio 2005 (cette version est la plus commune car elle est disponible par défaut sur un Windows XP SP3) ;
- 3.0 ;
- 3.5, utilisable par Visual Studio 2008 ;
- 4.0, utilisable par Visual Studio 2010.

Par ailleurs, .NET présente les mêmes propriétés de sécurité que Java : typage fort évitant les *overflows* en tout genre, *garbage collector* épargnant une gestion manuelle des ressources mémoire, bibliothèques sûres, politique de sécurité applicable par la machine virtuelle, signature de code, ...

Une application .NET présente les mêmes risques de sécurité qu'une application Java : décompilation possible de l'application, possibilité d'invoquer du code natif (si la politique de sécurité l'autorise), impact considérable d'une faille dans la machine virtuelle (ex. : MS07-040, MS09-061) ou les bibliothèques de base (ex. : Padding Oracle contre ASP.NET).

Le lecteur intéressé par la sécurité de la technologie .NET trouvera plus de détails dans la présentation effectuée lors de la conférence *SSTIC* 2010 **[23]**.

2.2 WMI

WMI (*Windows Management Instrumentation*) est une implémentation Microsoft des standards WBEM (*Web Based Enterprise Management*) et CIM (*Common Information Model*), disponible nativement depuis Windows 2000.

L'objectif est de faciliter l'administration de systèmes et de composants hétérogènes en présentant une interface d'administration unifiée. Ainsi, il est aussi facile pour un administrateur utilisant WMI de récupérer la version du BIOS d'une machine distante (classe `Win32_BIOS`) que de démarrer un service Windows (classe `Win32_Service`).

Cette abstraction est possible grâce à la notion de fournisseurs de services (*providers*). Des providers toujours plus nombreux sont fournis avec les révisions successives de Windows (il en existe une centaine dans Windows Seven), ce qui permet d'interroger ou de modifier à peu près tous les composants du système. Des tiers (comme les constructeurs informatiques) peuvent également être amenés à fournir des providers additionnels.

L'installation des « WMI Administrative Tools » (fournis gracieusement par Microsoft) permet d'appréhender la richesse de WMI, qui est une fonction d'administration majeure de Windows, pourtant souvent méconnue.

La syntaxe de base d'une requête WMI est le WQL, un langage très proche du langage SQL. WMI est accessible au travers de l'outil dédié WMIC.EXE, mais également par tout langage de script (ex. : VBScript, PowerShell) et par programmation (ex. : C#). Ma préférence va à l'outil Microsoft Scriptomatic, qui permet d'explorer tous les *namespaces* et toutes les classes, puis génère automatiquement le script VBScript ad hoc.

WMI supporte la notion d'événement et de *callback* - par exemple lors de la création d'un nouveau processus **[24]**. Il est donc possible de construire des *backdoors* sur WMI, une preuve de concept en est le Trojan Moth **[25]**.

Enfin, impossible de présenter WMI sans préciser qu'il a été sous les feux de la rampe grâce à StuxNet. En effet, StuxNet utilisait un fichier MOF pour transformer une faille d'écriture de fichiers arbitraires (MS10-061) en exécution de code à distance. Or les fichiers MOF (*Managed Object Format*) sont utilisés pour décrire les classes exposées par un provider WMI lors de son installation.

WMI repose massivement sur la technologie COM pour son implémentation. Ce point sera abordé plus en détail dans l'article suivant du dossier.

Conclusion

« *Un peuple qui oublie son passé se condamne à le revivre* » - Winston Churchill.

Espérons que la connaissance des circonstances qui ont présidé à la naissance de Windows permettra aux jeunes lecteurs de considérer d'un œil nouveau les tares dont on afflige souvent ce système et dont Microsoft ne saurait être tenu pour seul responsable.

Notons que si la famille Windows NT a été conçue au début des années 1990, toutes ses tares de conception étaient connues dès la fin des années 1990 (en particulier au niveau des mécanismes d'authentification). Il est regrettable qu'il ait fallu attendre la médiatisation de ces failles pour commencer à les voir corrigées ces dernières années.

Les réseaux d'entreprises restent aujourd'hui majoritairement exploités dans une configuration compatible avec Windows 95 - et ce sans aucune justification fonctionnelle - ce qui ne fait qu'aggraver la situation sur le front de la sécurité des réseaux internes. ■

NOTES

[1] http://en.wikipedia.org/wiki/Microsoft_BASIC

[2] http://fr.wikipedia.org/wiki/MS-DOS

[3] http://fr.wikipedia.org/wiki/XENIX

[4] http://fr.wikipedia.org/wiki/Microsoft_Windows

[5] Cf. fonctions Dpmi* dans les symboles de NTVDM.EXE

[6] http://www.blackhat.com/presentations/bh-dc-07/Heasman/Paper/bh-dc-07-Heasman-WP.pdf

[7] http://www.securityfriday.com/tools/SPC.html

[8] http://www.hexale.org/advisories/OCHOA-2010-0209.txt

[9] http://oss.coresecurity.com/projects/pshtoolkit.htm

[10] http://www.securityfocus.com/columnists/486

[11] http://blogs.technet.com/b/askds/archive/2009/10/08/ntlm-blocking-and-you-application-analysis-and-auditing-methodologies-in-windows-7.aspx

[12] DCE/RPC est cité par ailleurs comme exemple de protocole mal conçu car impliquant trop d'intervenants aux intérêts antagonistes : http://en.wikipedia.org/wiki/DCE/RPC

[13] Source : « DCE/RPC over SMB: Samba and Windows NT Domain Internals », Luke Kenneth Casson Leighton, page 39. Epuisé depuis longtemps, ce livre a été la référence absolue sur DCE/RPC pendant des années.

[14] http://www.hustlelabs.com/stuff/bh2009_dowd_smith_dewey.pdf

[15] http://msdn.microsoft.com/en-us/magazine/cc188708.aspx

[16] http://www.hsc.fr/ressources/articles/win_net_srv/rpcss_dcom_interfaces.html

[17] Clé de base de registre HKLM\Software\Microsoft\Rpc\Ports

[18] http://support.microsoft.com/kb/929851

[19] http://msdn.microsoft.com/en-us/library/aa720577(VS.71).aspx

[20] http://www.jose.it-berater.org/

[21] http://jacquelin.potier.free.fr/winapioverride32/

[22] http://labs.idefense.com/software/fuzzing.php#more_comraider

[23] http://www.sstic.org/media/SSTIC2010/SSTIC-actes/Audit_dotNet_et_OCS/SSTIC2010-Article-Audit_dotNet_et_OCS-ruff.pdf

[24] Un exemple fonctionnel est donné dans la newsletter HSC n°77 : http://www.hsc-news.com/archives/2011/000078.html

[25] http://www.security-assessment.com/files/presentations/The_Moth_Trojan.pdf

WINDOWS VU DU RÉSEAU

Aurélien Bordes – aurelien26@free.fr

mots-clés : IMPERSONATION / WMI / DCOM / NAMED PIPE / SMB / NETBIOS / TCP / IP

Le précédent article a présenté les principales technologies peuplant le monde de Windows. Celui-ci continue cette exploration en se concentrant sur les briques orientées réseau. Laissez-vous guider vers l'élaboration de la délicate tour de Babel WMI over DCOM over RPC over named-pipe over SMB over TCP over IP...

1 Mise en garde

Les éléments et paramètres présentés dans cet article peuvent apparaître ou varier d'une version à l'autre de Windows ou d'un service pack. En effet, Microsoft a, au fur et à mesure de l'évolution du système, renforcé sa sécurité. Dans tous les cas, il est recommandé de toujours vérifier si une fonctionnalité est présente ou activée.

2 Le protocole SMB

Le partage de fichiers sous Windows est assuré par le protocole SMB (*Server Message Block*). Ce protocole, très ancien[1], dispose de nombreuses variantes appelées dialectes (PCLAN1.0, LANMAN1.0, LM1.2X002, LANMAN2.1, NT LM 0.12, ...). Une liste des principaux dialectes et leurs différences ainsi que la description du protocole de base et des différents dialectes implémentés par Microsoft sont disponibles sur le site « Windows Open Protocols » (**[SMBDialect]**, **[CIFS]** et **[SMB]**).

Les principales commandes du protocole utilisées entre un client et un serveur sont :

- `SMB_COM_NEGOTIATE` : toute première commande envoyée lors de la mise en place d'une session SMB et négociation du dialecte entre les deux parties.
- `SMB_COM_SESSION_SETUP_ANDX` : commande de négociation des options de sécurité qui, optionnellement, offre au client la possibilité de s'authentifier auprès du serveur. Le protocole SMB repose entièrement sur la SSPI pour réaliser l'authentification (cf. ci-dessous). Les blocs d'authentification échangés entre les SSP sont contenus dans le champ `SecurityBlob`.
- `SMB_COM_TREE_CONNECT_ANDX` : connexion du client à un partage exporté par le serveur.
- `SMB_COM_NT_CREATE_ANDX` (création), `SMB_COM_OPEN_ANDX` (ouverture), `SMB_COM_READ_ANDX` (lecture), `SMB_COM_WRITE_ANDX` (écriture) : principales commandes de manipulation des fichiers.

La SSPI (*Security Support Provider Interface*) est l'interface de programmation pour toutes les fonctions d'authentification à distance. Lors de l'initialisation de la SSPI (via la fonction `InitializeSecurityContext`), le client ou le serveur doit spécifier un SSP (*Security Support Provider*) qui réalisera l'authentification. Le SSP est une bibliothèque implémentant un protocole d'authentification distant. Les principaux SSP intégrés en standard dans un système Windows sont : `msv1_0` (qui prend en charge les protocoles LM/NTLM), `Kerberos` ou `SPNego` (qui négocie l'utilisation de `msv1_0` ou `Kerberos`).

Afin d'être transporté, le protocole SMB doit reposer sur un protocole sous-jacent. Si *NetBIOS Frames Protocol* (NBF), *NetBeui* ou *NetBIOS over IPX* ont été utilisés dans les années 1990, avec la génération de TCP/IP, c'est *NetBios over TCP* (NBT) qui est maintenant principalement utilisé. NetBios met en œuvre différents protocoles de communication :

- *NetBIOS Name Service* (NBNS) sur TCP/UDP port 137 est en charge du nommage des machines et de leur localisation sur le réseau à partir de leur nom (c'est le voisinage réseau).

- *NetBIOS Datagram Service* (NBDS) sur UDP port 138 est utilisé pour le transport non connecté de données.
- *NetBIOS Session Service* (NBSS) sur TCP port 139 est utilisé pour le transport connecté de données.

Avec l'apparition de Windows 2000, Microsoft a permis le transport de SMB directement sur TCP (port 445) sans passer par NetBios. On parle alors de *Direct Hosting*. Dans une installation de base de Windows, les deux mécanismes de transport de SMB sont activés. Cependant, dans un environnement réseau supportant entièrement le Direct Hosting, il est fortement recommandé de désactiver le protocole NetBios sur TCP, qui est particulièrement « bruyant », avec ses annonces en *broadcast*, et qui permet facilement de connaître, par écoute passive, la version du système ou les utilisateurs connectés. L'impact le plus marquant de cette désactivation est la disparition du voisinage réseau lié à l'arrêt de NBSS. La connaissance de l'environnement réseau doit alors être assurée par l'*Active Directory*. Or, pour les petits réseaux avec quelques postes, il n'est pas toujours possible de disposer d'un tel annuaire. Il a fallu attendre Windows Vista et les protocoles *Link Layer Topology Discovery* (LLTD) et *Link-local Multicast Name Resolution* (LLMNR) pour disposer d'un remplaçant à NBSS.

Windows Vista a également vu l'apparition de SMBv2, le remplaçant de SMB[2]. Il s'agit d'une refonte complète du protocole rendue nécessaire afin de combler les nombreuses lacunes de SMB. Parmi les nouveautés les plus importantes, on peut noter des performances accrues (mutualisation des connexions ou des opérations, meilleure gestion de la latence, *handles* persistants) et une sécurité renforcée avec l'utilisation de HMAC-SHA256 pour la signature des échanges[3].

Au sein du système, les principaux composants en charge de la gestion de SMB sont :

- Le service LanmanServer (`srvsvc.dll`) et son pilote associé (`srv.sys`), responsables de la partie serveur. C'est en particulier ce service qui est sollicité lorsque l'on énumère les partages offerts par un système.
- Le service LanmanWorkstation (`wkssvc.dll`) ainsi que le redirecteur SMB (`mrxsmb.sys`) responsables de la partie cliente. Le service gère les sessions montées vers des partages distants (visibles par exemple par la commande `net use`). Quant au redirecteur, il intervient dans l'ouverture de fichiers sur un partage SMB directement à partir des API de manipulation de fichiers (`CreateFile`). Il est également responsable de la mise en cache des *credentials* si une authentification explicite est utilisée.

Lorsque SMB est utilisé pour accéder à un partage de fichiers, deux contrôles de droits d'accès sont effectués. Le premier intervient lors de l'ouverture du partage : chaque partage dispose d'une liste de droits d'accès où il est possible de spécifier des droits macroscopiques (lecture, modification, contrôle total). Le second a lieu lorsque le serveur accède à un fichier demandé par le client : il le fait avec le contexte de sécurité de la session d'authentification ouverte par ce dernier via le mécanisme d'incarnation (*impersonation*). Ainsi, ce sont les descripteurs de sécurité du système NTFS qui sont vérifiés.

Note

L'impersonation (dont la traduction française peut être l'emprunt d'identité, la représentation ou l'incarnation) est un mécanisme fondamental lié à l'authentification sous Windows. Après authentification d'un client, un serveur qui réalise une impersonation prend alors l'identité de ce dernier : le SSP qui a réalisé l'authentification crée un jeton de sécurité que le serveur, s'il dispose du privilège requis, assigne à un thread. Ainsi, toutes les actions du thread sont réalisées dans le contexte de sécurité du client et doivent être correctement autorisées. Si le client ne peut pas ou ne souhaite pas s'authentifier, il utilise un nom d'utilisateur et un mot de passe vide. Dans ce cas, un contexte d'impersonation anonyme est alors créé et l'on parle d'accès anonyme.

2.1 Mécanismes reposant sur SMB

Outre le classique partage de fichiers, deux mécanismes se basent à leur tour sur SMB comme support de transmission. Le premier est celui des *mailslots*. Celui-ci est très basique et offre simplement une communication interprocessus et unidirectionnelle via le réseau. Le fonctionnement est le suivant : un serveur crée un mailslot où un client vient écrire des messages à travers le réseau. Ces messages sont envoyés sous forme de commandes `SMB_COM_TRANSACTION`. Un client ne connaissant généralement pas l'emplacement du mailslot sur le réseau, ses messages sont envoyés sous forme d'annonce *broadcast*. Cette forme d'envoi rend difficile la confidentialité dans les échanges par mailslots.

Le second mécanisme est celui des canaux nommés (*named pipes*). Windows implémente le classique mécanisme d'IPC de canaux (*pipes*). Deux types de pipes existent : les *anonymous pipes*, qui permettent une communication entre un processus père et fils ou au sein d'un même processus, et les *named pipes* qui, en donnant un nom au pipe, permettent une communication entre tout processus se connectant sur le pipe et le processus créateur dudit pipe. Un named pipe est également accessible via le réseau au moyen du protocole SMB. Pour cela, un partage spécial (le fameux IPC$) est créé afin qu'une fois ouvert, un named pipe soit accessible à distance de manière transparente via les traditionnelles API d'écriture et de lecture de fichier. Point important, lorsque le serveur crée le canal nommé, il doit spécifier un descripteur de sécurité définissant les utilisateurs autorisés à l'ouvrir. Sachant qu'il existe déjà un descripteur de sécurité à ce niveau, le partage IPC$ est quant à lui systématiquement accessible en anonyme. La commande `net use \\X.X.X.X\IPC$ /user:"" ""` illustre l'ouverture anonyme de ce partage.

3 Remote Procedure Call

3.1 Présentation

Les *Remote Procedure Calls* (RPC) sont sans aucun doute l'un des mécanismes les plus importants dans les environnements Windows, car ils sont massivement utilisés par de nombreux composants du système. Dans son principe de base, RPC permet à un programme client d'appeler une fonction qui est exécutée en réalité sur un serveur distant. L'environnement d'exécution (*runtime*) RPC est alors responsable de la mise en relation du client et du serveur ainsi que de l'échange des données sur le réseau.

Afin de communiquer correctement, le client doit connaître les prototypes des fonctions exposées par le serveur. Ceux-ci sont définis via un langage spécifique, le MIDL (*Microsoft Interface Definition Language*), qui caractérise pour l'interface une référence unique (un numéro UUID et un numéro de version) et, pour chaque fonction, son prototype. Microsoft publie maintenant toutes les définitions IDL des interfaces RPC utilisées dans ses produits (voir un exemple d'IDL **[ExIDL]**).

Un serveur RPC peut exposer son interface via de nombreux protocoles de transport dont les principaux sont :

- `ncacn_ip_tcp` : l'interface RPC est accessible directement via un port TCP. Ce port peut être choisi par le serveur RPC ou être assigné dynamiquement par le système.
- `ncalrpc` : l'interface RPC est accessible via un port LPC (*Local Inter-Process Communication*)[4]. Ce mécanisme de communication non documenté permet une communication très rapide entre deux processus, mais uniquement au sein d'une même machine[5].
- `ncacn_np` : l'interface RPC est accessible via un canal nommé. Comme vu dans le chapitre précédent, c'est au final le protocole SMB (port 139 ou 445) qui est en charge du transport.
- `ncacn_http` : les échanges RPC sont encapsulés dans des requêtes HTTP, ce qui se révèle très utile dans certains environnements. Par exemple, un client Outlook peut accéder à un serveur Exchange à travers un proxy web **[RPCHTTP]**.

Lorsque le port TCP est assigné par le système, cela pose des problèmes de filtrage réseau, car les ports ne sont pas fixes. Par défaut, sous Windows XP, la plage de ports éphémères réservés est de 1024 à 5000, mais celle-ci peut être changée via l'utilitaire `rpccfg.exe`. Depuis Vista, la plage de ports est fixée de 49152 à 65535 et modifiable via l'utilitaire netsh (`netsh int ipv4 show dynamic tcp`).

L'environnement d'exécution RPC met à disposition du client et du serveur des services de sécurité supplémentaires. Tout d'abord, le serveur peut accepter une authentification de la part du client. De base, la fonction qui s'exécute sur le serveur utilise le contexte de sécurité du processus serveur. Mais la fonction, après authentification du client, a la possibilité de récupérer des informations d'authentification du client ou d'agir en son nom via l'impersonation. L'authentification est entièrement déléguée aux SSP[6]. Ensuite, si l'authentification est effective, il est possible d'activer la protection des échanges suivant différents niveaux allant d'un contrôle d'intégrité jusqu'au chiffrement intégral des communications. Là encore, ces opérations cryptographiques sont totalement transparentes pour l'utilisateur et sont dévolues au SSP ayant pris en charge l'authentification.

Un client souhaitant utiliser une interface (définie par son UUID et la version) va devoir avant tout trouver un des points d'accès du service RPC exposant cette interface. Pour cela, soit le client connaît par avance le point d'accès (s'il est fixe), soit il peut utiliser le service *Endpoint Mapper*, une sorte d'annuaire qui maintient au sein du système une liste des interfaces RPC disponibles ainsi que les canaux de communication associés. Ce service est lui-même un service RPC qui s'exécute dans le célèbre processus `rpcss.exe` et est accessible sur le port TCP/135[7] (celui-ci étant bien sûr fixe). Des outils offrent la possibilité d'énumérer les interfaces RPC enregistrées, par exemple PortQry du *Resource Kit* (`PortQry.exe -n X.X.X.X -e 135`). Cependant, tous les services RPC ne s'enregistrent pas systématiquement et cette méthode ne permet pas d'avoir la liste exhaustive des interfaces RPC du système. Pour cela, il faut utiliser `dbgrpc.exe` des *Debugging Tools* après avoir activé la génération des informations d'état de dépannage RPC (cf. la documentation des Debugging Tools).

RPC se révèle important dans l'environnement Windows car, de l'administration des utilisateurs jusqu'à la gestion de périphériques, en passant par la configuration des services, la quasi-totalité des composants d'un système Windows sont accessibles par RPC (de nombreuses listes très complètes sont disponibles sur Internet **[HSC]**). Cette utilisation ne se limite d'ailleurs pas qu'aux composants du système. Par exemple, toutes les communications MAPI entre un client Outlook et un serveur Exchange reposent sur des interfaces RPC.

4 Accès anonymes

La problématique des accès anonymes est fortement liée à RPC. En effet, suivant les paramètres du système et la configuration d'un serveur RPC, il doit être possible d'appeler une fonction RPC sans s'être préalablement authentifié. Cette problématique s'est longtemps vérifiée pour de nombreuses fonctions RPC intégrées à Windows. Avant Windows 2000, ceci était nécessaire du fait qu'un service s'exécutant dans le contexte SYSTEM n'avait pas

de compte associé et devait donc s'authentifier avec des informations d'authentification nulles. Avec l'arrivée de l'Active Directory, cette situation a évolué et il est possible pour certains processus SYSTEM d'utiliser le compte machine en s'authentifiant via Kerberos. Avec Windows XP est apparu le compte de service NETWORK_SERVICE, qui a la capacité de s'authentifier sur le réseau avec le compte de la machine toujours lorsque celle-ci est dans un domaine. Enfin, depuis Windows 7, la clé **UseMachineId**[8] définit si le compte SYSTEM a la possibilité d'utiliser ou non la session d'authentification de la machine avec le protocole NTLM.

Ainsi, de nombreuses fonctions RPC ont été accessibles anonymement. Parmi les plus utilisées par les outils d'intrusion, on peut citer **NetShareEnumAll** du service LanmanServer, qui liste les partages actifs sur un système, ou encore les fonctions **OpenDomain** et **QueryDisplayInfo** du processus LSASS énumérant les comptes utilisateurs.

Évidemment, Microsoft a, au fur et à mesure de différentes versions de Windows et Service Packs, restreint l'accès anonyme aux interfaces RPC. Ces ajouts de protections ont porté aussi bien sur les interfaces RPC que sur l'environnement d'exécution.

4.1 Rebond RPC

RPC souffre d'un défaut structurel inhérent à la séparation en les interfaces RPC et l'environnement d'exécution en charge des communications. Ainsi, dans un processus, il n'y a pas de lien direct entre les canaux de communication et les interfaces RPC. Lorsqu'une interface ouvre un canal de communication, elle le fait au niveau du processus et toutes les interfaces RPC du processus deviennent accessibles au travers de ce canal. Ceci permet de réaliser un « rebond RPC » dont le principe consiste à accéder une interface RPC au moyen d'un canal de communication moins sécurisé (port TCP, canal nommé autorisé à être accédé en anonyme[9], ...). Afin de se prémunir contre ce type d'attaque, l'interface RPC doit rigoureusement vérifier le moyen d'accès utilisé par ses clients.

5 Restreindre les accès anonymes

5.1 Protection sur l'environnement de communication

Si l'on souhaite restreindre les accès anonymes à une interface RPC, le choix du canal de communication est extrêmement important :

- **ncacn_ip_tcp** : le protocole TCP ne proposant nativement aucun mécanisme de sécurité, c'est à l'interface RPC de s'assurer de l'authentification et de l'autorisation du client.
- **ncalrpc** : chaque port LPC dispose d'un descripteur de sécurité. Cependant, si aucun descripteur n'est spécifié à l'enregistrement de l'interface (via l'appel de la fonction **RpcServerUseProtseq**), c'est théoriquement celui par défaut du processus qui est utilisé et qui n'autorise que l'entité **SYSTEM** et le créateur propriétaire.
- **ncacn_np** : dans ce cas, deux descripteurs de sécurité interviennent : celui du partage IPC$ et celui du canal nommé. Comme vu dans le chapitre précédent, le partage IPC$ est toujours accessible en anonyme. Quant à celui du canal nommé, c'est le même fonctionnement que pour le port LPC ci-dessus.

Dans la pratique, le descripteur par défaut n'est pas celui prévu. Par exemple, dans le cas d'un service RPC accessible par canal nommé, le descripteur de sécurité par défaut accordera l'accès au créateur ainsi qu'aux entités **Tout le monde** et **ANONYMOUS LOGON**.

Afin de restreindre plus efficacement les accès anonymes, il est préférable d'utiliser les clés de registre **NullSessionShares** et **NullSessionPipes**. La première liste les partages SMB qui sont accessibles anonymement. Toutefois, pour des raisons de compatibilité, le partage IPC$ est toujours ajouté implicitement. La seconde énumère les canaux nommés pouvant être accédés anonymement. Là aussi, jusqu'au SP2 de Windows XP et SP1 de Windows 2003, les canaux nommés utilisés par les principaux services du système étaient ajoutés implicitement et ne pouvaient être enlevés **[hardcoded]**.

Enfin, depuis Vista, la situation a considérablement évolué. Si aucun descripteur de sécurité n'est spécifié lors de l'enregistrement de l'interface RPC et que celui par défaut est utilisé, celui-ci possédera un niveau d'intégrité moyen (soit un niveau d'intégrité de 0x2000). Or, lors d'un accès anonyme, le niveau d'intégrité du contexte d'impersonation créé est marqué à « non approuvé » (*UNTRUSTED*, soit un niveau d'intégrité de 0x0000). Ainsi, l'accès n'est plus autorisé. Si un service RPC souhaite être accédé anonymement, il doit, lorsque les canaux de communications **ncalrpc** ou **ncacn_np** sont utilisés, explicitement descendre le niveau d'intégrité à non approuvé dans le descripteur de sécurité. Là encore, pour des raisons de comptabilité, ceux de LSASS (**lsass** ou **protected_storage**) sont mis à ce niveau pour permettre un éventuel accès anonyme. En revanche, les canaux nommés **srvsvc** et **wkssvc** sont positionnés au niveau faible (soit un niveau d'intégrité de 0x1000).

5.2 Protection sur les interfaces

Même si une interface RPC peut être accédée anonymement, elle peut décider à son niveau d'interdire l'exécution de telle ou telle fonction. C'est cette solution qu'a utilisé Microsoft il y a quelques années. Tout d'abord

est apparue la clé de registre **RestrictAnonymous**. Cette clé agit sur les services LanmanServer, LanmanWorkstation et LSASS afin de déterminer les droits d'accès aux fonctions RPC. Par exemple, les droits d'appel à la fonction **NetrShareEnum** de LanmanServer permettant d'énumérer les partages d'un système sont régis par des descripteurs de sécurité[10]. Lorsqu'elle est appelée, la fonction effectue un contrôle d'accès entre le jeton d'accès du contexte de sécurité du client (récupéré après Impersonation) et le descripteur de sécurité stocké dans la valeur **SrvsvcShareFileInfo**. Sous Windows XP, celui-ci contient les données suivantes :

```
Security Descriptor:
   Revision: 1, Size: 188
   Owner: S-1-5-18 (AUTORITE NT\SYSTEM)
   Group: S-1-5-18 (AUTORITE NT\SYSTEM)
   DACL:
      ACE ALLOWED: S-1-5-32-544 (BUILTIN\Administrateurs) (f0013)
      ACE ALLOWED: S-1-5-32-549 (f0013)
      ACE ALLOWED: S-1-5-32-550 (f0013)
      ACE ALLOWED: S-1-5-32-547 (BUILTIN\Utilisateurs avec pouvoir) (f0013)
      ACE ALLOWED: S-1-1-0 (\Tout le monde) (1)
      ACE ALLOWED: S-1-5-7 (AUTORITE NT\ANONYMOUS LOGON) (1)
```

On peut constater qu'il autorise les utilisateurs non authentifiés. Lorsque la clé **RestrictAnonymous** est positionnée, au démarrage du service, l'ACE correspondant au SID de **ANONYMOUS LOGON** est dynamiquement enlevée du descripteur interdisant ainsi l'appel anonyme.

Avec Windows XP, d'autres paramètres sont apparus, tels que la clé **RestrictAnonymousSam** qui interdit, si elle est positionnée, l'appel anonyme à certaines fonctions RPC de LSASS (si **RestrictAnonymous** n'est pas activé) ou encore le paramètre [Permettre la traduction de SID/noms anonymes] dans la politique de sécurité locale de la machine qui, s'il est désactivé, interdit la connexion anonyme à l'interface RPC de LSASS.

6 Renforcement de la sécurité des RPC

Si l'environnement de communication des interfaces RPC a évolué, l'environnement d'exécution RPC s'est également renforcé et propose maintenant de nombreuses options de sécurité.

Tout d'abord, une interface RPC peut définir une fonction de rappel de sécurité (*security callback*). Cette fonction est généralement appelée à la première connexion d'un client, ce qui offre la possibilité de vérifier différents paramètres tels que le canal de communication ou l'authentification.

Avec le SP2 de Windows XP et SP1 Windows 2003, une nouvelle option est apparue : **RestrictRemoteClients**. Cette clé active les restrictions au niveau de l'environnement d'exécution RPC concernant les accès anonymes à toutes les interfaces RPC et peut prendre les valeurs suivantes :

- 0 : il s'agit de la valeur par défaut pour les versions serveur de Windows (2003 et 2008). Dans ce cas, aucune restriction n'est appliquée.
- 1 : il s'agit de la valeur par défaut pour les versions poste client de Windows (XP à 7). Dans ce cas, les accès anonymes sont refusés pour les interfaces RPC, sauf celles qui se sont enregistrées avec l'option **RPC_IF_ALLOW_CALLBACKS_WITH_NO_AUTH**. Il faut noter que certains composants du système n'activent pas ce paramètre. C'est par exemple le cas du service *Endpoint Mapper*, qui ne permet donc plus d'énumérer anonymement les interfaces RPC enregistrées.
- 2 : il s'agit de la protection ultime contre les accès anonymes. Dans ce cas, aucune interface RPC du système ne peut être appelée anonymement.

RPC pose généralement des problèmes avec le filtrage réseau. Lorsque le protocole **ncacn_np** est utilisé, il faut bloquer ou autoriser le port TCP/445, ce qui agit également sur le partage de fichiers. De même, lorsque le protocole **ncacn_ip_tcp** est utilisé, les ports sont dynamiques, ce qui nécessite d'interdire ou d'autoriser des plages de ports. Cependant, le pare-feu intégré de Windows permet de filtrer ou d'autoriser plus finement certains protocoles basés sur RPC. Par exemple, sous Windows XP et l'utilisation des GPO[11], il est possible d'autoriser « l'exception d'administration à distance », ce qui autorise de nombreuses interfaces RPC d'administration. Avec Vista, la finesse a été améliorée avec l'apparition de nombreuses catégories spécifiques dans le pare-feu de l'architecture WPF (*Windows Filtering Platform*). Celui-ci est maintenant configurable via la stratégie de sécurité.

Vista apporte également une autre protection intéressante avec la possibilité de filtrer une interface RPC particulière toujours via le pare-feu intégré du système **[netshrpc]**. Cette possibilité se révèle très pratique afin d'interdire l'accès à une interface vulnérable en attendant un correctif de sécurité.

7 DCOM

Outre de nombreux services du système qui exportent des interfaces RPC, des technologies se basent sur RPC pour leurs échanges réseau. C'est le cas de DCOM (*Distributed Component Object Model*), une technologie qui permet l'instanciation d'objets COM à distance via le réseau.

La gestion de DCOM s'effectue via le service de composants, accessible via **comexp.msc**. Il est d'ailleurs possible, dans les propriétés du poste de travail, de désactiver complètement DCOM sur un système.

En matière de trafic réseau, DCOM génère deux flux successifs de communication, tous les deux étant basés

sur RPC. La première communication est l'instanciation de l'objet via la méthode `RemoteCreateInstance` de l'interface `IRemoteSCMActivator`. Cette interface s'exécute dans le processus `rpcss.exe` et est accédée via le port TCP/135. Cette fonction renvoie les canaux RPC par lesquels l'objet créé est accessible. La seconde communication est l'accès à l'objet DCOM via un canal reçu précédemment, TCP étant privilégié.

8 WMI

WMI est l'implémentation pour Windows de WBEM (*Web-Based Enterprise Management*), un mécanisme de gestion à distance de système. WMI offre une gestion du système d'exploitation ou des composants installés via un modèle objet. WMI étant un standard, et il existe des clients WMI sous Linux. Un des atouts de WMI est la possibilité d'effectuer des requêtes via un langage proche du SQL, le WQL (*WMI Query Language*). Dans les faits, WMI repose sur des objets COM, ce qui le rend très simple à utiliser, en particulier via des langages de script supportant l'automation, comme VB Script. En outre, l'utilisation des objets COM est possible à distance via DCOM, donc RPC.

La richesse des possibilités de WMI est méconnue : il existe de nombreux objets dans l'espace de nommage de WMI, qui permettent de connaître énormément d'informations sur un système. Trois exemples parmi tant d'autres :

- Dans l'espace `Root\CIMV2`, pour chaque interface réseau présente sur le système, il existe une instance de la classe `Win32_NetworkAdapterConfiguration`. Cette classe définit, entre autres, une propriété `SetTcpipNetbios` qui indique ou définit l'état de NetBios sur TCP. Il est donc extrêmement facile, via un script VBS, de désactiver NetBios sur TCP via WMI, localement ou à distance.
- Dans l'espace `Root\CIMV2`, les instances de `Win32_Product` ou de `Win32_QuickFixEngineering` permettent de lister les programmes ou les correctifs de sécurité installés.
- Dans l'espace `Root\SecurityCenter`[12], si des produits conformes aux exigences de Microsoft sont installés, il existe une instance de type `AntiSpywareProduct`, `AntiVirusProduct` ou `FirewallProduct` pour chaque produit de ce type installé. Leur inventaire est donc réalisable à distance toujours via WMI.

Afin de tester WMI, le plus simple est d'utiliser `wmic`, un utilitaire intégré à Windows permettant d'effectuer des

UseMachineId	Sécurité réseau : autoriser Système local à utiliser l'identité de l'ordinateur pour NTLM
NullSessionShares	Accès réseau : chemins et sous-chemins de Registre accessibles à distance
NullSessionPipes	Accès réseau : les canaux nommés qui sont accessibles de manière anonyme
RestrictAnonymous	Accès réseau : ne pas autoriser l'énumération anonyme des comptes et partages SAM
RestrictAnonymousSam	Accès réseau : ne pas autoriser l'énumération anonyme des comptes SAM
Descripteur de sécurité situé dans HKLM\SECURITY\Policy\SecDesc\(Default)	Accès réseau : permet la traduction de noms/SID anonymes

Tableau 1 : Correspondance entre les clés de registre et les paramètres de sécurité

requêtes WMI en ligne de commandes sur un système. Le paramètre **/NODE** indique en outre l'hôte distant sur lequel les requêtes doivent être exécutées. Par exemple, pour connaître les correctifs installés sur une machine, il suffit d'utiliser la commande **wmic qfe**[13]. WMI Explorer est un équivalent en version graphique et permet, en plus, de pouvoir énumérer toutes les interfaces et classes disponibles.

Conclusion

Bien que cet article ne soit sans doute pas exhaustif, il a tenté de présenter les principaux protocoles réseau sous Windows ainsi que les problématiques de sécurité associées. Un système Windows actuel met en œuvre de nombreux protocoles réseau dont certains peuvent être complexes (RPC) ou très utiles (WMI). Quant à la sécurité, même si elle a structurellement évolué (en particulier concernant la problématique des accès anonymes), rien de pourra remplacer un audit de sécurité d'un système Windows. ■

REMERCIEMENTS

Merci à Benjamin Caillat et Nicolas Ruff (c'est mon tour) pour leur relecture.

RÉFÉRENCES

[SMBDialect] http://msdn.microsoft.com/en-us/library/ee441843(PROT.10).aspx

[CIFS] http://msdn.microsoft.com/en-us/library/cc224428(PROT.13).aspx

[LPC] http://blogs.msdn.com/b/ntdebugging/archive/2007/07/26/lpc-local-procedure-calls-part-1-architecture.aspx

[ExIDL] http://msdn.microsoft.com/en-us/library/cc250320(PROT.10).aspx

[hardcoded] http://blogs.msdn.com/b/spatdsg/archive/2006/05/15/598260.aspx

[netshrpc] http://technet.microsoft.com/en-us/library/cc730626(WS.10).aspx

[HSC] http://www.hsc.fr/ressources/articles/win_net_srv/index.html.fr

[RPCHTTP] http://technet.microsoft.com/en-us/library/aa996072(EXCHG.65).aspx

RPC Security Considerations http://msdn.microsoft.com/en-us/library/ms818841.aspx

NOTES

[1] Les premières versions datent de la fin des années 1980.

[2] Windows 7/2008R2 supportant en plus SMBv2.1.

[3] Les observateurs avertis ajouteront également les vulnérabilités MS07-063 et MS09-050.

[4] Il y a toujours confusion sur la signification de LPC entre Local Inter-Process Communication et Local Procedure Calls. Au sein de Windows, LPC n'est pas directement lié aux RPC, c'est donc la première signification qui semble la plus exacte. Ceci est d'ailleurs confirmé par la « Microsoft Platforms Global Escalation Services team » [LPC].

[5] Remplacé à partir de Vista par les ALPC (A pour Advanced), qui apportent des fonctionnalités supplémentaires.

[6] Si les canaux nommés sont utilisés (`ncacn_np`), il n'y pas d'authentification au niveau de RPC et c'est l'authentification au partage IPC$ qui est utilisée.

[7] Mais également le canal nommé epmapper et le port LPC epmapper.

[8] Le tableau 1 indique la correspondance entre les clés de registre et les paramètres de la stratégie de sécurité locale.

[9] En particulier le canal browser du service Explorateur d'ordinateur.

[10] Ces descripteurs sont stockés dans la base de registre sous la clé `HKLM\SYSTEM\CurrentControlSet\services\LanmanServer\DefaultSecurity`.

[11] Configuration ordinateur/Modèle d'administration/Résau/Connexion réseau/Pare-feu Windows.

[12] SecurityCenter2 à partir de Windows Vista.

[13] qfe est en réalité un alias de « Select * from Win32_QuickFixEngineering ». La liste des alias est donnée par les instances de `MSFT_CliAlias` dans `\root\cli`.

Complétez votre collection des anciens numéros de...

Les 4 façons de commander !

Par courrier
En nous renvoyant ce bon de commande.

Par le Web
Sur notre site : www.ed-diamond.com.

Par téléphone
Entre 9h-12h & 14h-18h au 03 67 10 00 20 (paiement C.B.)

Par fax
Au 03 67 10 00 21 (C.B. et/ou bon de commande administratif)

MISC

MISC HORS-SÉRIE

Retrouvez tous les anciens numéros ainsi que nos offres spéciales sur notre site : http://www.ed-diamond.com

Bon de commande MISC Hors-série

Réf.	Désignation	Prix / N°s
MISCHS N°2	CARTES À PUCE - Découvrez leurs fonctionnalités et leurs limites	8,00 €

Bon de commande MISC

Réf.	Désignation	Prix / N°s
MISC N°44	Compromissions électromagnétiques	8,00 €
MISC N°45	La sécurité de Java en question	8,00 €
MISC N°46	Construisez et validez votre sécurité	8,00 €
MISC N°47	La lutte antivirale, une cause perdue ?	8,00 €
MISC N°48	Comment se protéger contre la peste spam ?	8,00 €
MISC N°49	Vulnérabilités Web et XSS - Des ennemis que vous sous-estimez !	8,00 €
MISC N°50	La sécurité des jeux	8,00 €
MISC N°51	La sécurité des jeux	8,00 €
MISC N°52	4 Outils indispensables pour tester votre sécurité !	8,00 €
MISC N°53	La sécurité du wi-fi, des paroles en l'air ?	8,00 €
MISC N°54	Anonymat sur internet : Risque ou nécessité ?	8,00 €

Bon de commande

à remplir (ou photocopier) et à retourner aux Éditions Diamond – MISC – BP 20142 – 67603 Sélestat Cedex

	Référence	Prix / N°	Qté	Total
EXEMPLE :	MISC N°42	8,00 €	1	8,00 €
			TOTAL :	
			FRAIS DE PORT FRANCE MÉTRO. :	+3,9 €
			FRAIS DE PORT HORS FRANCE MÉTRO. :	+6 €
			TOTAL :	

Voici mes coordonnées postales :

Nom :

Prénom :

Adresse :

Code Postal :

Ville :

Je choisis de régler par :

❑ Chèque bancaire ou postal à l'ordre des Éditions Diamond

❑ Carte bancaire n° |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|

Expire le : |_|_|_|_|

Cryptogramme visuel : |_|_|_|

Date et signature obligatoire

Retrouvez les sommaires et commandez tous nos magazines sur notre site :

http://www.ed-diamond.com

Avez-vous l'âme du collectionneur ?

Boostez votre collection !

Vous recherchez un magazine en particulier ? Allez sur www.ed-diamond.com pour voir le sommaire détaillé de chaque magazine et ensuite... Boostez votre collection avec les « Power packs x5 », soit 5 MISC pour 25€ et les « Power packs x10 », soit 10 MISC pour 40€, à choisir dans la liste ci-dessous :

Les 4 façons de commander !

Par courrier
En nous renvoyant ce bon de commande.

Par téléphone
Entre 9h-12h & 14h-18h au 03 67 10 00 20 (paiement C.B.)

Par le Web
Sur notre site : www.ed-diamond.com.

Par fax
Au 03 67 10 00 21 (C.B. et/ou bon de commande administratif)

5 Nos de MISC Multi-System & Internet Security Kbook 25€ ou 10 Nos de MISC Multi-System & Internet Security Kbook 40€

Choisissez vos numéros dans le tableau ci-dessous*
* Seuls les numéros ci-dessous sont disponibles pour une commande de Power Packs x5 et x10

- N°1 Les vulnérabilités du Web !
- N°2 Windows et la sécurité
- N°4 Internet un château construit sur du sable? ...ou les protocoles réseaux en question
- N°6 Sécurité du wireless ?
- N°7 La guerre de l'information - évaluation, riques, enjeux
- N°8 Honeypots - Le piège à pirate !
- N°9 Que faire après une intrusion ?
- N°10 VPN - Virtual Private Networt - Créez votre réseau sécurisé sur internet
- N°11 Test d'intrusion - Mettez votre sécurité à l'épreuve !
- N°12 La faille venait du logiciel
- N°13 PKI - Public Key Infrastructure
- N°14 Reverse Engineering - Retour au sources
- N°15 Authentification
- N°16 Télécoms - Les risques des infrastructures
- N°17 Comment lutter contre - Le spam, les malwares, les spywares ?
- N°18 Dissimulation d'information
- N°19 Les Déni de Services - La menace rôd
- N°20 Cryptographie malicieuse : quand les vers et virus se mettent à la crypto
- N°21 Limites de la sécurité
- N°22 Superviser sa sécurité
- N°23 De la recherche de faille à l'exploit
- N°24 Attaques sur le Web
- N°25 Bluetooth, P2P, Messageries instantanées : Les nouvelles cibles
- N°26 Matériel, mémoire, humain, multimédia : Attaques tous azimuts
- N°27 IPv6 : Sécurité, mobilité et VPN, les nouveaux enjeux
- N°28 Exploits et correctifs : Les nouvelles protections à l'épreuve du feu
- N°29 Sécurité du coeur de réseau IP : un organe critique
- N°30 Les protections logicielles
- N°31 Le risque VoIP
- N°32 Que penser de la sécurité selon Microsoft ?
- N°33 RFID - Instrument de sécurité ou de surveillance ?
- N°34 Noyau et rootkit
- N°36 Lutte informatique offensive - Les attaques ciblées
- N°37 Déni de service
- N°38 Code malicieux - Quoi de neuf ?
- N°39 Fuzzing - Injectez des données et trouvez les failles cachées
- N°40 Sécurité des réseaux - Les nouveaux enjeux
- N°41 LA CYBERCRIMINALITÉ ...ou quand le net se met au crime organisé
- N°42 LA VIRTUALISATION : Vecteur de vulnérabilité ou de sécurité ?
- N°43 La sécurité des web services

Numéros MISC épuisés :
N°3, N°5 et N°35

Bon de commande power packs

à remplir (ou photocopier) et à retourner aux Éditions Diamond – MISC – BP 20142 – 67603 Sélestat Cedex

Cochez ici ▲ POWER PACKS X5

OUI, je désire acquérir un power pack X5	1er 1PP* X5	2ème 2PP* X5	3ème 3PP* X5
1, MISC N°			
2, MISC N°			
3, MISC N°			
4, MISC N°			
5, MISC N°			
Total par série de POWER PACKS X5 :	25 €	50 €	75 €
Les hors-séries et les numéros spéciaux sont exclus des PP*	TOTAL :		
Ex : Achat d'un POWER PACK x5 : - France Métro. : Total = 25€ + 4€ de frais de port par pack. - HORS France Métro. : Total = 25€ + 9€ de frais de port par pack.	FRAIS DE PORT : FRANCE MÉTRO. : +4 € × (X PACK) HORS FRANCE MÉTRO. : +6 € × (X PACK)		
* PP= POWER PACK	TOTAL :		

Cochez ici ▲ POWER PACKS X10

OUI, je désire acquérir un power pack X10	1er 1PP* X10	2ème 2PP* X10	3ème 3PP* X10
1, MISC N°			
2, MISC N°			
3, MISC N°			
4, MISC N°			
5, MISC N°			
6, MISC N°			
7, MISC N°			
8, MISC N°			
9, MISC N°			
10, MISC N°			
Total par série de POWER PACKS X10 :	40 €	80 €	120 €
Les hors-séries et les numéros spéciaux sont exclus des PP*	TOTAL :		
Ex : Achat d'un POWER PACK x10 : - France Métro. : Total = 40€ + 8€ de frais de port par pack. - HORS France Métro. : Total = 40€ + 12€ de frais de port par pack.	FRAIS DE PORT : FRANCE MÉTRO. : +8 € × (X PACK) HORS FRANCE MÉTRO. : +12 € × (X PACK)		
* PP= POWER PACK	TOTAL :		

Voici mes coordonnées postales :
Nom :
Prénom :
Adresse :
Code Postal :
Ville :

Je choisis de régler par :

- ❑ Chèque bancaire ou postal à l'ordre des Editions Diamond
- ❑ Carte bancaire n° |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|

Expire le : |_|_|_|_|

Cryptogramme visuel : |_|_|_|

Date et signature obligatoire

CONTOURNEMENT DES SÉCURITÉS APPLICATIVES SOUS WINDOWS 7

Florent Hochwelker (@TaPiOn) – Consultant Sécurité chez XMCO – florent.hochwelker@xmco.fr
Axel Souchet (@0vercl0k) – 0vercl0k@tuxfamily.org
Mysterie

mots-clés : WINDOWS 7 / BYPASS / DEP / ASLR / SAFESEH / SANDBOX / EMET

Depuis l'apparition des premières failles applicatives, Microsoft a mis en place de nombreuses sécurités limitant leurs exploitations : GS cookie, SafeSEH, SEHOP, DEP, ASLR n'auront bientôt plus de secrets pour vous ! Ces protections ont leurs limitations et peuvent dans certains cas être contournées...

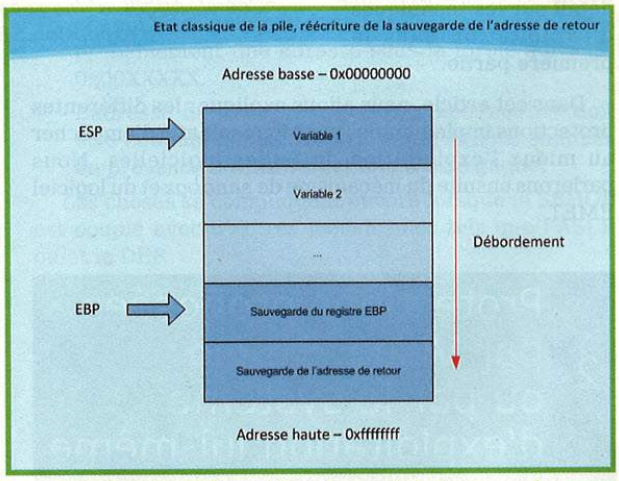
1 Introduction

Depuis des années, l'exploitation de failles de type « buffer overflow » a permis de compromettre des systèmes informatiques.

Le principe d'exploitation le plus connu est le suivant :

1) Corrompre la mémoire à l'aide d'erreurs présentes dans le logiciel, généralement un débordement de mémoire sur la pile. Lorsque le processeur appelle une fonction (instruction `call`), l'adresse de retour de la fonction appelante est stockée dans la pile. Or celle-ci est aussi utilisée pour le stockage des variables. Si un débordement survient lors de l'écriture dans un *buffer*, il est possible de modifier l'adresse de retour, et donc de rediriger le flux d'exécution à l'adresse de notre choix au moment de l'exécution de l'instruction `ret`.

2) Une fois le flux d'exécution contrôlé, il est possible de sauter sur un bloc de code assembleur (*payload*) précédemment injecté en mémoire. Celui-ci va permettre à l'attaquant de prendre la main sur la machine, lui donnant généralement une invite de commande (d'où l'appellation *shellcode*).



Cas classique d'un débordement de tampon dans la pile

Mais d'autres données que l'adresse de retour peuvent être écrasées dans le but de contrôler l'exécution. Citons par exemple le **SEH** (*Structured Exception Handling*) : lors de l'utilisation de **try-catch** dans un logiciel, une liste chaînée d'éléments appelée SEH est positionnée sur la pile. Chaque élément est composé d'une adresse vers un gestionnaire d'exceptions, appelé *handler* d'exception, et de l'adresse de la structure suivante, comme dans une liste chaînée classique.

Lorsqu'une erreur survient, la fonction **ntdll!KiUserDispatchException** va parcourir la liste chaînée et ainsi appeler un handler d'exception.

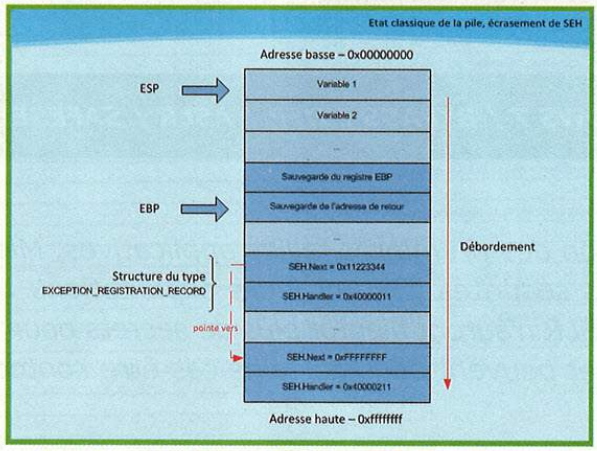
Note

Rappel de la structure :

```
typedef struct _EXCEPTION_REGISTRATION_RECORD
{
/*0x000*/     struct _EXCEPTION_REGISTRATION_RECORD* Next;
/*0x004*/     PVOID Handler;
}EXCEPTION_REGISTRATION_RECORD, *PEXCEPTION_REGISTRATION_RECORD;
```

Lors d'un débordement de mémoire sur la pile, il est possible d'écraser un des handlers d'exception afin de rediriger le flux d'exécution sur notre payload.

Dans le cas de débordement dans une zone mémoire différente de la pile, d'autres données, comme le **VEH** (*Vectored Exception Handling*) ou le **UEF** (*Unhandled Exception Filter*), peuvent être ciblées. Nous reviendrons en détail sur les techniques d'exploitation dans la première partie.



Cas classique d'une réécriture d'un gestionnaire d'exception SEH dans la pile

Dans cet article, nous allons expliquer les différentes protections implémentées par Microsoft afin d'empêcher au mieux l'exploitation de failles logicielles. Nous parlerons ensuite du mécanisme de *sandbox* et du logiciel EMET.

2 Protections proposées par Visual Studio ou par le système d'exploitation lui-même

2.1 /GS

Avec la sortie de Microsoft Visual Studio 2002, la première protection permettant de lutter contre l'exploitation des débordements sur la pile a été introduite. Celle-ci se base sur l'ajout d'un *cookie* sur la pile et a pour objectif de s'assurer que l'adresse de retour stockée dans celle-ci n'a pas été écrasée lors d'un débordement d'une copie dans les variables locales. Cette sécurité devait être explicitement activée avec le flag **/GS** lors de la compilation d'un programme. Lorsqu'un programme est compilé avec ce paramètre, toute fonction jugée dangereuse par le compilateur (utilisation de tableau sur la pile, par exemple) se voit légèrement modifiée au niveau de son prologue et de son épilogue. Au tout début de la fonction, une valeur de 4 octets (sur x86 32bits) est ajoutée sur la pile entre le buffer et les variables : **[buffer][cookie][var1][var2][ret]**. L'intégrité du cookie est vérifiée à la fin de l'exécution de la fonction. Si celui-ci diffère de **mon_programme!__security_cookie**, alors il y a eu une corruption de la mémoire et le processus s'arrête, empêchant toute exploitation. Depuis VS2005, il est possible de forcer l'ajout du cookie dans toutes les fonctions du programme en ajoutant la ligne dans le code source du programme :

```
#pragma strict_gs_check(on)
```

La valeur du cookie est calculée lors du chargement du programme par la fonction **__security_init_cookie** et est stockée dans la variable **mon_programme!__security_cookie**. Le cookie est une valeur de 4 octets dont 2 octets à **0x00**. Ceci permet de bloquer les fonctions de copie de chaîne de caractères (**strcpy**, **wcscpy**, ...). La fonction **__security_init_cookie**, qui initialise le cookie au démarrage du processus, se base sur des paramètres retournés par des fonctions comme : **kernel32!GetCurrentProcessId**, **kernel32!GetTickCount** ou encore **kernel32!QueryPerformanceCounter**.

Cette protection peut cependant être contournée en écrasant un pointeur dans la liste chaînée d'exceptions (SEH) située sur la pile. Dans un programme C++, il est également possible de réécrire la **virtual table** d'un objet **[!]**. Si nous pouvons exécuter un grand nombre de fois la tentative d'exploitation, il est aussi envisageable de tenter une attaque par brute-force sur le cookie. D'autres techniques, comme la réécriture du cookie dans **mon_programme!__security_cookie**, sont possibles, mais nécessitent une deuxième vulnérabilité permettant l'écriture arbitraire de 4 octets.

2.2 SafeSEH

Introduite lors de la sortie de Microsoft Visual Studio 2005, cette protection a pour objectif de protéger les SEH en cas de débordement sur la pile. SafeSEH n'a d'effet que sur une version récente de Windows (Windows XP SP2 ou mieux).

Ce mécanisme peut être activé en précisant simplement l'option **/SAFESEH** à l'éditeur de liens de Microsoft. Celui-ci va alors générer une table contenant les adresses vers les gestionnaires d'exceptions déclarés « sûrs ». À noter que celle-ci est bien évidemment spécifique au module en question. Cette table peut être localisée grâce à la structure **IMAGE_LOAD_CONFIG_DIRECTORY32** (pour la version 32bits) du binaire PE et plus particulièrement grâce aux champs **SEHandlerTable** et **SEHandlerCount**.

Lorsqu'une exception est générée par un thread, la fonction **ntdll!KiUserDispatchException** est déclenchée afin de sélectionner le gestionnaire d'exceptions adéquat, bien évidemment celui-ci doit être validé par **ntdll!RtlIsValidHandler** avant son appel. La protection apportée réside donc dans le code de la fonction **ntdll!RtlIsValidHandler** ; acceptant ou rejetant l'appel d'un handler.

L'exploitation d'un débordement de tampon en écrasant une structure SEH présente sur la pile se révèle alors a priori impossible. Cependant, il existe différents cas spécifiques où une fonction absente de la table des gestionnaires d'exceptions peut être appelée. Pour les connaître, nous vous recommandons la lecture du pseudo-code de la fonction **ntdll!RtlIsValidHandler** réalisée par Alexander Sotirov lors de la BH d'août 2008 **[2]**.

Plusieurs techniques sont connues pour passer au travers de ce mécanisme de sécurité. La principale est d'utiliser l'adresse d'un module n'étant pas protégé par le *SafeSEH* pour écraser un des handlers. Au vu de la fonction de validation, un handler situé dans un module non protégé est reconnu valide. Si un module non protégé est chargé dans le contexte du processus, celui-ci suffira à mettre à mal la sécurité de l'application.

Dans le cas où le dispositif DEP n'est pas actif, il est aussi possible d'utiliser l'adresse d'une zone mémoire n'appartenant pas à un module. Par exemple, le fichier **locale.nls** est chargé en mémoire par un grand nombre de processus et n'est pas exécutable. Il est possible de « sauter » dans sa zone de données seulement lorsque la sécurité DEP est désactivée.

2.3 SEHOP

Aussi connu sous le nom de « SEH Chain Validation », cette sécurité est une amélioration du SafeSEH. Présente depuis Microsoft Windows Vista SP1, son objectif est de contrôler l'intégrité de la chaîne des SEH. Les structures SEH forment une liste chaînée dans la pile (voir la définition de la structure un peu plus haut), le dernier maillon est un peu spécial, car son champ **Next** a la valeur 0xFFFFFFFF et son champ **Handler** est l'adresse de **ntdll!FinalExceptionHandler**. Le programme devra donc valider l'ensemble de la chaîne avant de déclencher un quelconque gestionnaire d'exceptions. Habituellement, l'attaquant réécrit la structure de façon à contrôler le champ **Handler** afin de diriger l'exécution du logiciel vers du code malveillant, mais pour cela, il aura forcément écrasé le champ **Next**, la liste chaînée est alors corrompue : c'est là que le mécanisme SEHOP **[3]** intervient.

Cependant, il ne faut pas oublier que cette protection n'est pas active par défaut sur Microsoft Windows 7 (à l'inverse de Microsoft Windows Server 2008) pour des problèmes de compatibilité. Dans le cas où vous souhaiteriez l'activer, vous avez deux choix :

- Installer le « fixit » **[4]** proposé par Microsoft afin de protéger l'ensemble de vos applications par le SEHOP.
- Paramétrer l'activation par processus. Voici un exemple de modification de la base de registre afin de l'activer pour Internet Explorer :

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\
CurrentVersion\Image File Execution Options\iexplore.exe]
"DisableExceptionChainValidation"=dword:00000000
```

Pour contourner cette protection, il faudra forger soi-même « la maille finale » dans une zone que nous avons sous contrôle, classiquement la pile, pour ensuite écraser le SEH de façon intelligente : c'est-à-dire en faisant pointer son champ **Next** vers notre dernier maillon. Il reste ensuite d'autres problématiques à gérer, et pour cela, nous vous renvoyons à l'article écrit par Heurs & Virtualabs **[5]**.

En revanche, nous tenons à être bien clairs : la technique exposée permet d'outrepasser le SEHOP sous certaines contraintes :

- La vulnérabilité doit permettre l'injection d'octets nuls afin de forger correctement le maillon final : son champ **Next** pointera dans la pile, il aura donc très probablement une adresse sous la forme suivante 0x00XXXXX.
- L'adresse de **ntdll!FinalExceptionHandler** doit être connue, autant vous dire que si nous sommes en présence d'ASLR, c'est loin d'être gagné.

Les choses se compliquent encore lorsque le SEHOP est couplé avec d'autres mécanismes tels que l'ASLR ou/et le DEP.

2.4 Protection pour les pointeurs de fonctions

2.4.1 ntdll!RtlEncodePointer

Pour protéger l'intégrité des pointeurs de fonctions utilisés par les mécanismes de gestion d'exceptions, le système utilise la fonction **RtlEncodePointer** afin de coder les pointeurs.

Celle-ci va, en interne, forger une « graine » de 4 octets (sur architecture x86 32bits) afin d'appliquer au pointeur de fonction l'opération mathématique réversible suivante :

```
ROR(pointeur ^ graine, graine & 0x1F)
```

Note

L'instruction assembleur ROR permet de faire une rotation de bit vers la droite, et son opération inverse est ROL ; celle-ci pivotant les bits vers la gauche.

En utilisant le champ `EPROCESS.Cookie`, la fonction va savoir si elle à déjà généré ou non une « graine ». Si le cookie vaut NULL, il faut donc en créer une. Celle-ci est construite une fois, pour toute l'exécution du processus, en utilisant une entropie basée sur : le nombre de défauts de page que le cœur sur lequel s'exécute le processus a comptabilisé (`KPRCB.MmPageFaultCount`), le nombre d'interruptions (`KPCRB.InterruptCount`) et d'autres éléments, comme le temps système, etc.

2.4.2 Unhandled Exception Filter

Il est possible de définir dans son programme une fonction chargée de gérer une exception qui n'a pas déjà été interceptée par un SEH. On parle de gestionnaire d'exceptions final ; celui-ci va s'occuper d'intercepter les exceptions qui n'ont pas pu être gérées par l'application. Ce gestionnaire peut être défini en appelant la fonction `kernel32!SetUnhandledExceptionFilter`, qui va stocker dans sa zone de données l'adresse de ce gestionnaire d'exceptions un peu spécial. Pour éviter que cette zone soit réécrite et utilisée par un attaquant ayant pour objectif de compromettre l'application, la fonction va coder ce pointeur par le biais de `ntdll!RtlEncodePointer`. Cela fragilise grandement la fiabilité d'une attaque et augmente considérablement la sécurité. En effet, il faut être capable de coder soi-même son pointeur avant de réaliser l'écrasement de sa valeur, or ceci est quelque chose de quasiment impossible.

2.4.3 VEH

Il existe un système complémentaire à celui-ci, appelé *Vectored Exception Handling*, souvent abrégé VEH. Ceux-ci ne sont pas stockés dans la pile, mais dans une liste chaînée dont la tête est stockée dans `ntdll!LdrpVectorHandlerList`. Ils seront tous appelés avant même les SEH lorsqu'une exception sera déclenchée. Dernier petit détail, les VEH sont valables dans le contexte d'un processus entier alors qu'un SEH l'est uniquement pour un seul « thread » de l'application. Les fonctions `kernel32!AddVectoredExceptionHandler` ou `kernel32!RemoveVectoredExceptionHandler` peuvent être utilisées pour ajouter ou supprimer un VEH. Comme tous les pointeurs de fonctions, celui-ci faisait office de cible pour les attaquants, lors d'exploitation de « heap-based overflow » **[6]**, par exemple. Depuis Microsoft Windows XP SP2, les pointeurs ajoutés dans la liste sont protégés par `ntdll!RtlEncodePointer`. Il est donc difficile de les utiliser à des fins malveillantes. Pour les personnes curieuses de voir à quoi ressemble une « pseudo-implémentation » des fonctions `ntdll!RtlAddVectoredExceptionHandler` et `ntdll!RemoveVectoredExceptionHandler`, nous vous renvoyons à l'article **[7]**.

2.5 Data Execution Prevention

Cette sécurité implantée depuis Microsoft Windows XP SP2 a pour objectif de prévenir l'exécution de code depuis des pages mémoire contenant des données. Il n'est alors plus possible d'exécuter un shellcode injecté dans la pile ou le tas. Cette protection repose sur deux parties.

- Niveau matériel :

  Le mécanisme est désigné par NX ou XD bit selon le type de processeur. Il permet de marquer la mémoire non exécutable et de lever une exception quand du code est exécuté depuis l'une de ces pages.

  L'implémentation matérielle de ce mécanisme nécessite l'activation du mode PAE (*Physical Address Extension*) lorsque le processeur s'exécute en mode 32 bits. Cette limitation n'existe pas lorsque le processeur s'exécute en mode 64 bits.

- Niveau logiciel :

  Le support NX (ou XD) doit être activé dans le BIOS de la machine. En effet, il est possible de désactiver cette fonctionnalité au travers d'un registre MSR du processeur.

Il existe également un « Software DEP » dans la littérature Microsoft, mais cette appellation est trompeuse : il s'agit simplement du mécanisme SafeSEH décrit précédemment. De plus, le système propose différentes options pour choisir quels binaires doivent être affectés par cette sécurité :

- `OptIn` - Appliqué aux processus système et à une liste de processus choisis par l'utilisateur seulement.
- `OptOut` - Appliqué à tout le système sauf à une liste de processus choisis par l'utilisateur.
- `AlwaysOn` - Appliqué sur tous les processus.
- `AlwaysOff` - Non appliqué.

Lors de la compilation, l'option `/NXCOMPAT` permet de spécifier si le binaire supporte le DEP. Lors de l'exécution du programme, cette protection peut être désactivée par le biais d'un appel à la fonction `SetProcessDEPPolicy`, on parle alors de DEP non permanent. Afin d'activer le DEP de façon permanente, le programme doit faire appel à la fonction `SetProcessDEPPolicy` avec comme paramètre le drapeau `PROCESS_DEP_ENABLE`. Il ne sera alors plus possible de désactiver la protection car la fonction ne peut être appelée qu'une fois par processus.

Il faut savoir qu'une application sécurisée par le DEP, lancée sur du matériel non compatible avec cette protection, permettra quand même à un attaquant d'exécuter du code sur des pages mémoire non exécutables.

Dans le cas contraire, il est possible de rediriger le flux d'exécution vers la section de code d'un module afin de chaîner des appels de fonctions comme `VirtualAlloc`, `VirtualProtect`, `HeapCreate`, `WriteProcessMemory`, `CreateFileMapping` et ainsi de passer au travers de la protection (exploitation de type *Return-oriented Programming*) **[8]**.

2.6 Address Space Layout Randomisation

Cette protection est intégrée à Microsoft Windows Vista et supérieur.

Elle a pour effet de placer de façon aléatoire les zones mémoire dans l'espace d'adressage. Ainsi, chaque processus affecté par l'ASLR aura son image, sa pile, son tas, TEB et PEB positionnés à des adresses non prédictibles lors de son chargement. Windows Seven génère de nouvelles adresses à l'exécution de chaque programme apportant une amélioration par rapport à Vista, qui ne les génère qu'au redémarrage du système. Ceci dans le but de limiter la fiabilité de l'exploitation d'une faille.

Il s'agit d'une protection à l'échelle du système, son paramétrage se fait par l'intermédiaire d'une clé dans la base de registre `HKLM/SYSTEM/CurrentControlSet/Control/Session Manager/Memory Management/MoveImages` qui, par défaut, n'existe pas. Elle peut prendre 3 valeurs, qui sont :

- 0 ASLR désactivé pour le système entier ;
- 1 ASLR activé pour tous les processus du système, même ceux qui ne sont pas compatibles avec cette protection.
- Pour toute autre valeur ou dans le cas d'une clé absente, l'ASLR ne sera appliqué qu'aux exécutables ou modules ayant la valeur `IMAGE_DLLCHARACTERISTICS_DYNAMIC_BASE` dans leur en-tête PE (`IMAGE_OPTIONAL_HEADER.DllCharacteristics`). Pour qu'un exécutable soit compatible avec cette protection, il est nécessaire de spécifier l'option `/dynamicbase` lors de sa compilation.

Il existe plusieurs méthodes pour contourner cette protection :

- La réécriture partielle de l'adresse de retour de la fonction vulnérable, afin de rediriger le flux d'exécution vers des instructions proches de celle-ci. En effet, l'*endianness* du processeur Intel fait en sorte que les bits de poids faible de l'adresse se trouvent écrasés en premier.
- Le chargement d'un module tiers non affecté par l'ASLR. Il est possible d'utiliser sa section de code afin d'effectuer un enchaînement de retours (*Return oriented programming*).
- L'utilisation d'une vulnérabilité permettant la lecture de parties de la mémoire. Il est alors possible, dans certains cas, de connaître l'emplacement d'un module ASLR.
- L'allocation d'un grand nombre de blocs mémoire exécutables contigus (*heap-spraying*) remplis d'instructions ne faisant rien (*nopsled*) et suivies d'un *shellcode*. Il est ensuite possible de rediriger le flux d'exécution vers le milieu du tas. Celui-ci devenant immense en mémoire, il y a donc plus de chance de rediriger le flux vers lui plutôt que dans une zone que nous ne contrôlons pas.
- L'utilisation de la structure `ntdll!_KUSER_SHARED_DATA`, qui est toujours au même endroit en mémoire (0x7ffe0300) et qui contient un pointeur vers `ntdll!KiFastSystemCall`. Cette structure permet, dans le cas où le registre EAX et la pile sont contrôlés, d'exécuter des appels système.

2.7 x64

La version 64 bits de Microsoft Windows Seven apporte des améliorations en termes de sécurité par rapport à la version 32 bits.

Voici une liste des principales différences complexifiant l'exploitation d'une vulnérabilité :

- Les adresses ne sont plus sur 32 bits (4 octets) mais sur 64 bits (8 octets) avec toujours au moins 2 octets à zéro (0x00). Il devient alors difficile d'exploiter toute vulnérabilité basée sur une corruption mémoire impliquant des fonctions comme `strcpy()` ou `wcscpy()`. Ces fonctions s'arrêtant au premier caractère NULL (0x00) (ou 2 pour les versions Unicode), il est donc impossible de copier des adresses 64 bits dans le but de sauter dans la section de code exécutable d'un module (Return oriented programming). Le GS cookie passe lui aussi a 8 octets, comprenant 2 octets NULL.
- Le mécanisme DEP est activé par défaut sur tous les processus 64 bits.
- La convention d'appel des API a elle aussi changé : sur 32 bits, tous les paramètres d'une fonction étaient copiés sur la pile, alors qu'en 64 bits, les 4 premiers paramètres doivent être placés dans les registres RCX, RDX, R8, R9, puis les suivants sur la pile. Les attaques de type ROP restent possibles en utilisant des « gadgets » de type [pop rcx + ret].
- L'adresse fixe 0x7ffe0300 ne contient plus de pointeur sur `ntdll!KiFastSystemCall`. Les appels système sont réalisés directement dans le module `ntdll.dll` à l'aide de l'instruction assembleur `sysenter`.
- Une erreur d'intégrité dans le tas provoque la fin immédiate du processus.
- Le mécanisme de gestion des exceptions est relativement différent (*stack unwinding*).

En ce qui concerne la protection ASLR, elle reste similaire, et bien que Windows dispose d'un adressage plus grand, le nombre de bits aléatoires n'est pas plus élevé.

À noter qu'un processus 32 bits tournant sous Windows Seven 64 bits peut être exploité comme s'il était sur un système 32 bits (exploit, shellcode) **[9]**.

3 Renforcer la sécurité des applications

3.1 EMET

Enhanced Mitigation Experience Toolkit (en abrégé EMET) est un outil créé par Microsoft pour mettre en échec les différentes techniques d'exploitation, et donc renforcer la sécurité du système. EMET en est aujourd'hui à sa version 2.0, celle-ci téléchargeable sur le site de Microsoft **[10]**.

Contrairement aux protections évoquées dans la première partie, celles-ci vont être mises en place par EMET lui-même.

EMET nous permet de contrôler l'activation des protections telles que le DEP, le SEHOP ou l'ASLR. Mais il est aussi possible de configurer de façon indépendante les paramètres de sécurité appliqués à un processus précis. L'avantage d'EMET réside dans sa capacité à mettre en place des sécurités présentes au sein des systèmes récents (SEHOP/ASLR) sur des systèmes anciens comme Microsoft Windows XP (seulement à partir du SP3 cependant).

L'outil propose donc six protections différentes, que nous allons aborder une à une afin de faire une brève présentation.

1) SEHOP - voir la partie précédente où nous avons abordé cette protection.

2) DEP - bien que ce mécanisme de protection soit disponible depuis Windows XP, EMET permet d'activer la protection même sur des programmes qui n'ont pas été compilés avec l'option de compatibilité `/NXCOMPAT`.

3) Anti-HeapSpraying - De nos jours, de nombreuses exploitations de vulnérabilités utilisent la technique de l'*heap-spraying* pour être plus fiables. Dans certaines conditions, il se peut qu'une vulnérabilité identifiée nous amène à un saut/appel sur une adresse mémoire invalide. Dans ce cas, il est nécessaire d'allouer la page mémoire contenant l'adresse afin de la rendre valide en remplissant le tas. EMET va « pré-allouer » les adresses mémoire souvent utilisées par les attaquants pour exécuter leur payload. De ce fait, l'espace mémoire n'est plus contrôlé par l'attaquant et est donc rendu inutilisable.

4) Anti-allocation de la page 0 - Permet d'éviter les exploitations de déréférencements de pointeur NULL ; une application ne pourra plus mapper la page 0 pour qu'un code noyau puisse brancher sur cette adresse et escalader ses privilèges.

5) ASLR - Tout comme pour le DEP, ici EMET permet d'activer une pseudo « randomisation » des adresses mémoire, même si le binaire n'a pas été compilé avec l'option de compilation `/DYNAMICBASE`.

6) *Export Address Table Filtering* (EATF) - Cette fonctionnalité bloque les accès de lecture sur la table d'exportation des modules `kernel32.dll` et `ntdll.dll`. En effet, les shellcodes génériques vont parcourir cette table afin de retrouver l'adresse des fonctions qu'ils comptent appeler tout au long de leur exécution. C'est là que le dispositif EATF intervient, en détectant l'origine de l'instruction qui tente de lire son contenu. Si EMET juge la provenance malicieuse, elle va mettre fin au processus.

Afin de parvenir à mettre en place ces protections sur les différents processus que nous avons sélectionnés, EMET va injecter dans chaque processus le module `EMET.dll` (ou `EMET64.dll` dans le cas d'un processus 64 bits). Pour cela, la version 2 d'EMET utilise le système de compatibilité de Microsoft Windows. En effet, depuis Windows XP, il est possible de lancer un programme en activant la compatibilité avec les versions des OS inférieurs. Le système va charger un module dans le contexte du processus en question afin de réaliser des *hooks* sur les fonctions qui n'existaient pas à l'époque de l'OS, par exemple. Sous Microsoft Windows 7, c'est à la bibliothèque `apphelp.dll` de faire en sorte que l'environnement soit bien préparé en installant des hooks (en modifiant l'*Import Address Table* du processus) ou en chargeant des modules. EMET va donc manipuler la base de registre et plus précisément `HKLM/SOFTWARE/Microsoft/Windows NT/CurrentVersion/AppCompatFlags/Custom/`.

3.2 Sandbox

Depuis Microsoft Windows Vista, une nouvelle sécurité est apparue : le mode protégé. Applicable à Internet Explorer 7 et supérieur, Chrome ou encore Adobe Reader, il permet d'exécuter ces applications avec des droits restreints. Pour comprendre son fonctionnement, il faut d'abord appréhender le concept de *Mandatory Integrity Control* (MIC) **[11]**.

Chaque objet système dit « sécurisable » (fichier, processus, etc.) possède une liste de contrôle d'accès (*Discretionary Access Control List* ou DACL). Dans cette liste va être ajoutée une entrée correspondant à l'un des quatre IL par défaut, chacun d'eux ayant un identifiant précis (Bas : « S-1-16-4096 », Moyen : « S-1-16-8192 », Haut : « S-1-16-12288 » et Système : « S-1-16-16384 »). Par défaut, un processus utilisateur est lancé avec un IL moyen, tout processus hérite par défaut de l'IL de son parent. Le MIC est un contrôle d'accès géré par le système complétant les groupes et utilisateurs.

Lors d'un contrôle d'accès sur un objet système, l'IL du demandeur est vérifié : s'il est supérieur ou égal à l'IL de l'objet, il aura les droits d'écriture et de destruction sur l'objet sous réserve d'être ensuite autorisé par la DACL. Dans le cas du mode protégé appliqué à Internet Explorer, toute interaction avec Internet sera effectuée par défaut avec un IL bas (*low integrity*). Internet Explorer s'exécutant avec un IL moyen, la communication entre un onglet ayant un IL bas se fait par « IPC » : grâce aux *Local Process Communication* (LPC).

Dans le cas d'une action demandant un IL supérieur depuis un onglet provenant de la zone internet, IE va d'abord l'exécuter dans un autre processus appelé « broker » et prendre une décision sur le résultat en fonction de la valeur de la clé suivante :

HKEY_LOCAL_MACHINE/SOFTWARE/Microsoft/Internet Explorer/Low Rights/ElevationPolicy

3 - L'action est lancée en IL Moyen.

2 - L'action est lancée en IL Moyen si l'utilisateur l'autorise.

1 - L'action est lancée en IL Bas.

0 - L'action ne sera pas lancée.

Il existe cependant plusieurs contournements **[12]** :

- Dans le cas où l'attaquant a la possibilité d'exécuter du code en IL bas, celui-ci a accès en lecture aux cookies et aux fichiers liés aux zones non sécurisées (Internet), mais aussi au presse-papiers récupérant ainsi les opérations de copier/coller. Il peut tenter de récupérer ou dupliquer des « handles » pour interagir avec des objets d'IL supérieurs et ainsi élever ses privilèges. S'il partage des objets avec un processus ayant un IL plus élevé, il peut en profiter pour tenter de corrompre ses objets (il faut savoir que les sockets ne sont pas sujets au MIC).
- La technique de « Squatting attack » : l'attaquant crée un objet avec un nom particulier, utilisé par une autre application qui ne vérifie pas forcement la provenance de celui-ci.
- Le « spoofing » de zones : en se faisant passer pour un site de la zone de confiance par le biais d'une XSS réflexive, persistante, ou en ayant la même adresse (*DNS rebinding*) qu'un site local. Il est aussi possible d'ouvrir un port et de créer un serveur web local à l'aide de notre payload. En appelant la fonction **IELaunchUrl()** avec en paramètre « http://localhost/exploit.html », nous pouvons relancer notre exploit qui sera soumis aux paramètres de la « Zone locale » (*Protected mode* à *Off* par défaut) si le domaine « localhost » a été validé comme une zone de confiance.
- Enfin, le noyau du système d'exploitation représente une surface d'attaque non négligeable et toujours exposée. Par exemple, le rendu des polices de caractères s'effectue en mode noyau.

Conclusion

Chacune des protections est complémentaire : le GS Cookie protège la pile, le SafeSEH protège la liste chaînée SEH utilisée pour contourner le GS Cookie, le SEHOP complique les techniques de contournement du SafeSEH, le DEP empêche l'exécution de code dans des zones de données et l'ASLR empêche l'utilisation de techniques comme le ROP. Il suffit d'une application, d'un module, d'un maillon faible pour passer au travers de toute la chaîne de sécurité. Un seul module non ASLR chargé en mémoire et c'est toute la protection ASLR qui est mise à mal.

Nous avons vu que le logiciel EMET peut être un outil efficace pour renforcer la sécurité des applications sensibles, telles que son navigateur web, son lecteur vidéo, ou encore son lecteur PDF.

Récemment s'est déroulé le challenge *PWN2OWN* organisé par *CanSecWest* 2011 à Vancouver. Durant celui-ci, l'expert informatique irlandais Stephen Fewer a réussi à passer au travers de l'ensemble des sécurités mises en place par Windows 7 x64 et par Internet Explorer 8 : ASLR, DEP et le Protected Mode.

Dans le prochain numéro, nous verrons comment écrire un exploit pour Internet Explorer 8 sous Seven en utilisant les dernières techniques de contournement du DEP et de l'ASLR (*teasing* !). ■

REMERCIEMENTS

Nous souhaitons remercier Benjamin Caillat, Ivanlef0u, x86, Nicolas Ruff, Eric Filiol, ainsi que l'équipe de XMCO pour leur relecture !

RÉFÉRENCES

[1] http://msdn.microsoft.com/fr-fr/library/8dbf701c.aspx

[2] http://ivanlef0u.fr/repo/expl0it/bh08sotirovdowd.pdf

[3] http://blogs.technet.com/b/srd/archive/2009/02/02/preventing-the-exploitation-of-seh-overwrites-with-sehop.aspx&yh

[4] http://go.microsoft.com/?linkid=9646972

[5] http://www.sysdream.com/articles/sehop_en.pdf

[6] http://www.madpowah.org/textes/heap-windows-exploitation.html

[7] http://www.codeproject.com/KB/exception/VEH.aspx

[8] http://www.uninformed.org/?v=2&a=4

[9] http://www.immunitysec.com/downloads/win64_confidence2010.pdf

[10] http://go.microsoft.com/fwlink/?LinkID=200220&fatfrancisco=37&clcid=0x409

[11] http://blogs.technet.com/b/steriley/archive/2006/07/21/442870.aspx

[12] http://www.verizonbusiness.com/resources/whitepapers/wp_escapingmicrosoftprotectedmodeinternetexplorer_en_xg.pdf

CODE INTEGRITY

Damien Aumaitre – SOGETI/ESEC – damien@security-labs.org, Ivanlef0u – SOGETI/ESEC – ivanlef0u@security-labs.org

mots-clés : WINDOWS / AUTHENTICODE / KERNEL / CERTIFICATS / SIGNATURE

Depuis Windows Vista, Microsoft a ajouté divers mécanismes de sécurité. Nous pouvons citer, par exemple, BitLocker, PatchGuard, les DRM, etc. Pour cet article, nous avons choisi de nous concentrer sur les nouveaux usages de la signature numérique. Par exemple, sous Windows Seven 64 bits, il est obligatoire que tous les drivers soient signés. Après avoir introduit les notions nécessaires à la compréhension de l'article, nous montrerons comment se servir des outils mis à notre disposition par Microsoft. Nous détaillerons ensuite le fonctionnement interne des dispositifs de signature.

1 Authenticode

Windows supporte depuis longtemps un nombre important de technologies de signature. Les plus utilisées étant Authenticode et le StrongName signing des applications .NET.

Authenticode permet de signer un binaire et d'assurer que celui-ci n'a pas été modifié depuis sa signature. Cette technologie combine une signature numérique avec plusieurs certificats afin d'assurer que l'application provient bien de la personne qui l'a signée.

Les signatures numériques sont utilisées dans de nombreux domaines de Windows :

- *Windows Setup* : Tous les fichiers installés sont signés par Microsoft.
- *Windows Update* : Les mises à jour logicielles installées par Windows Update possèdent un catalogue signé par Microsoft.
- WSUS (*Windows Server Update Services*) : Ce service à destination des entreprises utilise aussi des signatures numériques.
- *Windows Installer* (.msi) : si les signatures sont invalides, un avertissement est émis avant l'installation.
- SRP (*Software Restriction Policies*) : les administrateurs peuvent autoriser uniquement l'exécution de binaires signés.

Depuis Windows Vista, la signature de code a de nouveaux usages :

- UAC (*User Account Control*) est une fonctionnalité permettant d'élever de manière ponctuelle les droits d'une application. L'affichage et l'autorisation sont conditionnées par la présence de binaires signés.
- Les DRM (*Digital Rights Management*) et le PMP (*Protected Media Path*) nécessitent que tous les composants du noyau et certains composants utilisateurs (fournisseurs cryptographiques, par exemple) soient signés.
- Tous les installeurs téléchargés à partir d'Internet Explorer ne sont pas exécutés par défaut s'ils ne sont pas signés ou si la signature est invalide.
- Tous les composants kernel sur les versions 64 bits doivent être signés.

Le noyau assure que seuls des composants signés sont chargés en mémoire. Les versions 64 bits de Windows (à partir de Windows Vista) interdisent l'accès au kernel aux *drivers* non signés.

Authenticode est un format de signature utilisé pour déterminer l'origine et l'intégrité de binaires. Il utilise une signature au format PKCS #7 (*Public-Key Cryptography Standards*). Un certificat X.509 est lié à l'identité de son propriétaire via une signature par un tiers de confiance, une CA (*Certificate Authority*).

L'usage principal de la signature Authenticode est la signature de fichiers PE. La signature assure que le fichier provient bien d'une personne spécifique et qu'il n'a pas été modifié depuis sa signature. Par contre, il n'y a aucune information sur la qualité et/ou les intentions des concepteurs.

Authenticode a deux modes de fonctionnement :

- La signature peut être incluse dans une partie non exécutable du binaire.
- La signature est incluse dans un catalogue (.cat). Dans ce cas, c'est le catalogue qui contient la signature et les *hashs* des différents binaires. Le catalogue permet de détacher la signature et facilite la signature de plusieurs fichiers, y compris ceux dont le format n'est pas prévu pour contenir une signature (un fichier texte, par exemple).

La signature est constituée d'une structure PKCS #7 SignedData contenant :

- la valeur du hash du fichier PE ;
- la signature créée par la clé privée de l'émetteur ;
- un certificat liant l'émetteur à une entité racine.

La structure SignedData peut aussi contenir :

- une description de l'émetteur ;
- l'URL du logiciel ;
- un *timestamp* généré par une TSA (*TimeStamping Authority*), celui-ci assure que la signature existe à partir d'une date bien précise. De plus, le timestamp étend la durée de vie de la signature quand le certificat utilisé pour la signature expire ou est révoqué.

Dans un fichier PE, l'emplacement de la signature est indiqué par l'entrée *Certificate Table* dans les *Data Directories* de l'*Optional Header*.

Cette table contient une structure **WIN_CERTIFICATE**.

```
typedef struct _WIN_CERTIFICATE
{
    DWORD       dwLength;
    WORD        wRevision;
    WORD        wCertificateType;
    BYTE        bCertificate[ANYSIZE_ARRAY];
} WIN_CERTIFICATE, *LPWIN_CERTIFICATE;
```

- **dwLength** contient la longueur du tableau **bCertificate**.
- **wRevision** contient la version de la structure : la valeur actuelle est 0x200 (**WIN_CERT_REVISION_2_0**).
- **wCertificateType** contient le type de certificat utilisé : pour Authenticode, il s'agit de 0x2 (**WIN_CERT_TYPE_PKCS_SIGNED_DATA**).

Le lecteur intéressé pourra utiliser le script **disitool.py** de Didier Stevens **[disitool]** pour extraire la signature d'un binaire signé (à partir du champ **bCertificate**).

Le contenu du champ **bCertificate** peut être parsé par OpenSSL avec la commande suivante :

```
openssl pkcs7 -inform DER -in sig.bin -text
```

Il est aussi possible de décoder les structures ASN.1 :

```
openssl asn1parse -inform DER -in sig.bin -dump
```

Microsoft utilise des structures ASN.1 spécifiques pour le champ **ContentInfo** de la structure **SignedData** (celle-ci est décrite dans la RFC 2315). Le lecteur curieux trouvera leur description dans la spécification Authenticode fournie par Microsoft **[authenticode]**.

Les vérifications effectuées sur le certificat sont les suivantes :

- La chaîne de certification doit être valide et le certificat racine doit être présent dans le magasin de certificat *Trusted Root Certification Authorities*.
- Le certificat doit contenir un EKU (*Extended Key Usage*) valant **szOID_PKIX_KP_CODE_SIGNING** (1.3.6.1.5.5.7.3.3).
- Aucun des certificats de la chaîne de certification ne doit être dans le magasin *Untrusted Certificates*.
- La date de validité du certificat doit être valide ou la signature est signée par une TSA.
- La liste de révocation doit être valide.

Une fois que l'intégrité et l'identité de la signature ont été contrôlées, il reste à vérifier la validité du hash contenu dans la signature. Celui-ci est contenu dans le champ **Digest** de la structure **SpcIndirectDataContent** de la structure **DigestInfo**.

Le hash est calculé en excluant le champ **Checksum** et la Certificate Table. Comme nous allons le voir dans la suite de l'article, ces informations sont tirées des spécifications fournies par Microsoft et il se trouve que l'algorithme réellement implémenté est légèrement différent (avec des différences suivant que la vérification soit faite en noyau ou en *userland*).

2 Outils

Microsoft fournit un ensemble d'utilitaires à travers le SDK (*Software Development Kit*), le WDK (*Windows Driver Kit*) et Visual Studio afin de manipuler les certificats. Un certificat pour signer du code ne diffère pas des autres types de certificats (comme ceux utilisés par SSL/TLS), seul son rôle change. Ainsi, un développeur peut, dans un premier temps, créer des certificats auto-signés à des fins de tests. Ensuite, il effectue une demande auprès d'une CA reconnue afin d'intégrer des certificats valides lors du déploiement de ses applications.

Nous allons décrire quels sont les outils sous Windows **[tools]** servant à manipuler des certificats lors des phases de tests, puis nous verrons comment obtenir un certificat auprès d'une CA.

2.1 Création d'un certificat auto-signé

Le premier outil, **makecert.exe**, permet de créer un certificat X.509 qui sera stocké sous format DER (*Distinguished Encoding Rules*) binaire dans un fichier

suffixé .cer. Dans notre cas, l'option qui spécifie la création d'un certificat auto-signé est **-r**.

```
C:\cert>makecert /?
Usage: MakeCert [ basic|extended options] [outputCertificateFile]
Basic Options
 -sk  <keyName>      Subject's key container name; To be created if not present
 -pe                 Mark generated private key as exportable
 -ss  <store>        Subject's certificate store name that stores the output
                     certificate
 -sr  <location>     Subject's certificate store location.
                       <CurrentUser|LocalMachine>.  Default to 'CurrentUser'
 -#   <number>       Serial Number from 1 to 2^31-1.  Default to be unique
 -$   <authority>    The signing authority of the certificate
                       <individual|commercial>
 -n   <X509name>     Certificate subject X500 name (eg: CN=Fred Dews)
 -?                  Return a list of basic options
 -!                  Return a list of extended options
C:\cert>
C:\cert>makecert -n "CN=Test" -r -sv Test.pvk Test.cer
Succeeded
```

Lors du lancement, une pop-up nous invite à définir un mot de passe pour la clé privée (non obligatoire). Après création, nous retrouvons le certificat **Test.cer** avec, à côté, le fichier contenant la clé privée **Test.pvk**.

Nous pouvons dorénavant signer un fichier à l'aide de ce certificat. L'outil **signtool.exe** propose un wizard pour simplifier la procédure. Nous spécifions le fichier de certificat (**test.crt** ici), le fichier avec la clé privée associée et le type de hash à utiliser.

```
C:\cert>signtool /?
Usage: signtool <command> [options]

       Valid commands:
              sign       -- Sign files using an embedded signature.
              signwizard -- Launch the signing wizard.
              timestamp  -- Timestamp previously-signed files.
              verify     -- Verify embedded or catalog signatures.
              catdb      -- Modify a catalog database.

For help on a specific command, enter "signtool <command> /?"
C:\cert>
C:\cert>rem Avec le wizard
C:\cert>signtool signwizard test.exe
Successfully completed signing wizard: <>
```

Le binaire **test.exe** possède maintenant des informations de signature :

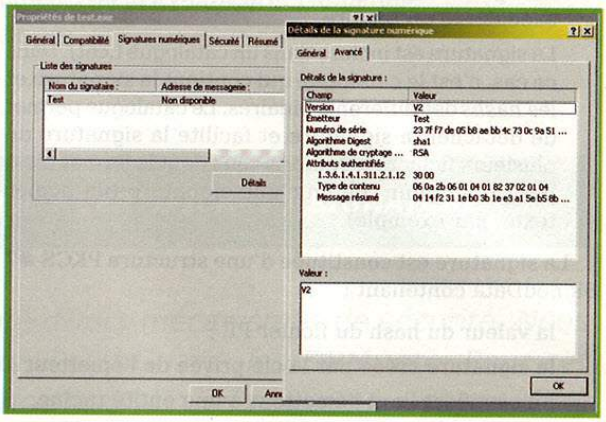


Figure 1 : Fichier test.exe signé avec un certificat embarqué

Cependant, il n'est pas encore reconnu valide par le système car la CA (*Certificate Authority*), représentée par le certificat **Test.crt**, n'est pas installée dans ses dépôts. Il existe 2 dépôts de CA de confiance, celui de l'utilisateur et celui du système. Voici les CA racines de confiance vues avec **certmgr.exe** (Figure 2).

La preuve en demandant une vérification de signature :

```
C:\cert>signtool verify test.exe
SignTool Error: A certificate chain processed, but terminated in a root
        certificate which is not trusted by the trust provider.
SignTool Error: File not valid: test.exe

Number of errors: 1
C:\cert>
```

Justement, l'installation dans le dépôt de certificats de l'utilisateur s'effectue de la manière suivante :

```
C:\cert>certmgr -add -c -s my Test.cer
CertMgr Succeeded
```

Après installation dans le dépôt, notre fichier **test.exe** est reconnu car la CA (sa propre CA en fait, vu qu'il est auto-signé) :

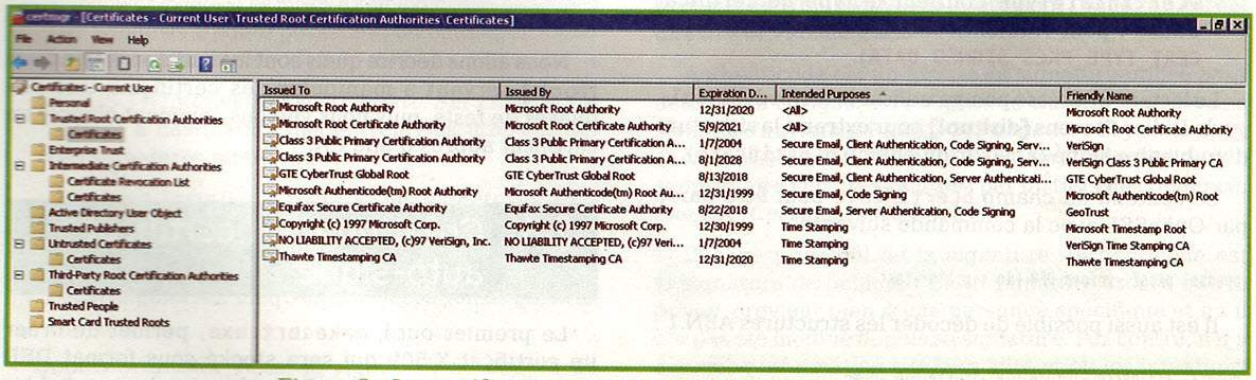


Figure 2 : Le certificate manager et les CA racines de confiance

```
C:\cert>signtool verify test.exe
Successfully verified: test.exe
C:\cert>
C:\cert>signtool verify /v /pa test.exe

Verifying: test.exe
SHA1 hash of file: B44A432E83C658E15808F3E1E00402CCCF954D27
Signing Certificate Chain:
    Issued to: Test
    Issued by: Test
    Expires:   01/01/2040 00:59:59
    SHA1 hash: 85D6241ACE2C7FA754C844A21A2BBEFCFFA74E48

File is not timestamped.
Successfully verified: test.exe

Number of files successfully Verified: 1
Number of warnings: 0
Number of errors: 0
```

Les outils **chktrust.exe** (du SDK) ou **SigCheck** de Sysinternals permettent aussi de vérifier à la main les signatures de fichiers.

Le tableau suivant indique dans quels *frameworks* se trouvent ces différents outils.

Sources for Code Signing and Related Tools			
Tool	**WDK**	**Platform SDK**	**Additional sources**
MakeCert	WDK	SDK	.NET SDK
CertMgr	WDK	SDK	.NET SDK
SignTool	WDK	SDK	
Capicom.dll v.2.1.0.1	WDK	SDK	Download Center
MakeCat	WDK	SDK	
Signability	WDK		
Inf2Cat			Winqual submission tools
PVK2PFX	WDK	SDK	
SelfSign_example	WDK		

Figure 3 : Outils pour la signature de code et où les obtenir

2.2 Obtenir un Software Publisher's Certificate

Un SPC (*Software Publisher's Certificate*) est un certificat fourni par un organisme (en général une CA commerciale). Il s'agit d'un conteneur PKCS#7 d'un ou plusieurs certificats X.509 signés par cette CA. En ce qui concerne la signature de code, Verisign propose un certificat Authenticode d'une durée d'un an pour 500 dollars. Les certificats de type Verisign Class-3 sont fournis à travers 2 fichiers, un .spc qui contient le certificat avec la clé publique et un .pvk qui contient la clé privée. On peut packager les 2 dans un .pfx, un document de type PKCS#12 (*Personal Information Exchange Syntax Standard*) qui sert à protéger avec un mot de passe le certificat et sa clé privée.

Il est possible de créer un .spc de test à l'aide de l'outil **cert2.spc** :

```
C:\cert>cert2spc Test.cer Test.spc
Succeeded
C:\cert>
```

Ce SPC contient des certificats auto-signés, seule une SPC est capable d'en fournir avec des certificats valides. Pour l'utiliser, il faut l'importer dans un des dépôts de certificats. La commande **certmgr.exe /add** distingue le dépôt de l'utilisateur (**currentUser**) ou celui du système (**localMachine**).

Cette méthode peut être utilisée pour signer des fichiers .exe (exécutable), .cab (archive d'exécutables), .dll (bibliothèque dynamique), .ocx (ActiveX) .msi (installers) et .sys (driver).

En ce qui concerne les drivers **[KMCS1] [KMCS2] [KMCS3]** et la politique de *Kernel Mode Code Signing* (KMCS), un éditeur peut aussi faire une demande de SPC et proposer des drivers signés. Il existe cependant un niveau « supérieur » de signature qui prouve à la fois la qualité du driver et sa validité. Le WHQL (*Windows Hardware Quality Labs*) permet à une compagnie de déployer des drivers directement via Windows Update, sous forme de .cab, mais cela requiert la validation d'une série de tests appelée « Windows Logo Program ». Bien entendu, un driver WHQL est conseillé par rapport à un driver KMCS.

Les CA « Microsoft Test Root Authority » permettent aux sociétés de vérifier leurs drivers avant sortie auprès du WHQL en bootant avec l'option ***Enable Test Signing***. Il existe aussi un service distant appelé le *Windows Quality Online Services* (Winqual) pour soumettre ses drivers aux tests du WHQL.

Signing options	Functionality verified to meet logo requirements	Identity verified	Intended use
Windows Logo Program	Yes	Yes	Release
KMCS by using an SPC	No	Yes	Release
WHQL Test Signature program	No	Yes	Testing
KMCS test signing	No	No	Testing

Figure 4 : Le KMCS et le programme WQHL

2.3 Les catalogues

Parfois, il est plus simple d'avoir un document qui certifie un ensemble de fichiers plutôt que d'embarquer un certificat dans chacun d'entre eux. C'est le rôle des catalogues, ces fichiers possèdent l'extension .cat et contiennent un dictionnaire qui associe à un nom de fichier son hash. Les catalogues sont sous la forme d'une structure SignedData PKCS#7 encodée en ASN.1.

On retrouve les catalogues installés sur le système dans **%systemroot%\system32\CatRoot\{F750E6C3-38EE-11D1-85E5-00C04FC295EE}**, où le dernier nom correspond au **GUID DRIVER_ACTION_VERIFY** qu'on retrouve dans **SoftPub.h**.

Les outils de vérification de signature cités plus haut ne proposent pas forcément une vérification de la signature via les catalogues. De plus, un catalogue peut aussi embarquer les hashs de fichiers non exécutables. Par défaut, Microsoft signe la plupart des hashs des fichiers présents sur le système lors de l'installation.

Par exemple, avec sigcheck :

```
C:\cert>sigcheck -q -h "c:\windows\NOTEPAD.EXE"
c:\windows\NOTEPAD.EXE:
        Verified:       Signed
        Signing date:   04:41 14/04/2008
        Publisher:      Microsoft Corporation
        Description:    Bloc-notes
        Product:        Système d'exploitation Microsoft Windows
        Version:        5.1.2600.5512
        File version:   5.1.2600.5512 (xpsp.080413-2105)
        MD5:    2dcc5c800f51d487178814ca9eada181
        SHA1:   69a0d228cd6f414b34f115fad2cca0e55e95e316
        SHA256:
10697adde4ba05d11191c9661e4722c2555a2b4225c5056ccda31242a18b6bb7

C:\cert>signtool verify /pa /v "c:\windows\NOTEPAD.EXE"

Verifying: c:\windows\NOTEPAD.EXE
SHA1 hash of file: 43D6AD6A9CC7F628A1485BA00FDD2157E3008513
SignTool Error: No signature found.
SignTool Error: File not valid: c:\windows\NOTEPAD.EXE

Number of files successfully Verified: 0
Number of warnings: 0
Number of errors: 1
```

Cependant, ces outils gèrent mal les fichiers non exécutables.

```
c:\windows\Mur de Santa Fe.bmp:
        Verified:       Signed
        Signing date:   04:41 14/04/2008
        Publisher:      n/a
        Description:    n/a
        Product:        n/a
        Version:        n/a
        File version:   n/a
        MD5:    eb3bfc14e41fbaa41b4fd4489aa82d39
        SHA1:   26e2e586634a4a28d5036ebde27760e4b1c3ed86
        SHA256: 6307d54f53affe07880cff3f478318893ec7b75276e239d3eb8875c3d8344655

C:\cert>
C:\cert>signtool verify /pa /v "c:\windows\Mur de Santa Fe.bmp"

Verifying: c:\windows\Mur de Santa Fe.bmp
SHA1 hash of file: 26E2E586634A4A28D5036EBDE27760E4B1C3ED86
SignTool Error: This file format cannot be verified because it is not recognized.
SignTool Error: File not valid: c:\windows\Mur de Santa Fe.bmp

Number of files successfully Verified: 0
Number of warnings: 0
Number of errors: 1

C:\cert>
```

Par exemple, sous XP X86, le hash du fichier `%systemroot%\Mur de Santa Fe.bmp` se situe dans le catalogue `%systemroot%\system32\CatRoot\{F750E6C3-38EE-11D1-85E5-00C04FC295EE}\NT5.CAT`. Les principaux catalogues sont `NT5.cat` et `SP3.cat`.

Sous 7 x86, les catalogues importants sont `ntexec.cat`, `nt5.cat` et `Microsoft-Windows-Foundation-Package~3Microsoft-Windows-Foundation-Package~31bf3856ad364e35~x86~~6.1.7601.17514.cat`.

2.4 Certificats : registre et fichiers

Le registre Windows contient les différentes CA racines présentes sur le système :

- `HKEY_CURRENT_USER\Software\Policies\Microsoft\SystemCertificates` ;
- `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\EnterpriseCertificates` ;
- `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\SystemCertificates` ;
- `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CertSvc`.

Les certificats personnels, quant à eux, sont stockés dans les fichiers utilisateurs : `%USERPROFILE%\Application Data\Microsoft\Systemcertificates\My`.

Les paramètres utilisés par le *cryptographic provider* (`crypt32.dll`) et le *trust provider* (`wintrust.dll`) sont définis dans `HKLM\Software\Microsoft\Cryptography\`.

Enfin, concernant les catalogues, nous avons :

- `%systemroot%\system32\CatRoot` : Contient les répertoires avec les catalogues racines, le principal étant `{F750E6C3-38EE-11D1-85E5-00C04FC295EE}`. Chacun contient un ensemble de .cat signant différents fichiers, voire des pages en mémoire.
- `%systemroot\system32\CatRoot2` : Utilisé par le service `CryptSvc` (implémenté dans `CryptSvc.dll`) pour stocker la base de données JET (*Joint Engine Technology*) du cache de catalogues. En fait, les API passent d'abord par ce service pour trouver un hash dans les catalogues avant de faire une recherche dans les catalogues de CatRoot.

3 Internals

Plusieurs modules sont impliqués dans la vérification des signatures. En userland, on retrouve principalement `wintrust.dll` et `crypt32.dll`, en kernel land, `ci.dll`.

3.1 Wintrust.dll

`Wintrust.dll` est la bibliothèque utilisée par Windows pour vérifier des certificats ou catalogues. Elle implémente aussi la gestion des CTL (*Certificate Trust List*) et CRL (*Certificate Revocation List*).

Ces principales fonctions sont `WinVerifyTrust` et `WinVerifyTrustEx`, qui servent à vérifier la signature d'un fichier. Elles autorisent aussi bien la vérification de

signatures de fichiers exécutables (depuis un catalogue ou un certificat embarqué) que de connexions SSL (certificats).

Les API suivantes permettent de manipuler les catalogues :

- `CryptCATAdminAcquireContext` / `CryptCATAdminReleaseContext` : Permet d'obtenir et de relâcher un *handle* sur les catalogues système.
- `CryptCATAdminCalcHashFromFileHandle` : Depuis un handle sur un fichier, calcule le hash du fichier en fonction de son type. Par exemple, s'il s'agit d'un fichier PE, la fonction calculera son hash en respectant la norme Authenticode.
- `CryptCATAdminEnumCatalogFromHash` : Énumère l'ensemble des catalogues qui contiennent le hash du fichier calculé précédemment. Pour chaque catalogue ayant le hash, on récupère un `HCATINFO` (comprendre un handle sur le catalogue).
- `CryptCATAdminResolveCatalogPath` : Retrouve le fichier du catalogue à partir d'un `HCATINFO`.

Les fonctions de cryptographie sont implémentées dans `crypt32.dll`, les plus courageux peuvent regarder le parseur ASN.1 dans `msasn1.dll` :)

3.2 ci.dll

La DLL `ci.dll` (*Code Integrity*) **[CI1] [CI2] [CI3]** est le composant noyau chargé de la vérification de l'intégrité des modules userland et kernel land qui sont chargés sur le système. Lorsqu'on charge un module, le kernel utilise `nt!MmCreateSection` afin de mapper le binaire en mémoire. C'est à ce moment que le système décide ou non de vérifier la signature du module. Elle est aussi très liée avec `peauth.dll`, le module kernel qui s'occupe en partie des DRM et qui utilise `ci.dll` pour vérifier l'intégrité des *protected processes* **[pprocesses]**.

Le module `ci` exporte les fonctions suivantes :

- `ci!CiCheckSignedFile` ;
- `ci!CiFindPageHashesInCatalog` ;
- `ci!CiFindPageHashesInSignedFile` ;
- `ci!CiFreePolicyInfo` ;
- `ci!CiGetPEInformation` ;
- `ci!CiInitialize` ;
- `ci!CiVerifyHashInCatalog`.

Lors de son initialisation, le kernel fait appel à `nt!SepInitializeCodeIntegrity`, dont le pseudo code est :

```
g_CiEnabled=InitIsWinPEMode()
if g_CiEnabled
        Flags=4|2
        if SepIsOptionPresent("DISABLE_INTEGRITY_CHECKS")
                Flags=0
        end
        if SepIsOptionPresent("TESTSIGNING")==0
                Flags|=8
        end
        CiInitialize(Flags, &g_CiCallbacks)
end
```

La variable globale `nt!g_CiEnabled` indique si la vérification de signature de code est active. Lorsqu'on boot en mode PE (*Preinstallation Environment*), elle n'est pas du tout présente. Cependant, on peut passer des options de boot via l'outil de configuration de démarrage `bcdedit.exe` :

- `DISABLE_INTEGRITY_CHECKS` : « bcdedit.exe /set nointegritychecks on » ;
- `TESTSIGNING` : La vérification de signature des modules est active, mais une chaîne de confiance valide n'est pas requise pour charger le module. Cela est utile pour les développeurs qui veulent signer les modules avec un certificat auto-signé « bcdedit.exe /set testsigning ON ».

Par contre, cette option affecte le comportement des DRM sous Vista et 7 :)

Bien sûr, il est toujours possible d'utiliser d'autres techniques pour désactiver la vérification de signatures :

- Boot en mode *debug* avec un débogueur attaché (« bcdedit.exe /bootdebug ON » ou « bcdedit.exe /debug ON »).
- Utiliser les options de démarrage avancé en appuyant sur F8 au boot et sélectionner ***Disable Driver Signature Enforcement***.

La variable globale `nt!g_CiCallbacks` sert au kernel à stocker 3 pointeurs de fonctions :

- `ci!CiValidateImageHeader` appelé par `nt!MiValidateImageHeader` -> `nt!SeValidateImageHeader`. Permet de vérifier si un module est signé.
- `ci!CiValidateImageData` appelé par `nt!MiValidateImagePages` -> `nt!MiValidateImagePfn` -> `nt!SeValidateImageData`. Permet de vérifier le hash des pages.
- `ci!CiQueryInformation` appelé par `nt!SeCodeIntegrityQueryInformation`. Cette dernière étant accessible via `nt!NtQuerySystemInformation`.

Le composant `ci.dll` est capable de lire des options depuis le registre via la clé `HKLM\System\CurrentControlSet\Control\CI`. La plus intéressante étant `DebugFlags` :

- 0x1 : Lève une exception si un *kernel debugger* est attaché. Le module est ensuite chargé en relançant l'exécution.

- 0x10 : Ignore la présence d'un débogueur et les modules non signés ne sont pas chargés.

Ce module embarque la bibliothèque `MinCrypt` pour effectuer toutes les opérations cryptographiques et vérifications de signatures. C'est l'équivalent de `wintrust.dll` et `crypt32.dll` en kernel.

Enfin, il est possible de connaître l'état de `ci.dll` via le *syscall* `nt!ZwQuerySystemInformation` et l'InformationClass 103. La structure passée en argument contient 2 DWORDs, le premier étant la taille de la structure en bytes (8 donc) et le second va contenir le retour de `ci!CiQueryInformation`. Il existe 3 valeurs de retour possibles :

- 0 : `ci.dll` fonctionne normalement par rapport à la signature de modules.

- 1 : `ci.dll` fonctionne sans `PEAUTH` (état du système non trusted). La désactivation de `PEAUTH` doit être spécifiée dans les `DebugFlags` par le flag 2 (non documenté) et un debugger kernel ne doit pas être présent.

- 2 : Equivalent au `TESTSIGNING` pour `ci.dll` (sans passer par le bcdedit). Les `DebugFlags` ont le flag 8 (non documenté) actif.

On retrouve dans `%systemroot%\system32\CodeIntegrity\Driver.stl` la liste des certificats de révocation pour les drivers.

Conclusion

Dans cet article, nous avons fait un état de l'art du fonctionnement des signatures numériques sous Windows ainsi que de leur impact sur le système.

Microsoft a fait le choix d'utiliser des composants classiques de la cryptographie à clé publique (certificats, standards PKCS). Cependant, la spécification Authenticode possède des structures ASN.1 non documentées. Bien que celle-ci existe depuis une dizaine d'années, très peu d'études publiques se sont penchées sur sa sécurité, que ce soit au niveau de sa conception ou de son implémentation. Nous avons vu que Windows fait usage de cette technologie à de multiples niveaux, il s'agit donc d'un composant critique du système.

Dans un prochain article, nous expliquerons comment le noyau met en place la chaîne de confiance depuis le démarrage. Nous montrerons que ce modèle possède des limitations si des mécanismes extérieurs de confiance comme le TPM ne sont pas utilisés. ■

RÉFÉRENCES

[disitool] http://blog.didierstevens.com/2008/01/11/the-case-of-the-missing-digital-signatures-tab/

[tools] *Tools to Create, View, and Manage Certificates*, http://msdn.microsoft.com/en-us/library/aa388150%28VS.85%29.aspx

[KMCS1] *Kernel-Mode Code Signing Requirements (Windows Vista and Later)*, http://msdn.microsoft.com/en-us/library/ff548239%28VS.85%29.aspx

[KMCS2] *Digital Signatures for Kernel Modules on Systems Running Windows Vista*, http://msdn.microsoft.com/en-us/library/bb530195.aspx

[KMCS3] *Kernel-Mode Code Signing Walkthrough*, http://download.microsoft.com/download/9/c/5/9c5b2167-8017-4bae-9fde-d599bac8184a/KMCS_Walkthrough.doc

[CI1] *A quick insight into the Driver Signature Enforcement*, http://j00ru.vexillium.org/?p=377

[CI2] *Code Integrity (ci.dll) Security Policy. pdf*, http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140sp/140sp890.pdf

[CI3] *Assessment of Windows Vista Kernel-Mode Security*, http://www.symantec.com/avcenter/reference/Windows_Vista_Kernel_Mode_Security.pdf

[pprocesses] *Protected Processes in Windows Vista*, http://msdn.microsoft.com/en-us/windows/hardware/gg463417

[pyasn1] http://pyasn1.sourceforge.net/

[authenticode] http://msdn.microsoft.com/en-us/windows/hardware/gg463180

[tld4] http://www.virusbtn.com/pdf/conference_slides/2010/Johnson-VB2010.pdf

[pkcs] http://en.wikipedia.org/wiki/PKCS

[asn1] http://en.wikipedia.org/wiki/Abstract_Syntax_Notation_One

NUMÉRO 2 **ENCORE + D'ÉLECTRONIQUE, ENCORE + DE HACK !**

DÉCOUVREZ

LE NOUVEAU MAGAZINE DES ÉDITIONS DIAMOND !

OPEN SILICIUM 2

LE MAGAZINE DE L'OPEN SOURCE POUR L'ÉLECTRONIQUE & L'EMBARQUÉ

APPRENEZ À EXPLOITER VOTRE MODULE ARDUINO ET À TIRER LE MEILLEUR DU MICROCONTRÔLEUR AVR D'ATMEL !

L'engouement suscité par les hors-séries de Linux Magazine spécialement consacrés au monde de l'embarqué et de l'électronique nous a naturellement conduits à concevoir un magazine uniquement dédié à l'univers des technologies embarquées et de l'open source : Open Silicium...

www.opensilicium.com

DISPONIBLE CHEZ VOTRE MARCHAND DE JOURNAUX DÈS LE 25 MARS 2011 ET SUR : WWW.ED-DIAMOND.COM

VOUS AVEZ MANQUÉ LE N°1 D'OPEN SILICIUM ? COMMANDEZ-LE SUR : WWW.ED-DIAMOND.COM

ALCASAR, LE PORTAIL CAPTIF QUI A FAIT SES PREUVES

Thierry Martineau – thierrymartineau@yahoo.fr

mots-clés : *PORTAIL CAPTIF / ALCASAR / GESTION DES JOURNAUX LOGS / AUTHENTIFICATION / INTERCEPTION / PREUVE NUMÉRIQUE*

Les portails captifs sont généralement utilisés sur des réseaux ouverts où les ordinateurs personnels viennent se connecter en mode Wi-Fi ou filaire. Le projet Alcasar a pour ambition de mettre en œuvre un dispositif de sécurité répondant aux besoins des utilisateurs, de l'administrateur et des obligations légales de journalisation des activités sur un réseau connecté à Internet. Cet article explique comment Alcasar améliore la sécurité des portails captifs classiques et aborde la notion de preuve numérique.

1 Introduction

1.1 Un portail captif pour capturer quoi ?

Un portail captif est un dispositif authentifiant les utilisateurs avant toute utilisation des ressources sur Internet et après l'acceptation d'une politique de sécurité, d'une charte ou d'un paiement. Déjà évoqué dans *MISC* n°53 **[HOTSPOT]**, certains réseaux d'entreprise s'appuient sur un portail captif pour assurer la gestion des postes invités, filaires ou Wi-Fi ne rentrant pas dans le périmètre habituel des clients Windows maîtrisés et intégrés à un domaine *Active Directory*. Concrètement, dès que le navigateur web est lancé, toute requête vers l'extérieur est automatiquement redirigée (statut HTTP 302) vers une page web d'authentification chiffrée HTTPS. On parle alors d'interception des flux applicatifs. Il reste des flux non traités par ce dispositif, notamment les flux téléphoniques de type 3G ou de type WiMax.

1.2 Il faut tracer !

Les obligations légales en matière de journalisation des événements sont applicables aux portails captifs **[LOG]**. Avec le développement des moyens d'investigation, l'exigence de vérité semble plus prégnante dans le monde du numérique. La collecte de la preuve dans un contexte de cyberdéfense répond cependant à deux principes : le principe de loyauté (les utilisateurs doivent être prévenus) et celui de proportionnalité (le respect des libertés individuelles). Le caractère immatériel, volatil et évolutif des logs d'un firewall sont autant d'obstacles à son élection au rang de preuve. Les juges **[CNEJITA]** s'attachent à distinguer la copie volontaire d'un fichier sur un ordinateur des copies automatiques. La portée des faits ainsi que les preuves ne seront pas les mêmes. C'est pourquoi il sera demandé aux experts de vérifier si les copies des pages d'un site (pédophile, par exemple) se trouvent ou non uniquement dans le cache du navigateur ou dans le dossier « Mes Images » et si une trace se trouve dans les journaux événements du dispositif de protection (portail captif filtrant dans notre cas). L'administrateur système devra s'attacher aux critères d'authenticité (chiffrement des logs), d'intégrité (hash pour éviter la perte ou l'altération des logs), de traçabilité (date des fichiers logs cohérente avec la date de gravure sur DVD) et de conservation (1 an - durée de vie limitée du support DVD contenant les logs).

1.3 Toujours les mêmes limites...

La sécurité des portails captifs couplés à des *hotspots* Wi-Fi est intrinsèquement contournable **[HOTSPOT]** **[HOTSPOT2]**.

Comment différencier les clients enregistrés des autres ? Le suivi des clients authentifiés sur le réseau est réalisé par le couple d'adresses MAC et IP ainsi qu'une fenêtre pop-up ouverte permettant de maintenir la session entre le client et le portail captif. Ce principe évite l'installation d'un logiciel agent supplémentaire.

Le responsable SSI de l'entreprise doit évaluer les risques :

- de contournement de son infrastructure : AP Wi-Fi factice avec SSID identique et de puissance plus forte permettant la récupération d'identifiants, usurpation d'adresses MAC/IP d'un client enregistré, installation de faux serveurs DHCP répondant plus rapidement aux sollicitations des clients, établissement de tunnels chiffrés ou cachés, impossibilité de prouver l'identité d'un utilisateur.
- d'attaque sur les clients du réseau interne : réseau filaire plus fiable que le Wi-Fi, vol de données personnelles ou des identifiants de connexion, client sans antivirus ou système d'exploitation obsolète.
- de non-mise à jour pour les équipements ou services ne disposant pas de navigateur (wsus, antivirus, etc.).

2 La réponse d'ALCASAR

Un Alcasar est un palais fortifié alliant des qualités militaires et d'agrément tel qu'aimaient se les faire construire les souverains musulmans sur la péninsule ibérique et dans le sud de la France. Aujourd'hui, certains lieux dits « ouverts » ne peuvent plus se passer d'un accès à Internet, comme les hôtels, les salles de conférence, les salles de classe, les zones d'hébergement pour étudiants, les fastfoods, les campings internationaux, sans pour autant détenir un personnel informaticien sur place et compétent pour assurer l'exploitation d'un dispositif de sécurité répondant aux obligations légales.

2.1 Un cas d'école

ALCASAR est un projet initié en 2008 par une équipe française du ministère de la Défense **[ALCASAR]**. Le résultat est une solution libre de contrôle d'accès à Internet, notamment pour les réseaux de type filaire et Wi-Fi, installée sur un PC bureautique récent doté de 2 cartes réseau. Utilisé en coupure, ce portail captif assure les services de passerelle d'interception, d'authentification, de contrôle des activités et d'imputation des usagers, conformément à la législation française **[LOG]**.

Les objectifs du projet sont de résoudre ou de limiter les risques et les faiblesses des portails captifs identifiés en introduction. Le cahier des charges répond à un besoin de déploiement rapide, de mise à jour automatique sans intervention d'un administrateur et d'une exploitation assurée par un personnel non informaticien, telle une appliance *plug and play*, mais de type boîte blanche.

Trois profils différents permettent de dissocier les rôles en entreprise :

- le profil *admin* permet d'accéder à toutes les fonctions d'administration du portail ;
- le profil *manager* est limité aux tâches de gestion des usagers ;
- le profil *backup* est dédié aux tâches de sauvegarde et d'archivage des fichiers journaux.

Le système d'exploitation installé est Linux Mandriva, épuré par un script d'installation et de configuration commenté en français. C'est pourquoi ALCASAR est utilisé comme outil pédagogique dans le cadre de plusieurs formations à la sécurité des systèmes d'information.

2.2 Les briques de la fortification

ALCASAR s'appuie sur une vingtaine de logiciels libres afin de constituer un portail captif authentifiant et sécurisé.

Système d'exploitation et pare-feu	Linux Mandriva et Netfilter
Passerelle d'interception	CoovaChilli
Serveur DHCP et DNS	Dnsmasq [DNSMASQ]
Serveur HTTP	Apache
Chiffrement flux HTTP	OpenSSL
Middleware	PHP PERL
Serveur d'authentification	FreeRadius [RADIUS]
Base de données usagers	Mysql
Cache WEB (proxy)	Squid
Serveur de temps	ntpd
Journalisation	Ulogd
Filtrage WEB par liste noire	DansGuardian
Statistiques de consultation	Awstat
Lecture des journaux du pare-feu	FirewallEyes
Archivage à chaud du système	Mondo Mindi
Chiffrement des fichiers journaux	Gnupg
Connexion distante sécurisée	openssh-server
Passerelle antivirus WEB	HAVP
Antivirus	LibClamav

Tableau 1 : Les briques Alcasar

Pour cet article, nous allons réaliser un focus sur la fonction interception/authentification (CoovaChilli, DNSMasq, Apache, Freeradius, MySQL) et la fonction traçabilité (log Netfilter et Squid, Freeradius et MySQL) implémentées sur Alcasar.

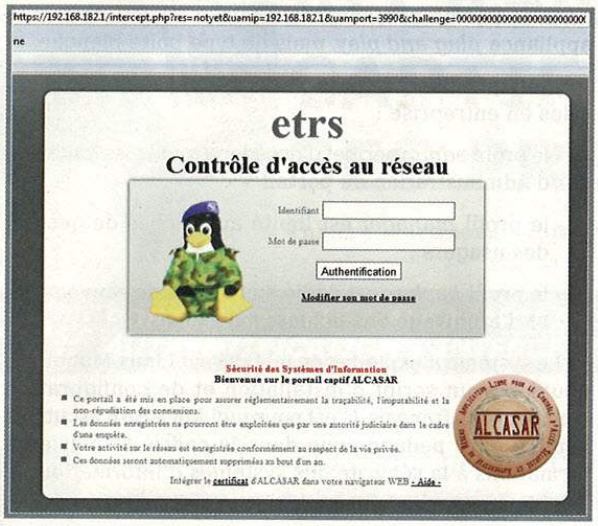


Fig 1 : Interception par CoovaChilli

2.3 Les limites repoussées

2.3.1 Déverrouillage

Par défaut, ALCASAR est configuré pour bloquer tous les flux réseau en provenance d'équipement sans usager authentifié. Il est possible d'autoriser le passage de certains flux non authentifiés vers des URL web ou IP de confiance. Cette possibilité permet par exemple :

- aux logiciels antivirus et système d'exploitation de se mettre à jour automatiquement.
- aux équipements sans navigateur de dialoguer (@MAC spécifiée), comme une borne ou un automate (Stuxnet vous guette !).

2.3.2 Vol de session

Cette technique exploite les faiblesses des protocoles Ethernet et Wi-Fi. La fonction interception/authentification s'appuie sur la passerelle CoovaChilli, le serveur web apache, le serveur d'authentification Freeradius et MySQL. CoovaChilli crée une interface réseau virtuelle TUN/TAP pointant sur l'interface physique ETH. Cet artifice lui permet de gérer sa propre table de résolution ARP en espace utilisateur sous Linux. L'intérêt de cette gestion consiste à verrouiller les couples @MAC/@IP rencontrés sur le réseau. Un empoisonnement du cache ARP par le réseau est alors impossible (*cache poisoning*).

Alcasar utilise également un script *watchdog* lancé toutes les 2 à 3 minutes (commande **arping**) permettant d'éviter l'usurpation d'adresses MAC et IP. Les stations non accessibles sur le réseau (oublis de déconnexion) sont déconnectées par CoovaChilli (commande **chilli_query logout** et filtrage Netfilter sur le port 3990). En outre, un

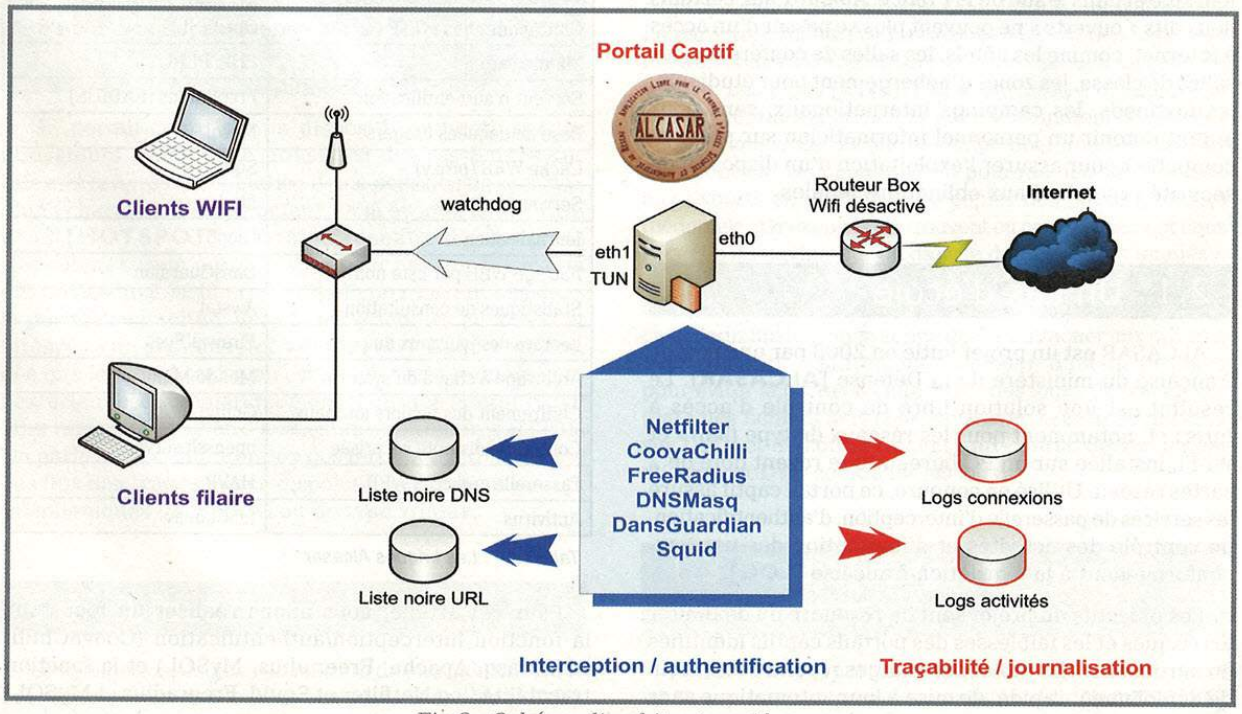


Fig 2 : Schéma d'architecture Alcasar

plugin permet de déconnecter automatiquement l'usager d'Alcasar à la fermeture de sa session Windows. Le risque de vol de session reste toutefois possible pendant 2 à 3 minutes en cas de client sous Linux (Figure 2).

2.4 Gestion des logs

Dans le cadre de la cybersurveillance et pour répondre aux exigences de la CNIL **[CNIL]**, la production des traces est associée aux mécanismes suivants :

- les flux liés à l'authentification des usagers sur ALCASAR sont chiffrés. Les mots de passe sont stockés et chiffrés dans la base interne MySQL. Les fichiers de traces peuvent être chiffrés. Ces précautions permettent de prévenir l'accusation d'un autre usager ou d'un administrateur d'avoir récupéré, exploité ou modifié des données.
- la consultation directe des activités internet nominatives est impossible. En effet, les journaux des connexions sont répartis dans différents fichiers. L'imputation des connexions n'est ainsi rendue possible qu'après un travail d'agrégat réservé aux autorités judiciaires.

Il est conseillé de chiffrer les logs du pare-feu Netfilter et de SQUID à l'aide d'un algorithme asymétrique (clé publique/clé privée). En fournissant la clé privée à un responsable de votre société, les administrateurs sont protégés de toute tentative de modification des logs.

L'architecture et la politique préconisées pour la gestion des journaux sont détaillées dans un document du CERTA **[CERTA]** applicable pour un portail captif.

2.5 La force du portail captif

La requête DNS initiale du poste client est récupérée par CoovaChilli qui teste sa conformité et la redirige vers le DNS de confiance interne fourni par le DHCP. La liste noire de domaine assure la fonction de filtrage (fonction **DNSMASQ**). Les tunnels DNS sont ainsi bloqués pour les utilisateurs authentifiés ou non. Le filtrage d'URL est alors assuré par DansGuardian (liste noire de Toulouse1). La requête DNS est alors transférée vers un DNS public, par exemple un OpenDNS ou le DNS de votre FAI.

La connexion directe par adresse IP (sans résolution de nom de domaine) pour établir un tunnel chiffré SSH ou VPN et ainsi contourner le dispositif de filtrage (procédé utilisé par certains logiciels anti-Hadopi comme Ultrasurf ou certains *botnets*) peut être contrôlée par liste blanche. Les logs d'Alcasar permettront de tracer les échanges sans pouvoir les imputer.

3 Emplois et perspectives

3.1 Bonnes pratiques

- Sécurité physique

Le poste hébergeant Alcasar doit être considéré comme un élément sensible du réseau. Toutes les précautions d'usage doivent être appliquées, comme le contrôle d'accès par clé ou digicode dans le local ondulé et

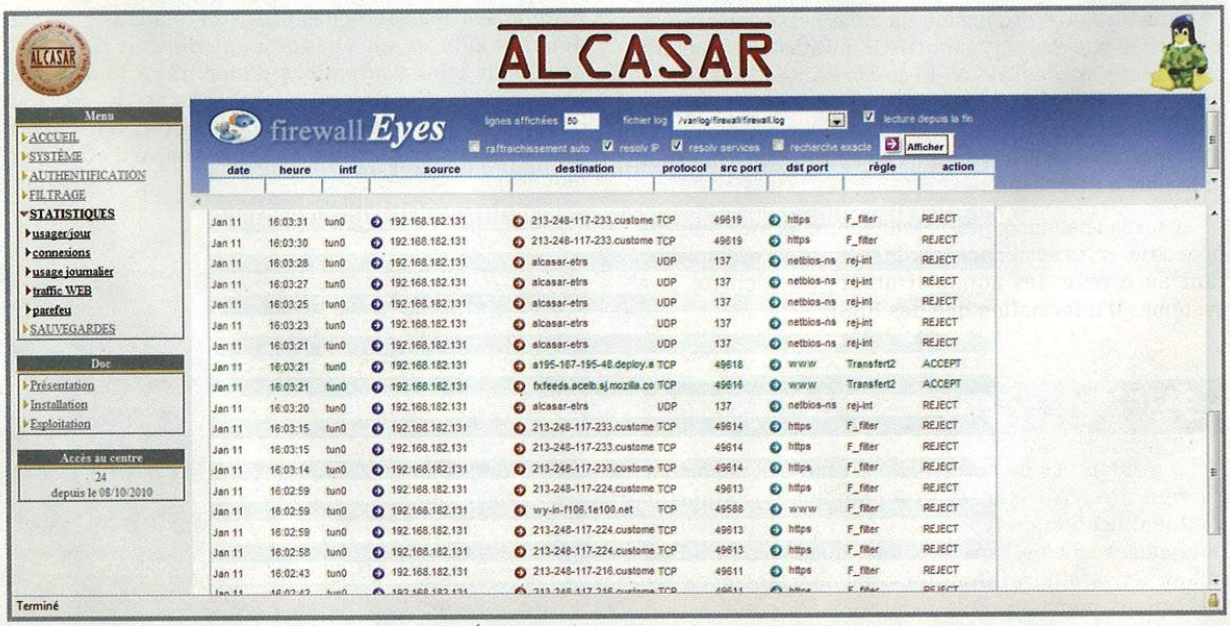


Fig 3 : Écran Alcasar pour les statistiques

climatisé, l'éloignement des supports de sauvegarde, y compris le CD de la distribution Mandriva utilisée. Le poste Alcasar ainsi que les moyens de connexion à Internet peuvent être redondés par un dispositif tiers pour une utilisation sans coupure de service.

- Conformité des clients

La défense en profondeur, terme emprunté à une technique militaire destinée à retarder l'ennemi, consiste à exploiter plusieurs techniques de sécurité afin de réduire le risque lorsqu'un composant particulier de sécurité est compromis ou défaillant. Pour les ordinateurs personnels, non maîtrisés par l'administrateur, il s'agit de mettre en œuvre un contrôle de conformité. Pour les postes sous Windows, Linux et Mac OS X, Microsoft propose la solution *Network Access Protection* **[NAP]** avec Windows 2008 server, imposant aux clients la mise à jour du système d'exploitation et du fichier de signature antivirus avant l'accès au Web. Pour les postes en entreprise (dits maîtrisés), une solution 802.1X couplée à un annuaire Active Directory est compatible avec le serveur d'authentification Freeradius inclus dans Alcasar.

3.2 Retour d'expérience

À titre d'exemple, un lycée militaire utilise deux Alcasar depuis 3 ans sur PC bureautique de 1 Go de RAM chacun, avec plus de 1000 comptes actifs. Les logs sont archivés sur DVD une fois par an. Les compétences de l'exploitant sont limitées à la gestion des comptes (import de fichier Excel) et aux mises à jour de la plateforme.

Dans un autre organisme, la zone hébergement est perturbée par des serveurs DHCP pirates, des joueurs en réseaux et des AP Wi-Fi fleurissantes. Certaines grandes écoles informatiques ont d'ailleurs délégué l'administration du réseau hébergement à un binôme d'élèves responsables, assurant l'administration technique et la police de proximité !

Au niveau pédagogique, l'architecture Alcasar retenue pour assurer les fonctionnalités de sécurité est exemplaire, tant au niveau des administrateurs de sécurité des systèmes d'information que des RSSI.

3.3 Perspectives

La gestion des comptes reste un facteur important de réussite : Alcasar va prochainement supporter l'authentification de type SSO CAS, technique utilisée notamment sur les Espace Numériques de Travail en milieu universitaire. L'intérêt est important pour les utilisateurs nomades où la fédération d'identité est un gage de sécurité des identifiants personnels.

Dans les prochaines semaines, selon le secrétaire général de la Haute Autorité (janvier 2011), l'ensemble du code source du logiciel HADOPI sera libre. Si la HADOPI constate un délit, c'est l'adresse IP de la box (ou côté WAN du poste Alcasar) qui sera incriminée. Espérons que ce code source permettra d'adapter Alcasar et assurera la cohérence des logs avec les journaux des clients Hadopi.

Alcasar est un ensemble d'outils s'appuyant uniquement sur la distribution Mandriva. À l'instar du logiciel Bastille-Linux (sécurisation pédagogique d'un serveur), le support des autres distributions comme Ubuntu reste attendu.

Conclusion

Il n'existe pas de solution de portail captif infaillible, mais nous avons vu que la solution Alcasar prenait en compte les principaux procédés de contournement (tunnel DNS) et d'usurpation d'identité (@MAC/IP). La gestion des logs et les mises à jour automatiques (OS, antivirus, listes noires) simplifient le travail de l'administrateur, lorsqu'il y en a un ! La version V2.0 (janvier 2011) du portail captif Alcasar répond aux principales limites des portails captifs dans un environnement professionnel comme personnel, notamment en présence d'une population de mineurs en appliquant des listes noires.

La démarche SSI de défense en profondeur, l'approche pédagogique des scripts et de la documentation, ainsi que l'architecture composée de briques répondant à des problématiques simples en font un projet exemplaire. Il reste la sensibilisation des utilisateurs à la connexion/déconnexion et au changement de mot de passe pour limiter les usurpations d'identité. À l'instar des correctifs de sécurité, la journalisation n'est pas une option de sécurité mais bien une impérative nécessité pour faciliter l'établissement d'une preuve d'effraction numérique. ■

■ RÉFÉRENCES

[ALCASAR] http://www.alcasar.info, **forge communautaire de l'administration française** www.adullact.net

[CERTA] Log CERTA : www.certa.ssi.gouv.fr/site/CERTA-2008-INF-005

[CNEJITAS] La preuve numérique à l'épreuve du litige – avril 2010 - http://www.cnejita.org/doc/CNEJITA-ACTES-COLLOQUE13042010-A5-V5.1.pdf

[DNSMASQ] *LINUX Magazine* 124 (Février 2010) : utilisez DNSMASQ pour interdire des sites

[HOTSPOT] *MISC* 53 (Janvier/Février 2011) : Sécurisation d'un réseau Wi-Fi d'entreprise - Sécurité des architectures hotspot

[HOTSPOT2] Limitations portail captif : http://sid.rstack.org/pres/0705_Rectorat_Hotspots.pdf (C. Blancher – 2007)

[NAP] *MISC* 32 (Juillet/Août 2007) : Microsoft Network access Protection

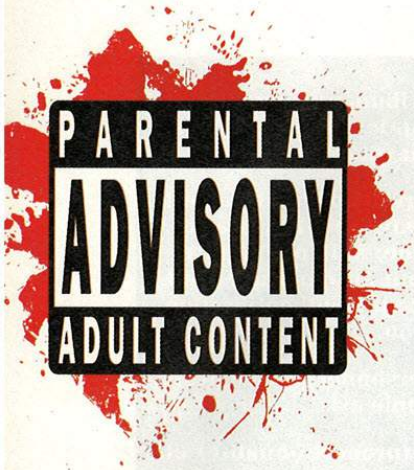
[RADIUS] *LINUX Magazine* 123 (Janvier 2010) : écriture d'un module RADIUS

[LOG] Décret nº 2006-358 du 24 mars 2006 relatif à la conservation des données des communications électroniques - Article R10-13

La loi n° 2006-64 du 23 janvier 2006 sur la lutte contre le terrorisme - Article 5

[CNIL] La CNIL et les tribunaux considèrent la cybersurveillance légale quand les trois conditions suivantes sont remplies :

- L'existence de la cybersurveillance doit d'abord avoir été portée à la connaissance des salariés, soit par voie d'affichage, soit par note de service. ALCASAR fournit automatiquement cette information sur la page d'authentification lors de chaque connexion.
- Les représentants du personnel doivent avoir été consultés (pour simple avis).
- Elle doit être justifiée (proportionnalité) et limitée à une surveillance de flux (volume de trafic, type de fichiers échangés, filtrage URL, etc.) sans accéder aux contenus des courriers électroniques, ni aux répertoires identifiés comme « personnels » sur le disque dur du poste de travail du salarié sous peine d'être poursuivi pour violation de correspondance privée. Les traces enregistrées par ALCASAR correspondent à cette exigence.



FILTRAGE DES FLUX WEB DANS L'ACADÉMIE DE NANCY-METZ POUR LA PROTECTION DES MINEURS (PARTIE 2)

Fabrice FLAUSS – DSI/DIT – Rectorat de Nancy-Metz – fabrice.flauss@ac-nancy-metz.fr

mots-clés : FLUX WEB / PROXY / ROUTAGE / FIREWALLS / GESTION CENTRALISÉE

La Division du système d'Information de l'académie de Nancy-Metz a souhaité mettre en œuvre une solution « plug and play », si possible installée par l'utilisateur lui-même, de filtrage des accès web en école primaire, à la manière des Box ADSL.

Lors de la première partie de cet article, diverses solutions ont été étudiées avant de retenir le boîtier Fast360 de la société Arkoon. Cette deuxième partie présente la mise en œuvre d'un boîtier unique puis son clonage en vu d'un déploiement industrialisé.

1 Préambule

L'académie de Nancy-Metz comporte plus de 2500 écoles primaires, cet article se focalise sur le filtrage des flux web pour la protection des mineurs au sein de celles-ci.

L'académie était déjà dotée d'une plateforme centralisée permettant de filtrer l'ensemble des flux web des collèges et lycées. Cet article se propose de démontrer comment industrialiser une solution à destination des écoles primaires et maternelles afin de rediriger ses flux web vers cette plateforme centralisée au moyen d'un équipement simple.

2 Rappel des objectifs

La solution doit être avant tout transparente pour l'utilisateur. Il ne s'agit, a priori, que d'une redirection des flux web provenant de l'école vers une plateforme centralisée telle que présentée sur la figure 1.

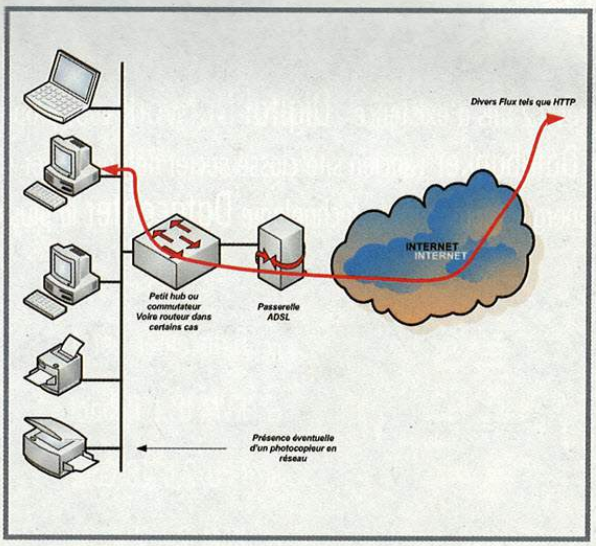


Figure 1 : Schéma « type » d'une école

Dans bon nombre de cas, l'école est dotée d'un accès ADSL avec ces caractéristiques :

- un fournisseur d'accès à Internet avec un modem réalisant dans certains cas de la translation d'adresses ;
- un adressage IP fixe ou dynamique ;
- un ou plusieurs postes de travail et, dans certains cas, des photocopieurs en réseau nécessitant une maintenance à distance par des tiers ;
- les paramètres réseau des équipements configurés en général via DHCP.

En cas d'adressage dynamique, nous allons devoir utiliser un mécanisme tel que *proxy-auth* afin d'authentifier l'école distante.

La figure 2 présente le fonctionnement global de l'architecture d'une l'école.

Pour le réseau local, le boîtier doit implémenter les points suivants :

- serveur DHCP ;
- router les flux en provenance du réseau local et à destination du port http (dans un premier temps) vers le port destination du proxy administré par l'académie ;
- transmettre les informations user/password via le mécanisme proxy-auth ;
- se connecter au dispositif de gestion centralisée (ici AMC, *Arkoon Management Center*) en présentant un certificat pour l'authentification.

3 Présentation et optimisation de la solution retenue

Le boîtier retenu pour l'ensemble des écoles primaires est le modèle SMALL 90 (noté par la suite de l'article S90) de la gamme d'Appliance Arkoon Fast 360. Il dispose de 6 ports Ethernet dont les 4 premiers sont configurés en bridge, de deux ports Ethernet et d'un port console, comme l'illustre la figure 3, page suivante.

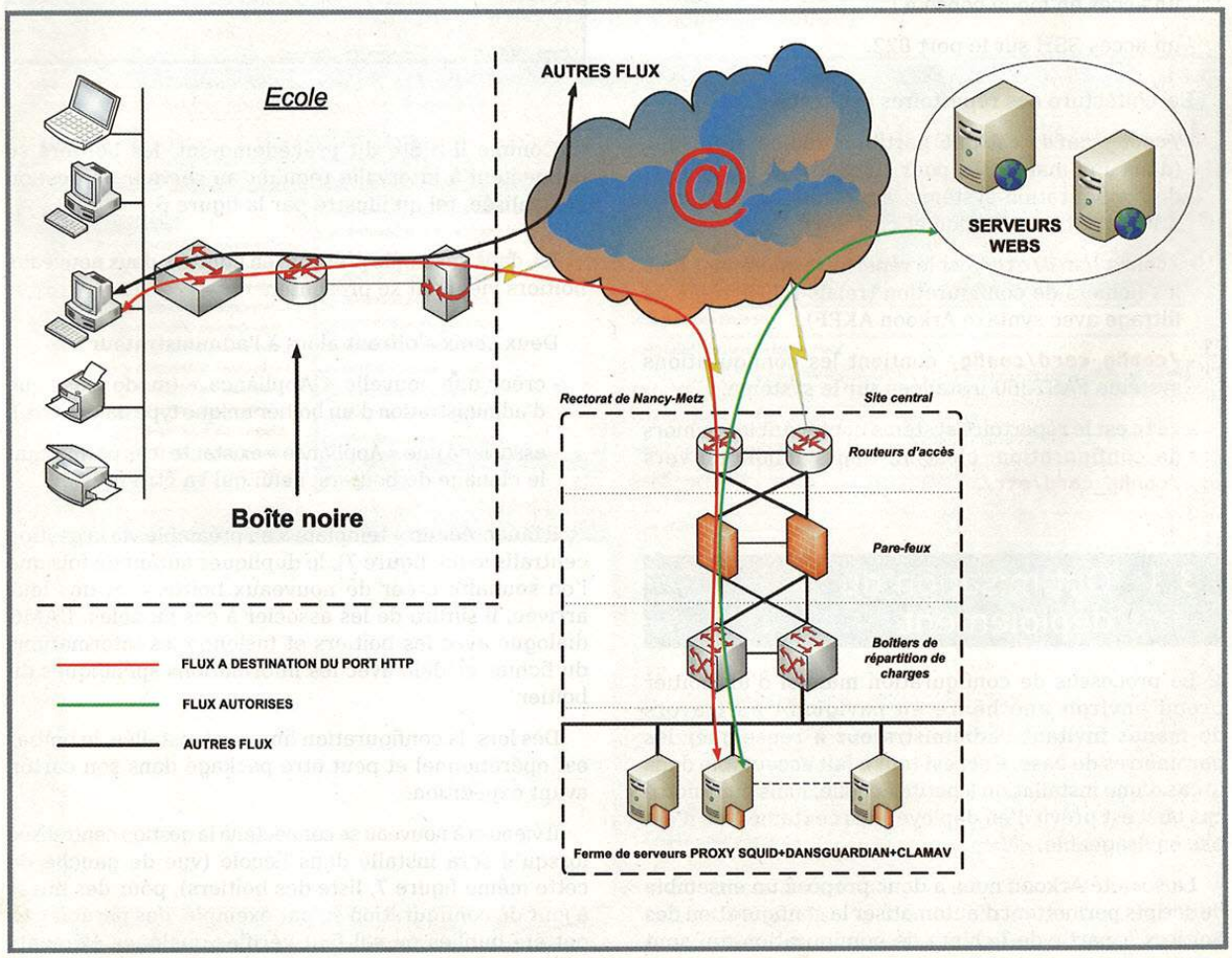


Figure 2 : Synopsis de fonctionnement

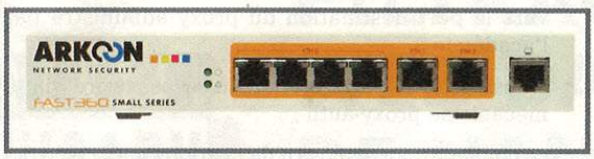


Figure 3 : Face avant d'un SMALL 90

Le boîtier est également doté à l'arrière d'un port USB (cf. figure 4) pour les mises à jour et la mise en place de la configuration.

Figure 4 : Face arrière d'un SMALL 90

Un boîtier « sorti du carton » se retrouve avec ces caractéristiques :

- adresse IP eth0 en 192.168.100.1 ;
- un accès en mode console ;
- un accès SSH sur le port 822.

L'architecture des répertoires est la suivante :

- **/config_card/** est une partition dédiée du disque (dans un Small, SSD) pour stocker tous les fichiers de configuration système, ainsi que les certificats X509 d'authentification et clés SSH.
- **/config_card/etc/** est le répertoire contenant tous les fichiers de configuration (relais HTTP, règle de filtrage avec syntaxe Arkoon AKPF).
- **/config_card/config/** contient les configurations système FAST360 installées sur le système.
- **/etc** est le répertoire système contenant les fichiers de configuration, c'est un lien symbolique vers **/config_card/etc/**.

3.1 Optimisation du déploiement

Le processus de configuration manuel d'un boîtier prend environ une heure en naviguant au travers de menus invitant l'administrateur à renseigner les paramètres de base. Ceci est tout à fait acceptable dans le cas d'une installation à petite échelle, mais dans notre cas où il est prévu d'en déployer des centaines, ce n'est pas envisageable.

La société Arkoon nous a donc proposé un ensemble de scripts permettant d'automatiser la configuration des boîtiers à partir de fichiers de configuration qui sont communs à toutes nos écoles.

3.2 Finalisation du boîtier via l'AMC

L'application de management des appliances S90 « Arkoon Management Center » se présente telle qu'illustrée en figure 5.

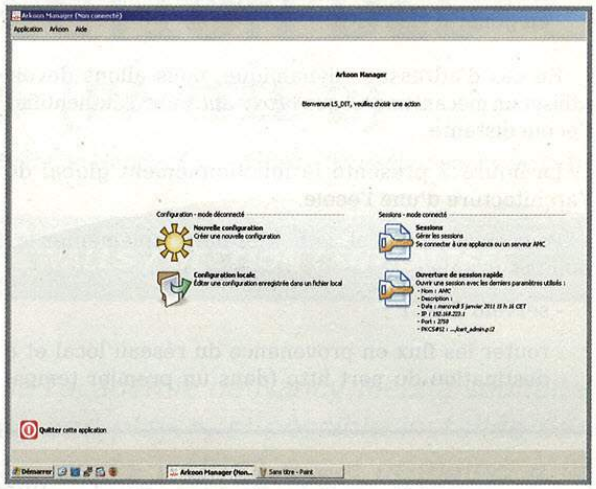


Figure 5 : écran d'accueil AMC

Comme il a été dit précédemment, les boîtiers se connectent à intervalle régulier au serveur de gestion centralisée, tel qu'illustré par la figure 6.

Ici, dans l'exemple présenté en figure 6, deux nouveaux boîtiers viennent se présenter.

Deux choix s'offrent alors à l'administrateur :

- créer une nouvelle « Appliance » (mode classique d'administration d'un boîtier unique type datacentre).
- associer à une « Appliance » existante (cas permettant le clonage de boîtiers, celui qui va être utilisé).

Il faut créer un « template » au préalable via la gestion centralisée (cf. figure 7), le dupliquer autant de fois que l'on souhaite créer de nouveaux boîtiers, et dès leur arrivée, il suffira de les associer à ces modèles. L'AMC dialogue avec les boîtiers et fusionne les informations du fichier modèle avec les informations spécifiques du boîtier.

Dès lors, la configuration finale est installée, le boîtier est opérationnel et peut être packagé dans son carton avant expédition.

Il viendra à nouveau se connecter à la gestion centralisée lorsqu'il sera installé dans l'école (vue de gauche de cette même figure 7, liste des boîtiers), pour des mises à jour de configuration si, par exemple, des paramètres ont été oubliés ou s'il faut vérifier quelques éléments en cas d'appel à l'assistance.

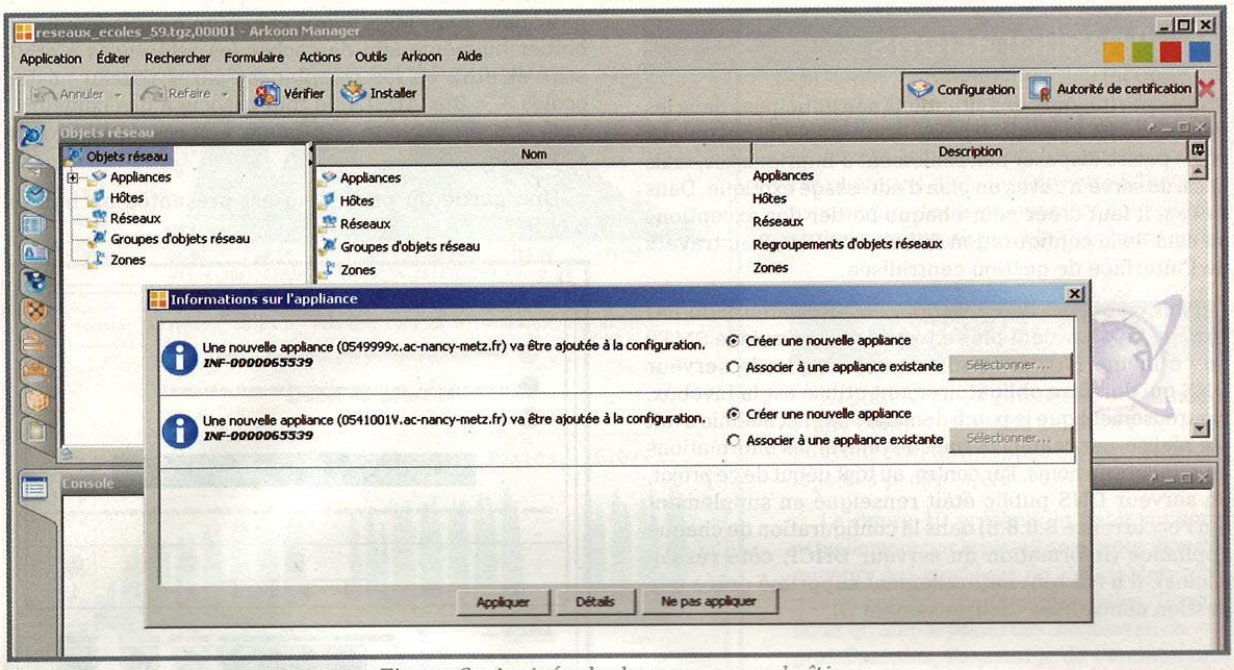


Figure 6 : Arrivée de deux nouveaux boîtiers

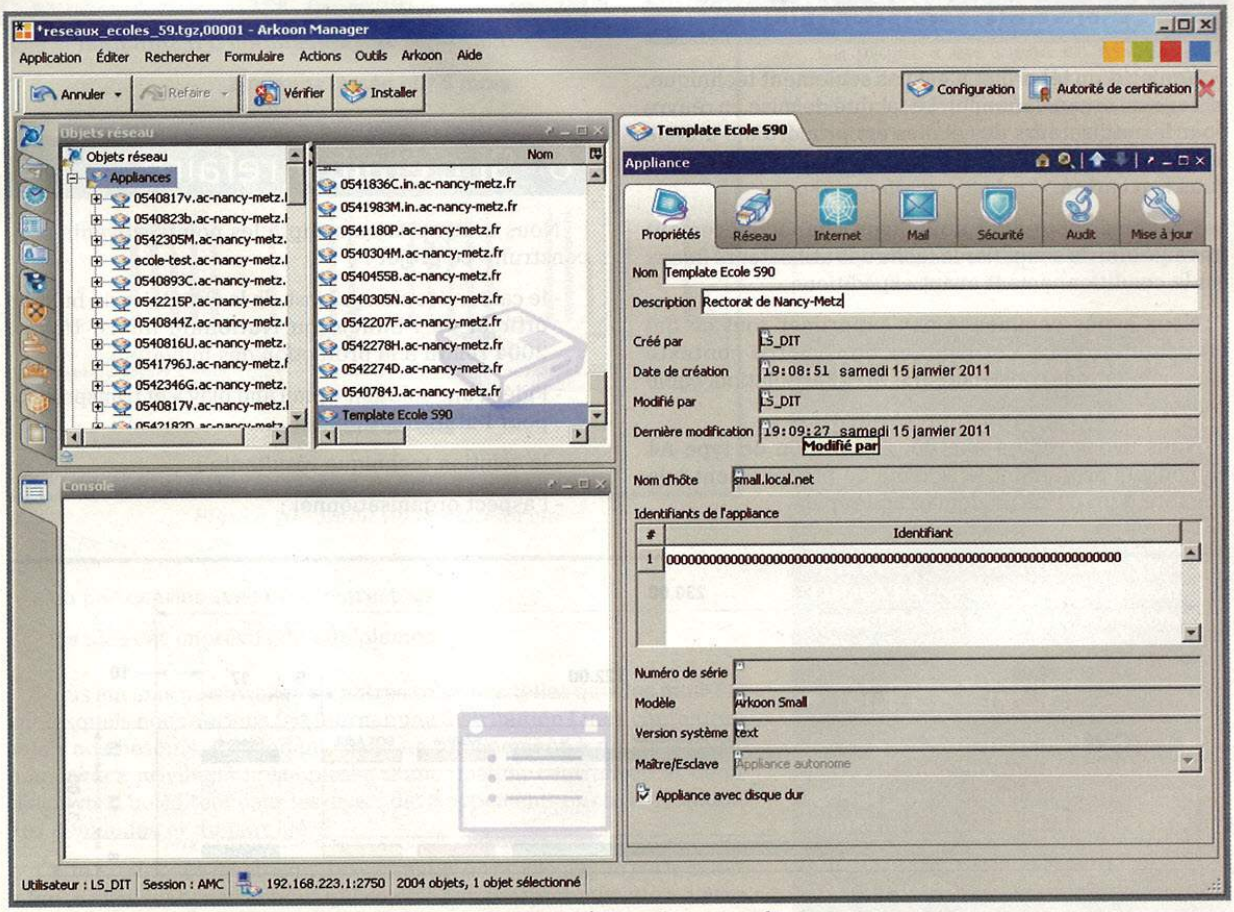


Figure 7 : Création d'un template

4 Améliorations

Il existe toujours des situations non anticipées dans les déploiements à grande échelle. Par exemple, le fait qu'une école puisse disposer non seulement d'imprimantes, mais aussi de serveurs avec un plan d'adressage exotique. Dans ce cas, il faut créer pour chaque boîtier des exceptions au sein de la configuration du serveur DHCP au travers de l'interface de gestion centralisée.

Pour ce qui est de l'anecdote, les utilisateurs du FAI Orange ne pouvaient plus envoyer de mails via le SMTP de ce même FAI, la raison en est simple, le serveur DNS qui doit être obligatoirement utilisé est la Livebox. Heureusement que le patch demandé par l'académie avait été réalisé, car celui-ci permet de relayer les informations de serveurs de noms. Par contre, au tout début de ce projet, un serveur DNS public était renseigné en supplément (en l'occurrence 8.8.8.8) dans la configuration de chaque appliance (information du serveur DHCP, côté réseau écoles), il a été tout naturellement supprimé grâce à la gestion centralisée (heureusement :)).

5 Aspect organisationnel

L'enjeu d'un tel projet n'est pas seulement technique, mais aussi organisationnel. La facilité de mise en œuvre pour les utilisateurs des écoles est primordiale afin de les faire adhérer au projet et que le boîtier ne reste pas dans son carton.

Un point sur lequel nous avons particulièrement travaillé afin de simplifier la tâche des utilisateurs finaux est le conditionnement avant expédition.

Nous avons commencé par supprimer tout ce qui était inutile pour l'utilisateur dans notre contexte (documentations constructeurs, CD d'installation, câble console, ...).

Nous avons rédigé une documentation de type A4 en couleur présentant le schéma de branchement à la manière d'un guide de démarrage rapide.

Et pour finir, nous avons conçu un habillage pour le boîtier lui-même afin de cacher les références à eth0, eth1 et eth2 en les remplaçant par « réseau postes écoles », « Box ADSL», le tout associé à des couleurs : vert « mode protégé », « jaune » accès ADSL, rouge « réservé », ...

Une partie du packaging est présentée en figures 8 et 9.

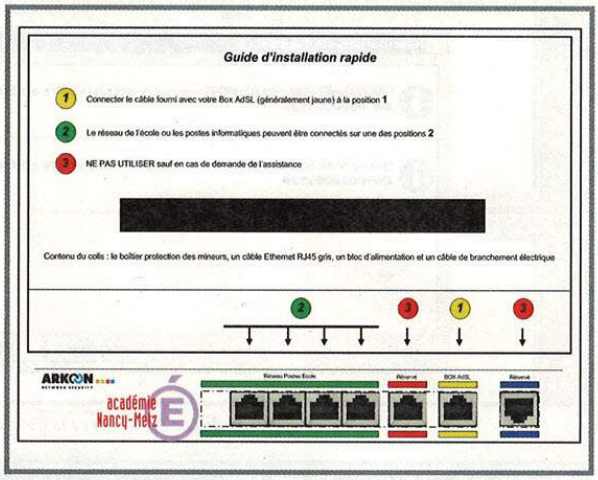


Figure 9 : Documentation d'installation

6 Si c'était à refaire...

Nous avons pris en compte les points suivants pour construire ce projet :

- le cahier des charges issu de la circulaire du bulletin officiel de l'Éducation Nationale du 18 février 2004 relatif à la protection des mineurs ;
- l'idée : une solution « plug and play » et transparente pour l'utilisateur ;
- la solution technique résultante ;
- l'aspect organisationnel ;

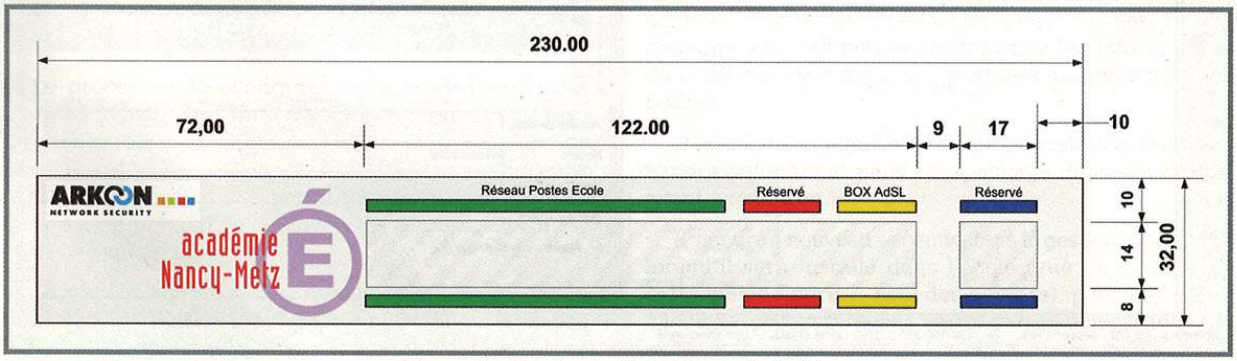


Figure 8 : Plastron d'habillage du boîtier

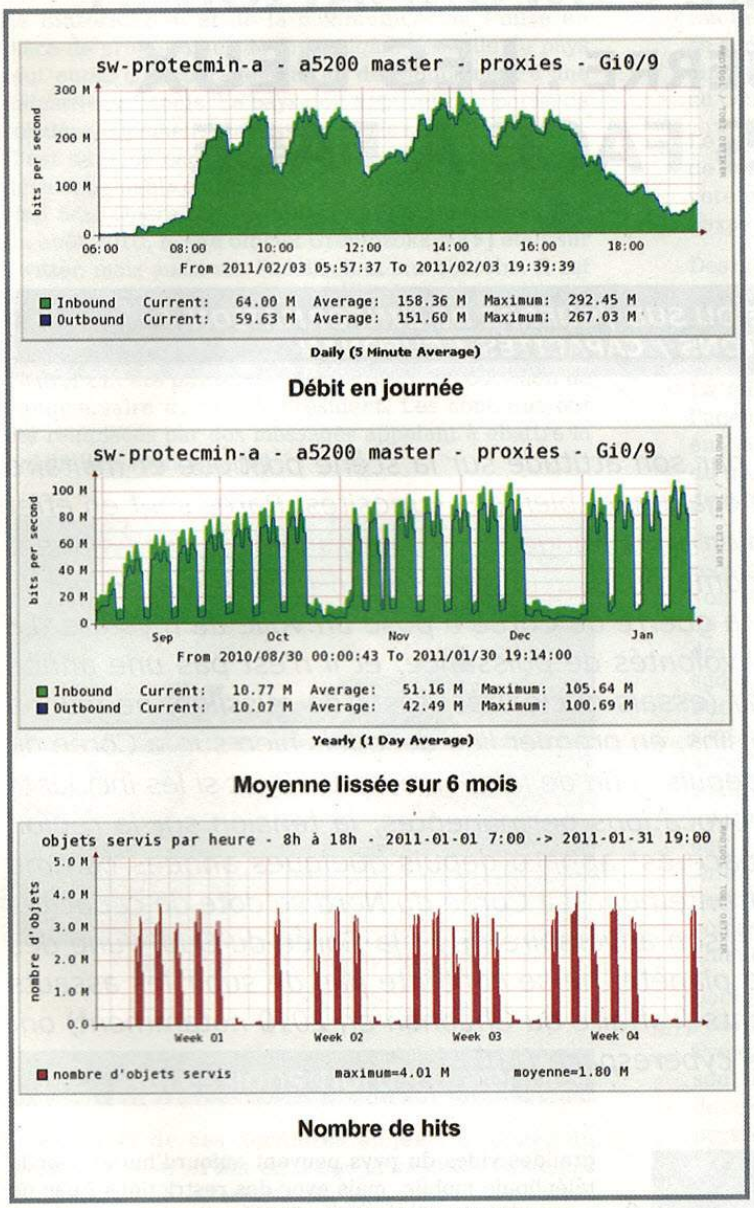


Figure 10 : Débit de la plateforme

- un partenariat avec un constructeur ;
- les aléas et imprévus du déploiement.

Nous aurions pu envisager d'autres solutions telles que des mini-PC sur lesquels nous aurions fait tourner une distribution Linux configurée selon nos besoins. Cependant, dans un déploiement à notre échelle, nous avons privilégié la souplesse d'une gestion centralisée et de la réactivité d'un éditeur dans les divers développements en cas d'évolution des demandes et du parc géré.

Cette expérience montre qu'il est essentiel de privilégier un partenariat « fort » avec un constructeur, qui se doit par contractualisation d'être réactif.

Si nous sommes satisfaits de la solution retenue, la principale faiblesse concerne certaines fonctionnalités perfectibles de l'interface de gestion centralisée (telle que l'absence de traitement par lots), qui se révèle le point le plus critique dans l'exploitation d'un parc de cette taille. Ceci est en cours de développement côté éditeur et devrait être disponible prochainement.

Conclusion

La plateforme de protection des mineurs étant centralisée, des tableaux de bord peuvent être développés en exploitant les logs des proxy Squid.

La figure 10 présente le débit supporté actuellement par la plateforme protection des mineurs de l'académie de Nancy-Metz ainsi que le nombre d'objets manipulés. Il est à noter qu'afin de pallier des montées en charge grandissantes, l'ensemble de l'architecture a été pensé en utilisant au maximum de l'agrégation de liens Ethernet gigabit.

Ce projet a été mené en essayant de prendre en considération la problématique des fournisseurs d'accès internet, minimiser les interventions sur site et permettre à l'utilisateur la possibilité d'effecteur des manipulations simples.

Pour conclure, à ce jour, le retour est positif, fidèle à ces objectifs de simplicité pour l'utilisateur. ■

REMERCIEMENTS

Merci à Dominique ALLIETTA et Laurent CHEYLUS de la société ARKOON pour leur aide précieuse, ainsi qu'à l'ensemble des relecteurs.

Des remerciements particuliers à Maxime MAZZINI, Pascal PIERRE, Nicolas THOUVENIN, ainsi qu'à l'ensemble des animateurs TICE de l'académie.

RÉFÉRENCES

FLAUSS (Fabrice), FOLL (Cédric) « Filtrage des flux web dans l'académie de Nancy-Metz pour la protection des mineurs (Partie 1) », *MISC* 51

Appliance ARKOON : http:/www.arkoon.net

GUERRE DE L'INFORMATION ET CYBERGUERRE : LES DEUX CORÉES FACE À FACE

Daniel Ventre – CNRS

mots-clés : *CORÉE DU NORD / CORÉE DU SUD /GUERRE DE L'INFORMATION / CYBERGUERRE / AGRESSIONS / CAPACITÉS / CHEONAN*

La Corée du Nord inquiète par son attitude sur la scène politique et militaire internationale. Elle suscite également bien des curiosités. Rares sont en effet ceux qui la connaissent bien ; les images et les informations qui émanent du pays sont relativement peu nombreuses : le repli sur lui-même imposé par le pouvoir en place au pays depuis la guerre de Corée a posé un voile de mystère. Le pays n'en affiche pas moins ses volontés de puissance, et il n'est pas une année sans qu'un événement significatif (essais nucléaires, tests de missiles) ne vienne perturber la stabilité des États voisins, en premier lieu desquels bien sûr la Corée du Sud. Les deux États se font face depuis la fin de la guerre en 1953, et si les incidents militaires sont nombreux, les provocations permanentes, la tension sur la région frontalière extrême, le cyberespace est apparu depuis quelques années comme l'un des nouveaux domaines d'affrontement. La Corée du Nord se dote de capacités qui lui permettent d'opérer contre son adversaire (§I) ; la Corée du Sud, l'une des nations les plus connectées de la planète, ne se contente pas de subir les assauts (§II). Des incidents majeurs récents (l'affaire du Cheonan en 2010 notamment) ont trouvé leur prolongement dans le cyberespace (§III).

1 La Corée du Nord

1.1 Le cyberespace nord-coréen

Depuis le début des années 2000, l'Internet nord-coréen, conçu comme un intranet, est accessible aux citoyens. Les contenus proposés sur cet Internet national sont des produits conçus à l'intérieur du pays. L'Internet mondial n'est pour sa part accessible qu'à une infime partie de la population, essentiellement des élites politiques et militaires **[1]**. Ces connexions sont réalisées via l'opérateur China Netcom et par satellites. La téléphonie mobile a été introduite en Corée du Nord dès 2002, par la société Loxley Pacifique et le gouvernement nord-coréen. Les habitants de la capitale et de quelques grandes villes du pays peuvent aujourd'hui utiliser la téléphonie mobile, mais avec des restrictions : pas de communications internationales directes, limites imposées aux communications avec des résidents étrangers **[2]**. Le *New York Times* rapportait récemment (juin 2010) que 1024 adresses IP avaient été enregistrées par la Corée du Nord, via une société enregistrée à Pyongyang. Faut-il y voir là la volonté des autorités de renforcer la présence nord-coréenne dans le cyberespace ? Sans doute. Déjà des serveurs localisés à l'étranger sont-ils utilisés pour diffuser de l'information officielle. Le pays veut assurer sa présence sur l'Internet mondial, user de cet outil à des fins de médiatisation. Le site de la *Korean Central News Agency* (www.kcna.co.jp) est hébergé à Tokyo, celui du *Pyongyang Times* (www.times.dprkorea.com) à Pékin. Récemment, la Corée du Nord s'est lancée activement dans un programme qui lui permettra de rattraper une partie de son retard dans le domaine des technologies

de l'information et de la communication : mise en place de programmes de formation à l'échelle du pays tout entier ; encouragement au développement d'une industrie nationale. Le pays doit appuyer une partie de son développement économique sur le secteur des NTIC. C'est dans ce cadre que s'inscrivent notamment des développements comme celui du système d'exploitation *Red Star Operating Systems* **[3]**, basé sur Linux **[4]**. En août 2010, le site officiel Uriminzokkiri **[5]** était sur Twitter, mais aussi sur Facebook et sur Youtube. Il est à noter que les nord-coréens n'ont bien entendu pas accès à ces pages qui sont destinées à la communauté internationale. Le site Uriminzokkiri et son compte Twitter ont été piratés en janvier 2011, à l'occasion de l'anniversaire du fils du Président. Les contenus ont été remplacés par des messages appelant à abattre la famille présidentielle **[6]**.

Red Star Operating System [7]

1.2 Agressions nord-coréennes

Au cours de ces dernières années, la Corée du Nord semble avoir fait un usage intensif des systèmes d'information pour mener des opérations de diverses natures, prenant pour cible ses adversaires que sont la Corée du Sud ou les États-Unis. De nombreux incidents lui sont en tous cas attribués par ces deux États :

- L'armée sud-coréenne confirmait en octobre 2005 avoir été victime d'intrusions nord-coréennes dans 33 des 80 réseaux de communication sans fil militaires utilisés lors d'exercices avec les États-Unis (*Ulchi-Focus Lens Exercise*).
- Les autorités nord-coréennes utiliseraient des sites internet de propagande pour diffuser des informations à l'intention de leurs espions à l'étranger.
- Des hackers nord-coréens ont tenté de s'introduire dans des systèmes d'information militaires américains et sud-coréens en 2006.
- L'unité 121 nord-coréenne (dédiée à la cyberguerre) se serait introduite dans les serveurs sud-coréens ainsi que du département de la défense américain au cours du mois de juillet 2006 **[8]**.
- Le conseil de sécurité des Nations Unies, informé de la réalisation de tests d'attaques informatiques, vote en octobre 2007 une résolution interdisant l'exportation d'ordinateurs vers la Corée du Nord.
- Des opérations d'espionnage sont rapportées par les autorités de Séoul en septembre 2008. Les militaires sud-coréens auraient reçu des chevaux de Troie dans leurs boîtes emails **[9]**.
- La NIS (*National Intelligence Service*), qui est l'agence de renseignement sud-coréenne, informait en 2008 le Premier Ministre que 130 000 documents du gouvernement avaient été piratés depuis 2004, de forts soupçons pesant contre Pyongyang.
- En mars 2009, un colonel de l'armée sud-coréenne recevait un fichier infecté par email, opération que l'on attribua à la Corée du Nord, toujours accusée de mener des opérations de renseignement, de déstabilisation de la défense. Les systèmes de l'armée sud-coréenne auraient été piratés. Des mots de passe auraient également été dérobés, permettant de s'introduire dans les serveurs de l'Institut National de la recherche environnementale **[10]** (NIER, *National Institute of Environmental Research*). Les hackers auraient dérobé des informations confidentielles sur le site du CARIS (*Chemical Accident Response Information System*) maintenu par le NIER. Près de 2 000 informations secrètes auraient ainsi été volées, dont les noms de 700 fabricants de produits chimiques toxiques.
- Lorsque la Corée du Sud confirma sa participation à l'exercice de cybersécurité Cyber Storm, en juin 2009, Pyongyang l'accusa de provocation **[11]**. Les États-Unis sont accusés pour leur part de préparer, au travers de ces exercices, des attaques préemptives contre les pays anti-américains. Les cyberattaques subies par la Corée du Sud et les États-Unis en juillet 2009 seraient une forme de réaction de Pyongyang à cette coalition armée dans le cyberespace **[12]**. Du 4 au 8 juillet 2009, les sites d'institutions financières, d'entreprises, d'agences et administrations gouvernementales américaines (Département du Trésor, services de renseignement, etc.) et sud-coréennes (bureau du Premier ministre, parlement, ministère de la Défense, ministère des Affaires étrangères) furent en effet paralysés par des cyberattaques. L'agresseur était-il vraiment nord-coréen ou un partisan du régime de Pyongyang ? La Corée du Nord était-elle d'ailleurs vraiment à l'origine de ces attaques ? La piste de hackers britanniques fut même évoquée **[13]**. De type DDoS, lancée à partir de serveurs localisés en Allemagne, en Autriche, en Géorgie, en Corée du Sud et aux États-Unis **[14]**, l'attaque impliqua quelque 12 000 ordinateurs en Corée du Sud et 8 000 dans

le monde selon la NIS **[15]**. D'autres rapports font état de 167 000 machines compromises dans 74 pays, utilisées pour l'attaque **[16]** ; d'autres parlent de 50 000 machines **[17]**. L'adresse IP à partir de laquelle l'attaque a été lancée aurait été celle des services de la Poste nord-coréenne **[18]**. L'attaque pouvait avoir plusieurs objectifs, notamment de tester la qualité des systèmes de sécurité informatique des pays cibles, ou de tester la résistance psychologique cette fois des États-Unis et de la Corée du Sud à des provocations, dont la Corée du Nord est coutumière.

- Les services de renseignement sud-coréens estiment que les hackers du nord ont volé les informations personnelles de près de 1,6 millions de citoyens importants en Corée du Sud **[19]** au cours des 5 dernières années.

www.auction.co.kr
www.chosun.com
www.hannara.or.kr
ebank.keb.co.kr
ezbank.shinhan.com
banking.nonghyup.com
www.assembly.go.kr
www.mofat.go.kr
www.mnd.go.kr
www.president.go.kr

Liste de quelques sites sud-coréens touchés entre les 4 et 8 juillet 2009 (d'après Pandalabs, 8 juillet 2009) [20]

1.3 Capacités nord-coréennes de guerre de l'information

Au début du mois d'avril 2009, la Corée du Nord procédait à des lancements de missiles longue portée. Pyongyang avertit alors le Conseil de sécurité des Nations Unies que si des sanctions étaient prises à son encontre, la Corée du Nord saurait réagir fermement **[21]**. La Corée du Nord serait-elle en mesure de mener, dans le cadre d'opérations agressives, des actions d'envergure et significatives dans le cyberespace ? En a-t-elle les moyens ?

La série d'incidents qui sont portés au compte de la Corée du Nord autorisent à penser que des capacités offensives puissent exister dans le pays, ou être mises à sa portée par des alliés.

Ces opérations consistent essentiellement en des actions de renseignement. La Corée du Sud n'a pas manqué d'insister sur le phénomène, dénonçant les nombreuses tentatives d'espionnage nord-coréennes (et chinoises) visant notamment les réseaux militaires (il y en aurait plus de 95 000 par jour !) **[22]**. Ces cyberattaques s'inscrivent dans le prolongement des manœuvres provocatrices de Pyongyang à l'encontre de Séoul et des États-Unis. Un document, du nom de code Oplan 5027, contenant des plans de guerre américains, aurait été volé par des hackers nord-coréens **[23]**. C'est ce que révélait en décembre 2009 le quotidien *Chosun Ilbo*. Ces documents fournissaient des informations sur la stratégie qu'adopteraient la Corée du Sud et les États-Unis confrontés à une guerre contre la Corée du Nord. Si l'incident fut comme à l'accoutumée minimisé par les victimes elles-mêmes, le quotidien s'inquiète de la facilité avec laquelle des intrusions semblent pouvoir être menées au sein des systèmes d'information de la Défense. Un nouvel incident de même nature, révélé à la presse en août 2010, serait survenu entre janvier et mars 2010 : des données sensibles auraient été dérobées sur les ordinateurs de l'armée sud-coréenne. Des *malwares* auraient été envoyés sur les ordinateurs de treize officiers sud-coréens afin de voler des informations secrètes **[24]**.

Le 16 octobre 2010, de nouveaux vols de documents sensibles furent révélés, ayant pris pour cible le ministère des Affaires étrangères et une fois encore le ministère de la Défense sud-coréen. Cette fois, on accuse la Chine **[25]**. Mais les documents concernés traitaient des relations entre la Chine et Pyongyang, ainsi que de la visite de Kim Jong-Il en Chine.

Un mois plus tard, peu avant la réunion du G20 à Séoul, les cyberattaques attribuées à la Corée du Nord augmentent. Le quotidien *Chosun Ilbo* affirme que les hackers nord-coréens sont localisés en Chine.

De tels incidents soulèvent des interrogations :

- sur le niveau de sécurité des forces armées sud-coréennes, tout d'abord, et notamment celles qui sont positionnées le long de la frontière. Les incidents sembleraient confirmer que la Corée du Nord est en mesure de tirer avantage du cyberespace pour mener des opérations de renseignement. La situation de l'armée sud-coréenne n'est toutefois pas un cas isolé. Elle n'est pas la seule qui soit victime d'intrusions, de vols et de pertes de documents et informations sensibles.
- La cybermenace nord-coréenne semble tirer quelques ressources de la Chine, d'où des unités exerceraient ou s'entraineraient. Pyongyang trouverait en Chine des ressources lui faisant défaut, des compétences particulières. Suite à l'affaire du Cheonan, des hackers ont pris pour cible les sites internet du gouvernement sud-coréen. La piste des hackers a mené les enquêteurs jusqu'à la Chine, où 120 serveurs auraient été utilisés pour cette opération.

Il est extrêmement difficile de se faire une idée précise, voire même simplement approximative, des capacités nord-coréennes en matière de guerre de l'information. Les informations dont nous disposons pour en parler sont essentiellement des déclarations qui émanent de diverses sources sud-coréennes, loin d'être impartiales en la matière. Les accusations portées et les informations

publiées le sont souvent sur la base de révélations faites par l'Agence de renseignement sud-coréenne **[26]** (en anglais, la NIS : *National Intelligence Agency*). Dans le pire des cas, les informations émanent de sources non identifiées ou non précisément nommées **[27]**.

Ainsi, sur cette base, attribue-t-on à la Corée du Nord :

- Très tôt, dès 1981, la formation de hackers, au sein du Collège du Mirim.
- La création en 1997 de la *Moranbong University*, pour former des experts en cyberespionnage **[28]**.
- La création d'une école de pirates informatiques, en février 2003 **[29]**.
- L'existence de l'unité 121, créée en 1998 au sein de l'armée, dont la mission serait d'accroître les capacités militaires en développant les capacités de guerre asymétrique et de cyberguerre.
- Un potentiel compris entre 500 et 1000 individus investis dans les cyberattaques de cibles étrangères, au sein de l'armée **[30]**. Ce groupe porterait le nom de code Lab 110 **[31]**. Il aurait reçu l'ordre, le 7 juin 2009, d'attaquer et détruire les réseaux sud-coréens **[32]**. Les chiffres relatifs au potentiel humain sont peu précis. En 2004, des articles font état de l'existence de 600 hackers en Corée du Nord **[33]**. D'autres observateurs parlent de 12 000 hackers et d'un budget équivalent à plus de 56 millions de dollars US, quand dans le même temps l'agence Yonhap News fait état de 100 hackers et non plus de 1 000, au sein de l'unité 121.
- Courant 2010, le général sud-coréen Bae Deuk-Shik déclarait que l'armée nord-coréenne disposait d'une unité militaire de hackers de haut niveau **[34]**.
- Des capacités de nuisance très élevées. Des analystes américains estiment que les unités de cyberpirates nord-coréens sont en mesure de paralyser l'*U.S. Pacific Command*, de causer des dommages importants dans les réseaux américains (vol de documents sensibles, attaques virales) **[35]**. Les sources de compétences en *hacking* se trouveraient dans les principales universités du pays : la *Pyongyang Automation University* (Mirim University) qui appartient à l'armée, la *Kim Chaek University of Technology*, la *Pyongyang University of Computer Technology*.

Quels que soient les chiffres avancés, la Corée du Nord apparait de fait comme un acteur non négligeable en matière de guerre de l'information. Certains s'avancent même à affirmer qu'elle est juste derrière la CIA en termes de capacités de piratage **[36]**. En 2009, elle serait au huitième rang mondial des cybermenaces **[37]**.Ces capacités sont des estimations émanant de la Corée du Sud, mais que les États-Unis ne peuvent confirmer **[38]**.

L'implication de la Corée du Nord dans le cyberespace repose notamment sur le développement de ses capacités de communication, ce qu'elle fait parfois avec l'aide de la communauté internationale. La Corée du Sud fournit ainsi son voisin du nord en fibre optique (45 km de fibres livrés au cours de l'année 2009 ; 37 km de câbles cuivrés avaient été envoyés entre 2002 et 2007), en terminaux pour réseaux fibre optique et en instruments de mesure. Or cette fibre optique serait détournée de l'usage initialement prévu (réseaux civils), au bénéfice des réseaux de l'armée, rendant ainsi par la même occasion la tâche des services de renseignement sud-coréens plus difficile **[39]**. La Corée du Nord aurait demandé de nouvelles livraisons, mises en attente par Séoul (courant 2010).

Il est intéressant de constater qu'un État qui ne dispose pas d'une infrastructure réseau internet très développée, dont la population n'est pas très connectée, ne dispose pas non plus d'une infrastructure de réseaux de télécommunications *high tech*, comparativement aux pays industrialisés, est supposée en mesure de lancer des opérations d'envergure contre la Corée du Sud, contre les États-Unis et autres adversaires, dont le Japon. Un pays faiblement connecté mais disposant d'alliances peut se poser en défi majeur. Ces capacités consolideraient la faisabilité de la mise en œuvre d'une stratégie ou au moins d'une tactique efficace du faible au fort. Mais si les capacités reposent sur la mise à disposition par un tiers de moyens, cela signifie encore que cette stratégie est fragile, car non autonome.

Si de son côté la Corée du Nord est en mesure d'attaquer des cibles à l'étranger, est-il tout aussi aisé d'attaquer des cibles nord-coréennes via les réseaux ? Le fait que le pays soit faiblement connecté ne facilite pas la tâche. Il y a peu de cibles accessibles. Le développement d'infrastructures réseau en Corée du Nord est sans doute dans l'intérêt des puissances étrangères.

2 La Corée du Sud

2.1 Le cyberespace sud-coréen

La Corée du Sud est l'un des pays les plus avancés au monde en matière de développement des technologies de l'information et de la communication. Le réseau TCP/IP a été inauguré le 15 mai 1982. Il s'agit de l'un des développements de l'Internet les plus précoces au monde. La première connexion relia un ordinateur du département des sciences informatiques de l'université nationale de Séoul, à un ordinateur de l'Institut coréen des technologies électroniques (aujourd'hui ETRI), à Gumi. En janvier 1983, un troisième ordinateur était connecté au KAIST (*Korea Advanced Institute of Science and Technology*), formant ainsi les bases d'un vrai réseau. En juillet 1986, les premières adresses IP pour la Corée furent assignées.

En 2010, la ville de Séoul arrivait en tête du classement annuel Rutgers concernant les municipalités les plus avancées en termes de e-government **[40]**. La ville été

AUTOUR DE L'ARTICLE...

▪ LES CRÉATIONS DE CYBER-UNITÉS DANS LE MONDE

La Corée du Nord, pas davantage que la Corée du Sud d'ailleurs, ne sont des cas isolés en matière de développement de capacités de guerre de l'information. Rappelons simplement les créations suivantes :

- **Mars 2009, le Royaume-Uni annonçait la création d'une unité dédiée à la cyberguerre [1], le Centre d'Opérations de Cyber Sécurité (*Cyber Security Operations Center* - CSOC), au sein du GCHQ. Le centre devait être doté d'un peu moins de 20 personnes. En janvier 2011, le général Sir David Richards exprimait le souhait de créer un cyber commandement, sur le modèle américain, pour protéger le pays des cyberattaques et être capable de lancer ses propres cyberattaques. Cette démarche s'inscrit dans le prolongement des conclusions du *Strategic Defense and Security Review* publié au Royaume-Uni en octobre 2010 [2].**
- **Août 2009 : 24ème Air Force, localisée sur la base de Lackland (Texas, États-Unis). Cyber unité de l'Air Force, pleinement opérationnelle depuis octobre 2010, et chargée de défendre les réseaux de l'Air Force contre les cyberattaques [3].**
- **2009 : l'Allemagne crée une unité de cyberguerre, installée dans la ville de Rheinbach, en 2009 [4].**
- **Mai 2010 : *US Cyber Command*.**
- **Octobre 2010 : création au sein de l'US Cyber Command, de l'*Army Cyber Command* (ARCYBER) qui centralise les moyens de l'*US Army* dans le domaine de la protection de ses réseaux.**
- **Janvier 2010 : la Corée du Sud crée un centre de commandement pour la cyberguerre, pour contrer d'éventuelles attaques de la Corée du Nord et de la Chine [5].**

également première en 2003, 2005 et 2007. En juin 2010, la Corée du Sud comptait quelque 39 440 000 internautes, soit 81,1 % de la population (48 636 068 habitants, selon le Bureau national du recensement) **[41]**.

Le nombre d'internautes n'a cessé d'augmenter au cours des dix dernières années : il représentait un peu moins de 40 % de la population en 2000, et 81 % en 2010. De fortes différences se font jour selon les tranches d'âge : 95 % des 6-29 ans se connectent quotidiennement, pour seulement 27 % des plus de cinquante ans. 70 % des citadins se connectent, contre seulement 46 % des populations rurales. En 2005, 75% des foyers connectés l'étaient en haut débit.

Depuis 2001 une loi controversée régule les contenus internet dans le pays. La Commission pour la sécurité de l'Internet coréen décide des sites qui doivent être bloqués. La même année, quelque 120 000 sites entrèrent sous le coup de ces nouvelles mesures et furent bloqués. Mais les rapports divergent quant au nombre de sites réellement bloqués et quant à la rigueur de l'application de la censure. Notamment, des tests effectués par l'*OpenNet Initiative* en 2006 concluaient que la censure n'est pas aussi stricte qu'on l'affirme généralement.

2.2 La cybersécurité

Les incidents avec la Corée du Nord ont été évoqués. Des tensions existent également avec le Japon, qui prennent forme dans le cyberespace au travers d'attaques de sites. Les exemples sont nombreux. Ne rappelons que les plus récents :

- En février 2010, un étudiant sud-coréen est assassiné en Russie. Des internautes japonais prennent le parti des meurtriers. Des hackers sud-coréens répliquent.
- Lors des jeux olympiques de Vancouver, en patinage artistique, un duel oppose la Corée du Sud et le Japon. La patineuse coréenne est accusée d'avoir payé les jurys pour remporter la victoire. Un site internet est alors ouvert pour créer une communauté d'internautes dont la mission consistera à lancer une cyberattaque contre le site de la télévision japonaise.

Ces affrontements entre communautés, entre citoyens, entre groupes de hackers, ne mettent pas en péril la stabilité nationale, ni fondamentalement la sécurité des réseaux, mais elles sont le reflet des tensions qui perdurent parfois entre les peuples, pour des raisons historiques.

Si l'on considère que la Corée du Sud est l'un des pays les plus avancés en matière de réseaux haut débit, et que les technologies de l'information y tiennent une place prépondérante, on ne peut écarter l'éventualité d'un scénario à l'estonienne visant le pays : pays ultra connecté, pays dépendant de ses systèmes d'information, pays ayant des adversaires immédiats, ... Les conditions sont réunies pour faire des systèmes d'information et donc de la société coréenne des cibles d'attaques importantes.

Les tensions politiques entre la Corée du Sud et ses voisins de la région se traduisent dans le cyberespace essentiellement par des vagues de défigurations de site **[42]**, mais aussi des intrusions dans les systèmes à des fins de renseignement. Des attaques perturbatrices, destructrices, ne semblent pas avoir été subies.

Les attaques subies en provenance de la Corée du Nord, dont nous avons cité quelques exemples dans le premier chapitre de l'article, ainsi que celles pouvant émaner du Japon, de la Chine et du reste du monde, démontrent la relative fragilité de la cybersécurité sud-coréenne. Comme partout ailleurs dans le monde, s'imposent le discours sur la cybersécurité, son caractère impératif, les besoins financiers qui lui sont associés.

Car si aujourd'hui l'adversaire majeur n'a pas les moyens de porter un coup fatal, ce sera peut-être le cas demain. Les intentions de Pyongyang sont sans ambiguïté et les moyens auxquels le régime pourrait recourir ne s'embarrasseront probablement pas des règles du droit international.

Les multiples incidents de sécurité subis par les forces armées sud-coréennes inquiètent. Ces incidents sont possibles en raison de l'absence de protocoles de sécurité interdisant, par exemple, aux militaires de recourir à leurs ordinateurs personnels pour traiter les dossiers sensibles, ou encore d'utiliser les moyens de l'armée à des fins personnelles. Ainsi, en 2005, un lieutenant de l'armée sud-coréenne a-t-il téléchargé un rapport confidentiel sur une clé USB, puis l'a consulté sur son ordinateur personnel, où il fut piraté avant d'être divulgué sur la Toile.

La même année, de nouveaux documents militaires classifiés furent dérobés et diffusés sur le net. En 2006, un lieutenant exposa des documents classés « confidentiel défense », alors qu'il téléchargeait une vidéo sur son ordinateur. En août 2009, les responsables de la sécurité du département de la Défense lancèrent une alerte pour informer d'attaques de hacking lancées contre l'Intranet des militaires. Malgré l'alerte, une partie de l'Intranet fut paralysée plusieurs heures et un millier de documents classifiés furent piratés. En octobre 2009, des hackers supposés appartenir à une unité de cyberguerre nord-coréenne ont dérobé des mots de passe dans l'ordinateur d'un haut responsable militaire sud-coréen et les ont ensuite utilisés pour dérober des informations classifiées au sein de l'Institut National de la Recherche sur l'Environnement, parmi lesquelles les noms de 700 compagnies ou entités étatiques fabriquant des produits chimiques toxiques **[43]**.

Le niveau de sécurité du cyberespace sud-coréen est comme dans de nombreux pays montré du doigt pour ses lacunes encore importantes. Lors d'une audition à l'Assemblée Nationale, le 22 octobre 2009, il fut révélé que des pans entiers des réseaux de communication de 109 agences dépendant de 10 ministères n'étaient pas préparés à faire face à des attaques DDoS. La Bourse de Séoul, l'aéroport international d'Incheon, l'Institut de l'Information géographique nationale (qui gère les données GPS), font partie des institutions non préparées - ou insuffisamment préparées - à faire face à de telles attaques **[44]**.

De nombreux niveaux de sécurité sont pourtant implémentés dans le pays : rappelons l'existence du CERT, du NCSC (*National Cyber Security Center*) **[45]**, plate-forme créée en février 2004, qui coordonne les secteurs public et privé, civil et militaire dans la lutte contre les cybermenaces. Cette instance travaille sous la tutelle de l'agence du renseignement national (*National Intelligence Service*, NIS).

La Corée du Sud organise également sa défense au travers de coopérations avec des partenaires internationaux. C'est ainsi qu'elle a signé en mai 2009 un accord avec les États-Unis sur la cyberdéfense **[46]**.

AUTOUR DE L'ARTICLE...

- **2010 : la Suisse projetait de créer deux unités dédiées à la cyberguerre au sein du Centre des Opérations Électroniques de l'Organisation de Soutien au Commandement des Forces Armées.**
- **Janvier 2011 : le ministère de la Défense estonien proposait la constitution d'unités de cyberdéfense sur la base des expertises présentes dans les communautés d'informaticiens du pays (organisations de volontaires) [6].**

La liste ne cessera de s'allonger. Israël serait dotée de sa célèbre Unité 8200, basée dans le désert du Negev. On prête à l'Iran la volonté de développer des capacités de cyberguerre. La France renforce et élargit les attributions de l'ANSSI par décret du 11 février 2011 [7]. La Chine, depuis longtemps déjà, est supposée s'être dotée d'unités spécialisées dans le domaine. L'OTAN compte à son niveau 15 centres d'excellence de cyberdéfense coopérative (CCD COE). Il s'agit de centres de prévention des attaques, de cyberdéfense.

Le qualificatif de « cyber-unité », souvent utilisé pour parler de ces structures, recouvre des réalités fort différentes : militaires, civiles, à vocation strictement défensive, à vocation également offensive, etc. Les États sont-ils uniquement en train de renforcer leurs protections, ou de fourbir leurs nouvelles armes ?

Notes

[1] http://www.theregister.co.uk/2009/11/12/csoc_date/

[2] http://www.direct.gov.uk/prod_consum_dg/groups/dg_digitalassets/@dg/@en/documents/digitalasset/dg_191634.pdf?CID=PDF&PLA=furl&CRE=sdsr

[3] http://www.signonsandiego.com/news/2010/oct/01/air-force-declares-cyberwarfare-unit-operational/#

[4] http://benmazzotta.wordpress.com/2009/02/11/new-german-cyber-warfare-unit/

[5] http://www.theregister.co.uk/2010/01/12/korea_cyberwarfare_unit/

[6] http://www.defensenews.com/story.php?i=5556484

[7] http://www.linformaticien.com/Actualit%C3%A9s/tabid/58/newsid496/10342/anssi-naissance-du-cybercommand-francais/Default.aspx

2.3 Capacités sud-coréennes de guerre de l'information

Face aux menaces persistantes du Nord, en matière de cyberattaques comme d'attaques conventionnelles, Séoul déploie de nouveaux moyens de défense. Suite aux cyberattaques subies en juillet 2009, et cédant partiellement à la tentation de la protection illusoire par la forteresse, des « bunkers » numériques **[47]** ont temporairement été déployés pour protéger les moyens du gouvernement et l'économie nationale des attaques DDoS. Chaque incident est ainsi prétexte à un nouveau déploiement et renforcement des capacités.

En janvier 2010, la Corée du Sud a mis sur pied une nouvelle unité de cyberguerre **[48]**, chargée de contrer les agressions qui viennent le plus souvent de Chine et de Corée du Nord, et en particulier les quelque 95 000 agressions quotidiennes recensées contre les réseaux militaires. La création de cette unité par les autorités coréennes est présentée comme une réaction aux attaques coordonnées contre les ordinateurs et réseaux du pays. Des attaques menées contre les systèmes d'information du pays à l'occasion du G20 qui s'est tenu à Séoul en 2010 vinrent appuyer et justifier cette décision.

Les compétences en matière offensive ne sont bien sûr jamais mises en avant. Seul le caractère défensif des unités est affiché. Les autorités justifient la création de cette nouvelle unité de cyberdéfense comme une réaction légitime à des attaques subies et à des menaces bien identifiées. Le processus est identique à celui de nombreux autres pays : c'est toujours par nécessité sécuritaire, dans la position de la victime agressée qui exerce un droit légitime de défense, que sont instituées les entités nouvelles. On argumente sur des faits (concrets, ou présentés comme tels) qui légitiment l'existence et l'urgence à la fois de ces nouvelles organisations.

Le processus répond aussi à la logique d'une course aux armements, mais une course imposée par l'adversaire, par la menace. Ce processus qui consiste à renforcer un acteur affaiblit automatiquement les autres, lesquels seront amenés à renchérir. Il est donc probable que les moyens nord-coréens et ceux des pays de la région soient augmentés dans les prochains mois, y compris ceux de pays alliés. Ces développements capacitaires constituent pour Pyongyang un argument : si le Sud développe ses moyens de cyberguerre, le Nord doit en faire de même.

Il n'est pas assuré que les nouvelles unités soient davantage en mesure de contrer les attaques. La question de l'efficacité des unités de défense est rarement l'objet des bilans : les analyses portent plus généralement sur le niveau croissant des attaques.

Les développements récents s'inscrivent dans une volonté de longue date. La Corée du Sud n'a jamais vraiment caché sa volonté de développement de capacités de guerre de l'information. En 2000, un rapport du ministère de la Défense révélait que l'armée disposait de 177 centres de formation informatique. Bien sûr, tous n'ont pas vocation à former des cybercombattants, mais tous en ont potentiellement les capacités.

La question se pose également du niveau d'interopérabilité entre l'organisation du système sud-coréen et celle des États-Unis, dans le cadre des relations particulières qui lient les deux pays en matière de défense.

3 L'affaire du Cheonan

Une commission d'enquête internationale confirma le 20 mai 2010 que la Corée du Nord était bien à l'origine de l'attaque menée contre le navire de guerre sud-coréen, le Cheonan, le 26 mars de la même année. Le navire avait été torpillé, l'attaque se soldant par 46 victimes sud-coréennes. L'incident a mobilisé la communauté internationale de nombreux mois durant.

Cet incident grave n'était ni le premier, ni le dernier d'une longue série. Entre les deux pays, les tensions se traduisent régulièrement par des actes d'agression armés depuis la fin de la guerre de Corée (1950-1953). En 1998, un semi-submersible nord-coréen fut détruit par les forces sud-coréennes. En 1999, la Corée du Sud ouvre le feu sur des patrouilleurs nord-coréens qui ont pénétré dans une zone de pêche revendiquée par Séoul ; la même année, l'armée sud-coréenne ouvre le feu sur un patrouilleur nord-coréen. En 2002, la Corée du Nord coule un patrouilleur sud-coréen. Fin novembre 2010, la Corée du Nord procède à des bombardements contre l'île de Yeonpyeong.

Les tensions entre les deux pays s'expriment également au travers de manœuvres de propagande et autres opérations d'information.

L'affaire du Cheonan est intéressante à plusieurs titres. Tout d'abord parce qu'elle n'a pas été accompagnée de « vagues » de cyberattaques, attribuables à un acteur étatique ou à des hacktivistes des pays impliqués directement ou indirectement. Les faits restent au contraire, au moins officiellement, relativement isolés. Une attaque DDoS fut lancée depuis 120 serveurs localisés en Chine courant juin 2010 **[49]**, en lien avec l'affaire du Cheonan. Un site internet du gouvernement sud-coréen a été attaqué **[50]**. Nombre de cyberattaques ont pu être rapportées dans les deux Corées au cours de l'année 2010, sans que l'on puisse précisément en relier un nombre à l'affaire du Cheonan. Peu de temps après que la commission d'enquête internationale eut conclu à l'attaque nord-coréenne du Cheonan, des hackers nord-coréens, selon les services de renseignement de Séoul, auraient volé des identifiants **[51]** pour poster des rumeurs dans des blogs sud-coréens. Ces rumeurs accusaient les autorités de Séoul de fabriquer de toutes pièces les preuves tendant à accuser Pyongyang **[52]**. Les autorités y étaient qualifiées de traitres. Les contenus des « post » reprenaient les termes d'un communiqué publié par la Commission nationale de Défense nord-coréenne,

posté sur le site officiel Uriminzokkiri. Le site internet de *Free North Korea Radio* (radio basée à Séoul) a enregistré une hausse importante du nombre de messages qualifiant l'incident du Cheonan de « montage de toutes pièces ». En juin 2010, les militaires sud-coréens ont appelé à maîtriser les auteurs de ces rumeurs qui remettaient en question les versions officielles. Les remises en question sont qualifiées de cyberterrorisme **[53]**. Suivant cette demande, la police sud-coréenne a lancé une enquête pour trouver les auteurs des rumeurs. Les autorités craignent que des citoyens ne se montrent sensibles aux arguments de la propagande de Pyongyang, laquelle ne passe pas uniquement par le biais de l'Internet, mais simplement parfois par des campagnes d'envois de fax, comme ce fut le cas fin mai 2010. Au cours de cette campagne, le nord envoya des fax à de nombreuses organisations sud-coréennes, dont l'association pour la réunification nord-sud **[54]**. Les autorités s'empressent d'interdire les manifestations de sympathie à l'égard du nord. En août 2010, le nouveau compte Twitter nord-coréen a été bloqué en raison de l'information illégale qu'il était susceptible de diffuser **[55]**. Ont également été bloqués 64 sites nord-coréens ou pro-Corée du Nord. Fin novembre 2010, la division de lutte contre la cybercriminalité de la police sud-coréenne a procédé à l'arrestation de 22 personnes accusées d'avoir diffusé des rumeurs selon lesquelles la Corée du Sud aurait attaqué la Corée du Nord, et avoir déclaré elles-mêmes appartenir à l'armée sud-coréenne **[56]**. La police recherchait également les membres d'un groupe sévissant sur le net, se faisant appeler « Cyber quartier général pour la défense du peuple », qui a posté sur Naver - le principal portail coréen - des messages à la gloire de l'attaque nord-coréenne contre Yeonpyeong le 24 novembre 2010. La censure et les poursuites s'inscrivent dans le cadre de la loi sur la sécurité, qui interdit notamment tout contact non autorisé avec la Corée du Nord, et toute manifestation de sympathie à l'égard de la Corée du Nord.

Dans les jours qui suivirent l'incident du Cheonan, les militaires sud-coréens ont également pointé du doigt les limites des capacités des technologies de l'information, et spécifiquement en matière de surveillance satellitaire dans le cadre de la protection nationale. Le Cheonan aurait-il pu être torpillé par un sous-marin que les satellites

AUTOUR DE L'ARTICLE...

▪ IDENTIFICATION, ATTRIBUTION, ACCUSATIONS, ...

Le contexte coréen est spécifique. Les deux Corées ne sont pas officiellement en situation de paix. De fait, l'ennemi est tout désigné, et les incidents subis de part et d'autre sont quasi-automatiquement attribués au voisin d'en face. En matière de cyberattaques, il est donc naturel que les soupçons se portent immédiatement sur l'adversaire désigné. Mais du soupçon à la preuve, à l'attribution, nous savons que le chemin est grand en matière informatique, en Corée comme partout ailleurs. Identification, attribution, accusations portées contre les coupables : de manière générale, quelles sont les méthodes utilisées et les attitudes adoptées par les victimes des cyberattaques ?

Il y a tout d'abord celles qui recourent aux outils « forensics ». Ces analyses techniques permettent de tracer les attaques. Le travail d'enquête s'avère toutefois relativement long à mener. Il n'est pas inefficace : à preuve les quelques cybercriminels placés derrière les verrous chaque année. Mais il a ses limites (techniques et judiciaires notamment) : il demeure le plus souvent difficile sinon impossible de donner l'identité de l'auteur de l'attaque. On ne peut jamais affirmer que la source identifiée soit bien le maillon ultime de la chaîne que l'on a remontée. Quand bien même y arrive-t-on, cette information ne permet pas d'attribuer assurément l'opération (dire qui a vraiment agi, qui est le commanditaire).

L'absence de certitudes n'empêche pas nombre de victimes de porter des accusations : l'attaque a été menée par la Chine, par la Russie, par la Corée du Nord, etc. On procède par raccourcis : le serveur qui a permis l'attaque est sur le territoire de X, donc X est coupable. L'accusateur considère que le pays qui a fourni une partie des ressources de l'attaque est le coupable. Il se fonde sur des convictions : il déduit de la nature de la cible, du contexte, du moment, que l'auteur est X. Il pense d'ailleurs rarement à regarder chez lui en premier lieu. Les accusations portent souvent contre une entité « pays ». Mais accuser « la Chine », est-ce accuser le gouvernement en place, l'armée chinoise, les services de renseignement, des entreprises, des délinquants, le peuple, ou tout à la fois ? Or les mots. Or affirmer que « le pays X » est coupable, renvoie à des questions de relations internationales, aux rapports de force entre États, évoque la volonté manifeste du gouvernement d'un État de mener des opérations spécifiques contre un autre État. Dire « des hackers de nationalité X sont coupables » peut simplement renvoyer à des questions de cybercriminalité, appeler à un dialogue international sur la question, placer les autorités des divers États devant leurs responsabilités face à la communauté internationale, etc.

Une méthode, tout aussi aléatoire que le recours aux simples convictions, est celle qui s'appuie sur la question « à qui profite le crime ». La réponse est supposée fournir une idée assez précise du nom du coupable. La méthode n'en est bien sûr pas une, elle se fonde elle aussi sur des soupçons, des indices faibles, des opinions, la perception d'un contexte.

D'autres États peuvent choisir quant à eux de garder secret le nom du coupable, quand bien même le connaîtraient-ils. Tout ceci afin de ne pas froisser un partenaire privilégié.

d'observation n'auraient su voir ? Il semble en effet que les sous-marins nord-coréens soient coutumiers de l'intrusion dans les eaux territoriales sud-coréennes. En 1998, l'un d'entre eux fut même pris dans des filets de pêche. L'analyse de son carnet de bord révéla qu'il avait pénétré les eaux territoriales à plusieurs reprises au cours des derniers mois. Or aucun satellite ne l'avait détecté. Les militaires expliquent que les satellites détectent les sous-marins, mais que la surveillance ne peut être faite 24h/24. Des images sont prises de 1 à 3 fois par jour **[57]**, ce qui n'est manifestement pas suffisant pour assurer la surveillance du territoire.

L'affaire du Cheonan est également intéressante par les analogies qu'elle permet d'établir avec l'environnement des cyberattaques. L'attaque qui a coulé le navire coréen a touché sa cible par surprise. Le milieu d'où elle est arrivée assurait une invisibilité parfaite à l'agresseur. Ce scénario a rendu l'attribution de l'attaque fortement difficile, même si des soupçons pesaient fortement sur l'identité de l'auteur. Nous retrouvons les éléments principaux des cyberattaques :

- un environnement qui permet de se dissimuler.
- un environnement qui rend l'attribution difficile à impossible.
- un processus d'accusations sans preuves irréfutables (sur la base d'indices faibles et de convictions potentiellement sources d'erreurs), alliances (agresseur et victime ont fait jouer leurs alliances, leurs partenariats dans cette affaire, pour y trouver des moyens d'attaque, des soutiens politiques, etc.), rejet systématique des accusations (argumentant sur la base de l'absence de preuves).
- un grand nombre d'acteurs impliqués. Dans l'affaire du Cheonan, la communauté internationale a été mobilisée, comme témoin, comme enquêteur, comme force de proposition de solutions.
- un incessant rapport de forces.

Cette affaire nous permet de rappeler que l'un des aspects essentiels du problème auquel sont confrontées les victimes d'attaques - à savoir l'attribution des attaques - n'est pas spécifique au cyberespace.

Conclusion

Les tensions entre le nord et le sud se sont naturellement reportées dans le cyberespace, mais ce déplacement ne signifie pas que le cyberespace est un substitut et que les affrontements dans le monde réel se sont effacés. Nous sommes dans une configuration de crise permanente, de guerre même - puisqu'officiellement la paix n'a pas été conclue entre les deux États – qui trouve dans le cyberespace un lieu où les belligérants peuvent prolonger leur action, à la poursuite de buts politiques. La Corée du Nord, nettement moins à la pointe que son voisin du sud en termes de nouvelles technologies de l'information, sait toutefois utiliser ce déséquilibre à son avantage. Car si la Corée du Nord peut atteindre des cibles au sud ou chez les alliés du sud, le contraire n'est pratiquement pas possible, en tous les cas très limité, en raison de la faible étendue des réseaux dans le pays.

La situation en Corée du Sud et Corée du Nord, au regard des capacités de cyberaffrontements, n'est que trop rarement analysée et débattue. Pourtant, il ne fait aucun doute que les alliances régionales possibles, les développements capacitaires des deux acteurs, pourront faire du cyberespace un de leurs champs de bataille. Ceci d'autant plus facilement au regard de la communauté internationale, que cette dernière ne dispose pas d'un cadre juridique permettant de réglementer les attaques. Le cyberespace offre des atouts indéniables à Pyongyang pour poursuivre son action de déstabilisation du sud. Dès lors que Pyongyang sera dotée des capacités nécessaires, il est probable qu'elle choisira de mener des opérations nettement plus agressives et destructrices que la simple défiguration de sites ou quelques attaques sporadiques de type DDoS ou intrusions à des fins d'espionnage. Les provocations permanentes entre les deux Corées, qui se soldent par des attaques militaires, des dizaines de morts chaque année des deux côtés, sont un exemple du niveau de violence que deux États sont en mesure d'accepter. Nous pouvons toutefois nous interroger sur le seuil d'acceptabilité (tolérance, résilience) de l'un et l'autre, dans l'éventualité où ils seraient confrontés à des cyberattaques d'envergure. Un tel acte pourrait-il justifier la rupture du cessez-le-feu, être le degré ultime qu'il ne fallait pas franchir ? Quelle serait la réaction de Séoul et de la communauté internationale si la Corée du Sud devait subir le même sort que l'Estonie en 2007 ? ■

NOTES

[1] www.brookings.edu/views/papers/fellows/oh20040601.pdf

[2] http://news.yahoo.com/s/pcworld/20100818/tc_pcworld/uspushesnorthkoreaontwitterfreedom

[3] http://rt.com/Top_News/2010-03-01/north-korea-cyber-weapon.html

[4] http://ashen-rus.livejournal.com/4300.html

[5] www.uriminzokkiri.com/2010/index.php et http://twitter.com/uriminzok

[6] http://www.rfi.fr/asie-pacifique/20110120-guerre-hackers

[7] http://pics.livejournal.com/ashen_rus/pic/0003fcpe/

[8] www.scribd.com/doc/15078953/Cyber-Threat-Posed-by-North-Korea-and-China-to-South-Korea-and-US-Forces-Korea

[9] http://news.softpedia.com/news/South-Korean-Military-Equipment-Development-Secrets-Compromised-by-Hackers-94876.shtml

[10] http://english.chosun.com/site/data/html_dir/2009/11/04/2009110400775.html

[11] www.foxnews.com/story/0,2933,530900,00.html

[12] www.timesonline.co.uk/tol/news/world/asia/article6667440.ece

[13] www.guardian.co.uk/technology/2009/jul/15/hackers-internet-attack

[14] www.huffingtonpost.com/2009/07/11/north-korea-army-lab-110-_n_229986.html

[15] www.foxnews.com/story/0,2933,530645,00.html

[16] Selon l'entreprise vietnamienne BKis, qui affirme également que le serveur maître se trouvait au Royaume-Uni.

[17] Selon une estimation de l'expert Joe Stewart, de la société SecureWorks.

[18] http://english.chosun.com/site/data/html_dir/2009/11/02/2009110200788.html

[19] http://english.chosun.com/site/data/html_dir/2010/06/03/2010060302076.html

[20] Liste des sites attaqués, en date du 8 juillet 2009, publiée sur http://pandalabs.pandasecurity.com/ddos-attacking-us-and-south-korea-government-sites/

[21] http://defensetech.org/2009/04/20/north-korea-poised-for-cyber-salvo/

[22] www.foxnews.com/story/0,2933,530900,00.html

[23] www.guardian.co.uk/world/2009/dec/18/north-south-korea-hackers

[24] http://techinfo.co.in/wordpress/hackers-steal-army-secrets-from-south-korea

[25] http://techinfo.co.in/wordpress/chinese-hackers-seized-sk-defense-secrets

[26] www.foxnews.com/story/0,2933,531637,00.html

[27] www.foxnews.com/story/0,2933,531637,00.html

[28] www.strategypage.com/htmw/htiw/articles/20100118.aspx

[29] www.wired.com/politics/law/news/2003/06/59043

[30] http://english.chosun.com/site/data/html_dir/2009/07/10/2009071000588.html?FORM=ZZNR3

[31] www.huffingtonpost.com/2009/07/11/north-korea-army-lab-110-_n_229986.html

[32] www.foxnews.com/story/0,2933,531637,00.html

[33] www.securityfocus.com/news/9649

[34] http://www.allheadlinenews.com/articles/7020646606#ixzz1EQ4h2D60

[35] www.asiamedia.ucla.edu/article.asp?parentid=25233

[36] http://english.chosun.com/site/data/html_dir/2009/07/10/2009071000588.html?FORM=ZZNR3

[37] http://defensetech.org/2009/04/20/north-korea-poised-for-cyber-salvo/

[38] http://defensetech.org/2009/04/20/north-korea-poised-for-cyber-salvo/

[39] http://english.chosun.com/site/data/html_dir/2010/04/14/2010041400876.html

[40] http://news.rutgers.edu/medrel/news-releases/2010/07/seoul-and-prague-ach-20100713

[41] www.internetworldstats.com/asia/kr.htm

[42] Les défigurations de sites recensées sur la base zone-h.org n'ont que peu de rapports avec la situation politique régionale. La base n'est pas exhaustive, et n'offre sans doute qu'une vision partielle de la réalité du phénomène en Corée du Sud.

[43] http://english.chosun.com/site/data/html_dir/2009/11/02/2009110200788.html

[44] http://english.chosun.com/site/data/html_dir/2009/11/02/2009110200788.html

[45] www.ncsc.go.kr

[46] http://www.defensenews.com/story.php?i=4072075

[47] www.zdnet.co.uk/news/security-management/2010/11/18/south-korea-builds-digital-bunkers-against-ddos-attacks-40090902/

[48] www.theregister.co.uk/2010/01/12/korea_cyberwarfare_unit/

[49] www.earthtimes.org/articles/news/328218,south-korea-government-website.html

[50] www.businessweek.com/news/2010-06-10/south-korea-says-cyber-attacks-came-from-china-sites-update1-.html

[51] http://english.chosun.com/site/data/html_dir/2010/06/02/2010060200550.html

[52] http://joongangdaily.joins.com/article/view.asp?aid=2921288

[53] http://english.yonhapnews.co.kr/news/2010/06/08/0200000000AEN20100608005500315.HTML

[54] http://english.chosun.com/site/data/html_dir/2010/06/02/2010060200550.html

[55] http://www.foxnews.com/scitech/2010/08/17/south-korea-bans-north-korean-twitter/#

[56] http://www.allheadlinenews.com/articles/7020646606

[57] http://english.chosun.com/site/data/html_dir/2010/04/06/2010040601060.html

DESCRIPTION ALGÉBRIQUE DE L'ADVANCED ENCRYPTION STANDARD

Michel Dubois – Chercheur doctorant au sein du laboratoire de Cryptologie et de Virologie Opérationnelle de l'ESIEA

mots-clés : CRYPTOGRAPHIE / AES / ALGÈBRE / CRYPTANALYSE

Afin de contrer les méthodes de cryptanalyse usuelles, basées sur des méthodes statistiques, les concepteurs de l'Advanced Encryption Standard (AES) lui ont donné une forte architecture algébrique. Ce choix élimine brillamment toute possibilité d'attaque statistique, mais il pourrait également être sa faiblesse, en le rendant susceptible d'être compromis par ces mêmes structures algébriques.

1 Introduction

Depuis le 26 novembre 2001, l'algorithme de chiffrement par bloc « Rijndael », dans sa version 128 bits, est devenu le successeur du DES sous le nom d'*Advanced Encryption Standard* (AES).

Issu d'un concours lancé par le *National Institute of Standards and Technology* (NIST) en 1997, Rijndael **[7]** a franchi toutes les étapes de sélection et est maintenant un standard fédéral américain enregistré sous le numéro FIPS 197 **[9]**. Inscrit sur la suite B de la *National Security Agency* (NSA)[1], l'AES a vocation, promu par le gouvernement américain, à devenir un standard pour l'échange sécurisé des informations classifiées, aux Etats-Unis et entre les Etats-Unis et leurs partenaires. De fait, il est aujourd'hui l'algorithme symétrique de chiffrement par bloc le plus couramment utilisé en Occident[2].

Ses concepteurs, Joan Daemen et Vincent Rijmen, ont utilisé des outils algébriques pour fournir à leur algorithme un niveau de garantie inégalé contre les techniques de cryptanalyse statistique standards.

De récents travaux tendent à montrer que ce qui est censé faire la robustesse de l'AES pourrait se révéler être son point faible. En effet, selon ces études, cryptanalyser l'AES pourrait se « résumer » à résoudre un système d'équations quadratiques symbolisant la structure du chiffrement de l'AES.

1.1 Description de l'AES

L'AES un est algorithme de chiffrement symétrique par bloc. Il chiffre et déchiffre des blocs de données à partir d'une seule clé.

Contrairement au DES, basé sur un réseau de Feistel, l'AES s'appuie sur un réseau de substitution et de permutation, *SP-network* ou réseau SP. Ce dernier est constitué de fonctions de substitution non linéaire appelées *S-Box* ou boîtes S et de fonctions de permutation linéaire que l'on pourrait appeler par analogie *P-Box* ou boîtes P. Chaque boîte prend un bloc de texte et la clé en entrée et fournit un bloc de texte chiffré en sortie. Le cheminement de l'information dans une suite définie de plusieurs P-Box et S-Box forme un tour.

Ce mécanisme met en œuvre les principes de diffusion et de confusion développés par Shannon **[13]**. L'objectif de la diffusion est de dissiper la redondance statistique d'un texte en clair dans le texte chiffré. Ce sont les opérations de permutation qui permettent de garantir la diffusion. L'objectif de la confusion est de rendre difficile la relation entre le texte clair, la clé et le texte chiffré. La confusion est obtenue par des substitutions choisies avec soin.

Historiquement, l'AES a deux prédécesseurs. Le premier est l'algorithme de chiffrement par bloc Shark **[11]** publié en 1996 par Vincent Rijmen, Joan Daemen, Bart Preneel, Anton Bosselaers et Erik de Win. Shark utilise des blocs de 64 bits et une clé de 128 bits. À l'instar de l'AES, il utilise un SP-network avec six tours. Le deuxième

algorithme s'appelle Square, il a été publié en 1997 par Joan Daemen et Vincent Rijmen. Il utilise un SP-network avec huit tours et travaille sur des blocs de 128 bits et une clé de 128 bits également.

1.2 Fonctionnement de l'AES

Les entrées et sorties de l'AES sont des blocs de 128 bits et la longueur de la clé peut-être 128, 192 ou 256 bits. L'unité de base de l'algorithme est l'octet. Les blocs de données fournis en entrée sont transformés en tableaux de quatre colonnes et quatre lignes, chaque case contenant un octet, soit 4*4*8=128 bits par tableau.

Au début des opérations de chiffrement et de déchiffrement, le bloc d'octets en entrée est copié dans le tableau d'état[3]. Les opérations de chiffrement et de déchiffrement sont effectuées sur ce tableau, puis le résultat est copié dans un tableau de sortie.

1.2.1 Opérations de chiffrement

Les opérations de chiffrement (voir figure 1) s'appuient sur quatre fonctions prédéfinies : **AddRoundKey**, **SubBytes**, **ShiftRows** et **MixColumns**. Chacune de ces fonctions est exécutée sur le tableau d'état. Le cycle de chiffrement comprend une transformation initiale, des tours intermédiaires et un tour final.

La transformation initiale consiste à appliquer la fonction **AddRoundKey** au tableau d'état. Les tours intermédiaires exécutent, dans l'ordre, les fonctions **SubBytes**, **ShiftRows**, **MixColumns** et **AddRoundKey** sur le tableau d'état. Le tour final diffère des tours intermédiaires par la suppression de la fonction **MixColumns** dans le cycle des transformations.

Le nombre de tours dans l'AES est dépendant de la taille de la clé. Ainsi, pour une clé de 128 bits, le nombre de tours est 10, de même, nous avons 12 tours pour une clé de 192 bits et 14 tours pour une clé de 256 bits.

1.2.1.1 SubBytes

La transformation **SubBytes** effectue une substitution d'octets non linéaire en utilisant une fonction de substitution représentée sous forme de table (S-Box). Ainsi, chaque élément $S_{r,c}$, $0 \leq r, c \leq 3$ du tableau d'état est substitué selon la fonction suivante :

$$S_{r,c} \mapsto \text{S-Box}[S_{r,c}]$$

1.2.1.2 ShiftRows

La transformation **ShiftRows** effectue une rotation des lignes du tableau d'état. Ainsi, chaque élément $S_{r,c}$ du tableau d'état est remplacé selon la fonction :

$$S_{r,c} \mapsto S_{r,c-r}$$

1.2.1.3 MixColumns

La transformation **MixColumns** agit sur les colonnes du tableau d'état. Chaque colonne est considérée comme un polynôme sur sur le corps fini à 256 éléments $GF(2^8)$ et est multipliée, modulo x^4+1, par le polynôme :

$$a(x) = \{0x03\}x^3 + \{0x01\}x^2 + \{0x01\}x + \{0x02\}$$

Cette opération revient à multiplier chaque colonne du tableau d'état par une matrice carrée d'ordre 4 selon l'équation suivante :

$$\begin{pmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{pmatrix} \mapsto \begin{pmatrix} 0x02 & 0x03 & 0x01 & 0x01 \\ 0x01 & 0x02 & 0x03 & 0x01 \\ 0x01 & 0x01 & 0x02 & 0x03 \\ 0x03 & 0x01 & 0x01 & 0x02 \end{pmatrix} \begin{pmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{pmatrix}$$

1.2.1.4 AddRoundKey

Dans la transformation **AddRoundKey**, l'AES ajoute, par un **XOR**, la clé du tour au tableau d'état. Cette opération s'effectue colonne par colonne selon la fonction équivalente :

$$S_{r,c} \mapsto S_{r,c} + K_{i,4c+r}$$

avec $S_{r,c}$ un élément du tableau d'état, $0 \leq i \leq 13$ le numéro du tour et K_i la clé du tour obtenue par la fonction d'expansion de l'AES.

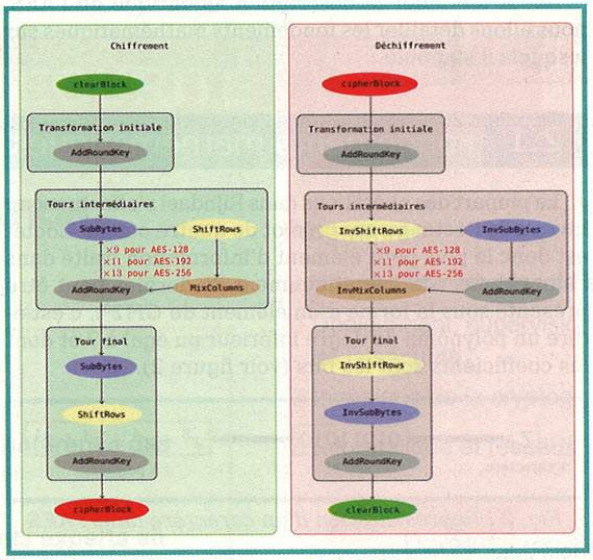


Fig. 1 : Les processus de chiffrement et de déchiffrement de l'AES

En conclusion, la fonction **SubBytes** effectue une substitution sur chaque octet du tableau d'état, elle correspond à la couche de substitution. La succession des fonctions **ShiftRows** et **MixColumns** correspond à la couche de diffusion au travers du tableau d'état. Enfin, la fonction **AddRoundKey** correspond à l'introduction de la clé de chiffrement. Ainsi, l'AES répond aux grands principes d'un algorithme de chiffrement moderne, en garantissant la diffusion et la confusion de l'information dans le message chiffré.

1.2.2 Opérations de déchiffrement

Le déchiffrement (voir figure 1) est réalisé en effectuant les opérations inverses des quatre fonctions de chiffrement, dans l'ordre inverse.

Ainsi, chaque fonction utilisée dans les opérations de chiffrement dispose de sa fonction inverse, utilisée pour le déchiffrement : **InvShiftRows**, **InvSubBytes** et **InvMixColumns**. La fonction **AddRoundKey** reste inchangée. À l'instar du chiffrement, le processus de déchiffrement comprend une transformation initiale, des tours intermédiaires et un tour final.

La transformation initiale consiste à appliquer la fonction **AddRoundKey** au tableau d'état. Les tours intermédiaires exécutent, dans l'ordre, les fonctions **InvShiftRows**, **InvSubBytes**, **AddRoundKey** et **InvMixColumns** sur le tableau d'état. Le tour final diffère des tours intermédiaires par la suppression de la fonction **InvMixColumns** dans le cycle des transformations.

2 L'architecture algébrique de l'AES

Avant d'étudier la description algébrique de l'AES, nous allons détailler les fondements mathématiques sur lesquels il s'appuie.

2.1 Fondements mathématiques

La plupart des opérations dans Rijndael s'effectuent au niveau de l'octet ou sur un mot de quatre octets. L'octet est donc le plus petit élément d'information traité dans l'algorithme. Mathématiquement ce dernier peut être présenté sous la forme d'un élément de $GF(2^8)$, c'est-à-dire un polynôme de degré inférieur ou égal à 7 et dont les coefficients sont les bits (voir figure 2).

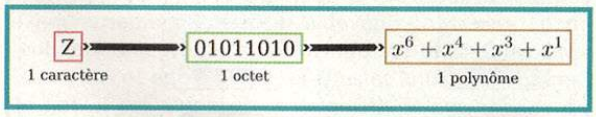


Fig. 2 : Représentation d'un caractère dans l'AES

2.1.1 Le corps de Galois GF(2^8)

La conception de l'AES s'articule autour des corps de Galois. Un corps de Galois est un corps contenant un nombre fini d'éléments. Toutes les opérations utilisées dans l'AES sont donc décrites par des opérations algébriques sur des corps finis de mêmes caractéristiques. Avant d'appréhender les opérations arithmétiques de base dans les corps de Galois, décrivons les structures algébriques nécessaires à la compréhension des propriétés des corps finis.

2.1.1.1 Les groupes

Les groupes constituent la structure algébrique de base des mathématiques, puisqu'à partir de ceux-ci sont créés les anneaux, les corps, les espaces vectoriels, ...

Soit G un ensemble non vide doté d'une loi de composition interne :

$$\circ : G \times G \mapsto G$$

Alors, G est un groupe noté $(G,\circ)$, si la loi de composition interne est associative, s'il existe un élément neutre e[4] tel que $e \circ g = g \circ e = g, \forall g \in G$ et si $\forall g \in G$, il existe un unique $g^{-1} \in G$ tel que $g \circ g^{-1} = g^{-1} \circ g = e$ (voir figure 3).

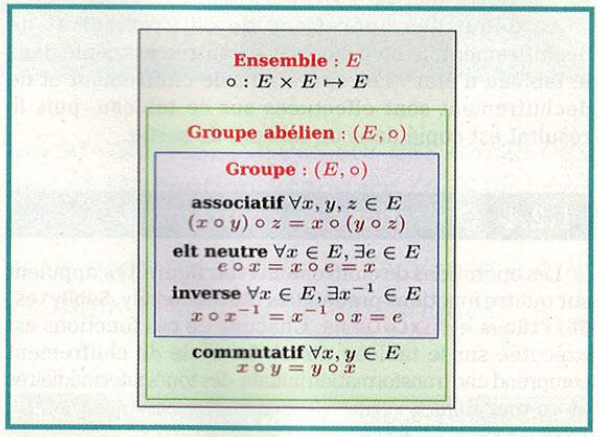


Fig. 3 : Architecture d'un groupe

L'ordre de $(G,\circ)$ est la cardinalité de l'ensemble G. Si l'ordre de $(G,\circ)$ est fini, alors $(G,\circ)$ est un groupe fini. Enfin, un groupe dont la loi de composition interne est commutative est un groupe commutatif ou encore appelé groupe abélien.

À titre d'exemple, l'ensemble des entiers relatifs $\mathbb{Z}$ muni de l'opération d'addition forme un groupe abélien noté $(\mathbb{Z}, +)$.

Une permutation sur un ensemble non vide E est une bijection de $E \to E$. L'ensemble des permutations de E, muni de l'opération de composition, forme un groupe nommé groupe symétrique de E et noté $\mathfrak{S}_E$. Si E est fini et de cardinalité $n > 0, n \in \mathbb{N}$, le groupe symétrique de cet ensemble E est nommé groupe symétrique d'indice n, il est noté $\mathfrak{S}_n$. L'ordre de $\mathfrak{S}_n$ est $n!$, les éléments de $\mathfrak{S}_n$ sont des permutations. Un élément de $\mathfrak{S}_n$ qui permute deux éléments de E et laisse les autres éléments inchangés est appelé une transposition.

Soit deux groupes $(G,\bullet)$ et $(H,\star)$, la fonction $f : G \to H$ est un homomorphisme de groupe si $\forall g,g' \in G$ alors $f(g \bullet g') = f(g) \star f(g')$. Un homomorphisme de groupe bijectif est appelé isomorphisme. Dans ce cas, G et H sont dits isomorphes, ce qui se note par $G \cong H$. Deux groupes isomorphes ont la même structure algébrique et peuvent être considérés comme représentant le même objet algébrique.

2.1.1.2 Les anneaux

Soit A un ensemble non vide associé à deux opérations binaires + et • de $A \times A \to A$. Alors $(A,+,\bullet)$ est un anneau si $(A,+)$ est un groupe abélien, si l'opération • est associative, si elle est distributive sur + et s'il existe un élément $1 \in A$ tel que $1 \bullet a = a \bullet 1 = a$, $\forall a \in A$ (voir figure 4).

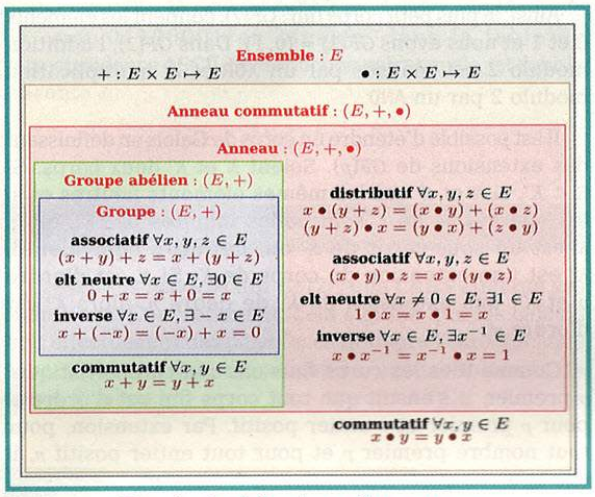


Fig. 4 : Architecture d'un anneau

L'élément identité de $(A,+)$ est 0 et constitue le zéro de l'anneau $(A,+,\bullet)$. L'élément 1 est l'élément identité de l'anneau $(A,+,\bullet)$. Enfin, un anneau est commutatif si sa seconde loi est commutative.

L'ensemble des entiers relatifs $\mathbb{Z}$ muni des opérations d'addition et de multiplication forme un anneau commutatif noté $(\mathbb{Z},+,\bullet)$.

Un anneau commutatif $(A,+,\bullet)$ est un anneau intègre s'il ne contient aucun diviseur de zéro, c'est-à-dire si $a \bullet a' \neq 0 \ \ \forall a,a' \in A\backslash\{0\}$.

Un élément non nul $a \in A$ est dit inversible s'il existe $a^{-1} \in A$ tel que $a \bullet a^{-1} = a^{-1} \bullet a = 1$.

Soit l'anneau $(A,+,\bullet)$ et I un sous-ensemble non vide de A, I est un idéal de A, noté $I \lhd A$, si $(I,+)$ est un sous-groupe de $(A,+)$ et si $\forall i \in I$ et $\forall a \in A$ alors $i \bullet a \in I$ et $a \bullet i \in I$.

Quel que soit $k \in \mathbb{Z}$, $k\mathbb{Z}$ est un idéal de $(\mathbb{Z},+,\bullet)$.

Soient l'anneau A et I un idéal de A, nous pouvons définir une relation d'équivalence $\sim$ dans A telle que $\forall a,a' \in A$, $a \sim a'$ si et seulement si $a' \sim a \in I$. Dans ce cas, la relation $\sim$ est une relation de congruence et nous disons alors que a et a' sont congruents modulo I. La classe d'équivalence[5] de l'élément $a \in A$ est définie par : $[a] = \{a + i \mid i \in I\}$. L'ensemble de toutes les classes d'équivalence de ce type, noté A/I, forme un anneau appelé anneau quotient.

Si S est un sous-ensemble non vide de l'anneau A, alors l'idéal engendré par S est noté $\langle S \rangle$ et consiste en toutes les sommes finies de la forme $\sum a_i \bullet s_i$ où $a_i \in A$ et $s_i \in S$.

L'idéal I de l'anneau $(A,+,\bullet)$ est dit principal s'il existe un élément $a \in A$ tel que $I = a.A$. Autrement dit, I est principal s'il peut être engendré par un élément $a \in A$.

Un anneau commutatif est dit principal si tous ses idéaux sont principaux.

Un anneau intègre dans lequel chaque idéal est un idéal principal est appelé anneau principal.

2.1.1.3 Les corps

Un corps est un anneau dans lequel tout élément non nul est inversible. Autrement dit, le corps K est l'anneau $(K,+,\bullet)$ tel que $(K,+)$ et $(K\backslash\{0\},\bullet)$ sont des groupes abéliens. L'ordre d'un corps est le nombre d'éléments qu'il contient.

L'ensemble des nombres relatifs $\mathbb{Q}$, l'ensemble des nombres réels $\mathbb{R}$ et l'ensemble des nombres complexes $\mathbb{C}$ sont des corps munis des opérations d'addition et de multiplication.

2.1.1.4 Les anneaux polynomiaux

L'analyse algébrique de l'AES fait un usage intensif des anneaux polynomiaux. Un anneau polynomial est un cas particulier d'anneau commutatif.

Un polynôme à une variable x, ou polynôme univarié en x, dans un corps K est une combinaison linéaire finie sur K de monômes en x dont l'expression formelle est de la forme :

$$c_d x^d + c_{d-1} x^{d-1} + \cdots + c_2 x^2 + c_1 x + c_0 \qquad (1)$$

avec d un entier positif, $c_d, \ldots, c_o \in K$ et si $d > 0$ alors $c_d \neq 0$.

L'ensemble de tous les polynômes à une variable x dans un corps K forme un anneau, noté $K[x]$, muni des opérations standards d'addition et de multiplication polynomiales. Cet anneau est un anneau principal nommé anneau des polynômes univariés sur K.

Soit $f(x) \in K[x]$ un polynôme univarié tel que :

$$f(x) = c_d x^d + c_{d-1} x^{d-1} + \cdots + c_i x^i + \cdots + c_2 x^2 + c_1 x + c_0 \qquad (2)$$

le degré de $f(x)$, noté $deg(f(x))$, est l'entier maximum d tel que $c_d \neq 0$, les $c_i x^i$ sont les termes de $f(x)$ et c_i est le coefficient du monôme x^i.

Soient $f(x), g(x) \in K[x]$, il existe $q(x), r(x) \in K[x]$ avec $deg(r(x)) < deg(g(x))$ tel que $f(x) = q(x) \bullet g(x) + r(x)$. Un polynôme $f(x) \in K[x]$ avec $deg(f(x)) > 0$ est dit irréductible s'il n'y a pas de factorisation de la forme $f(x) = q(x) \bullet g(x)$ avec $q(x), g(x) \in K[x]$ et $deg(g(x)) > 0$ et $deg(q(x)) > 0$.

2.1.1.5 Les corps finis ou corps de Galois

L'ensemble $\mathbb{Z}_p = \{0, \ldots, p-1\}$, noté $\mathbb{Z}/p$, muni des opérations d'addition et de multiplication modulo p forme un corps fini si et seulement si p est premier. Ce corps est nommé corps de Galois d'ordre p et noté $GF(p)$.

Ainsi, le plus petit corps fini : $GF(2)$, contient les éléments 0 et 1 et nous avons $GF(2) = \{0, 1\}$. Dans $GF(2)$, l'addition modulo 2 est réalisée par un **XOR** et la multiplication modulo 2 par un **AND**.

Il est possible d'étendre un corps de Galois en définissant des extensions de $GF(p)$. Soient K et K' deux corps. Si $K \subset K'$, si K et K' ont les mêmes éléments neutres et si les opérations de K sont celles induites par K', alors K est un sous-corps de K' ou, de façon équivalente, K' est une extension de corps de K. Si K est d'ordre p et K', une extension de K, de degré n, alors K' est d'ordre p^n.

Comme tous les corps finis ont une caractéristique[6] p premier, il s'ensuit que tout corps fini est d'ordre p^n pour p premier et n entier positif. Par extension, pour tout nombre premier p et pour tout entier positif n, il existe un corps fini, noté $GF(p^n)$, d'ordre p^n.

Lorsque $n > 1$, $GF(p^n)$ peut être représenté comme le corps des classes d'équivalence des polynômes dont les coefficients appartiennent à $GF(p)$.

Un sous-corps de $GF(p^n)$ a un ordre p^d tel que d est un diviseur de n. En outre, il y a un sous-corps d'ordre p^d pour chaque diviseur d de n. Ainsi, le corps de Galois $GF(2^8)$ a $GF(2^4)$, $GF(2^2)$ et $GF(2)$ comme sous-corps.

2.1.1.6 Construction des corps de Galois

Soit $K[x]$ un anneau polynomial formé de l'ensemble de tous les polynômes à une variable x sur le corps K et muni des opérations d'addition et de multiplication polynomiales. L'anneau quotient $\frac{K[x]}{\langle f(x) \rangle}$ est un corps si et seulement si $f(x)$ est irréductible dans $K[x]$.

Soient K un corps fini d'ordre $q = p^n$ et $f(x) \in K[x]$ un polynôme irréductible de degré d. L'anneau quotient $A = \frac{K[x]}{\langle f(x) \rangle}$ est un corps d'ordre $q^d = p^{nd}$ qui est une extension de degré d de K. Ces éléments peuvent être présentés sous la forme :

$$c_{d-1} x^{d-1} + \cdots + c_2 x^2 + c_1 x + c_0$$

avec $c_i \in K$. Tout corps fini d'ordre p^{nd} est isomorphe à A.

Il existe d'autres façons de construire et de représenter les éléments d'un corps fini. Nous retiendrons qu'à toute puissance première correspond un corps fini et que, par conséquent, toutes les représentations de $GF(2^8)$ sont isomorphes. Cependant, malgré cette équivalence, la représentation choisie a un impact sur la complexité de

l'implémentation. Le standard de l'AES a choisi d'utiliser la représentation polynomiale.

Ainsi, un octet contenant les bits $b_7\ b_6\ b_5\ b_4\ b_3\ b_2\ b_1\ b_0$ est considéré comme un polynôme avec des coefficients dans {0,1} :

$$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$$

Par exemple, le caractère 'a' a la valeur hexadécimale 61, soit 01100001 en binaire, dans la table de correspondance ASCII. Sa traduction polynomiale est donc la suivante :

$$x^6 + x^5 + 1 \quad (3)$$

2.1.2 Opérations dans GF(2^8)

C'est dans le corps fini $GF(2^8)$ que l'AES effectue ses opérations. Concrètement, cela signifie que les calculs sont effectués sur des polynômes de degré 7 à coefficients dans {0,1}, modulo un polynôme irréductible $m(x)$ de degré 8 valant $x^8 + x^4 + x^3 + x + 1$ pour l'AES.

2.1.2.1 L'addition

L'addition de deux éléments dans $GF(2^8)$ est réalisée en ajoutant les coefficients des puissances correspondantes des polynômes représentant ces éléments. Cet ajout s'effectuant modulo 2, cela revient à appliquer un **XOR** sur les coefficients. Ainsi, nous avons $1 \oplus 1 = 0$, $1 \oplus 0 = 0 \oplus 1 = 1$ et $0 \oplus 0 = 0$.

Par exemple, si nous souhaitons additionner les octets 01010111 et 10000011 à l'aide de leur représentation polynomiale dans $GF(2^8)$, nous effectuons l'opération suivante :

$$\begin{array}{rl} & 0x^7 + 1x^6 + 0x^5 + 1x^4 + 0x^3 + 1x^2 + 1x + 1 \\ \oplus & 1x^7 + 0x^6 + 0x^5 + 0x^4 + 0x^3 + 0x^2 + 1x + 1 \\ \hline & 1x^7 + 1x^6 + 0x^5 + 1x^4 + 0x^3 + 1x^2 + 0x + 0 \end{array} \quad (4)$$

Le résultat de l'addition est donc l'octet : 11010100.

2.1.2.2 La multiplication

La multiplication dans $GF(2^8)$ s'effectue modulo un polynôme irréductible. Nous avons vu que pour l'AES, ce polynôme est $m(x) = x^8 + x^4 + x^3 + x + 1$. La réduction modulaire permet de garantir que le résultat de la multiplication sera un polynôme de degré inférieur à 8, et donc, que ce dernier correspondra bien à la représentation d'un octet.

La multiplication de deux polynômes s'effectue en deux étapes. Dans un premier temps, la multiplication est effectuée classiquement, et dans un deuxième temps, la réduction polynomiale est appliquée au résultat.

En reprenant les octets de l'exemple utilisé ci-dessus pour l'addition, les étapes de leur multiplication seront les suivantes :

$$\begin{array}{rcl} (x^6 + x^4 + x^2 + x + 1)(x^7 + x + 1) & = & x^{13} + x^7 + x^6 + x^{11} + x^5 + x^4 + \\ & & x^9 + x^3 + x^2 + x^8 + x^2 + x + \\ & & x^7 + x + 1 \\ & = & x^{13} + x^{11} + x^9 + x^8 + x^6 + \\ & & x^5 + x^4 + x^3 + 1 \end{array} \quad (5)$$

$$x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1 \pmod{x^8 + x^4 + x^3 + x + 1} = x^7 + x^6 + 1 \quad (6)$$

Le résultat de la multiplication est donc l'octet : 11000001.

2.1.2.3 Les polynômes à coefficients dans GF(2^8)

Certaines opérations dans l'AES sont réalisées sur des mots de 4 octets. En reprenant le même principe de représentation d'un octet par un polynôme de degré 7, un mot de 4 octets peut être représenté par un polynôme de degré 3 à coefficients dans $GF(2^8)$. Cette représentation prend alors la forme suivante :

$$a(x) = a_3x^3 + a_2x^2 + a_1x + a_0$$

Ce qui donne, si on utilise les octets suivants (01000001), (01100001), (01011010), (011111010), correspondant au mot AaZz :

$$a(x) = (01000001)x^3 + (01100001)x^2 + (01011010)x + (011111010)$$

Dans ce cas, l'addition correspond à l'addition des coefficients de même puissance de x. Ainsi, si nous avons $b(x) = b_3x^3 + b_2x^2 + b_1x + b_0$, alors :

$$a(x) + b(x) = (a_3 \oplus b_3)x^3 + (a_2 \oplus b_2)x^2 + (a_1 \oplus b_1)x + (a_0 \oplus b_0)$$

La multiplication se déroule en deux étapes, dans un premier temps le produit des deux polynômes est calculé et le polynôme résultant $c(x) = a(x) \bullet b(x)$ est de degré 6 et ses coefficients sont :

$$\begin{array}{ll} c_0 = a_0 \bullet b_0 & c_4 = a_3 \bullet b_1 \oplus a_2 \bullet b_2 \oplus a_1 \bullet b_3 \\ c_1 = a_1 \bullet b_0 \oplus a_0 \bullet b_1 & c_5 = a_3 \bullet b_2 \oplus a_2 \bullet b_3 \\ c_2 = a_2 \bullet b_0 \oplus a_1 \bullet b_1 \oplus a_0 \bullet b_2 & c_6 = a_3 \bullet b_3 \end{array}$$

Dans un deuxième temps, le polynôme résultant est réduit modulo un polynôme irréductible sur $GF(2^8)$ de degré 4 qui, pour l'AES, est $m(x) = x^4 + 1$.

2.2 Description algébrique de l'AES

Il existe plusieurs représentations algébriques de l'AES, nous allons nous intéresser à celle exposée dans **[8]**.

La S-Box de l'AES est la composition de trois opérations algébriques simples: une inversion dans $GF(2^8)$ avec $0 \rightarrow 0$, une fonction de permutation linéaire et l'addition de la constante 63. La fonction d'inversion est donnée par $x \rightarrow x^{(-1)} = x^{254}$ et la fonction de permutation est donnée par la fonction polynomiale $x \rightarrow 05x^{254} + 09x^{253} + f9x^{251} + 25x^{247} + f4x^{239} + 01x^{223} + b5x^{191} + 8fx^{127}$.

Ainsi, la S-Box est définie par la fonction polynomiale sur F :

$$S(x) = 63 + \sum_{d=0}^{7} w_d x^{255-2^d} \quad (7)$$

où les w_d sont définis dans la fonction polynomiale ci-dessus.

L'équation 7 peut-être simplifiée en effectuant les approximations suivantes :

- suppression de la constante 63 et remplacement par l'ajout d'une constante spécifique à la clé de chiffrement ;
- suppression de la puissance 255 en partant du principe que $x^{255} = 1$ pour tous les x, sauf quand $x = 0$.

L'équation 7 devient alors :

$$S(x) = \sum_{d=0}^{7} w_d x^{-2^d} \quad (8)$$

Si l'on détaille le fonctionnement d'un tour, nous avons, après l'application de la S-Box à chaque octet du tableau d'état par la fonction **SubBytes** :

$$s_{i,j}^{(r)} = S(a_{i,j}^{(r)}) = \sum_{d_r=0}^{7} w_{d_r} (a_{i,j}^{(r)})^{-2^{d_r}} \quad (9)$$

où $a_{i,j}^{(r)}$ est l'octet situé dans la ligne i, colonne j du tableau d'état, et $s_{i,j}^{(r)}$ est le tableau d'état après l'application de la fonction **SubBytes**.

Après l'application de la fonction **ShiftRows**, le tableau d'état peut alors s'écrire sous la forme :

$$t_{i,j}^{(r)} = s_{i,j-i}^{(r)} = \sum_{d_r=0}^{7} w_{d_r} (a_{i,j-i}^{(r)})^{-2^{d_r}} \quad (10)$$

Après l'application de la fonction **MixColumns**, le tableau d'état s'écrit sous la forme :

$$\begin{aligned} m_{i,j}^{(r)} &= \sum_{e_r=0}^{3} v_{i,e_r} \sum_{d_r=0}^{7} w_{d_r} (a_{e_r,j-e_r}^{(r)})^{-2^{d_r}} \\ &= \sum_{e_r=0}^{3} \sum_{d_r=0}^{7} w_{i,e_r,d_r} (a_{e_r,j-e_r}^{(r)})^{-2^{d_r}} \end{aligned} \quad (11)$$

où les $v_{i,j}$ sont les coefficients de la matrice de diffusion.

Enfin, la dernière étape est réalisée par la fonction **AddRoundKey** qui ajoute la clé :

$$a_{i,j}^{(r+1)} = k_{i,j}^{(r)} + \sum_{e_r=0}^{3} \sum_{d_r=0}^{7} w_{i,e_r,d_r} (a_{e_r,j-e_r}^{(r)})^{-2^{d_r}} \quad (12)$$

Soit :

$$a_{i,j}^{(r+1)} = k_{i,j}^{(r)} + \sum_{e_r=0}^{3} \sum_{d_r=0}^{7} \frac{w_{i,e_r,d_r}}{(a_{e_r,j-e_r}^{(r)})^{2^{d_r}}} \quad (13)$$

où $k_{i,j}^{(r)}$ est la clé du tour r à la position (i, j).

L'équation 13 est une expression algébrique décrivant les transformations du tableau d'état durant un tour.

La version la plus simple de l'AES compte 10 tours, ce qui, en utilisant l'expression précédemment décrite, correspond au final à une expression algébrique comptant environs 2^{50} termes[7]. Cette technique représente malgré tout un avantage. En effet, le DES pouvait également être décrit sous la forme d'une expression algébrique comptant 2^{64} termes alors qu'il n'était basé que sur des blocs de 64 bits et sur une clé de 56 bits.

3 Application pratique

3.1 L'AES sous forme de système d'équations

« *Breaking a good cipher should require as much work as solving a system of simultaneous equations in a large number of unknowns of a complex type.* » **[13]**

Cette phrase[8] célèbre de Claude Elwood Shannon est au coeur de ce que nous allons décrire à présent. En effet, décrire algébriquement l'AES nous permet d'arriver à définir un environnement dans lequel il est possible de décrire un système d'équations quadratiques multivariées.

Dans **[4]** Cid, Murphy et Robshaw définissent deux variantes réduites de l'AES notées $SR(n,r,c,e)$ et $SR^*(n,r,c,e)$. Ces deux variantes ont les mêmes paramètres, à savoir : n le nombre de tours pour le chiffrement, r et c définissent respectivement le nombre de lignes et de colonnes de la table d'entrée et e la taille d'un mot en bits. SR et SR^* diffèrent par la forme du dernier tour. Ces deux environnements travaillent sur des blocs de taille $r \cdot c \cdot e$ et le tableau d'état est un tableau $r * c$ contenant des mots de e bits. Enfin, la version complète de l'AES-128 est décrite par : $SR^*(10,4,4,8)$.

3.1.1 Le Big Encryption System

L'existence d'un système d'équations quadratiques multivariées a été montrée **[10]** en définissant le *Big Encryption System* (BES). Le BES est un système de chiffrement par bloc utilisant des blocs de 128 octets et une clé de 16 octets. L'AES et le BES utilisent

tous les deux un tableau d'état contenant des octets, tableau qui est transformé durant les différentes étapes des tours.

Les espaces d'état de l'AES et du BES sont respectivement les espaces vectoriels $A = GF(2^8)^{16}$ et $B = GF(2^8)^{128}$. Tous les textes clairs, textes chiffrés et clés possibles sont des éléments de A pour l'AES et de B pour le BES. Ainsi, l'AES est un sous-ensemble du BES et la relation définissant l'image de l'AES dans le BES est donnée par :

$$B_A = \phi(A) \subset B \tag{14}$$

où $\phi() : GF(2^8) \rightarrow GF(2^8)^8$ est le vecteur conjugué de mapping entre A et B défini par : $\phi(a)=\left(a^{2^0}, a^{2^1}, a^{2^2}, a^{2^3}, a^{2^4}, a^{2^5}, a^{2^6}, a^{2^7}\right)$. Ainsi, chaque élément $a \in GF(2^8)$ peut être inclus comme un élément $\left(a^{2^0}, a^{2^1}, a^{2^2}, a^{2^3}, a^{2^4}, a^{2^5}, a^{2^6}, a^{2^7}\right) \in GF(2^8)^8$ avec $\phi()$.

3.1.2 Système d'équations pour le chiffrement

L'espace d'état de BES est $B = GF(2^8)^{128}$ et l'équation d'un tour de chiffrement dans BES est donnée par :

$$b \mapsto M_B(b^{(-1)}) + k_{B_i} \tag{15}$$

où M_B est la matrice de diffusion linéaire définie dans **[5]** et k_{B_i} est la clé BES pour le tour i.

Le chiffrement du texte clair $p \in B$ vers le texte chiffré $c \in B$ par le BES est décrit par[9] :

$$w_0 = p + k_0 \tag{16a}$$
$$x_i = w_i^{(-1)} \quad [i = 0, \cdots, 9] \tag{16b}$$
$$w_i = M_B x_{i-1} + k_i \quad [i = 0, \cdots, 9] \tag{16c}$$
$$c = M_B^* x_9 + k_{10} \tag{16d}$$

où $w \in B$ et $x \in B$ sont les vecteurs d'états respectivement avant et après l'opération d'inversion et M_B^* est la version modifiée de la matrice M_B pour le tour final.

3.2 Implémentation de SR

3.2.1 Le logiciel SageMath

Afin d'étudier les systèmes d'équations précédemment décrits, nous allons utiliser le logiciel Sage[10]. Sage est l'acronyme de *Software for Algebra and Geometry Experimentation*, c'est un logiciel open source basé sur le langage Python. Il a pour objectif de fournir une alternative viable à Magma, Maple, Mathematica et Matlab.

Sage est fourni par défaut avec un certain nombre de modules comprenant notamment des outils pour travailler sur des constructions algébriques. Ainsi, nous pouvons créer un anneau polynomial univarié sur l'anneau $\mathbb{Z}$:

```
sage: Z. = ZZ[]
sage: Z
Univariate Polynomial Ring in x over Integer Ring
sage:
```

À partir de cette définition, plusieurs opérations sont possibles :

```
sage: Z.random_element()
-x^2 - x + 3
sage: Z.characteristic()
0
sage: Z.is_finite()
False
```

De même, il est possible de travailler avec des anneaux polynomiaux sur $GF(2)$:

```
sage: A. = GF(2)[]
sage: A
Univariate Polynomial Ring in x over Finite Field of size 2
sage: (x^2 + 1).is_irreducible()
False
sage: (x^3 + x + 1).is_irreducible()
True
sage: x.degree()
1
sage: x.list()
[0, 1]
sage:
```

3.2.2 Étude d'une version allégée

En 2007, Martin Albrecht[11] a commencé à développer un module pour le logiciel Sage implémentant l'environnement *SR* détaillé ci-dessus.

Commençons par travailler avec $SR(1,1,1,4)$, c'est-à-dire avec un chiffrement AES sur 1 tour avec un tableau d'état de 1 colonne et de 1 ligne et un mot de 4 bits.

Construction de $SR(1,1,1,4)$ et affichage du résultat :

```
sage: sr = mq.SR(1, 1, 1, 4)
sage: sr
SR(1,1,1,4)
sage:
```

Affichage des variables :

```
sage: print sr.R.repr_long()
Polynomial Ring
  Base Ring : Finite Field in a of size 2^4
       Size : 20 Variables
    Block 0 : Ordering : degrevlex
              Names    : k100, k101, k102, k103, x100, ...
sage:
```

Calcul et affichage du polynôme irréductible :

```
sage: sr.base_ring().polynomial()
a^4 + a + 1
sage:
```

Affichage de la S-Box :

```
sage: sr.sbox()
(6, 11, 5, 4, 2, 14, 7, 10, 9, 13, 15, 12, 3, 1, 0, 8)
sage:
```

Calcul du système d'équation polynomiale :

```
sage: sr.polynomial_system()
(Polynomial System with 40 Polynomials in 20 Variables,
{k003: a, k002: a^2 + 1, k001: a + 1, k000: a^2})
sage:
```

3.2.3 Quelques données pour l'AES

Comme nous l'avons vu plus haut, dans l'environnement *SR*, l'AES-128 est décrit par : *SR**(10,4,4,8). En utilisant le module *SR* de Sage, nous obtenons les données suivantes :

Construction de *SR**(10,4,4,8) et affichage du résultat :

```
sage: sr = mq.SR(10, 4, 4, 8, star=True)
sage: sr
SR*(10,4,4,8)
sage: sr.base_ring()
Finite Field in a of size 2^8
sage:
```

Calcul et affichage du système polynomial correspondant :

```
sage: sr.polynomial_system()

(Polynomial System with 8576 Polynomials in 4288 Variables,
 {k001507: a^7 + a^6 + a,
  k001506: a^6 + a^5 + a + 1,
  k001505: a^5 + a^4 + a^2 + a,
  k001504: a^5 + a^4 + a^3 + 1,
  k001503: a^5 + a^3 + a^2 + a + 1,
  k001502: a^7 + a^6 + a^4 + a + 1,
  k001501: a^6 + a^5 + a^2 + a,
  k001500: a^5 + a^4 + a^2 + 1,
...
sage:
```

Il est possible d'aller plus loin en développant ses propres scripts Python utilisant les bibliothèques de Sage. Pour cela, le *shebang* du script et les modules importés sont les suivants :

```
#!/usr/bin/env sage -python
# -*- coding: utf-8 -*-

from sage.all import *
from files_aes import *
```

Ensuite, la fonction suivante permet de réaliser des tests de chiffrement à partir des vecteurs donnés dans **[9]**.

```
def aes_128(round):
""" AES: FIPS 197 implementation """
aes = mq.SR(round, 4, 4, 8, star=True, gf2=True,
      allow_zero_inversions=True)
print aes
print aes.base_ring()
print aes.base_ring().base()
print aes.base_ring().degree()
print aes.base_ring().cardinality()
print aes.base_ring().polynomial()
print aes.base_ring().ideal()
print aes.base_ring().zero_element()
print aes.sbox_constant()
print aes.block_order()
print
print '#### key schedule ####'
key = '2b7e151628aed2a6abf7158809cf4f3c'
print 'key:', key

temp = [aes.base_ring().fetch_int(ZZ(key[i:i+2], 16)) for i in
       range(0, len(key), 2)]
key = aes.state_array(temp)
print key
print aes.hex_str(key)
for r in range(aes.n):
  key = aes.key_schedule(key, r+1)
  print aes.hex_str(key)
print
print '#### first test of AES cipher ####'
plain = '3243f6a8885a308d313198a2e0370734'
key = '2b7e151628aed2a6abf7158809cf4f3c'
set_verbose(2)
cipher = aes(plain, key)
set_verbose(0)
print
print '#### second test of AES cipher ####'
plain = aes.vector([0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1,
        1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0,
        1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0,
        1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1,
        0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0,
        1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0,
        0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0]) # fips 197 vector
key = aes.vector([0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0,
        0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1,
        0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1,
        0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0,
        1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0,
        0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1,
        1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0]) # fips 197 vector
set_verbose(2)
cipher = aes(plain, key)
set_verbose(0)
```

La fonction **aes_128** ci-dessus prend en paramètre le nombre de tours à effectuer. Pour exécuter un chiffrement AES complet, il faut réaliser 10 tours. Les lignes 4 à 13 affichent des informations sur le corps dans lequel est défini l'AES. Les lignes 15 à 25 affichent les clés des différents tours. Les lignes 26 à 32 réalisent le chiffrement d'un vecteur de texte clair avec une clé, tous deux fournis dans **[9]**. Enfin, les lignes 33 à 38 réalisent le chiffrement du même vecteur de texte clair avec la même clé, mais donnés sous la forme d'une chaîne de 128 bits.

Conclusion

Décrire l'AES sous la forme d'un système d'équations quadratiques multivariées est une avancée importante pour sa cryptanalyse à partir de méthodes algébriques. Aujourd'hui, les recherches portent sur les méthodes de résolution de grands systèmes d'équations comme les bases de Groebner. Nous l'avons vu ci-dessus, le système d'équation décrit par *SR** permet d'appréhender sa taille : 8576 polynômes et 4288 variables.

En introduction, nous avons expliqué que l'architecture de l'AES avait été conçue afin d'éliminer toute tentative de cryptanalyse statistique. Les recherches actuelles butent sur la résolution de grands systèmes d'équations à variables multiples[12].

En 2002, Nicolas Courtois et Josef Pieprzyk **[6]** ont présenté l'algorithme XSL permettant de cryptanalyser l'AES. Cette attaque est basée sur la résolution d'un système de 8000 équations quadratiques avec 1600 variables pour l'AES-128. Une attaque XSL montée sur le système BES, détaillé ci-dessus, donne un système d'équations qui briserait l'AES avec une complexité de l'ordre de 2^{100}. Plusieurs études **[2,1]** ont montré que cette voie est une impasse: « *Our conclusion is that if XSL works on BES, then it is worse than brute force.* »[13] **[1]**.

Pour autant, la cryptanalyse de l'AES, en s'appuyant sur sa structure algébrique, n'est pas une impasse. Une autre voie pourrait consister à réécrire l'AES sous la forme de systèmes d'équations construites à partir des tables de vérité de ses fonctions internes. En utilisant, par la suite, les algorithmes couramment utilisés dans les codes correcteurs, il est possible d'obtenir un système d'équations plus simple. ■

NOTES

[1] http://www.nsa.gov/ia/programs/suiteb_cryptography/index.shtml

[2] La Russie, par exemple, utilise les algorithmes de chiffrement définis par les standards GOST 28147-89, GOST R 34.10-2001, etc.

[3] Le tableau d'état est un tableau d'octets à deux dimensions comprenant n lignes et m colonnes. Pour l'AES, $n = m = 4$.

[4] L'élément e est unique et est aussi nommé élément identité.

[5] Soit un ensemble S et une relation $\sim$ dans S, la classe d'équivalence d'un élément $a \in S$ est le sous-ensemble, noté $[a]$, de tous les éléments de S qui sont équivalents à a : $[a] = \{s \in S | s \sim a\}$.

[6] La caractéristique p d'un corps fini est le plus petit entier naturel tel que l'addition de 1 itérée p fois soit égal à 0. La caractéristique d'un corps fini est un nombre premier.

[7] Pour l'AES 256, qui compte 14 tours, le nombre de termes s'élève à 2^{70}.

[8] Briser un bon algorithme de chiffrement doit exiger autant de travail que la résolution d'un système d'équations avec un grand nombre d'inconnues de type complexe.

[9] Système d'équation pour la version à 10 tours de l'AES.

[10] http://www.sagemath.org/index.html

[11] http://www.informatik.uni-bremen.de/~malb/ et http://www.informatik.uni-bremen.de/cgi-bin/cgiwrap/malb/blosxom.pl/

[12] Résoudre un système d'équations quadratiques multivariées est un problème de complexité NP.

[13] « Notre conclusion est que si XSL fonctionne sur le BES, alors il est pire que l'attaque par force brute. »

RÉFÉRENCES

[1] Lim Chu-Wee and Khoo Khoongming. *An analysis of xsl applied to bes. Lecture Notes in Computer Science*, 4593:242–253, 2007.

[2] Carlos Cid and Gaëtan Leurent. *An analysis of the xsl algorithm. Lecture Notes in Computer Science*, 3788:333-335, 2007.

[3] Carlos Cid, Sean Murphy, and Matthew Robshaw. *Computational and algebraic aspects of the advanced encryption standard. Seventh International Workshop on Computer Algebra in Scientific Computing*, CASC' 2004:93-103, 2004.

[4] Carlos Cid, Sean Murphy, and Matthew Robshaw. *Small scale variants of the advanced encryption standard. Fast Software Encryption 2005*, LCNS 3557:145-162, 2005.

[5] Carlos Cid, Sean Murphy, and Matthew Robshaw. *Algebraic Aspects of the advanced encryption standard*. Springer, 2006.

[6] Nicolas Courtois and Josef Pieprzyk. *Cryptanalysis of block ciphers with overdefined systems of equations. Lecture Notes in Computer Science*, 2501:267-287, 2002.

[7] Joan Daemen and Vincent Rijmen. *AES proposal: Rijndael*. http://csrc.nist.gov/archive/aes/rijndael/Rijndael-ammended.pdf, 1999.

[8] Niels Fergusson, Richard Schroeppel, and Doug Whiting. *A simple algebraic representation of rijndael. Selected Areas in Cryptography*, LCNS 2259:103-111, 2001.

[9] FIPS. *Advanced encryption standard*. http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf, 2001.

[10] Sean Murphy and Matthew Robshaw. *Essential algebraic structure within the aes. Crypto 2002*, LCNS 2442:11-16, 2002.

[11] Vincent Rijmen, Joan Daemen, Bart Preneel, and Antoon Bosselaers. *The cipher shark*. https://www.cosic.esat.kuleuven.be/publications/article-55.pdf, 1996.

[12] Adi Shamir, Jacques Patarin, Nicolas Courtois, and Alexander Klimov. *Efficient algorithms for solving overdefined systems of multivariate polynomial equations. Lecture Notes in Computer Science*, 1807:392–407, 2000.

[13] Claude Elwood Shannon. *Communication theory of secrecy systems. Bell System Technical Journal*, 28, 1949.

[14] Ilia Toli and Alberto Zanoni. *An algebraic interpretation of aes-128. Advanced Encryption Standard - AES*, 4th International Conference, AES 2004:84-97, 2004.

[15] Eric Weisstein. Field theory. *MathWorld - A Wolfram Web Resource*, 2010.

CENTRALISEZ LA GESTION DES AUTHENTIFICATIONS X.509 + SSH

LM 138
Actuellement en kiosque !

SOMMAIRE :

DISPONIBLE CHEZ VOTRE MARCHAND DE JOURNAUX
JUSQU'AU 27 MAI 2011 ET SUR :
www.ed-diamond.com

8, 9 et 10 juin 2011, Rennes

SSTIC

www.sstic.org

SYMPOSIUM
SUR LA SÉCURITÉ
DES TECHNOLOGIES
DE L'INFORMATION
ET DES COMMUNICATIONS

MISC Multi-System & Network Security Cookbook
cea
esiea
EADS
Supélec
ANSSI Agence nationale de la sécurité des systèmes d'information
technicolor
UNIVERSITÉ DE RENNES 1
Lab-STICC