Linux et virus

Eric Martel

7 février 2004

1 Introduction

Le 26 janvier 2004 était découvert le virus MyDoom¹, lequel s'est répandu comme une traînée de poudre par son habileté — habituelle pour les virus récents — de s'auto-envoyer par courriel à ceux apparaissant dans le carnet d'adresse Outlook ou Outlook Express de l'ordinateur infecté. Et le 27 janvier, je vécus une situation quelque peu frustrante pour l'heureux utilisateur Linux² que je suis...

En effet, voulant contribuer à l'effort de guerre (même si elle ne me touchait pas vraiment), j'avais envoyé au matin du 27 janvier au responsable de la sécurité informatique de l'Université Laval un message auquel j'avais joint des copies de messages infectés que j'avais reçus et dont l'adresse IP d'origine pointait vers le sous-réseau de l'Université Laval, histoire d'aider à débusquer les machines infectées. Or, j'avoue humblement que je n'avais aucunement imaginé le scénario qui s'ensuivit : mon message fut intercepté par l'Université comme étant porteur du virus, et il n'en fallait pas plus pour que mon poste soit déclaré infecté, ce qui conduisit à l'arrêt *manu militari* de mon poste par mon responsable réseau, lequel est totalement étranger au monde UNIX. Lorsque je lui expliquai que mon poste *ne pouvait* être infecté par un virus Windows, il me dit : « Qu'est-ce que tu en sais ? Vous autres, vous dites toujours que Linux est plus sécuritaire et tout, mais tu ne sais pas : peut-être que le virus peut se servir de ta partition Windows ou quelque chose du genre. » (Je suis en *dual boot* avec WindowsNT.³)

Estomaqué devant cette phrase (qui marquait l'absence de connaissance de la structure fine de la gestion des permissions, de la mémoire et du format des exécutables des différents systèmes d'exploitation), je lui dis que ce n'est pas une question de religion, qu'il n'y a pas de « foi » làdedans, que c'est purement technique et facilement démontrable. Ce à quoi il a dit que, non, selon lui ça ressemble un peu à une religion et que, oui, selon lui il y a un peu de foi là-dedans.

Troublé, je me demandai ce que je pouvais faire pour tenter d'expliquer le pourquoi du comment des choses. Alors me vint l'idée d'écrire ce document.

Dans la foulée, je me suis dit que, tant qu'à faire, pourquoi ne pas écrire un document qui pourra profiter au plus grand nombre de gens possible? Voilà pourquoi je m'évertuerai ici à expliquer le plus simplement possible ce qu'il y a à savoir en virologie informatique, en espérant que les

¹http://securityresponse.symantec.com/avcenter/venc/data/w32.novarg.a@mm.html

²Pour ceux qui ne sauraient pas ce qu'est Linux, il s'agit d'un système d'exploitation au même titre que MS-Windows et MacOS; voir la section 4.1.

³Ceci signifie que je peux, au démarrage de mon ordinateur, choisir d'une fois à l'autre entre WindowsNT et Linux comme système d'exploitation.

débutants autant que les intermédiaires pourront y trouver de quoi combler leurs lacunes en ce domaine.

N'étant pas moi-même « gourou » à proprement parler dans le domaine de la virologie informatique, il ne faudra pas m'en vouloir si jamais j'omets quelques détails par ci et là.

2 La notion de virus informatique

J'aimerais juste ici rappeler quelques notions de base, pour savoir de quoi on parle exactement, et définir les différentes classes de virus informatique.

2.1 Un virus est un programme

Un virus informatique, c'est d'abord *un programme informatique*, au même titre que Microsoft Word, Microsoft Outlook ou Internet Explorer, pour ne nommer que quelques-uns des plus connus en général. Un programme, c'est une suite de commandes qu'on donne à l'ordinateur. Par exemple, dans le cas de MS-Word, le programme donne à l'ordinateur des commandes du genre « Affiche des icônes dans le haut de la fenêtre ; si l'utilisateur clique sur l'icône *Caractères gras*, mets les caractères en gras », etc. (évidemment, je schématise ici). Ce qui distingue un virus des programmes habituels, c'est qu'un virus informatique est un *programme malveillant*, destiné à donner des commandes indésirables et nuisibles à l'ordinateur (par exemple, « Efface le contenu du disque dur »).

Dans les faits, définir un virus n'est pas si simple, et un consensus absolu n'existe pas vraiment quant à ce que l'on devrait véritablement appeler un virus informatique; plusieurs distinguent notamment les virus des chevaux de Troie et d'autres programmes malicieux.

2.2 Virus, vers, chevaux de Troie et logiciels espions

Par analogie avec les virus naturels, plusieurs puristes croient qu'on ne devrait appeler virus que les programmes informatiques capables de se répliquer et de se répandre que par l'infection d'un autre programme dont ils utilisent le statut d'exécutable. Un peu comme le virus du SIDA qui est capable de s'incorporer à l'ADN humain pour se cacher et se faire répliquer, les « vrais » virus vont s'insérer au travers du code d'autres programmes — idéalement sans compromettre leur action normale — qui leur serviront alors de tremplin. Les plus vieux se rappelleront du virus Michelangelo⁴ qui était de ce type et dont les médias avaient beaucoup parlé. Ce type de « vrai » virus est cependant de plus en plus rare : il a eu son âge d'or avant Internet, lorsque le principal mode de propagation était par disquette : ces virus infectaient habituellement le *boot sector* des disquettes et des disques durs d'où ils pouvaient par la suite infecter d'autres disquettes et des programmes situés sur les ordinateurs où ils étaient insérés. Un concept important à retenir, c'est que ces virus sont cachés et ne requièrent pas une action humaine délibérée pour s'exécuter (outre l'exécution de programmes légitimes) : ils s'exécutent et se reproduisent à l'insu des utilisateurs jusqu'au moment où, selon le bon loisir de leur programmeur, ils manifesteront leur existence par une action visible quelconque (s'ils le font jamais).

⁴http://securityresponse.symantec.com/avcenter/venc/data/stoned.michelangelo.html

Avec l'avènement d'Internet, le « vrai » virus a plutôt « évolué » en ver. Le ver n'infecte pas d'autres programmes et se réplique en utilisant Internet comme voie de propagation. En fait, beaucoup ne s'entendent pas sur ce que l'on devrait appeler un ver : certains disent que, dès qu'il utilise Internet pour se propager, il s'agit d'un ver; d'autres disent qu'en plus de cela il doit aussi et nécessairement être autonome. Un exemple récent de cette dernière définition plus restrictive fut MSBlast⁵ qui, sans aucune intervention humaine, pouvait exploiter une faille dans les systèmes d'exploitation Windows 2000 et XP pour infecter une machine d'où il pouvait ensuite se propager, seul, vers d'autres machines en utilisant Internet. Les mêmes puristes classeront MyDoom plutôt dans la catégorie des chevaux de Troie⁶, car il faut absolument pour l'exécuter que quelqu'un le fasse délibérément. Ainsi, MyDoom se présente sous les traits d'un fichier attaché dans un courriel que l'utilisateur doit explicitement exécuter : le programme malicieux ici n'est pas caché dans le code d'un autre programme : le fichier attaché est le programme malicieux. Le tout est de convaincre l'utilisateur de l'exécuter (on dirait que peu de gens se font prier, mais bon je ne me lancerai pas là-dessus...). Vous devinerez que ceux pour qui un ver est simplement défini par sa capacité d'utiliser Internet à ses fins de propagation définiront un cheval de Troie comme un programme malicieux que l'on doit explicitement exécuter, mais qui ne ne se répliquera ni ne se propagera d'aucune façon.

Finalement, beaucoup pensent que les logiciels espions devraient aussi être considérés comme des virus. Beaucoup de ces logiciels s'installent à l'insu de l'utilisateur avec certains gratuitiels et rapporteront des informations plus ou moins personnelles à leur auteur, comme les habitudes de navigation Web, etc. En fait, beaucoup définissent un virus comme étant n'importe quel sorte de programme qui fait des actions non désirées par l'utilisateur et à son insu.⁷

Dans la suite de ce document, le terme « virus » sera utilisé pour désigner globalement les « vrais » virus, les vers et les chevaux de Troie. Un terme plus précis sera utilisé lorsque jugé opportun.⁸

3 La notion de programme informatique

J'ai expliqué en 2.1 qu'un virus est un programme informatique, et j'ai expliqué qu'un programme était une suite d'instructions données à un ordinateur. Je veux ici développer plus en détail ce concept en expliquant *comment* ces instructions sont données à l'ordinateur; tous les moyens qui seront décrits ici de donner des instructions à un ordinateur peuvent être utilisés par un virus.

3.1 Le langage machine

Chaque ordinateur est muni d'au moins un processeur. Un processeur — ou unité principale de contrôle (*Central Processing Unit* ou CPU en anglais) — est chargé de recevoir et d'exécuter

⁵http://securityresponse.symantec.com/avcenter/venc/data/w32.blaster.worm.html

⁶Pour ceux qui auraient oublier leurs notions de mythologie grecque et qui seraient intéressés à connaître l'origine de cette expression, voir http://www.ac-versailles.fr/pedagogi/anti/troie/troie0.htm.

⁷À ce titre, les logiciels de la suite MS-Office pourraient être considérés comme des virus; voir http://www.transfert.net/a8977.

⁸Pour les intéressés, vous trouverez sur le Web plus de détails sur l'histoire des virus, comme sur http://www.cknow.com/vtutor/vthistory.htmpar exemple.

ce qu'on appelle des *instructions élémentaires*. Chaque type de processeur comprend un certain nombre d'instructions élémentaires, dont l'ensemble constitue un *langage machine*. Ces instructions varient d'un processeur à l'autre, de telle sorte que deux processeurs différents ne comprendront pas le même langage. Ainsi, un processeur PowerPC, qu'hébergent les ordinateurs Macintosh, ne pourra pas exécuter les instructions élémentaires écrites pour un processeur Pentium qu'hébergent certains ordinateurs de type PC⁹.

Bien qu'on ait attribué à chaque instruction élémentaire une étiquette permettant à un humain de plus facilement la reconnaître et la manipuler (dont l'ensemble est appelé *langage assembleur*), écrire un programme en assembleur est extrêmement long et fastidieux. Néanmoins, il fut un temps où c'était la norme et où de nombreux virus étaient écrits en assembleur. Aujourd'hui, des langages de plus haut niveau permettent aux humains d'écrire des programmes de façon plus simple : des *compilateurs* et des *interpréteurs* se chargeront de traduire un langage de haut niveau en langage machine (le seul que l'ordinateur, ultimement, peut comprendre).

3.2 Les langages compilés

Un langage compilé est un langage de programmation de haut niveau qu'un humain peut facilement lire et comprendre (pourvu qu'il connaisse le langage, évidemment). Un compilateur est un programme déjà fini et exécutable (donc, vous comprendrez que les tout premiers compilateurs étaient écrits en assembleur...) capable de comprendre un certain langage de haut niveau, de le traduire en langage machine et d'en faire un fichier appelé *exécutable binaire* (on précise « binaire » pour bien signifier qu'il s'agit de langage machine — nous verrons plus loin que d'autres types d'exécutables existent). Un exécutable binaire est donc un programme compilé, c'est-à-dire un fichier contenant des instructions élémentaires pour l'ordinateur.

Des exemples connus de langages compilés : Fortran, BASIC, Pascal, C, C++, Cobol

3.3 Les langages interprétés

Les langages interprétés sont des langages de très haut niveau qui ne seront pas directement traduits en fichiers exécutables binaires. Tout comme pour un langage compilé, le langage interprété sera lu par un programme déjà existant (l'interpréteur dans ce cas ci); cependant, en gros, ce programme transmettra directement les instructions élémentaires au processeur plutôt que de les inscrire dans un fichier exécutable binaire. Un langage interprété est habituellement plus souple qu'un langage compilé et est aussi plus simple à concevoir, de là la possibilité d'écrire très rapidement des programmes qui prendraient beaucoup plus de temps à écrire avec la plupart des langages compilés. Un autre avantage est qu'un programme interprété sera directement portable, c'est-à-dire qu'il pourra être exécuté sur n'importe quel ordinateur sur lequel un interpréteur adéquat est installé, peu importe le type de processeur ou le système d'exploitation à partir duquel il a été initialement écrit (alors qu'un programme écrit en langage compilé devra nécessairement l'être sur un ordinateur du type ciblé). De tels programmes écrits en langage interprété sont appelés scripts. Comme la plupart des systèmes d'exploitation où l'interpréteur adéquat sera présent seront

⁹Il est à noter ici que certains processeurs sont dits *compatibles* si, bien que construits différemment, ils partagent les mêmes instructions élémentaires de base; par exemple, les processeurs Athlon de la compagnie AMD sont compatibles avec les processeurs Pentium de la compagnie Intel, bien que leur architecture soit différente.

en mesure d'exécuter un script, on répertorie parfois également ces programmes sous le terme général d'exécutables non binaires.

Des exemples connus de langages interprétés : VBScript, JavaScript, Perl, awk, tcl

3.4 Les langages semi-compilés

En fait, personnellement, je n'en connais qu'un seul, soit le Java (il y en a peut-être d'autres, mais bon je n'ai pas trop cherché...). Le Java est particulier en ce sens qu'un programme écrit en Java sera d'abord « compilé » dans un fichier, lequel ne sera cependant ni un fichier exécutable binaire, ni un fichier script. Par contre, il s'agira d'un fichier exécutable non binaire qui devra être passé à un interpréteur Java pour être exécuté. L'avantage perçu de ce système est qu'il permet de cacher le code Java initial (protection de la propriété intellectuelle) tout en permettant d'écrire sur un ordinateur quelconque un programme qui pourra être exécuté sur n'importe quel autre type d'ordinateur grâce à l'interpréteur (portabilité).

4 Le système d'exploitation

4.1 Définition

Un système d'exploitation est, somme toute, une sorte de gros programme servant à gérer les ressources tant matérielles que logicielles d'un ordinateur. Des exemples connus de systèmes d'exploitation sont : DOS¹⁰, MS-Windows, MacOS, Linux, FreeBSD, UNIX.

Le système d'exploitation est le premier programme qu'exécute un ordinateur au démarrage¹¹. Pour des raisons évidentes, il s'agit nécessairement d'un exécutable binaire. Le système d'exploitation sert somme toute d'interface entre l'utilisateur et l'ordinateur, lui facilitant diverses tâches telles qu'accéder au disque dur, afficher de l'information à l'écran, interpréter les mouvements de la souris, etc. Sans un système d'exploitation, « exploiter » un ordinateur serait loin d'être à la portée du premier venu! Ce qu'il faut comprendre aussi, c'est que plusieurs systèmes d'exploitation très différents peuvent très bien être conçus pour un même type d'ordinateur et de processeur. Prenez WordPerfect et MS-Word, par exemple, qui sont tous deux des logiciels de traitement de texte permettant, en gros, d'effectuer les mêmes tâches, mais pas nécessairement de la même facon (interfaces différentes, menus différents, etc.); eh bien il en va de même pour les systèmes d'exploitation qui permettront, en gros, d'effectuer les mêmes tâches, mais pas nécessairement de la même façon (pour l'utilisateur). Ainsi, MS-Windows et Linux sont deux systèmes d'exploitation conçus pour fonctionner sur un ordinateur de type PC¹² qui vous permettront tous deux, en gros, d'effectuer les mêmes opérations avec votre ordinateur, mais de façon différente. Donc, si vous pensiez que vous étiez « obligé » d'avoir MS-Windows sur votre ordinateur, vous vous êtes trompé! Vous pourriez tout aussi bien avoir Linux ou d'autres systèmes d'exploitation encore. 13

¹⁰J'abrège volontairement ici toute la gamme des MS-DOS, PC-DOS, DR-DOS, etc.

¹¹Pour les pointilleux, je mets dans cette note que le vrai premier programme exécuté est en fait la ROM du BIOS, et qu'on pourrait très bien faire exécuter autre chose qu'un OS, mais bon je ne voulais pas trop entrer dans les détails...

¹²En fait, il existe aussi différentes versions de ces systèmes d'exploitation pour différents types d'ordinateurs.

¹³Et encore, plusieurs systèmes d'exploitation peuvent même être présents en même temps sur votre ordinateur (bien que ne pouvant pas être exécutés en même temps cependant — utilisateurs de VMWare, SVP ne pas intervenir...;)

4.2 La gestion de la mémoire et des exécutables

Un concept important que je veux maintenant introduire est celui de la façon dont un exécutable — binaire ou script — interagit avec le système d'exploitation. En effet, la plupart des systèmes d'exploitation récents exerceront un farouche contrôle sur les exécutables, afin justement d'éviter le plus possible que des dommages soient provoqués. Le but n'est pas seulement de protéger contre les virus, mais aussi contre les bogues (erreurs de programmation) et les tentatives d'intrusion/de piratage.

Puisque les systèmes d'exploitation veulent exercer un contrôle, ils doivent pour cela exiger des exécutables qu'ils respectent des normes très strictes en ce qui a trait à la structure qu'ils occuperont en mémoire¹⁴. Ces normes n'ont normalement pas à être exhaustivement connues du programmeur (sauf s'il programme en assembleur), car c'est le compilateur ou l'interpréteur, selon le cas, qui s'assurera que ces normes soient respectées en fonction du système d'exploitation auquel le programme est destiné. Ces normes seront différentes d'un système d'exploitation à l'autre, selon les règles que ses concepteurs auront bien voulu définir. Si un exécutable binaire ne rencontre pas ces normes, son exécution sera impossible puisque le système d'exploitation ne pourra pas le charger en mémoire. Ce qui amène un nouveau concept d'incompatibilité : en 3.1, j'avais introduit la notion d'incompatibilité entre processeurs, à cause des différents langages machines; j'introduis maintenant la notion d'incompatibilité entre systèmes d'exploitation, à cause des différentes normes de structures mémorielles. Autrement dit, même si deux exécutables binaires sont conçus pour le même processeur, et donc ont été traduits dans les mêmes instructions élémentaires, ils ne pourront s'exécuter que s'ils respectent en plus la structure attendue par le système d'exploitation ciblé pour leur mise en mémoire.

Évidemment, les lecteurs les plus futés se diront sans doute : « Mais, on vient de dire en 4.1 que le système d'exploitation lui-même est un programme exécuté au démarrage de l'ordinateur : il ne peut donc sûrement pas contrôler sa propre structure! » En effet, et c'est là le seul lieu ou l'incompatibilité entre systèmes d'exploitation n'a plus lieu, car si un virus est écrit pour prendre le contrôle d'un ordinateur au moment du démarrage, avant que le système d'exploitation ne soit exécuté, eh bien la seule incompatibilité demeurant est celle du langage machine : un tel virus ne pourra s'exécuter que sur certains processeurs et pas d'autres, mais sans égard au(x) système(s) d'exploitation utilisé(s). Cependant, le virus devra d'abord aller s'installer dans un endroit stratégique, ce qui implique une exécution initiale par l'intermédiaire d'un système d'exploitation : à moins que vous ne vous serviez d'une disquette de démarrage infectée, on ne se tire donc pas vraiment de la nécessité pour le virus d'adopter les normes mémorielles d'au moins un type de système d'exploitation.

4.3 Les librairies

Je parlais en 3.2 et 3.3 de langages de programmation dits de haut niveau. Souvent, les programmes écrits dans ces langages « ne réinventeront pas la roue », c'est-à-dire que s'il existe déjà sur l'ordinateur des exécutables binaires capables d'effectuer certaines tâches, pourquoi ne pas d'en servir ? Les systèmes d'exploitation fournissent aux programmes de tels exécutables précompilés et prêts à servir appelés *librairies* (les fichiers .dll sous Windows, entre autres, et .so

¹⁴En effet, je ne l'ai pas mentionné précédemment, mais pour pouvoir être exécutées par le processeur, les instructions élémentaires définies par un programme, quel qu'il soit, doivent d'abord être chargées en mémoire vive (RAM).

sous Linux). Ce ne sont habituellement pas des programmes en soit, mais plutôt des collections de fonctions regroupées dans un seul fichier : les programmes externes peuvent ensuite accéder à une fonction spécifique incluse dans une librairie et l'exécuter pour leurs besoins. De ce fait, si un programme est écrit pour utiliser une certaine fonction d'une certaine librairie d'un certain système d'exploitation, on se retrouve avec un niveau d'incompatibilité encore plus grand, car cette librairie, essentielle pour le programme, ne se retrouve peut-être pas dans toutes les versions d'un système d'exploitation donné (ce qui explique principalement, d'ailleurs, pourquoi certains programmes peuvent être exécutés sous WindowsXP, par exemple, mais pas sous Windows95). Et chez deux familles de systèmes d'exploitation complètement différentes, par exemple Windows et Linux, aucune des librairies de l'une n'est retrouvée chez l'autre (en plus des différences de structure mémorielle).

5 Les virus par rapport à Linux et MS-Windows

5.1 Un virus Windows ne peut infecter Linux

Si vous avez bien suivi les explications données jusqu'à maintenant, vous comprendrez pourquoi un virus binaire conçu pour s'exécuter sous MS-Windows ne peut être exécuté sous Linux : d'abord et surtout en raison des incompatibilités de structure mémorielle, ensuite en raison de la parfaite et totale inexistence de librairies MS-Windows sous Linux (dans le cas où un virus en aurait besoin, ce qui est souvent le cas chez les virus récents, notamment pour aller modifier la base de registre de MS-Windows). Et qu'en est-il des virus scripts, que j'ai affirmés portables en 3.3 ? Eh bien, effectivement, si l'interpréteur est présent sur le système, le virus pourra s'exécuter, peu importe sur quel système d'exploitation il se trouve. Cependant, il y a quand même quelques points à considérer :

- un virus script sera nécessairement non binaire, donc devra nécessairement être sciemment exécuté par l'utilisateur (cheval de Troie obligatoire 15);
- les virus scripts habituellement rencontrés sous MS-Windows sont surtout écrits dans le langage VBScript, or, ce langage n'est pas supporté par Linux (aucun interpréteur VBScript n'existe pour Linux).

De plus, il y a encore beaucoup d'autres obstacles à franchir ; par exemple, la structure de fichiers et de répertoires est différente sous Linux : un virus qui chercherait le « C : » sous Linux chercherait longtemps !

5.2 Des virus pour Linux? Oui, mais...

Linux est-il invulnérable contre les virus ? Pas du tout ! Le simple fait qu'il soit absolument et complètement immunisé contre les virus MS-Windows ne signifie pas pour autant que quelqu'un ne pourrait pas décider d'écrire un virus ciblant Linux ! Et, d'ailleurs, quelques-uns ont déjà été

¹⁵À noter cependant que de nombreuses failles de sécurité dans Outlook, Outlook Express, Internet Explorer et MS-Windows ont été découvertes par les années et mois passés qui permettaient l'exécution de programmes scripts à la simple ouverture d'un courrier électronique ou d'une page Web. Ces problèmes étaient tous dûs à la présence dans MS-Windows du *Windows Script Engine* (WSE); étant donné l'absence sous Linux d'un tel moteur de prise en charge des scripts, ce dernier est donc immunisé contre ce genre de vulnérabilité.

produits. ¹⁶ Seulement, où sont-ils ? Pourquoi n'y a-t-il pas une sorte de panique générale de même nature que celle que l'on observe pour les virus MS-Windows ? Pour plusieurs raisons. D'abord, il est clair que la rareté des ordinateurs Linux par rapport aux ordinateurs MS-Windows ne donnerait pas grand-chance à un virus Linux de faire beaucoup de ravages. Mais il y a également des considérations techniques.

5.2.1 Le système de permissions UNIX

Microsoft n'a pas cru bon d'offrir la possibilité à ses clients résidentiels de protéger leur système avant la venue de WindowsXP familial vers la fin de 2001. Sur Linux, depuis les tout débuts (1991), tout le système d'exploitation repose sur un système de gestion des permissions basé sur UNIX, lequel avait depuis longtemps fait ses preuve en termes de sécurité. Une de ces contraintes fait que chaque fichier reçoit, entre autres, la permission ou non de pouvoir s'exécuter et ce en fonction de l'utilisateur qui tente l'exécution. Pour faire la comparaison, traditionnellement dans MS-Windows le caractère exécutable d'un fichier est simplement donné par l'extension de son nom! De plus, tous les fichiers systèmes (incluant les fichiers des différents programmes) appartiennent par défaut à l'administrateur du système et sont verrouillés en écriture (mais pas nécessairement en exécution, puisqu'ils sont légitimes), empêchant du même coup toute infection et toute corruption par d'éventuels virus qui voudraient justement corrompre le système, infecter des exécutables binaires ou s'installer dans les fichiers de démarrage automatique¹⁷. Tous les utilisateurs de MS-Windows 95/98/ME, encore très nombreux, n'ont aucun moyen d'empêcher un virus d'aller s'enregistrer dans la base de registres ou même (j'en vois encore parfois!) dans l'autoexec.bat de manière à être exécuté à chaque nouveau démarrage de MS-Windows.

Même WindowsXP familial n'est pas la solution parfaite, car combien de propriétaires de ce système l'utilisent en mode de simple utilisateur, hmmm?¹⁸

5.2.2 Logiciels de courriel et fichiers attachés

Aussi, il y a la difficulté de trouver un consensus d'utilisation de logiciels. Par exemple, pour le courrier électronique, il est clair que, sur MS-Windows, Outlook et Outlook Express sont définitivement les rois : à cause de cela, tous les virus qui utilisent leurs carnets d'adresses pour s'envoyer par courriel à d'autres personnes sont sûrs d'avoir un certain succès. Mais, sur Linux, il n'y a pas un tel consensus : les logiciels de courrier électronique sont aussi nombreux que variés et, surtout, ne permettent pas d'exécuter un programme attaché simplement en cliquant dessus! Personnellement, certains logiciels de courrier que je connais sous Linux permettent d'ouvrir un fichier attaché avec un logiciel quelconque déjà présent sur le système, mais pas d'exécuter un programme (script ou binaire), car un programme attaché sera importé par le système d'exploitation comme un simple fichier n'ayant aucune permission d'exécution par qui que ce soit, y compris le propriétaire du fichier.

¹⁶Le plus connu est sans doute Bliss: http://www.europe.f-secure.com/v-descs/bliss.shtml

¹⁷Il est vrai cependant qu'un virus Linux pourrait modifier le .bashrc/.rshrc/.login de l'utilisateur et ainsi être exécuté à chaque fois que l'utilisateur ouvre une session.

¹⁸À remarquer d'ailleurs que beaucoup actuellement critiquent l'approche prise par la compagnie Lindows (http://www.lindows.com), laquelle vend des distributions Linux qui, par défaut, tout comme Windows, s'utilisent en mode administrateur.

Donc, imaginons le scénario suivant : un utilisateur Linux reçoit un courrier électronique dans lequel se trouve attaché un virus sous forme d'un exécutable (script ou binaire) pour Linux. L'utilisateur, pour activer le virus, doit faire les étapes suivantes :

- 1. sauvegarder le fichier attaché dans son répertoire de travail (le seul en principe auquel il a accès en écriture);
- 2. modifier les permissions sur le fichier pour le rendre exécutable ;
- 3. exécuter volontairement le fichier.

Voyons maintenant les étapes que doit suivre un utilisateur MS-Windows qui reçoit un courrier électronique dans lequel se trouve attaché un virus sous forme d'un exécutable (script ou binaire) pour MS-Windows :

1. cliquer sur le fichier attaché.

On voit tout de suite la différence, non?

5.2.3 Reproduction difficile

Si maintenant notre utilisateur Linux s'est rendu jusqu'à l'étape 3, le virus devra encore essayer de se répliquer; oui, mais comment? Un binaire qui voudrait infecter des fichiers ne pourrait qu'infecter les fichiers de l'utilisateur, mais pas corrompre les fichiers du système ni ceux d'autres utilisateurs. S'envoyer par courriel? Oui, mais à quelles adresses? Sous MS-Windows, il est très facile d'accéder au carnet d'adresses d'Outlook et Outlook Express, mais sous Linux, un tel virus n'aurait d'autre choix que d'explorer les sous-répertoires afin d'y débusquer un éventuel carnet d'adresses Mozilla, Evolution ou Kmail (pour ne nommer que ceux-là); et, évidemment, il devra en connaître le format!

5.2.4 L'exploitation de failles

Bref, bien que possible, l'écriture d'un virus Linux qui aurait la capacité de se propager aussi rapidement et avec autant de ravages que les virus MS-Windows actuels aurait peu de chance de survenir. En fait, la seule véritable avenue qui pourrait s'offrir est la même qui s'est présentée dans le cas de MSBlast, soit l'exploitation d'une faille de sécurité ne requérant pas l'intervention de l'utilisateur. Linux et les autres logiciels libres 19 ne sont absolument pas à l'abri des failles de sécurité, et même que plusieurs virus ont déjà été écrits pour exploiter des failles passées dans ces derniers. Seulement, la grande majorité de ces failles se trouvaient dans des logiciels serveurs, c'est-à-dire des logiciels servant explicitement de « porte d'entrée » sur un ordinateur : Monsieur et Madame Tout-Le-Monde n'ont pas besoin de ces logiciels et seraient donc passablement à l'abri d'infections autonomes du type MSBlast. Par comparaison, MSBlast exploitait une faille dans un serveur laissé actif par défaut dans MS-Windows, alors que sous Linux il faut explicitement installer et activer les logiciels serveurs désirés. 20

¹⁹Les logiciels libres sont des logiciels libres de toute propriété commerciale : la plupart sont le fruit d'un travail collaboratif international dans le but d'offrir des logiciels de haute qualité que tous peuvent adapter à leur façon et améliorer constamment, car le code source est rendu public ; au contraire, le code source des logiciels propriétaires est caché de telle sorte que seule la compagnie qui le produit peut y apporter des modifications, corrections et améliorations. Il existe plusieurs logiciels libres disponibles sous MS-Windows, tels que Mozilla (http://www.mozilla.org/) et OpenOffice (http://www.openoffice.org/), entre autres.

²⁰Une exception ici pourrait être le serveur X servant à l'affichage graphique.

6 Conclusion

Pour des raisons strictement techniques, Linux est totalement immunisé contre les virus MS-Windows, mais pas contre les virus dont il serait explicitement la cible; cependant, même dans ce cas, l'exécution et la reproduction d'un virus Linux seraient des étapes plus difficiles à achever. Ultimement, à moins du surgissement d'une tendance à utiliser Linux en administrateur, seuls les virus exploitant des failles de sécurité de logiciels serveurs auraient une véritable chance de succès dans un monde Linux, mais encore une fois à une échelle sans doute beaucoup moins grande que les virus MS-Windows actuels en raison de la probable faible prévalence des logiciels serveurs sur les ordinateurs domestiques. Néanmoins, je ne suis pas devin, et rien ne dit qu'un jour les gens ne diront pas qu'ils en ont assez de « Linux et de ses failles de sécurité »! Mais, pour le moment, Linux est véritablement plus sécuritaire à utiliser que MS-Windows. Et beaucoup moins coûteux...