Formats de fichiers et code malveillant

Philippe Lagadec

DGA/CELAR
philippe.lagadec@ifrance.com

Résumé Cet article se penche sur le cas particulier des fichiers, qui peuvent pénétrer sur un intranet par de nombreux moyens différents (web, e-mail, dis- quette, CDROM, ...), la plupart du temps sans réel filtrage. Une fois qu'il est ouvert par un utilisateur, un fichier peut facilement agir sans aucun contrôle sur un intranet et mettre en jeu sa sécurité. Afin de préciser les menaces, quelques-uns des formats de fichiers les plus classiques sous Windows seront présentés : exécutables, scripts, HTML, documents Office, PDF, ... Cela permet d'ébaucher une classification des différents formats de fichiers, afin de distinguer ceux qui sont inoffensifs et ceux qui présentent une menace potentielle. Il est ainsi possible de déterminer une politique de filtrage adaptée au réseau à protéger. Cet article présente enfin divers moyens techniques et organisationnels disponi- bles aujourd'hui pour contrer au mieux ces menaces.

1 Introduction

La plupart des réseaux d'entreprise connectés à Internet sont aujourd'hui relativement bien sécurisés : les éléments de filtrage tels que routeurs, pare-feux ou DMZ évoluées se sont démocratisés, avec selon les cas antivirus et détection d'intrusion.

Cependant même si la sécurité de niveau réseau est assurée, certains aspects des couches applicatives ne sont pas forcément bien maîtrisés. Cet article se penche sur le cas particulier des fichiers, qui peuvent pénétrer sur un intranet par de nombreux moyens différents (web, e-mail, disquettes, CDROMs, ...) la plupart du temps sans réel filtrage. Or une fois qu'il est ouvert par un utilisateur, un fichier peut facilement agir sans aucun contrôle sur un intranet et mettre en jeu sa sécurité.

Nous préciserons tout d'abord ces menaces liées à l'importation de fichiers, puis nous étudierons les formats de fichiers classiques dans un environnement Windows, afin de dresser une classification suivant les problèmes de sécurité qu'ils peuvent poser. Nous aborderons ensuite les moyens disponibles pour sécuriser un réseau vis-à-vis de ces fichiers.

2 Menaces liées aux fichiers

2.1 Importation de fichiers

Tout utilisateur importe régulièrement des fichiers sur son poste de travail, son ordinateur portable ou son PC personnel. Cela peut se faire par divers moyens :

- Depuis Internet, par e-mail, HTTP, FTP ou transfert de fichiers peer-to-peer.
- Depuis des disquettes et des CDROMs de provenances variées.
- Depuis des supports amovibles comme Zip, des disques USB, ...
- Depuis un ordinateur portable connecté au réseau.

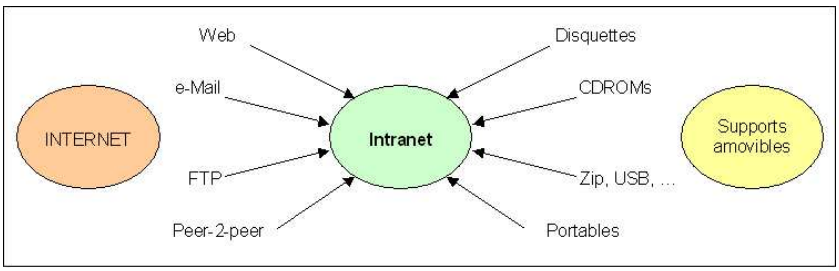

Fig. 1. Importation de fichiers sur un réseau.

Pour les utilisateurs les plus consciencieux, ces fichiers sont d'abord vérifiés par un ou plusieurs antivirus, mis à jour toutes les semaines dans le meilleur des cas.

Ensuite, l'utilisateur ouvre les fichiers importés pour s'en servir. A ce stade il connaît uniquement le nom des fichiers, ce qui lui indique leur format à priori. En général il ne connaît pas vraiment la nature du contenu avant de les avoir ouverts. La seule certitude, c'est que les fichiers ne contiennent pas de virus connu.

Une fois ouverts, ces fichiers s'exécutent avec les même droits que l'utilisateur sur le réseau, sans que celui-ci puisse véritablement contrôler ce qui se passe. S'ils contiennent du code (exécutable, script ou macro), ce code peut agir au nom de l'utilisateur.

Un tel fichier peut donc faire beaucoup de choses, à l'insu de l'utilisateur et de l'administrateur, par exemple :

- Envoi de données confidentielles vers Internet.
- Installation d'une porte dérobée, permettant un accès réseau depuis l'extérieur.
- Installation d'un logiciel de commande à distance.
- Création d'un compte administrateur avec un mot de passe connu, si l'utilisateur en question est lui-même administrateur.
- Destruction ou falsification de données.

– Ecoute des mots de passe saisis au clavier ou circulant sur le réseau.

Dans ce cas, comment garantir l'intégrité d'un système d'information, voire sa disponibilité et sa confidentialité ?

Il existe de très nombreux formats de fichiers différents. Certains comme les exécutables posent des problèmes de sécurité évidents, car ils contiennent directement du code machine, et il est difficile de déterminer leur comportement sans les exécuter ou les désassembler.

D'autres formats peuvent dans certains cas contenir du code, comme les documents Office ou les pages HTML. Enfin, certains formats comme les images bitmap JPEG ou PNG ne présentent aucun problème de sécurité car ils ne peuvent contenir aucun code actif. Il est donc possible de classifier ces formats de fichiers, par exemple dans le but de filtrer les fichiers importés sur un réseau.

2.2 Définition d'un code malveillant

Avant d'étudier les formats de fichiers et les codes malveillants qu'ils peuvent contenir, il est nécessaire de préciser ce qu'est un code malveillant.

Un tel code agit sur le système d'information (soit sur le poste local soit sur le réseau), sans que l'utilisateur l'ait voulu ou bien qu'il y soit autorisé. Dans le domaine de la sécurité, il peut essentiellement porter atteinte au système :

– En intégrité : modification de fichier, de la base de registre, de la mémoire, d'une base de données, d'un annuaire (LDAP, Active Directory), d'autres services d'information, ...
– En disponibilité : atteinte à un service du poste local ou du réseau, afin de le rendre inaccessible ou de perturber son fonctionnement.
– En confidentialité : émission d'informations confidentielles du SI vers l'extérieur.

Un code malveillant fait donc généralement appel à des fonctions du système d'exploitation pour accéder aux fichiers, à la base de registre, au réseau, ...

La frontière entre un code normal et un code malveillant n'est cependant pas évidente. Elle est liée au contenu du système d'information et à sa politique de sécurité globale (ce que chaque utilisateur peut voir ou modifier sur le SI, les flux d'informations autorisés à sortir ou rentrer par le réseau, ...) Elle dépend aussi de l'intention de l'utilisateur au moment où il ouvre un fichier, et de sa connaissance du contenu de ce fichier.

Pour un fichier de type exécutable (binaire ou script), distinguer un code normal d'un code malveillant est quasiment impossible sans désassemblage, processus long, complexe et manuel.

Par contre dans le cas des fichiers de type " documents" qui contiennent du code, il est en théorie plus facile de distinguer les deux : en général un code non offensif ne fait que de l'affichage, du calcul ou de la fusion de données, ne modifie pas d'informations sensibles, et ne transmet pas d'informations confidentielles vers l'extérieur.

2.3 Classification des codes malveillants

Lorsqu'on ouvre un fichier, plusieurs cas peuvent se produire :

- S'il s'agit d'un fichier binaire exécutable (EXE, COM), il sera directement chargé en mémoire et exécuté.
- S'il s'agit d'un script (BAT, CMD, VBScript, JScript, ...), l'interpréteur correspondant sera appelé pour l'exécuter. Notons que certains interpréteurs ne sont pas installés par défaut sur un système Windows classique. (Perl, Python, TCL, ...).
- S'il s'agit d'un document, l'application associée sera ouverte. Si le document contient du code, celui-ci pourra être exécuté avec ou sans confirmation de l'utilisateur suivant les cas.

Du point de vue de la sécurité vis-à-vis d'un code malveillant contenu dans le fichier, nous pouvons donc distinguer plusieurs niveaux de risque, afin de classifier les formats :

1. Le format de fichier contient **toujours** du code, qui est **directement** exécuté à l'ouverture du fichier (cas des exécutables et scripts).
2. Le format contient **parfois** du code, qui peut s'exécuter **directement** (par exemple HTML avec Vbscript ou Javascript utilisant des vulnérabilités d'IE).
3. Le format contient **parfois** du code, qui ne peut s'exécuter qu'après *confirmation* de l'utilisateur (macros Word et Excel).
4. Le format contient **parfois** du code, qui ne peut s'exécuter qu'après une **action volontaire** de l'utilisateur (documents Office contenant des objets OLE).
5. Le format ne peut **jamais** contenir de code (images bitmap GIF, JPEG, PNG, ...)[1].

2.4 Associations de types de fichiers

Dans un environnement Windows, l'action effectuée lorsqu'un utilisateur ouvre un fichier dépend principalement de son extension. Chaque extension correspond à un type de fichier, auquel est associé un programme ou une ligne de commande. Par exemple, l'extension ".HTML" est par défaut associée à Internet Explorer, dont l'exécutable est "iexplore.exe". Cette association est inscrite dans la base de registre, dans la ruche "HKEY_CLASSES_ROOT". Sous Windows NT/2000/XP, les commandes FTYPE et ASSOC permettent d'afficher les associations de fichiers. Par exemple, pour afficher l'application associée aux fichiers ".VBS", on peut utiliser les deux commandes suivantes :

```
Assoc .vbs => cela donne ".vbs=VBSFile"
Ftype VBSFile => on obtient Wscript.exe
```

C'est un peu plus compliqué, mais il est aussi possible de lister toutes les extensions de fichiers associées avec un programme en utilisant les commandes suivantes dans un fichier batch. Voici un exemple pour Word :

[1] Nous considérons ici des codes malveillants directement activables

```
ftype|find /i "word.exe" >types.tmp
FOR /F "delims==" %%t IN (types.tmp) DO (
assoc|find /i "%%t" >ext.tmp
FOR /F "delims==" %%e IN (ext.tmp) DO echo %%e
)
```

Cela montre que de nombreux types de fichiers peuvent être associés à un même programme. Dans le cas de Word, un document peut être renommé avec diverses extensions (doc, dot, dochtml, rtf, wbk) en ayant le même effet lorsqu'il est ouvert.

3 Formats de fichiers

Ce chapitre présente quelques formats de fichiers classiques dans un environnement Windows standard, et montre certains problèmes de sécurité qu'ils peuvent poser. Cette liste est loin d'être exhaustive, étant donné les milliers de formats de fichiers qui existent. Elle constitue cependant une bonne base pour définir une politique de filtrage pour une passerelle Web ou e-mail.

Elle pourrait être facilement étendue aux environnements Unix, Linux ou Macintosh en y ajoutant les formats de fichiers spécifiques à ces systèmes.

3.1 Fichiers texte ou binaires

On distingue généralement deux grands types de formats de fichiers : Les fichiers texte contiennent uniquement des caractères imprimables (codes ASCII 32 à 255 : lettres, chiffres, signes, ...) et des caractères spéciaux de mise en page (certains codes inférieurs à 32 : saut de ligne, tabulation, ...) Les octets qui les composent correspondent donc à un sous-ensemble des codes ASCII.

Les fichiers binaires sont tous les autres fichiers qui ne correspondent pas à cette définition. Ils peuvent donc contenir l'intégralité des codes ASCII (codes 0 à 255), en particulier les codes de 0 à 31.

3.2 Exécutables binaires : EXE, COM, SCR,...

Ces fichiers contiennent du code binaire directement exécutable par le processeur. Lorsqu'un utilisateur les ouvre, le code est immédiatement chargé et exécuté dans le contexte de l'utilisateur.

Ce sont donc les formats qui présentent le plus important problème de sécurité, puisque le code qui s'y trouve a un accès complet au système avec tous les droits de l'utilisateur. La plupart des virus et des chevaux de Troie sont des exécutables. Les fichiers EXE et leurs dérivés (SCR, CPL, OCX, DLL, ...) ont un format binaire, et possèdent une entête dont la structure est fixée. En particulier, les 2 premiers octets sont toujours "MZ" (initiales de Michael Zimmer, un des concepteurs de MS-DOS), sinon Windows refuse de les exécuter. Certains fichiers EXE au format MS-DOS 16 bits peuvent également débuter par "ZM". Un fichier EXE peut donc être reconnu par son nom et son contenu.

Les fichiers COM quant à eux ne possèdent pas d'entête, leur contenu est simplement du code exécutable. On peut noter qu'un fichier COM peut être conçu de manière à n'utiliser que des caractères imprimables, et avoir ainsi l'apparence d'un fichier texte. Seul le nom permet de détecter un fichier COM.

3.3 Fichiers de commande MS-DOS/Windows : BAT, CMD

Les fichiers BAT et CMD (appelés aussi "fichiers batch") sont au format texte et contiennent une liste de commandes qui sont exécutées séquentiellement lorsqu'on les ouvre. Ces commandes peuvent porter atteinte à la sécurité du système dans une certaine mesure, en lançant des outils présents sur le système.

A titre d'exemple, les 2 lignes suivantes créent un nouvel utilisateur "toto" avec un mot de passe "toto", et l'ajoutent au groupe des administrateurs (doit être lancé par un administrateur) :

```
net user toto toto /add
net localgroup administrateurs toto /add
```

3.4 PIF (Program Info File)

Un fichier PIF sert normalement à Windows pour stocker des informations sur un programme MS-DOS, afin de préciser dans quelles conditions celui-ci doit être exécuté : en fenêtre ou plein écran, s'il a besoin de mémoire étendue, etc... Il a un format binaire, et contient la ligne de commande de l'exécutable à lancer. Il peut donc exécuter une commande portant atteinte au système.

Il est important de noter que l'extension PIF est toujours masquée dans l'explorateur Windows, même si la case "masquer les extensions pour les types de fichiers connus" a été désactivée.

Un fichier .COM ou .EXE peut être renommé en .PIF. Dans la plupart des cas celui-ci sera alors directement exécutable même s'il ne possède pas la structure normale d'un fichier PIF.

3.5 Raccourcis Windows : LNK

D'une manière similaire aux fichiers PIF, les fichiers LNK contiennent une ligne de commande qui sera exécutée quand on double-clique sur le raccourci, commande qui peut porter atteinte au système.

L'extension LNK est également toujours masquée, et on peut observer dans certains cas qu'un fichier EXE ou COM renommé en LNK reste exécutable. Scripts du Windows Scripting Host : VBS, JS, VBE, JSE, WSF, WSH

Le Windows Scripting Host est un outil optionnel (installé par défaut sur les systèmes récents), qui permet d'ajouter de nouveaux langages de script à Windows. Il contient deux langages de base qui sont Vbscript et Jscript, mais il est possible d'en ajouter d'autres comme Perl ou TCL.

Ces langages sont plus évolués que les fichiers de commandes batch, et ils ont la possibilité d'accéder aux contrôles ActiveX installés sur le système, afin

de contrôler toute application qui en offre la possibilité. Il est possible de manipuler les fichiers et la base de registre, mais aussi d'accéder aux logiciels comme Outlook express, Word ou Excel avec un langage de haut niveau. Par exemple le tristement célèbre virus "I Love You" était écrit en Vbscript, pour accéder facilement au carnet d'adresses d'Outlook Express et se répliquer par e-mail.

Les fichiers VBS et JS s'exécutent directement lorsqu'on les ouvre. Ils sont très rarement utiles pour les utilisateurs normaux, il est donc souvent recommandé de désactiver voire désinstaller le Windows Scripting Host pour éviter ce problème de sécurité.

Voici un exemple de code Vbscript qui crée un fichier, ce qui est une action potentiellement dangereuse :

```
Dim fso, tf
Set fso =
Wscript.CreateObject("Scripting.FileSystemObject")
Set tf = fso.CreateTextFile("c:\temp\test.txt", True)
tf.WriteLine("test VBscript.")
tf.Close()
```

Le même code, cette fois-ci en Jscript :

```
var fso, tf;
fso = new ActiveXObject("Scripting.FileSystemObject");
tf = fso.CreateTextFile("c:\\temp\\test.txt", true);
tf.WriteLine("Ceci est un test.") ;
tf.Close();
```

Les fichiers VBE et JSE correspondent respectivement à des fichiers VBS et JS qui ont été chiffrés par l'outil "Microsoft Script Encoder". Leur code n'est pas lisible en clair, cependant ils s'exécutent de la même façon.

Un fichier WSF est un fichier au format XML pouvant contenir un ou plusieurs scripts Vbscript ou Jscript. S'il est ouvert, les scripts seront exécutés séquentiellement. Voici un exemple de fichier WSF contenant 2 scripts :

```
<Job id="vbs">
<Script language="VBScript">
ok = msgbox("script 1")
</Script>
<Script language="VBScript">
ok = msgbox("script 2")
</Script>
</Job>
```

Un fichier WSH est associé à un fichier VBS ou JS : il sert à stocker des options pour ce script, comme par exemple le temps maximal d'exécution. Il est au format "INI' de Windows, et contient notamment une variable Path qui indique le chemin du script associé. Lorsqu'on lance le fichier WSH, le script associé est exécuté. Voici un exemple de fichier WSH qui fait référence à un script situé sur une autre machine du réseau :

```
[ScriptFile]
Path=\\serveur\pirate\script.vbs
[Options]
Timeout=0
DisplayLogo=0
```

3.6 Autres scripts : Perl, Python, AWK, KiXtart, PHP, TCL, bash, ...

Il existe de nombreux autres langages de script, cependant chacun nécessite que l'interpréteur correspondant soit installé sur la machine locale pour qu'il puisse être exécuté par l'utilisateur. On les retrouve rarement sur un système Windows standard.

Ces types de fichiers présentent donc un risque bien moindre que Vbscript, Jscript ou les fichiers de commandes batch. Ils doivent cependant être pris en compte si des interpréteurs sont installés sur le réseau.

3.7 Application et applets Java : fichiers .java, .class et .jar

Le langage Java est pseudo-compilé : les fichiers source ".java" ne sont pas directement interprétés comme les scripts, mais transformés en pseudo-code dans des fichiers ".class". Ces fichiers .class peuvent être alors interprétés par la machine virtuelle Java, appelée "JVM". Suivant sa conception, un fichier .class peut être soit une application Java autonome, soit une applet Java intégrée dans une page HTML.

Une applet Java s'exécute normalement dans une "sandbox" (bac à sable), et la JVM lui interdit toute action qui pourrait porter atteinte à la sécurité du système : lecture/écriture de fichiers ou du registre, accès réseau à d'autres machines que le serveur d'où l'applet a été téléchargée, etc... Il est cependant possible de réduire ces contraintes pour des applets de confiance, en utilisant la signature de code et des certificats cryptographiques. Ce système de sandbox est réputé fiable, même si certaines JVM ont souffert de quelques vulnérabilités qui permettaient à des applets d'échapper dans certains cas aux contrôles de sécurité.

Une application Java a quant à elle un accès complet au système local, tout comme les fichiers exécutables ou les scripts. Sur un système normal, l'extension .class n'est pas associée à la JVM, donc il ne se passe rien si l'utilisateur essaye d'ouvrir directement un tel fichier. Pour l'exécuter, le JRE (Java Runtime Environment) doit être installé, ce qui n'est pas le cas sur un système Windows par défaut. Il doit lancer explicitement l'application, en tapant une ligne de commande du type "java application.class". Un fichier .class pris tout seul ne pose donc pas de problème de sécurité en soi, sauf s'il a été volontairement associé au JRE, et que celui-ci est installé.

Les fichiers .jar, pour "Java Archive"», sont des archives compressées similaires aux fichiers Zip, qui contiennent plusieurs fichiers .class et d'autres fichiers nécessaires à leur fonctionnement. Si un JRE est installé, il peut prendre en

charge un fichier .jar et exécuter l'application Java qui s'y trouve, mais là encore ce n'est pas le cas par défaut.

Les fichiers .java contiennent uniquement du code source Java, et ne sont pas directement exécutables. Ils ne posent donc aucun problème de sécurité.

3.8 Documents MS Office : Word, Excel, Powerpoint, Access, ...

Les documents Microsoft Office peuvent contenir des macros écrites en langage VBA (Visual Basic for Applications), qui ont souvent été utilisées par les concepteurs de virus. Le langage VBA permet de nombreuses manipulations sur les documents, ainsi que sur le système lui-même (fichiers et registre).

En nommant une macro de façon particulière, il est possible de la faire s'exécuter de façon automatique dès l'ouverture du document, voire même qu'elle s'installe dans le logiciel. (AutoOpen ou Document_Open pour Word, Auto_Open pour Excel, autoexec pour Access,...). Une macro peut aussi être associée à un objet dans le document, à un bouton, une touche du clavier ou un autre événement.

Les versions récentes d'Office intègrent quelques garde-fous pour protéger l'utilisateur contre les macros malveillantes. Par défaut, lorsqu'on ouvre un fichier Word, Excel ou Powerpoint contenant des macros, une boîte de dialogue s'ouvre pour permettre d'activer ou désactiver les macros. Cette sécurité peut cependant être ignorée voire retirée par l'utilisateur. Depuis Office 2000, il est toutefois possible de renforcer la sécurité, en n'autorisant que des macros ayant été préalablement signées. Les rapports [1] et [2] détaillent les problèmes de sécurité et les contre-mesures des documents Office 97 et 2000.

Il est important de remarquer que toutes les applications Office ne sont pas homogènes pour la gestion des macros. En particulier Access (au moins jusqu'à la version 2000) ne demande pas confirmation, et une macro nommée "autoexec" sera systématiquement exécutée à l'ouverture de la base de données. Access ne fait cependant pas partie de la version standard d'Office.

Les fichiers Office (sauf Access) possèdent une particularité : lorsqu'on renomme un fichier Office en lui donnant une extension qui n'est associée à aucun programme, par exemple ".xyz", et qu'on essaye d'ouvrir le fichier depuis l'explorateur, la bonne application (Word, Excel ou Powerpoint) sera quand même lancée. Cela n'est pas le cas avec les autres formats de fichiers, et cela ne se produit pas si on tente d'ouvrir le fichier Office renommé depuis l'invite de commande. Microsoft Office installe donc un module spécifique à l'explorateur qui examine le contenu d'un fichier lorsqu'il a une extension inconnue. En conclusion, pour reconnaître un fichier Office il ne suffit pas de regarder son nom.

3.9 Objets OLE

De nombreuses applications permettent l'utilisation du protocole OLE (Object Linking and Embedding), afin d'inclure un document de toute nature à l'intérieur d'un autre document. Il est par exemple possible d'inclure un tableau Excel dans un document Word. Il est également possible d'inclure un exécutable, qui sera lancé par un double-clic sur l'objet OLE.

La sécurité liée à l'exécution des objets OLE est assurée par chaque application. Ainsi Word demande confirmation à l'utilisateur avant de lancer un exécutable inclus, ce qui n'est pas le cas de WordPad.

Tout format de fichier supportant OLE est donc un conteneur, qui peut servir à camoufler un code malveillant.

3.10 Documents RTF

Le format RTF (Rich Text Format) est un format de Microsoft, qui permet d'exporter un document Word sous une forme "plus portable" car simplifiée. Certaines mises en forme sont supprimées, les liens hypertextes et les macros VBA sont retirées. RTF est donc souvent présenté comme un format statique, idéal pour l'échange de données car sans problème de sécurité. Il est cependant possible d'inclure un fichier exécutable dans un document RTF, sous forme d'objet OLE qui peut être activé par un double-clic. Le format RTF est donc un conteneur.

3.11 Shell Scrap : SHS

Lorsque l'on sélectionne un objet OLE dans un document, et qu'on le glisse avec la souris dans l'explorateur ou sur le bureau, on obtient un fichier SHS. Cette extension SHS est toujours masquée dans l'explorateur, et l'icône du fichier ressemble à celle d'un fichier texte. Si l'on essaye d'ouvrir ce fichier SHS, le contenu de l'objet OLE d'origine est directement ouvert, sans confirmation. S'il s'agit d'un exécutable, il est directement lancé.

3.12 HTML

Sur la plupart des systèmes Windows, les fichiers HTML sont associés à Internet Explorer. De nombreux logiciels comme Outlook Express font également appel à Internet Explorer pour afficher des contenus HTML, par exemple lorsqu'ils sont inclus dans des e-mails.

Les fichiers HTML et les e-mails au format HTML affichés par IE peuvent contenir des scripts (Vbscript ou Jscript, équivalent de Javascript), et faire appel à des objets ActiveX ou des applets Java.

Comme nous l'avons vu plus haut, les applets Java sont des fichiers .class ou .jar externes au fichier HTML, qui y fait juste référence. De plus les applets non signées s'exécutent dans un environnement sécurisé, la sandbox. A part s'il existe une vulnérabilité dans la JVM (et cela est déjà arrivé), un fichier HTML ne peut utiliser une applet Java comme code malveillant pour accéder au système.

Pour les objets ActiveX et les scripts, IE utilise un système de zones de sécurité, afin d'appliquer des restrictions plus ou moins fortes suivant la provenance des pages HTML. Pour un fichier HTML, lorsqu'un script tente d'effectuer une action pour accéder au disque, à la base de registre ou au réseau, ou lorsque la page HTML essaye de charger un objet ActiveX non signé, l'utilisateur

reçoit une demande de confirmation, qui lui permet de bloquer le script ou le chargement de l'objet ActiveX. Internet Explorer a cependant un lourd passé : de nombreuses vulnérabilités ont permis de contourner le système de zones de sécurité, et ces failles ont été mises à profit dans la plupart des virus récents pour exécuter du code malveillant sans confirmation. Comme les anciennes versions d'Internet Explorer sont toujours largement répandues, il est nécessaire de prendre en compte le risque d'exécution de code malveillant depuis les pages HTML, malgré les mesures de protection.

Les scripts peuvent être placés à de nombreux endroits dans un code HTML, notamment :

- Entre des balises `<SCRIPT>` et `</SCRIPT>`
  (exécution dès le chargement de la page).
- Dans l'URL d'un lien hypertexte : `<A HREF="javascript:...">`
  (exécution quand l'utilisateur clique sur le lien).
- Dans l'événement "Onload" de la zone Body : `<BODY ONLOAD="javascript:...">`
  (exécution dès le chargement de la page).
- Dans de nombreux évènements associés à des objets, par exemple :
  `<IMG SRC="une_image.gif" ONMOUSEOVER="javascript:...">`
  (exécution quand l'événement apparaît).

Le langage Javascript, développé au départ par Netscape, a des fonctionnalités limitées à l'affichage, au calcul, à la gestion des cookies et au contrôle du navigateur. Vis-à-vis du système, il ne pose donc pas à priori de problème de sécurité.

Par contre le langage Jscript, qui possède une syntaxe similaire à Javascript, offre des fonctions supplémentaires d'accès au système de fichiers, à la base de registre et aux contrôles ActiveX installés sur le système. Tout code Javascript est exécuté par Internet Explorer en tant que Jscript, il est donc possible d'écrire un code malveillant dans un simple lien `"javascript:..."`. Le langage Vbscript possède une syntaxe différente de Jscript (dérivée de Visual Basic), cependant les fonctionnalités et les risques sont identiques.

3.13 XHTML

XHTML est une nouvelle version du langage HTML, dont la syntaxe a été corrigée pour être conforme au standard XML. Cette syntaxe est plus stricte, et interdit certaines formes d'écritures pouvant mener à l'exploitation de vulnérabilités ou a du camouflage de code.

Les possibilités d'inclusion de scripts sont cependant les mêmes que pour HTML, il convient donc de traiter les fichiers XHTML avec autant de précautions.

3.14 XML

XML est un format conçu pour contenir des données, structurées sous forme d'arbre. A la différence de HTML, un fichier XML ne contient que des données brutes et pas de mise en forme. L'interprétation de ces données dépend de chaque application, qui peut définir son propre schéma (structure de données). Il est

donc impossible de déterminer dans l'absolu si un fichier XML contient du code malveillant, à moins de savoir exactement comment les données seront exploitées par l'application associée.

Par défaut sur un système Windows, l'extension ".XML" est associée à Internet Explorer. Lorsqu'on ouvre un fichier XML brut, IE affiche simplement les données sous forme d'arbre à l'écran. Il n'y a pas de possibilité d'exécution de code dans ce cas.

Cependant XML introduit la notion de feuilles de style XSL. Une feuille de style XSL décrit les règles de transformation à appliquer pour afficher les données du fichier XML, par exemple pour obtenir une page HTML avec mise en forme. Si un fichier XML possède dans son entête une référence à une feuille de style XSL, et que le fichier XSL en question est accessible (soit en local, soit par le réseau), alors Internet Explorer applique automatiquement la transformation, et le résultat est affiché. Dans ce cas, le résultat peut inclure du code malveillant, comme toute page HTML. De plus, il est possible d'inclure la feuille de style XSL à l'intérieur du fichier XML, ainsi un simple fichier XML peut très bien contenir du code malveillant HTML, qui s'exécutera à l'ouverture dans IE. Voici un très court exemple :

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="#xsl" ?>
<xsl:stylesheet id="xsl"
xmlns:xsl="http://www.w3.org/TR/WD-xsl">
   <xsl:template match="/">
     <HTML><BODY><SCRIPT language="javascript">
       alert('Ceci est un jscript contenu dans XML.');
     </SCRIPT></BODY></HTML>
   </xsl:template>
</xsl:stylesheet>
```

Les fichiers XML posent donc les mêmes problèmes de sécurité que les fichiers HTML, avec des possibilités supplémentaires de camouflage à cause du mécanisme des feuilles de style XSL.

3.15 MHT (MIME HTML)

Les fichiers MHT correspondent au format MHTML défini dans [3]. Il s'agit d'un format similaire aux e-mails avec pièces jointes, permettant d'encapsuler une page HTML et les autres fichiers qui la composent (images, sons, ...). Il est donc possible d'y inclure un nombre quelconque de fichiers, de tous formats. Ces fichiers peuvent apparaître en clair, ou être codés suivant le standard MIME (Base64 ou quoted-printable, par exemple).

C'est notamment le format utilisé par Internet Explorer lorsqu'on sauvegarde une page HTML en tant qu' "Archive Web".

Par défaut sur un système Windows, l'extension ".MHT" est associée à Internet Explorer. Lorsqu'on ouvre un fichier MHT, IE affiche directement le premier fichier contenu, qui est normalement une page HTML.

Le format MHT est donc un conteneur, dans lequel peuvent être cachés des scripts ou tout autre fichier codés au format Base64.

Voici un exemple de fichier MHT, lançant un simple script non codé :

```
MIME-Version: 1.0
Content-Type: text/html
Content-Transfer-Encoding: 7bit

<body onload="alert('Ceci est un script dans une page
HTML, dans un fichier MHT.')">
```

3.16 Adobe Acrobat : PDF

Le format PDF a été créé par Adobe pour produire des documents lisibles sur de nombreux systèmes, en conservant une mise en forme homogène à l'écran et à l'impression. Le logiciel d'affichage Acrobat Reader est fourni gratuitement, et il est aujourd'hui extrêmement répandu, PDF est donc devenu un standard.

A l'instar de RTF, PDF est souvent considéré comme un format sûr (plus sûr que Word et HTML), permettant d'échanger des documents riches sans risque d'infection par un virus.

Ce format possède cependant une fonction méconnue, qui permet à un document PDF d'ouvrir un fichier externe (y compris un exécutable) lorsqu'un événement se produit, par exemple à l'ouverture ou à la fermeture du fichier PDF. Avant d'exécuter une telle action qui pourrait porter atteinte au système, Acrobat Reader demande confirmation à l'utilisateur. Cette sécurité peut être considérée comme faible, car l'action par défaut est d'accepter l'exécution, et l'utilisateur peut choisir d'ignorer toutes les confirmations suivantes (tant qu'Acrobat Reader reste en mémoire).

Un fichier PDF peut également contenir des fichiers attachés, et ceux-ci peuvent être lancés par un double-clic de l'utilisateur après confirmation, tout comme les objets OLE dans Word. Ces fichiers attachés ne sont cependant accessibles que dans la version complète d'Acrobat, et non dans Acrobat Reader. Ce logiciel étant beaucoup moins répandu, cette fonctionnalité est difficilement utilisable pour un code malveillant.

Le document [4] fourni par Adobe détaille ces deux problèmes de sécurité, et apporte quelques éléments de sécurisation.

Le format PDF doit donc également être considéré avec précaution, même si le risque est à priori moins grand que pour les formats vus précédemment.

3.17 Synthèse

Le tableau suivant permet de synthétiser les caractéristiques des différents formats abordés du point de vue de l'exécution de code malveillant. La colonne risque correspond aux 5 niveaux de classification des codes malveillants listés en début de chapitre :

1. Le format de fichier contient **toujours** du code, qui est **directement** exécuté à l'ouverture du fichier.
2. Le format contient **parfois** du code, qui peut s'exécuter **directement**.
3. Le format contient **parfois** du code, qui ne peut s'exécuter qu'après **confirmation** de l'utilisateur.
4. Le format contient **parfois** du code, qui ne peut s'exécuter qu'après une **action volontaire** de l'utilisateur.
5. Le format ne peut **jamais** contenir de code.

Format	Extensions	Risque	Conteneur	Type
exécutable binaire	EXE, SCR	1	oui	binaire
exécutable binaire	COM	1	oui	binaire/texte
fichier batch	BAT,CMD	1		texte
raccourci MS-DOS	PIF	1		binaire
raccourci Windows	LNK	1		binaire
scripts WSH	VBS, JS, VBE, JSE, WSF, WSH	1		texte
code source Java	JAVA	5		texte
classe Java	CLASS	4		binaire
archive Java	JAR	4	oui	binaire
Word	DOC, DOT, WBK, DOCHTML	3	oui	binaire
Excel	XLS, XL ?, SLK, XLSHTML,	3	oui	binaire
Powerpoint	PPT, POT, PPS, PPA, PWZ, PPTHTML, POTHTML	3	oui	binaire
Access	MDB, MD ?, MA ?, MDBHTML	1	oui	binaire
RTF	RTF	4	oui	texte
Shell Scrap	SHS	1	oui	binaire
HTML	HTML, HTM, ...	2		texte
XHTML	XHTML, XHT	2		texte
XML	XML, XSL	2		texte
MHTML	MHT, MHTML	2	oui	texte
Adobe Acrobat	PDF	3	oui	texte

4 Solutions

Il n'existe pas aujourd'hui de solution unique pour se protéger globalement du risque d'exécution de code malveillant depuis des fichiers. Il est nécessaire de mettre en place de nombreuses mesures pour améliorer la sécurité de chaque système. Les paragraphes suivant listent les principales solutions envisageables à l'heure actuelle.

4.1 Sécurisation du poste de travail

La sécurisation de chaque poste de travail est la première mesure technique à mettre en place. Elle implique la sécurisation du système d'exploitation et des logiciels installés, ainsi que leur mise à jour régulière.

4.2 Mise à jour du système et des logiciels

Le passé a montré que la plupart des virus et des attaques mettant en jeu du code malveillant tiraient parti de vulnérabilité dans le système d'exploitation Windows et notamment dans son navigateur Internet Explorer. Il est donc crucial de mettre à jour chaque système d'exploitation pour corriger les failles connues. Cela nécessite un suivi régulier et rigoureux, qui prend beaucoup de temps.

C'est également le cas de la suite Microsoft Office et des autres logiciels qui sont installés sur la plupart des postes clients, comme Acrobat Reader.

4.3 Sécurisation du système et des logiciels

Les systèmes Windows et les logiciels sont rarement installés avec une sécurité optimale par défaut. Il est donc important d'appliquer les recommandations pour améliorer la sécurité de l'ensemble, et combler certaines failles. De nombreuses check-lists de sécurité sont disponibles sur Internet pour mener à bien ce travail.

4.4 Internet Explorer et Outlook Express

La sécurisation d'IE et d'OE passe essentiellement par le renforcement de la sécurité des zones d'Internet Explorer. Le site [5] propose une sécurisation de ces deux logiciels.

4.5 Acrobat Reader

Certaines clés de registre peuvent être positionnées pour empêcher l'exécution automatique de fichiers externes, comme le montre le document [4]. Ces clés varient suivant la version d'Acrobat ou d'Acrobat Reader installée, par exemple :

- Pour Acrobat Reader 4.0 :
  ```
  HKEY_CURRENT_USER\Software\Adobe
  \Acrobat Reader\4.0\AdobeViewer\SecureOpenFile = 1
  ```
- Pour Acrobat Reader 5.0 :
  ```
  HKEY_CURRENT_USER\Software\Adobe
  \Acrobat Reader\5.0\AdobeViewer\SecureOpenFile = 1
  ```
- Pour Acrobat 5.0 :
  ```
  HKEY_CURRENT_USER\Software\Adobe
  \Adobe Acrobat\5.0\AdobeViewer\SecureOpenFile = 1
  ```

4.6 Office

La sécurisation d'Office 2000 passe avant tout par la mise en place du niveau de sécurité pour les macros, afin de n'autoriser que les macros signées (menu Outils/Macro/Sécurité).

Les rapports [1] et [2] apportent une liste de contre-mesures détaillées pour améliorer la sécurité d'Office 97 et 2000.

4.7 Antivirus

Un antivirus mis à jour très régulièrement (toutes les semaines pour les postes clients) est évidemment une mesure indispensable pour se protéger des codes malveillants connus. Certains antivirus intègrent une méthode de détection heuristique capable de détecter certains codes ayant un comportement de virus, cependant cette approche ne suffit pas pour détecter un code malveillant dans le cas général.

4.8 Passerelle de filtrage réseau : e-Mail, Web, FTP, ...

Il existe de nombreuses solutions de filtrage réseau à base de passerelles ou proxies, afin de s'assurer qu'aucun contenu actif ne peut pénétrer par e-mail, consultation web ou transfert de fichiers FTP.

La plupart des solutions proposent une analyse antivirus à la volée ainsi qu'un filtrage par type de fichier importé. Cependant ce filtrage est souvent basé uniquement sur le nom des fichiers ou sur leur type MIME déclaré. Or nous avons vu qu'un type de fichier peut souvent être renommé de diverses manières, ou même que certains types peuvent avoir une extension quelconque tout en restant actifs (documents Office). De plus, de nombreux formats de fichiers jouent le rôle de conteneurs, et peuvent donc camoufler du code malveillant.

Un filtrage de fichiers efficace nécessite donc une analyse par contenu et par nom. Cette analyse doit être récursive afin de traiter les formats conteneurs.

4.9 Suppression de code, conversion de format

Pour les codes inclus dans des documents Office ou HTML, il est en théorie possible de distinguer un code inoffensif d'un code potentiellement dangereux, car le langage de programmation est connu, et les commandes "à risque" qui accèdent au système de fichier, à la base de registre ou au réseau peuvent être identifiées. Dans la pratique, les outils qui proposent ce type de détection possèdent cependant des limites, dues à l'évolution des langages de script et aux possibilités de camouflage de code.

Certains produits de filtrage (comme eSafe ou MimeSweeper) et certains antivirus (comme F-Prot) proposent la suppression complète des portions de code dans les formats connus, en particulier HTML et les documents Office. Cette suppression n'est généralement pas parfaite et peut parfois être contournée

par des techniques de camouflage, cependant il s'agit d'une fonction intéressante pour assurer un service tout en évitant de nombreux risques.

Pour supprimer le code de ces formats de fichiers, il est également possible de convertir les fichiers dans un autre format compatible présentant moins de risques, comme RTF ou PDF, ou encore le format texte pur. Cette conversion est cependant souvent très imparfaite (perte de mise en forme), et très contraignante pour l'utilisateur.

4.10 Supports amovibles

Pour un système très sensible, il est parfois possible de neutraliser tous les périphériques amovibles : lecteur de disquette, CDROM, ports USB, série et parallèle, SCSI, ... Cependant ce n'est pas envisageable dans un cadre général. Certaines solutions logicielles existent, par exemple pour limiter l'accès aux supports amovibles à l'administrateur.

Une solution intermédiaire serait sans doute plus adaptée, par exemple pour assurer un filtrage des types de fichiers importés depuis les lecteurs amovibles. Cela nécessiterait un outil fonctionnant sur le principe des "antivirus temps réel", qui vérifierait chaque fichier avant ouverture, avec suppression éventuelle des codes exécutables.

4.11 Contrôle des actions au niveau système

Pour obtenir un niveau de sécurité plus élevé, et empêcher tout fichier importé d'exécuter une action malveillante, l'idéal serait à priori de contrôler chaque action effectuée au niveau système, en distinguant les fichiers suivant leur provenance et la confiance accordée par l'utilisateur ou l'administrateur. Cela pourrait par exemple passer par la signature de code ou des fichiers de données, en associant à chaque fichier des droits.

Il existe des produits comme Tiny Personal Firewall qui associent un parefeu personnel (filtrage réseau par application) à une sandbox (environnement restreint) pour l'exécution des programmes. Chaque exécutable du système est reconnu de façon unique par une empreinte de type MD5, et il est associé à un profil d'exécution qui lui donne accès à tout ou partie des fonctions du système. Il est ainsi possible de laisser les outils et applications de confiance fonctionner normalement, tout en restreignant les autres exécutables. Dès qu'un exécutable inconnu est lancé par l'utilisateur, celui-ci est prévenu par un message d'alerte et il peut lui attribuer un profil ou bien le bloquer.

Cette approche est pour l'instant relativement nouvelle dans l'environnement Windows, et les outils actuels sont difficilement déployables dans un cadre général. L'attribution optimale des profils est une tâche complexe, et l'utilisateur peut facilement trouver le système trop contraignant s'il est mal configuré. De plus, le filtrage est exclusivement réservé aux fichiers exécutables, il ne couvre donc pas toutes les menaces abordées dans cet article.

4.12 Contrôle d'intégrité régulier

Il existe de nombreux logiciels capables de vérifier l'intégrité d'un système. Cette protection se limite généralement aux fichiers qui ne doivent pas être modifiés, en particulier les exécutables du système d'exploitation et des applications, ainsi que les fichiers de configuration sensibles.

Il est possible de mettre à profit ces outils pour détecter toute apparition de fichier exécutable ou de script dans des zones de disque où cela est anormal. Certains outils proposent également la vérification de clés de registre, ce qui permet de détecter les codes malveillants utilisant les clés de type "Run" pour être activés à chaque démarrage du système.

Le contrôle d'intégrité doit cependant être considéré comme une mesure complémentaire, surtout destinée à détecter les symptômes d'une attaque réussie.

4.13 Mesures organisationnelles

Des mesures organisationnelles sont évidemment nécessaires, car le risque d'exécution de code malveillant provient avant tout de l'utilisateur qui importe les fichiers ou qui les ouvre. Voici quelques mesures utiles :

- Sensibilisation régulière des utilisateurs, si possible avec démonstration concrète de code malveillant dans des documents standards.
- Sensibilisation renforcée des administrateurs : ils doivent éviter au maximum d'ouvrir des fichiers importés lorsqu'ils sont connectés sous un compte d'administration. Ils doivent donc utiliser au maximum un compte utilisateur nominatif non privilégié.
- Interdiction d'ouvrir des fichiers importés ou des e-mails sur un serveur.
- Tout fichier importé doit être vérifié par au moins un antivirus à jour.
- Tout fichier suspect doit être signalé à la personne en charge de la sécurité du réseau.

5 Conclusion

La menace liée aux fichiers importés sur un intranet est réelle, que ceux-ci proviennent des connexions réseau avec l'extérieur ou de supports amovibles. En effet de nombreux formats de fichiers peuvent contenir du code malveillant, qui s'exécute avec les droits de l'utilisateur qui les ouvre.

Cependant cette menace est habituellement mal ou sous-évaluée, et les outils disponibles sont aujourd'hui peu adaptés pour assurer la sécurité globale d'un réseau d'entreprise vis-à-vis des fichiers importés. Il reste donc beaucoup de recherche à mener dans ce domaine.

Il existe cependant de nombreuses mesures techniques et organisationnelles qui peuvent être mises en place afin d'améliorer la sécurité et sensibiliser les utilisateurs au problème du code malveillant dans les fichiers.

Références

1. *Microsoft Office 97 Executable Content Security Risks and Countermeasures*, NSA. *http ://www.nsa.gov/snac/emailexec/guides/eec-3.pdf*
2. *Microsoft Office 2000 Executable Content Security Risks and Countermeasures*, NSA. *http ://www.nsa.gov/snac/emailexec/guides/eec-4.pdf*
3. RFC 2557, *MIME Encapsulation of Aggregate Documents, such as HTML (MHTML)*
4. Carl Orthlieb, *Adobe Acrobat Security Issues : The Open File Action and File Attachment Annotations*, *http ://www.adobe.com/products/acrobat/pdfs/OpenFilenAttach.pdf*
5. Patrick Chambet, *Sécurisation d'Internet Explorer et d'Outlook Express*, *http ://www.chambet.com/publications/ie-oe-security/index.html*