LinuxFocus article number 245
http://linuxfocus.org

Sécuriser un réseau hétérogène avec des outils libres

par Georges Tarbouriech
<georges.t/at/linuxfocus.org>

L´auteur:

Georges est un vieil utilisateur d'Unix. Il remercie la communauté du logiciel libre de nous fournir une telle quantité de très bons outils de sécurité.

Résumé:

Cet article a été publié dans un numéro spécial sur la sécurité de Linux Magazine France. L'éditeur, les auteurs, les traducteurs ont aimablement accepté que tous les articles de ce numéro hors-série soient publiés dans LinuxFocus. En conséquence, LinuxFocus vous "offrira" ces articles au fur et à mesure de leur traduction en Anglais. Merci à toutes les personnes qui se sont investies dans ce travail. Ce résumé sera reproduit pour chaque article ayant la même origine.

Préambule

La sécurité informatique est certainement l'une des grandes questions technologiques du 21ème siècle. Mais comme pour tous les domaines préoccupants, tout le monde en parle mais ceux qui devraient se sentir les plus concernés ne semblent pas encore avoir perçu l'étendue du désastre potentiel. Les "plus concernés" sont bien évidemment les grands éditeurs de logiciels, de systèmes. Le plus bel exemple, vient une fois de plus de Redmond, où la sécurité semble n'être qu'un mot, au demeurant beaucoup moins bien maîtrisé que "marketing", par exemple.
Par chance, les deux dernières décennies du 20ème siècle ont connu l'avènement du logiciel libre et de sa philosophie. Si vous "souhaitez" améliorer la sécurité de vos machines, de vos systèmes, de vos réseaux... c'est là qu'il faudra chercher. La communauté du logiciel libre a plus fait pour la sécurité que tous les grands éditeurs réunis.

Cela dit, les outils ne font pas tout, et la sécurisation d'un réseau, par exemple, est un travail quasi permanent : tout change, tout le temps ! Ca signifie qu'on ne pourra jamais garantir qu'un réseau est sûr à 100%. On ne pourra que diminuer les risques. Ce que nous abordons ici, n'est qu'une partie des moyens de limiter ces risques. Après avoir lu ce numéro spécial (NDA : rappelez-vous, cet article faisait partie d'un numéro hors-série de Linux Magazine France sur la sécurité), vous en saurez sans doute plus sur la sécurité, mais en aucun cas vous ne pourrez affirmer que votre réseau est sûr. Vous voici prévenus. Dernière précision : un tel article ne peut être exhaustif. Il existe une énorme quantité de littérature sur le sujet et elle n'a toujours pas fait le tour du problème, loin de là. Par conséquent, n'attendez pas d'un article qu'il aborde tous les cas de figure, qu'il s'agisse d'OS, d'outils, de configuration, d'utilisation... Pour conclure ce préambule, nous devons ajouter que certains paragraphes sont empruntés à des articles parus dans le magazine en ligne LinuxFocus, mais rassurez-vous, avec l'agrément de l'auteur : c'est le même !

Présentation

Nous aborderons tout d'abord l'architecture d'un réseau très hétérogène, composé de systèmes plus ou moins répandus. Plus les OS sont nombreux, plus la tâche est complexe, ces systèmes n'étant bien évidemment pas égaux face à l'adversité... De plus les machines serveurs ont souvent un rôle différent au sein d'un réseau : nous essaierons donc de varier leurs attributions. Nous traiterons ensuite d'une panoplie d'outils indispensables à l'amélioration de la sécurité. Le choix sera arbitraire : il en existe beaucoup trop pour ne serait-ce que les mentionner tous. Bien évidemment nous en profiterons pour expliquer la sécurisation des machines et du réseau grâce à ces outils. Le chapitre suivant servira à faire un tour d'horizon des particularités de différents systèmes durant la phase de sécurisation. Nous conclurons par la "relativité" de la sécurisation, pour essayer de montrer à quel point la route est longue, sans pour autant faire de prospective.

Exemple de réseau hétérogène

Le protocole TCP/IP possède comme avantage premier d'être "parlé" par tous les OS de la planète. Grâce à lui des systèmes totalement différents sont capables de communiquer, d'échanger. Compte-tenu du type de réseau que nous allons traiter, TCP/IP sera toujours le protocole utilisé. En clair, ne seront pas abordés les protocoles propriétaires, ou les moins répandus ou les carrément désuets. Nous ne nous attarderons pas sur la structure physique, autrement dit le type de connexion, la catégorie utilisée, etc. Donc, ce réseau, nous allons y mettre un peu de tout. Bien sûr, nous allons y trouver de l'Unix, commercial et libre : par exemple, du Solaris 2.6 ou SunOS 5.6, si vous préférez, de l'Irix 6.5, du Linux (RH 6.2), du Mac OS X. Nous aurions pu ajouter QNX ou NeXTstep, ou encore NetBSD ou OpenBSD. Côté "conventionnel", nous intégrerons l'inénarrable Non Terminé 4.0 (non pas les autres, ils sont encore pires). Là-aussi, nous aurions pu ajouter OS2 qui est quand même beaucoup moins catastrophique. Enfin, nous allons ajouter du "non conventionnel", à savoir BeOS et AmigaOS (si, ça existe... enfin, plus vraiment !).
Alors, il y en a déjà qui se plaignent : quoi, pas d'AIX, de HP-UX ? Ben non ! Si nous devions traiter tous les Unixes (oui, Unices, c'est plus classe) du marché, ce serait un article en dix volumes. Toutefois, les règles de base sur la sécurité sont applicables à tous les systèmes sans exception.

Maintenant, qu’allons-nous leur demander ?
Par exemple, nous allons supposer que Solaris soit serveur d’applications. Irix va gérer les sauvegardes. NT va servir d’autres applications. Linux va servir de passerelle vers le monde extérieur. Un autre Linux pourra servir de serveur http ou de serveur de bases de données. Considérons que tout le reste, ce sont des clients. Nous allons considérer qu’il s’agit d’un petit réseau d’une trentaine de machines de manière à utiliser une authentification par fichiers. Nous aurions pu choisir quelque chose de plus élaboré pour l’authentification : NIS (ex Yellow Pages) ou encore LDAP ou Kerberos... Faisons simple, l’exemple n’en aura que plus de valeur ! Nous n’utiliserons pas non plus NFS, même si ce peut être parfois très pratique. Question sécurité, on fait mieux, même s’il y a eu certains progrès dans le domaine.
Selon le bon vieil adage "ne mettons pas tous nos oeufs dans le même panier", les services ou protocoles "douteux" mais indispensables n’existeront qu’à un seul exemplaire, sur des machines qui ne feront que ça. Par exemple, un seul serveur ftp, un seul serveur http, de préférence sous Unix, bien sûr. Nous aurons également quelques serveurs SSH sur les machines Unix et des clients SSH sur les autres systèmes. Nous y reviendrons. Enfin, l’adressage IP sera statique : en clair, pas de DHCP. En résumé, nous allons faire "basique" !
Cela bien sûr, n’est applicable que sur un réseau de moins de 50 machines : au-delà, ça deviendrait vite cauchemardesque. (On pourrait même dire 30 machines !).

Outils et sécurisation

Comme toujours, il y a plusieurs façons de faire (TIMTOWDI). Le cas idéal serait de partir de zéro, avec des machines à installer et le réseau à mettre en place. Mais, ça c’est dans les films ! Dans notre cas, considérons que le réseau est en production depuis longtemps, que les machines vont et viennent au fil du temps. En raison de l’escalade ambiante, par exemple, les machines Intel d’aujourd’hui ne vivent pas très longtemps. Au bout de trois ans, environ, il devient difficile de trouver des pièces détachées. Ainsi, soit vous recyclez les machines vers d’autres fonctions "subalternes", soit vous vous en débarrassez : triste mais vrai ! Heureusement, d’autres durent beaucoup plus longtemps et valent la peine d’être améliorées à moindre coût... quoi que. Contrairement aux apparences, ce n’est pas hors-sujet : un administrateur n’a pas le droit de risquer la panne et les problèmes qui en découlent.

Le B A BA

Dans ces conditions, le premier travail est celui que nous pourrions nommer "généralités". Il consiste à supprimer tout ce qui est inutile sur chaque machine : ce n’est pas une mince affaire ! Chaque OS, Unix compris, installe par défaut un nombre incalculable de services, de protocoles dont vous ne vous servirez jamais. Le mot d’ordre, c’est : virez-les ! Une chose simple et brutale, consiste à tout mettre en commentaire dans /etc/inetd.conf sous Unix. C’est déjà ça de moins. Bien sur, c’est un peu exagéré, mais sur de nombreuses machines ce sera parfaitement réalisable. C’est fonction de vos besoins. Sous Linux et quelques autres, vous pourrez utiliser la commande *chkconfig* pour désactiver certains services.
Vérifiez aussi les fichiers SUID/SGID root et n’hésitez pas à supprimer les bits en question ou à désactivez le programme concerné. Une commande telle que :
find / -user root -a \(-perm -4000 -o -perm -2000 \) -print
vous fournira la liste. Ensuite, à vous de voir. Pour désactiver les bits "s", tapez *chmod a-s nomduprogramme*.
Supprimez les programmes "dangereux" ou connus pour être vulnérables. Les "remote commands" telles

que rsh, rlogin, rcp, par exemple. SSH les remplacera très bien.
Vérifiez les permissions dans les répertoires tels que /etc, /var... Plus, les droits sont restrictifs, mieux c'est. Par exemple, un petit *chmod -R 700* sur le répertoire contenant les fichiers de démarrage (/etc/rc.d/init.d sous différents Unixes) n'est pas une mauvaise idée.
Le même raisonnement s'appliquera à tous les systèmes faisant partie du réseau : supprimer tout ce qui ne sert pas ou au moins désactiver. Pour NT, n'hésitez pas à "taper" dans les services du panneau de configuration. Il existe une infinité de choses basiques à faire.

Les outils

A tout seigneur tout honneur : commençons par Unix, et pour cause, c'est le seul à prendre vraiment en considération les problèmes de sécurité. Ensuite la panoplie d'outils libres est "colossale" et fonctionne sur toutes les saveurs d'Unix (presque).

Pour l'instant, nous allons travailler sur les machines, puisque sécuriser un réseau, c'est avant tout sécuriser ses composants. L'installation de ces outils étant "classique", nous ne la décrirons pas. Leur paramétrage dépend également des systèmes utilisés, des besoins... A vous de voir comment appliquer cela à votre cas.
Le premier outil indispensable, c'est *shadow utils*. C'est un moyen de crypter les mots de passe. Par chance, il fait maintenant partie intégrante de la plupart des Unixes. Le fichier /etc/shadow se "substitue" alors au fichier /etc/passwd.

Encore mieux, *PAM* (Pluggable Authentication Modules) permet de restreindre les accès aux utilisateurs par service. Tout se gère dans le répertoire contenant les fichiers de configuration par service, généralement /etc/pam.d. De nombreux services peuvent être "pilotés" par PAM tels que ftp, login, xdm, etc, permettant ainsi à l'administrateur de choisir qui a le droit de faire quoi.

L'outil suivant est incontournable sur un réseau : *TCPWrapper* . Il fonctionne également sous toutes les moutures d'Unix ou presque. Pour faire court, il permet de restreindre l'accès aux services à certains hôtes. Ces hôtes sont acceptés ou rejetés par l'intermédiaire de deux fichiers : /etc/hosts.allow et /etc/hosts.deny. TCPWrapper peut être configuré de deux manières différentes : soit en déplaçant les démons, soit en modifiant le fichier /etc/inetd.conf. Nous verrons plus loin que TCPWrapper se marie à la perfection avec d'autres outils. Vous trouverez TCPWrapper sur ftp://ftp.porcupine.org/pub/security

Nous pouvons mentionner un autre outil intéressant : xinetd. Encore une fois, pour faire court, xinetd est un remplacement de inetd en plus performant. Compte-tenu de ce qui a été dit sur inetd un peu plus haut, nous ne nous attarderons pas. S'il vous intéresse, vous le trouverez sur http://www.xinetd.org.

Sous Linux, il existe un outil incontournable, nommé Bastille-Linux. Vous le trouverez sur http://www.bastille-linux.org.
Cet outil, écrit en perl a la particularité d'être très didactique en plus d'être très efficace. Après lancement d'un script, vous répondez à différentes questions et Bastille-linux agit en conséquence. Toutes les questions sont expliquées et des réponses par défaut sont proposées. Vous pouvez supprimer les modifications effectuées, recommencer une nouvelle configuration, voir ce qui a été fait... Tout y est ! Il propose même de configurer un pare-feu : nous y reviendrons. Au moment d'écrire ces lignes, Bastille-linux est en version 1.1.1, mais la version 1.2.0 est déjà disponible en "release candidate". Elle est encore améliorée, et propose même une interface graphique basée sur Tk et son module perl.
(NDA : cet article a été écrit plusieurs mois plus tôt. La version actuelle de Bastille-Linux est en réalité

la 1.3.0).

Les outils de détection d'intrusion sont également indispensables. Les deux poids-lourds de la catégorie se nomment snort et portsentry. Le premier peut être téléchargé sur http://www.snort.org et le second à partir du site du projet Abacus, http://www.psionic.com.
L'approche de chacun des outils est différente : le premier est un NIDS (Système de Détection d'Intrusion Réseau) qui repose surtout sur l'information, le second peut être considéré comme plus orienté machine et il est plus "actif". snort bénéficie d'un nombre incalculable d'options vous permettant de "surveiller" tout ce qui se passe sur le réseau. Vous pouvez "écouter" tout ce que vous voulez : le trafic entrant, sortant, à l'intérieur du pare-feu, à l'extérieur. Le revers de la médaille : ça génère des logs énormes, mais il faut savoir ce qu'on veut ! Il possède ausi une version Win 32, ce qui a son importance, vu le nombre de logiciels libres disponibles sur ces "systèmes".

portsentry a une fonctionnalité particulièrement intéressante : il peut bloquer les ports "scannés", selon votre choix. Vous pouvez soit rediriger l'attaquant vers une adresse inutilisée de votre réseau, soit rediriger la route vers le pare-feu. Bien sûr, vous avez la possibilité de choisir qui bloquer et qui ne pas bloquer. C'est là que nous revenons à TCPWrapper : portsentry est capable d'écrire dans le fichier /etc/hosts.deny si vous le décidez. portsentry est donc particulièrement efficace. Nous ne nous lancerons pas dans le débat sur la philosophie consistant à lier les ports à une application comme le fait portsentry. A vous de faire votre choix après avoir approfondi le sujet. Sachez que porsentry peut rendre une machine "invisible" : ce n'est pas négligeable ! Dernière précision : portsentry a plusieurs modes de fonctionnement; le plus élaboré étant réservé à Linux (pour l'instant).

Nous ne pouvons pas parler de sécurité, sans aborder le cryptage. Celui-ci n'est toujours pas libéralisé et il est même encore totalement interdit dans certains pays.
Sachez qu'en France, la situation en vigueur date de 1999, c'est-à-dire "sous le régime 128 bits". Il se trouve que ssh utilise du 168 bits (avec 3DES), donc soumis à autorisation préalable. Ca signifie que si vous utilisez OpenSSL (256 bits) ou OpenSSH tels quels, vous êtes dans l'illégalité... c'est toujours bon à savoir !
La libéralisation est bien prévue... mais il faudra attendre encore un peu (de nouvelles élections pour que la loi sur la société de l'information soit votée).
Merci à Bernard Perrot pour avoir "éclairé mes lanternes".
Pour ceux qui l'ignore, Bernard Perrot est la personne à qui nous devons de pouvoir utiliser légalement ssh en France. Sa version de ssh, nommée ssf, est la seule prenant en charge la déclaration d'utilisation et l'autorisation qui en découle... et qui est obligatoire dans notre pays.
Vous trouverez ssf à http://ww2.lal.in2p3.fr/~perrot/ssf/Si vous souhaitez rester dans la légalité, sans tracasserie administrative, c'est la version qu'il vous faut. Cela dit, c'est comme vous le sentez, mais vous devrez en assumer le risque.
Conclusion : installez serveur et client ssf sur vos machines Unix.

Pour en terminer avec les outils Unix, mentionnons ceux qui sont spécifiques à certains Unixes propriétaires. Pour Solaris, vous disposez de ndd, de aset; pour Irix, vous pouvez utiliser ipfilterd. Mac OS X fournit en standard certains outils issues de la communauté du logiciel libre : ssh, ipfwadm... Nous y reviendrons plus loin.

Passons maintenant au seul et unique (heureusement !) Non Terminé 4.0.
Là, nous ne pouvons pas vraiment parler d'outils libres... mais PetitMou nous fournit de quoi améliorer les fonctionnalités du système gratuitement (et non pas de quoi corriger les bogues, puisqu'il n'y en a

pas !). Question sécurité, NT 4.0 est un modèle... d'incongruité. Plus gruyère, on peut se demander si c'est possible : ça fait peur ! Glissons. Donc, pour NT 4.0, il existe plusieurs solutions. Soit, vous êtes courageux et vous téléchargez le dernier Service Pack (le 6 au moment de cet article) ou vous téléchargez les HotFixes qui sont des patches de sécurité. Ensuite... Il existe quelques outils libres (dans le sens de gratuits, et sans les sources !) mais réservés aux versions anglaises. Voilà, c'est tout.

Pour les autres systèmes, il faudra chercher. A la décharge d'AmigaOS, le développement n'intéresse plus grand monde et la couche TCP/IP date un peu. Cela dit le Domaine Public contient toujours de quoi vous occuper. Pour BeOS, ce n'est guère mieux : ce fabuleux OS semble avoir un avenir compromis et sa couche réseau nommée Bone n'est toujours pas finie.
(NDA : malheureusement, depuis, BeOS est mort. Un petit groupe essaie de le garder "vivant" en tant que produit libre... et il fait un très beau travail.)
Mais là-aussi, vous trouverez des outils issues du monde Unix pour améliorer les choses.

La sécurisation des hôtes

Maintenant, il va falloir configurer tout ça !
Une fois de plus, considérons que toutes les machines Unix sont "équipées" des shadow-utils, de PAM, de TCPWrapper et que tous les services inutiles sont désactivés ou supprimés, que les permissions sont suffisamment restrictives sur les répertoires "délicats", etc.

Sur les machines Linux, c'est le moment de lancer Bastille-Linux. (Cet outil devrait fonctionner avec la plupart des distributions Linux, mais il a été conçu à l'origine pour RedHat et Mandrake). N'hésitez pas à répondre aux questions du script de manière très restrictive.
Sur la machine Linux servant de passerelle, il faut que le système soit minimaliste. Vous pouvez supprimer la plupart des serveurs : http, ftp, etc. Retirez X11 : vous n'en avez aucun besoin ! Enlevez tous les logiciels qui ne seront jamais utilisés... c'est-à-dire, presque tous. Arrêtez les démons inutiles. Vous devez vous retrouver avec un système sur lequel l'écran de la console ne sera même pas rempli après une commande *ps ax*.
Si vous utilisez le masquage IP, la commande *lsof -i* devrait afficher une seule ligne : celle du serveur à l'écoute (nous supposons qu'il ne s'agit pas d'une connexion permanente).
Arbitrairement, nous installerons portsentry sur ces machines Linux et il sera lancé au démarrage, bien évidemment, en mode "stealth" (le mode avancé réservé à Linux, soit les options -atcp et -audp). Cela suppose qu'un pare-feu soit installé, ainsi que TCPWrapper, nous y reviendrons.

Pour Solaris, nous utiliserons les commandes *aset* et *ndd*. Nous en reparlerons également. portsentry sera aussi installé. Nous pourrions ajouter IP Filter et remplacer la version standard de RPCBIND par la version 2.1 disponible sur porcupine.org.
Pour Irix, nous choisirons ipfilterd pour filtrer les paquets, comme son nom l'indique. Il existe en standard dans les distributions Irix, mais il n'est pas installé par défaut.

Concernant NT, ça se complique... La solution "fasciste" consiste à bloquer les ports 137 et 139, à savoir le fameux NetBIOS (ou encore mieux supprimer NetBIOS)... mais vous n'avez plus de réseau (c'est-à-dire de réseau Windos), ce qui est gênant pour un serveur d'applications ! Vous pouvez aussi installer snort, mais ça n'empêchera pas les machines en question d'être des passoires. En conséquence, à part limiter au maximum les permissions par partition, par répertoires... à condition de travailler en NTFS, of course, il y a peu de possibilités. Il existe un programme gratuit, susceptible de supprimer l'utilisateur "invité", mais il ne fonctionne que pour les versions anglaises... et bien sûr les sources ne

sont pas disponibles. Moralité : installez tous les patches de sécurité que vous pouvez trouver ! Enfin, et surtout, prenez-vous par la main pour essayer de rendre ce machin moins vulnérable. Ca vaut le parcours du combattant, mais c'est indispensable.

Pour les OS "exotiques", ce sera à vous de chercher et de choisir. Comme toujours, avant toute autre chose, la règle de base devra être appliquée : moins il y aura de services actifs, mieux ce sera.

Protection du réseau

Si les hôtes ont été bien "préparés", vous avez fait la plus grosse partie du chemin. Il faut malgré tout aller un peu plus loin.
Puisque nous sommes dans le logiciel libre, nous allons choisir un pare-feu libre, pour la passerelle, puisque c'est la machine qui vous permet d'accéder au monde extérieur. Nous avons arbitrairement choisi une machine Linux : nous pouvons donc utiliser le pare-feu proposé par Bastille-Linux. Celui-ci fait appel à ipchains ou ipfwadm selon votre version de noyau. Ce sera encore différent si vous êtes déjà au noyau 2.4.

Une petite parenthèse : ce n'est jamais une bonne idée "d'essuyer les plâtres" lorsqu'il s'agit de sécurité. La course à la dernière version du noyau risque de mener à une situation plus négative qu'autre chose. Je ne mets pas en cause la qualité des travaux effectués sur les nouveaux noyaux, mais leur "mariage" avec des outils existants qui n'ont pas été prévus pour fonctionner de cette manière. Petit conseil : patience ! La nouvelle "formule" de pare-feu intégrée au noyau 2.4 est certes prometteuse mais sans doute prématurée par rapport à ce qui l'entoure. Cela dit, faites comme vous le sentez...

Donc, le pare-feu proposé par Bastille-Linux est à la fois simple et efficace. Toutefois, il existe un outil beaucoup plus élaboré, même un petit peu "usine à gaz" sur les bords, il s'agit de T.REX disponible sur http://www.opensourcefirewall.com. Si vous recherchez un outil libre très sophistiqué, le voici.

Il existe d'autres solutions telles que les proxys, mais elles ne sont pas forcément meilleures. Une petite parenthèse : les proxys sont souvent appelés "firewalls". Ce sont pourtant deux choses différentes. Les "firewalls" dont nous parlons fonctionnent par filtrage de paquets et n'offrent pas de moyens d'authentification. Les serveurs proxy sont de deux types : application ou socks. En bref, un proxy application, fait le travail pour vous en gérant les communications et permet l'authentification des utilisateurs. Pour cette raison, il est plus gourmand qu'un pare-feu en terme de ressources.
Répétons inlassablement, que ce genre d'outil ne peut protéger "efficacement" que sur un laps de temps relativement court. Un pare-feu peut être "craqué" en une quinzaine de minutes. C'est toujours bon à savoir, n'est-il pas ? D'où l'intérêt de bien sécuriser les hôtes composant le réseau : baser la sécurité d'un réseau uniquement sur un pare-feu ou un proxy est une hérésie !

Autre moyen de "limiter les dégâts" sur le réseau : le cryptage. Utiliser telnet, par exemple, consiste à faire marcher les "pirates" sur un tapis rouge. Ca sert à leur fournir les clés du magasin. Non seulement, ils peuvent voir tout ce qui circule mais surtout, ils bénéficient du mot de passe en clair : sympathique, non ?
En conséquence, n'hésitez pas à utiliser ssh (ssf) à outrance avec les protocoles "douteux" (ou à la place). Si vous devez impérativement utiliser telnet, faites circuler les données par un canal sécurisé. En clair, redirigez le port telnet vers un port sécurisé. Pour plus ample explication, lisez l'article *Par le*

tunnel (www.linuxfocus.org/Francais/May2001/article202.shtml). (Publicité gratuite !)

Alors, c'est bien joli de tenter de sécuriser, mais encore faut-il vérifier l'efficacité du travail réalisé. Pour cela, nous allons nous transformer en "pirates" de pacotille : nous allons utiliser leurs outils. C'est laid, hein ?
Dans ce domaine, il existe aussi une belle collection de programmes, aussi nous allons, encore arbitrairement, en choisir deux : nmap et nessus. Il n'y a pas redondance, dans la mesure, où, par exemple, le second a besoin du premier. Ces outils sont des scanners de ports, même si nessus va plus loin. Celui-ci vous informe des vulnérabilités d'un système en comparant les résultats obtenus pendant le scan à sa base de données de vulnérabilités. Lancer ces outils sur une machine de votre réseau vous permettra de découvrir les vulnérabilités de chaque hôte le composant, quel que soit l'OS concerné. Le résultat est très révélateur et suffit à rendre ces outils indispensables. Vous trouverez nmap sur http://www.insecure.org et nessus sur http://www.nessus.org.

Nous parlons depuis le début de la sécurisation d'un réseau local dont certaines machines sont ouvertes au monde extérieur. Le cas du réseau d'un prestataire de services Internet sera bien sûr différent et nous n'aborderons pas le sujet en détail. Disons simplement que tout ce qui précède reste valable mais qu'en plus il faut utiliser des méthodes beaucoup plus élaborées, telles que les VPN (Virtual Private Network), LDAP pour l'authentification (par exemple), etc. C'est presque un autre sujet puisque les contraintes sont encore plus nombreuses selon le cas. Ne parlons surtout pas des sites de commerce en ligne, où là, ça relève de l'inconscience. Sites sécurisés qu'ils disent ! Ben voyons... Vous envoyez votre numéro de carte de crédit sur Internet vous ? Si oui, vous êtes plutôt courageux. Suggestion : allez faire un tour sur le site http://www.kitetoa.com, ça vaut son pesant de cacahuètes.

Particularités des systèmes

Comme nous l'avons déjà mentionné, les systèmes ne sont pas égaux face à l'adversité. Certains ont de très bonnes dispositions, d'autres sont de véritables gruyères. Paradoxalement (quoi que !), les OS dits libres sont au-dessus du lot. Les différents BSD (OpenBSD, NetBSD, FreeBSD...), les différents Linux disposent d'atouts sérieux lorsqu'il s'agit de sécurité. Répétons-le, c'est le résultat de l'extraordinaire travail de la communauté du logiciel libre. Tous les autres, même estampillés Unix, sont un ton au-dessous. Lorsqu'ils ne sont pas Unix, c'est largement pire !

Tous les outils cités dans cet article ont été développés pour les systèmes libres. Les systèmes Unix propriétaires peuvent dans la plupart des cas en bénéficier. Toutefois, ces OS propriétaires possèdent souvent leur propre panoplie.
Par exemple, pour Solaris, nous avons cité *ndd* et *aset*. Contrairement à une idée reçue, les systèmes de Sun ne sont pas des modèles de sécurité. Un outil comme aset, permet d'améliorer un peu les choses pour ce qui concerne les droits d'accès. aset propose trois niveaux de protection : bas, moyen et élevé. Vous pouvez le lancer par une commande shell ou par cron. Sur un réseau, ce qui est vrai à 17 heures, ne l'est peut-être plus à 17 heures 30. D'où l'intérêt de lancer des commandes de manière cyclique pour garantir une certaine homogénéité. C'est pour cela qu'aset peut être géré par cron. Ainsi il vérifiera toutes les 30 minutes ou toutes les heures, ou ce que vous voulez, l'état des permissions des répertoires, des fichiers...
Pour sa part, ndd permet de modifier les paramétres de la pile. Cela permet par exemple de cacher les empreintes du système. Un système identifié est un système plus vulnérable, puisque les "crackers"

savent mieux où "taper". Grâce à ndd, vous pouvez modifier la taille maximum des segments TCP (MSS). Par défaut, cette taille est à 536 sous Solaris 2.6. La commande *ndd -set /dev/tcp tcp_mss_def 546* la fait passer à 546. Plus ce MSS est élevé, mieux c'est (pas trop quand même !). nmap, par exemple, est capable d'exploiter cette faiblesse. En utilisant ndd, vous lui coupez l'herbe sous les pieds. Si vous avez des machines sous Solaris, n'hésitez donc pas : servez-vous de ndd. Vous disposez d'innombrables possibilités : voir la page de manuel.
Vous pouvez également utiliser IP Filter, un système de filtrage de paquets. Il est disponible sur ftp://coombs.anu.edu/pub/net/ip-filter.

Pour ce qui est d'Irix, c'est encore autre chose. A la décharge de SGI (ex Silicon Graphics), comme son nom l'indique, son sytème a été prévu pour le graphisme. La sécurité, ce n'était pas vraiment la préoccupation majeure. Nécessité faisant loi, il a bien fallu proposer des moyens de limiter les risques. ipfilterd a donc été proposé dans Irix, même s'il n'est pas installé par défaut : il faudra donc aller le chercher ! ipfilterd sert bien sûr à filtrer les paquets, permettant ainsi de refuser l'accès à qui vous le souhaitez. Il repose sur un fichier de configuration nommé ipfilterd.conf, et c'est là que ça se complique. La syntaxe de ce fichier est pour le moins particulière et surtout elle ne supporte pas les espaces inattendus ou les lignes vides. Ainsi, pour permettre à la machine "mars" de causer à la machine "jupiter" qui n'est autre que la station SGI, il faudra taper une ligne du style : *accept -i ec0 between jupiter mars*. Les machines ne figurant pas dans ce fichier ne pourront pas accéder à jupiter (et donc encore moins à sa cuisine). Encore pire : si vous ne modifiez pas le paramètre *ipfilterd_inactive_behavior* par systune, plus personne n'aura accès à la machine ! Ca c'est de l'efficacité ! Donc, par défaut ce paramètre est à 1, et il faudra le mettre à 0 par la commande *systune -i ipfilterd_inactive_behavior 0*.
Autre chose bien connue, mais qu'il faut peut-être rappeler, Irix possède une vulnérabilité de choc, nommée fam (File Alteration Monitor). Ce programme sert à rendre Irix très agréable en permettant la communication entre différents démons. C'est lui par exemple qui permet d'obtenir des icônes d'enfer dans le gestionnaire de fichiers. Pourtant, il n'y a qu'une chose à faire : le désactiver ! Dommage, mais c'est ainsi.

Pour en terminer avec les systèmes Unix, signalons que QNX est extrêmement vulnérable mais qu'il peut bien sûr bénéficier des outils libres. Mac OS X profite déjà de certains de ces outils.

Il faut quand même parler un peu du système réseau par excellence : the one and lonely NT 4.0. Sécuriser ce machin relève de l'utopie quoiqu'en dise PetitMou (et plein d'autres). Si vous simulez une attaque avec nessus par exemple, vous allez en avoir des cauchemars. Tant que NetBIOS est actif, nessus vous donne les noms de toutes les machines constituant le domaine et des utilisateurs qui vont avec, administrateurs compris. Moralité : supprimez NetBIOS ! Oui, mais comme déjà mentionné, si plus NetBIOS, plus réseau... Il faut choisir son camp. nessus vous informera aimablement qu'il peut aussi se loger par l'intermédiaire de l'utilisateur invité par une session NULL (soit avec un nom d'utilisateur NULL et un mot de passe NULL). Donc, supprimez-le ! Oui, mais comment ? Et c'est tout comme ça ! En conclusion, prenez-vous par la main et limitez les accès au maximum par partition (NTFS), par répertoire. Pour les partitions FAT... pas de solution. Toutefois, selon les logiciels utilisés vous pouvez être obligés d'en avoir des partitions FAT : il y en a plein des logiciels qui ne fonctionnent pas sous NTFS. Pour en finir, évitez l'inénarrable IIS, surtout comme serveur ftp. En fait, ne l'installez pas. S'il existe aujourd'hui tant de prestataires suffisamment inconscients pour utiliser ce truc, nous ne pouvons que leur conseiller de passer à Apache, mais bon... Ne nous attardons pas sur IIS, la littérature sur ses "tares" est suffisamment dense, n'en rajoutons pas. En réalité, il existe une marche à suivre pour transformer la passoire en filtre (les trous sont plus petits !). Le problème c'est qu'elle est plutôt du style

"longue marche" et le magazine entier n'y suffirait pas. Concentrons-nous sur l'essentiel. Bien évidemment, il ne s'agit plus de sécuriser avec du logiciel libre : il s'agit du monde M$!
La première suggestion serait d'utiliser MSCE (Microsoft Security Configuration Editor) disponible sur le ServicePack 4 avec MMC (M$ MAnagement Console). Par contre, si c'est votre choix, prudence ! Avec cet outil si vous vous "loupez", c'est la catastrophe garantie. De plus il est en version anglaise, et le mélange des "langues" n'a jamais franchement donné de bons résultats dans le monde de Redmond. Vous voilà prévenus.
Ensuite, parmi les différentes mesures indispensables, vous devez "sécuriser" le compte administrateur, voire le désactiver. Penchez-vous sur *passprop* disponible à partir du SP 3. Vous pouvez aussi renforcer les mots de passe avec la dll *passfilt* par l'intermédiaire de la base des registres (j'ai toujours pensé que les individus qui avaient inventé ce truc étaient sous l'influence de champignons hallucinogènes, mais bon...). Désactivez le fameux invité. Ca ne sert pas à grand chose (voir plus haut), mais c'est moins pire. Par contre, vous pouvez restreindre son accès aux logs par la base des registres. Dans "HKEY_LOCAL_MACHINE", créez les clés *System\CurrentControlSet\Services\EventLog\Application,* Security et System (ces deux derniers à la place d'Application). Leur nom c'est "RestrictGuestAccess", le type REG_SZ et la valeur 1.
Vous pouvez crypter les mots de passe avec *syskey*. Attention, c'est une opération irréversible !
Quand même, une bonne nouvelle : vous pouvez restreindre l'accès au fameux invité. Encore une fois, jouons avec la base des registres, toujours dans "HKEY_LOCAL_MACHINE". Cette fois, la clé se nomme *System\CurrentControlSet\Control\Lsa*. Le nom de la valeur est "RestrictAnonymous", le type "REG_DWORD" et la valeur 1. Mais le monde M$ est très taquin : sachez que cette modification peut altérer certains services réseau...
Parmi les choses importantes, vous pouvez également ne permettre l'accès qu'à certains ports, par l'intermédiaire de l'application "Réseau" dans le panneau de configuration. Dans les propriétés TCP/IP, sélectionnez "Avancé" et cochez la case "activer sécurité" (je crois que ça s'appelle comme ça, mais je n'ai pas ce genre de truc chez moi pour vérifier). Dans la fenêtre "Sécurité", cochez "permettre seulement" et choisissez les ports que vous souhaitez activer. Là-aussi, prudence. Il vaut mieux savoir ce que vous faites, sinon, certains services ne fonctionneront plus.
Beaucoup d'autres choses sont envisageables, mais disons qu'il s'agit de l'essentiel. Pour en savoir plus, visitez sans.org : une grande quantité de documents y est disponible.

L'insoutenable relativité des choses

Bon, vous avez fait tout ça. Vous lancez nessus pour scanner le réseau entier et vous vous retrouvez encore avec des trous de sécurité. Nous ne dirons pas d'où ils viennent... on le sait déjà ! Il ne vous reste plus qu'à essayer de leurrer ces succédanés de systèmes. Ca ne supprimera pas les trous dûs à NetBIOS, mais ça limitera les dégâts. Créez des sous-domaines. Ne vous logez pas en tant qu'administrateur. Patchez comme des bêtes. Enfin, essayez de cacher tout ça derrière des machines Unix transformées en passerelles. Malheureusement, la relativité de la sécurité ne vient pas que des produits conçus "in Redmond". Un réseau, c'est vivant : il s'y passe plein de choses. Partant donc du principe qu'un bon administrateur est un administrateur "parano", vérifiez de manière quasi permanente l'état des lieux. Ecrivez des scripts pour automatiser toutes ces vérifications. Par exemple pour contrôler régulièrement les programmes SUID/SGID root, les fichiers critiques, les logs...
Histoire de se faire quelques amis supplémentaires, verrouillez les lecteurs des utilisateurs (disquettes, CD-Rom). Interdisez les téléchargements sans votre aval, surtout lorsqu'il s'agit d'exécutables comme c'est toujours le cas dans le monde M$. Interdisez l'ouverture des documents joints au format Word,

Excel... par l'intermédiaire de filtrage de courrier.
Oui, c'est facho, mais on fait quoi contre les virus macro ? N'utilisez pas des produits comme Outlook. Encore une fois, il faut savoir ce qu'on veut ! Je suis très conscient de me battre contre les moulins à vent, mais on ne peut plus parler de sécurité avec des produits semblables. Le fameux "I love you" n'a servi à rien, et surtout pas de leçon.
Pour ce qui concerne Unix, les téléchargements doivent aussi être contrôlés. Les checksums et autres clés n'ont pas été mis là par hasard.

Prenez l'habitude de contrôler régulièrement l'état du réseau par l'intermédiaire des logs, de scripts, de scans... Vous verrez : ça change vite et pas forcément dans le bon sens. Enfin, nous n'en avons pas parlé, mais n'oubliez pas les sauvegardes. La stratégie est immuable : quotidienne, hebdomadaire et mensuelle. Même une machine Unix peut rencontrer des soucis, même si c'est très rare. Et puis, les utilisateurs font parfois quelques erreurs... mais c'est rare. C'est la faute à la machine ou à l'informatique, c'est bien connu.

C'est enfin terminé !

Si vous êtes arrivés jusque là, c'est que vous êtes très courageux. Le problème, c'est que nous n'avons fait que survoler le sujet !
La sécurité est un domaine sans fin et elle ne concerne pas que les réseaux. Des applications vulnérables peuvent gravement compromettre un réseau. Un pare-feu mal paramétré est plus dangereux que pas de pare-feu du tout. Une machine Unix contient souvent des dizaines de milliers de fichiers. Qui peut affirmer qu'aucun d'entre eux n'est vulnérable ? Qui pense qu'un "cracker" va essayer de casser une clé de 128 bits ? Que nenni : il choisira une porte dérobée. Encore et encore, vous pouvez installer tous les outils de sécurité de la création, si vous laissez un petit trou c'est par là que passera l'indésirable.

La sécurité est aussi un comportement : il faut suivre de très près ce qui se passe. Visitez par exemple régulièrement les sites consacrés à la sécurité, les sites de vos différents OS... Sun, par exemple publie des patches recommandés tous les mois. SGI sort une nouvelle version d'Irix environ tous les six mois. Microsoft propose des ServicePacks ou des patches de sécurité fréquemment. Les distributeurs de Linux publient des correctifs pour chaque nouvelle vulnérabilité. Même chose pour ce qui concerne les différents BSD.
Si vous n'utilisez pas les produits concernés par les correctifs retirez-les de votre disque. Et ainsi de suite : la liste des choses à faire est très longue. En clair, la corporation ne risque pas le chômage technique.

Enfin, répétons-le, tout ce qui précède ne contribuera qu'à rendre votre réseau moins vulnérable. N'imaginez pas que vous pourrez obtenir un réseau sûr à 100%, même à un moment donné (sauf quand toutes les machines sont arrêtées). Cela dit, il n'est pas nécessaire d'être "parano" pour faire ce métier... mais ça aide ! Reste à ne pas l'être dans la vie, ce sera beaucoup plus agréable pour votre entourage...

Références

http://www.linuxsecurity.com

http://www.sans.org

http://www.infosyssec.org

http://www.securityfocus.com

http://www.cs.purdue.edu/coast/hotlist/

La vie est trop triste : rions un peu !

Incontournable ! La vie quotidienne d’un sysadmin...

Si vous causez la langue de Shakespeare Délirant, mais hilarant (de la Baltique)

Site Web maintenu par l´équipe d´édition LinuxFocus	Translation information: fr --> -- : Georges Tarbouriech <georges.t/at/linuxfocus.org>

2005-01-14, generated by lfparser_pdf version 2.51