Formats de fichiers et code malveillant

SSTIC 03 – 10/06/2003

Philippe Lagadec
DGA / CELAR
philippe.lagadec@ifrance.com

Sécurité d'un réseau d'entreprise

- Essentiellement aux **interconnexions réseau**:
  - Filtrage réseau et applicatif
    - Routeur filtrant
    - Pare-feu
    - DMZ, passerelle
  - Antivirus: détection des virus connus
    - Serveurs et postes clients
    - Serveurs de messagerie
    - Proxy Web
  - Détection d'intrusion

- …Mais les **fichiers importés** sont rarement filtrés et peuvent constituer une menace pour le système d'information.

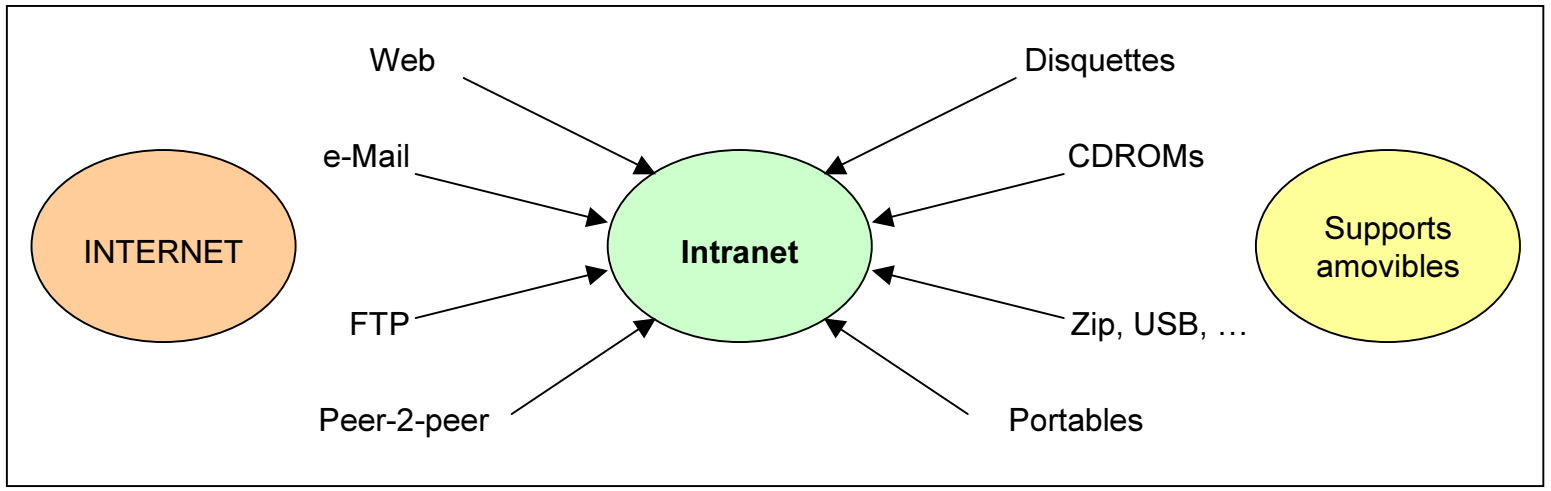
Sommaire

- Cette présentation se penche sur le cas d'un utilisateur qui ouvre un fichier quelconque sur son poste de travail Windows standard.
  - poste Windows NT/2000/XP standard avec MS Office
  - Fichier ouvert par double-clic dans Explorer, ou menu Fichier/Ouvrir dans une application

- **1) Menaces liées aux fichiers et au code**
- **2) Formats de fichiers**
- **3) Solutions de sécurité**

Importation de fichiers

- **Internet**: Web, e-mail, FTP, peer-2-peer, …
- **Supports amovibles**: disquette, CDROM, Zip, USB, portable connecté au réseau, …



Scénario d'ouverture d'un fichier

1. L'utilisateur importe ou reçoit un fichier.
2. Le **nom du fichier** lui indique la nature de son contenu a priori.
3. L'utilisateur **ouvre le fichier** (par double-clic ou dans une application).
4. Vérification préalable par un **antivirus** (automatique ou volontaire) => les virus connus sont bloqués.
5. Si le fichier contient du code, celui-ci peut s'exécuter **avec les droits de l'utilisateur**, et agir en son nom.
   - Problème: Cela peut être un **code malveillant**, agissant à l'insu de l'utilisateur.

Définition: code

- Un code est une information qui peut être interprétée ou exécutée pour déclencher des **actions** sur le système
- Diverses formes de code:
  - Code binaire exécutable (code machine)
  - Script ou macro
  - Ligne de commande
  - …
- L'exécution d'un code dépend de l'application qui ouvre un fichier.

Formats de fichiers et code

- **Exécutables ou scripts**
  - Contiennent toujours du code: problème de sécurité évident.
  - Majorité des virus et chevaux de Troie connus.
- **Documents actifs**: MS Office, HTML, …
  - Peuvent contenir du code (macros, scripts)
- **Documents statiques**: images, texte brut, …
  - Ne contiennent jamais de code, de par leur structure
    - …sauf si code « dormant » activé par un autre fichier
    - ou débordement de buffer (exemple:MP3 et Windows XP)

Menaces liées au code malveillant

- Possibilités d'actions malveillantes:
  - 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 est administrateur).
  - Destruction ou falsification de données.
  - Ecoute des mots de passe saisis au clavier ou circulant sur le réseau.
  - …

Définition d'un code malveillant

- Agit sur le système d'information sans que l'utilisateur l'ait voulu
  - Notion intuitive, mais difficile à formaliser
    - Dépend de la volonté de l'utilisateur…
- Porte atteinte au système d'information
  - En **intégrité**: modification de fichier, registre, mémoire, base de données, annuaire, …
  - En **disponibilité**: perturbe ou rend inaccessible un service du poste ou du réseau
  - En **confidentialité**: émission d'informations confidentielles vers l'extérieur

Code malveillant / code normal

- Frontière très difficile à déterminer automatiquement
  - Liée au contenu du SI et à sa politique de sécurité… et à la volonté de l'utilisateur !

- Pour détecter un code malveillant:
  - **Fichier Exécutable**: nécessite désassemblage (long et manuel, résultat non garanti)
  - **Document actif** (macro/script): en théorie plus facile car langage limité
    - un code non offensif se limite généralement à de l'affichage, du calcul ou de la fusion de données
    - Donc les mots-clés du langage correspondant à des **actions « à risque »** (actions sur fichiers, registre, réseau, …) peuvent être détectés

Formats de fichiers

Quelques exemples de formats de fichiers classiques dans un environnement **Windows** standard avec MS Office.

Remarque: La même étude pourrait être menée sur Linux ou un autre OS.

Associations de types de fichiers

- Sous Windows, le type d'un fichier est indiqué par son **extension**
  - Exemple: « rapport2003.**html** »
- Ce type détermine **l'application** lancée lorsque l'utilisateur ouvre le fichier
  - Pour HTML: Internet Explorer par défaut
- Association type / application stockée dans la base de registre.
  - Sous Windows NT/2000/XP, les commandes « assoc » et « ftype » affichent les types enregistrés.

Exécutables binaires

- Extensions: **EXE, COM, SCR, CPL, OCX, DLL…**
- Code directement exécutable par le processeur
- Formats qui posent le plus grand problème de sécurité (majorité des virus)
- EXE, SCR: format toujours binaire, avec une entête commençant toujours par « MZ » ou « ZM »
- COM: format brut sans entête, binaire ou texte
- CPL, OCX, DLL: même format que EXE, mais ne peuvent être exécutés directement par un double-clic de l'utilisateur

Fichiers de commandes Windows

- Extensions: **BAT, CMD**
- Format texte
- Suite de commandes exécutées directement, qui peuvent porter atteinte à la sécurité du système
  - Exemple: Création d'un compte administrateur avec un mot de passe connu
    - Net user pirate pirate /add
    - Net localgroup administrateurs pirate /add
  - Activation d'un service de commande à distance
    - Net start telnet (sous Windows 2000, 2003 ou XP)

Raccourcis et « Program Info File »

- Extensions: **LNK**, **PIF**
- LNK: raccourci Windows
- PIF: Informations pour le lancement d'un programme MS-DOS sous Windows
  - Fenêtre ou plein écran, mémoire étendue, …
- Lance une ligne de commande: risque similaire aux fichiers BAT
- Extension PIF/LNK toujours masquée dans Explorer
- Un fichier EXE ou COM renommé en PIF/LNK reste généralement exécutable !

Scripts du Windows Scripting Host

- Extensions: **VBS, JS, VBE, JSE**
- WSH outil optionnel, installé par défaut
- Interpréteur de scripts, par défaut **VBScript et Jscript** (possibilité d'ajouter Perl, TCL, …)
- Langages plus évolués que Batch, avec accès aux fichiers, au registre, aux applications par ActiveX
  - Exemple: virus ILOVEYOU, qui accédait au carnet d'adresses d'Outlook Express et se répliquait par e-mail
- WSH rarement employé par utilisateurs normaux: peut être généralement désactivé
- VBE, JSE: scripts chiffrés par MS Script Encoder

Exemples de VBScript et JScript

- Scripts qui créent un fichier:

VBScript:

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

JScript:

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

Autres scripts

- **Nombreux langages de scripts**:
  - Python, Perl, AWK, TCL, PHP, bash, KiXtart, …
- Chacun nécessite que l'interpréteur correspondant soit installé, sinon pas d'exécution possible
- Donc risque moins important que Vbscript et Jscript
  - Mais cela dépend des applications installées

HTML

- Extensions: **HTML, HTM**, …
- Document texte pouvant contenir des **scripts**, et faire appel à des objets ActiveX.
- Ouvert par défaut par **Internet Explorer:**
  - Support de **VBScript** et **Jscript**
    - Jscript est un sur-ensemble de Javascript, avec les mêmes fonctions que VBScript, et donc les mêmes problèmes de sécurité
  - **Zones de sécurité**, qui permettent d'interdire des actions dangereuses, ou demander confirmation.
    - …Mais nombreuses **failles** qui permettent de contourner cette protection, et accéder au système. (mise à jour régulière d'IE indispensable)
  - Si autre navigateur (Netscape, Mozilla, …): seul Javascript est accepté, il ne peut accéder aux fichiers et au registre.

HTML et scripts

- Un script peut être placé à de nombreux endroits dans un page HTML:
  - Entre les balises <SCRIPT> et </SCRIPT>
  - Dans une URL: <A HREF=«javascript:…»>
    - Note: pour IE, javascript signifie toujours Jscript, qui est aussi dangereux que VBScript.
  - Pour un événement sur un objet, comme « onLoad » ou « onMouseOver »:
    - <BODY ONLOAD=« … »>
- Si action sur fichiers ou registre, par défaut demande de confirmation car utilise ActiveX
  - Dépend du paramétrage d'IE

Exemple de script dans HTML

- HTML avec Script qui crée un fichier:

```
<html>
<SCRIPT>
var fso, tf;
fso = new ActiveXObject("Scripting.FileSystemObject");
tf = fso.CreateTextFile("c:\\temp\\test.txt", true);
tf.WriteLine(" test JScript.") ;
tf.Close();
</SCRIPT>
Voici une simple page HTML avec un script.
</html>
```

Documents MS Office

- Extensions: **DOC, XLS, PPT, MDB**, et nombreuses autres…
- Format binaire commun « CDA »
- Peuvent contenir des **macros**, selon leur nom elle peuvent être lancées dès l'ouverture du document
- Par défaut, **demande de confirmation** à l'utilisateur si présence de macros.
  - … sauf pour Access !
- **Niveau de sécurité** paramétrable pour les macros:
  - Confirmation (par défaut)
  - Seulement les macros signées (depuis Office 2000)
  - Ou pas de confirmation du tout

Documents MS Office renommés

- Particularité des documents MS Office:
  - Ils peuvent être **renommés avec une extension inconnue**, et la bonne application est quand même lancée par Explorer !
  - Exemple: rapport**.doc** → rapport**.xyz**
  - Explorer (ou plutôt une DLL installée par MS Office) vérifie donc le **contenu** d'un fichier pour choisir l'application à lancer lorsqu'il ne connaît pas son **extension**.
- La même fonction peut être employée par d'autres applications que MS Office.
- **Conclusion**: le nom d'un fichier ne suffit pas toujours à déterminer son type.

Objets OLE

- De nombreuses applications permettent l'inclusion d'objets OLE dans des documents: Word, Excel, WordPad, …
  - Exemple: inclusion de tableau Excel dans Word
- Un **objet OLE « Package »** peut contenir tout fichier (y compris exécutable), qui sera ouvert si on double-clique sur son icône.
- C'est à chaque application d'assurer la sécurité, par exemple une demande de confirmation.
- Exemples:
  - Word demande bien confirmation, avec mise en garde.
  - WordPad ne demande rien, exécution directe !
  - => problème des fichiers Word renommés en .wri

Documents RTF, Rich Text Format

- Extension: **RTF**
- Format texte « portable », pouvant reproduire la majorité des mises en forme de Word.
- Meilleure sécurité, car absence de macros et de liens URL.
  - Souvent recommandé comme format d'échange car plus sûr vis-à-vis des virus.
- Cependant le contenu n'est pas 100% statique, car on peut y inclure des **objets OLE**.
- De plus un document RTF peut être renommé en WRI pour être ouvert par WordPad et non Word.

Shell Scrap ou « fichier Bribes »

- Extension: **SHS**
- Format binaire correspondant à un objet OLE indépendant
  - Obtenu par glisser-déposer depuis WordPad vers Explorer, par exemple (sous Windows NT)
- **Multiples problèmes de sécurité**:
  - Extension SHS toujours masquée
  - Lancement direct du fichier inclus
  - Icône ressemblant à celle d'un fichier texte
  - Le fichier peut être renommé, par exemple « rapport.txt.shs »

Acrobat PDF

- Extension: **PDF**
- Format portable, jugé sûr
  - Largement utilisé pour l'échange de documents
- Cependant, possibilité méconnue de lancer des **commandes** externes à l'ouverture ou la fermeture du document.
  - Simple confirmation demandée
  - Possibilité de sécurisation par une clé de registre.

Autres formats

- De nombreux autres formats classiques peuvent poser des problèmes de sécurité:
  - XML, WSH, WSF, MHT, HLP, CHM, REG, EML, MSG, URL, …
- La même étude pourrait être menée sur Linux ou d'autres systèmes, avec des résultats similaires

Synthèse: Ouverture d'un fichier sous Windows

- Mécanisme d'ouverture d'un fichier sous Windows avec MS Office installé:

1) Exécutable ? (EXE, COM, …) → Lancement direct

↓

2) Extension connue, associée à une application ? → Ouverture du fichier par l'application

↓

3) Fichier MS Office renommé, reconnu par son contenu ? → Ouverture du fichier par l'application Office ad hoc

↓

4) Sinon, **fichier non reconnu** → Erreur, ou boîte de dialogue pour choisir une application

Classification des formats de fichiers vis-à-vis du code

- **Risque classé de 1 à 5:**
  - 1) Format contenant toujours du code
  - 2) Format contenant parfois du code, et celui-ci peut s'exécuter directement
  - 3) Format contenant parfois du code, et celui-ci ne peut s'exécuter qu'après confirmation
  - 4) Format contenant parfois du code, et celui-ci ne peut s'exécuter qu'après action volontaire de l'utilisateur
  - 5) Format ne contenant jamais de code

- **Conteneur**:
  - format pouvant contenir d'autres fichiers (archives zip, documents Word, …)

Synthèse : Classification des formats de fichiers

Format	Extension(s)	Risque	Conteneur	Type
Exécutable binaire	EXE, SCR	1	X	Binaire
Exécutable binaire	COM	1	X	Binaire/Texte
Fichier batch	BAT, CMD	1		Texte
Scripts WSH	VBS, JS, VBE, JSE, …	1		Texte
Base Access	MDB	2	X	Binaire
HTML, XML	HTM, HTML, XML, …	2		Texte
Document Word	DOC, DOT, WBK, …	3	X	Binaire
Document PDF	PDF	3		Texte
Document RTF	RTF	4	X	Texte
Image Bitmap	GIF, JPEG, PNG, BMP	5		Binaire
…	…	…	…	…

Synthèse : Classification des formats de fichiers

- Ce type de classification est très utile pour effectuer une **analyse de risque** concernant les fichiers importés dans un système d'information.

- Elle permet de bâtir une **politique de sécurité ou de filtrage** basée sur les formats de fichiers, adaptée aux besoins.

Solutions

- **1) Sécurisation au niveau des entrées de fichiers:**
  - Filtrage des **entrées réseau** (e-mail, web, …)
  - Sécurisation des **supports amovibles**
- **2) Sécurisation du poste de travail**
  - Système d'exploitation
  - Applications
- **3) Mesures organisationnelles**

1) Sécurisation des entrées de fichiers

- **Passerelle de filtrage réseau applicatif**
  - Filtrage des e-mails, pages web, transferts FTP, …
  - Blocage ou nettoyage des fichiers avec contenu actif (exécutables, scripts, macros)
  - Un filtrage simple par nom ou type MIME ne suffit pas: **analyse par contenu** et **récursive** nécessaire.
    - Car formats conteneurs et renommages possibles

Nettoyage des fichiers avec code

- **Suppression directe de code**
  - Suppression de macros (exemple: F-Prot)
  - Suppression de scripts dans HTML

- **Conversion de format**
  - Exemples: Word ou HTML → RTF ou PDF
  - Efficace, mais très contraignant pour l'utilisateur (perte de mise en forme voire d'information)

1) Sécurisation des entrées de fichiers

- **Supports amovibles:**
  - À neutraliser physiquement pour un système très sensible (rarement applicable)
  - Neutralisation logicielle: l'accès peut être réservé à l'administrateur
  - Pas vraiment de solution satisfaisante aujourd'hui pour protéger l'utilisateur

2) Sécurisation du poste de travail

- **Paramétrage sécurisé** du système d'exploitation et des applications
  - Exemples: Office, Acrobat, IE
- **Mise à jour régulière** du système et des applications (IE, Office, …)
- **Antivirus** sur chaque poste, avec mise à jour automatique

2) Sécurisation du poste de travail

- **Contrôle des actions au niveau système**
  - Exécution des applications dans un « bac à sable », comme la sandbox de Java
  - Exemple: pare-feu personnel Tiny Personal Firewall v4
    - Efficace, mais **complexe** à mettre en œuvre et **limité aux exécutables**
    - Peut bloquer tout exécutable inconnu avant son exécution: pallie certains problèmes des pare-feux personnels classiques.

- **Contrôle d'intégrité régulier**
  - Détection des fichiers exécutables inconnus
    - Détection a posteriori, ce n'est pas une protection.

3) Mesures organisationnelles

- **Sensibilisation** régulière des utilisateurs et administrateurs
  - Démonstrations concrètes pour marquer les esprits
- Procédures de **mise à jour** des OS, logiciels, antivirus
- Interdiction aux administrateurs d'ouvrir des fichiers importés sous leur compte d'administration

Autres solutions ?

- Le problème de base est que les **fichiers de contenu inconnu** sont ouverts avec les même droits que les **fichiers de confiance**
  - Signature des fichiers de confiance ?
  - Détecter les fichiers inconnus à leur ouverture, et leur appliquer un filtrage/nettoyage comme sur une passerelle réseau ?
    - Solution pour supports amovibles
  - Exécuter les applications qui ouvrent ces fichiers dans un « bac à sable » pour limiter les risques ?

Conclusion

- **La menace liée aux fichiers importés par réseau ou support amovible est bien réelle, et souvent sous-estimée. Il faudrait l'aborder de façon plus globale.**
  - Cette menace provient des nombreux formats de fichiers pouvant contenir du **code malveillant,** et du fait que de nombreux fichiers au contenu inconnu sont ouverts **sans contrôle** par les utilisateurs.
- Les solutions de sécurité disponibles aujourd'hui ne sont pas vraiment adaptées et exhaustives.
  - Cependant quelques mesures techniques et organisationnelles sont déjà envisageables.
  - La conception de **nouvelles solutions** est nécessaire.

Questions