KONRAD ZUWAŁA

Analyse après l'attaque

Degré de difficulté

Après avoir découvert une activité indésirable sur l'ordinateur, notre objectif consiste le plus souvent à détecter les traces d'une activité d'un utilisateur non autorisé et à apprendre que s'est réellement passé sur notre ordinateur. C'est le but de l'analyse après l'attaque.

CET ARTICLE EXPLIQUE...

- Qu'est-ce qu'une analyse après l'attaque,
- comment la faire,
- comment fonctionne un système de fichiers,
- à quoi ressemble une analyse d'un programme suspect.

CE QU'IL FAUT SAVOIR...

- connaître les notions de l'administration d'un système UNIX,
- connaître les notions du langage C,
- avoir des connaissances générales sur les systèmes informatiques et leur fonctionnement

L'analyse après l'attaque a beaucoup de points en communs avec une analyse détectiviste. Il faut déterminer l'heure, l'endroit, lier de nombreux faits entre eux. Pour obtenir des informations, nous sommes souvent obligés de parcourir des endroits les plus étranges du système, męme le plus sombres. Les informations obtenues seront maintes fois partielles, imprécises, nous devrons faire beaucoup d'effort pour les lier entre elles et trouver des relations. C'est en quoi consiste une analyse après l'attaque : à utiliser les informations que nous avons réussi à trouver dans un système compromis.

Afin de commencer l'analyse, il faut dans un premier temps nous préparer à la question du point de vue théorique. Il est également recommandé de préparer le système d'exploitation à cet objectif d'une manière appropriée, nécessaires pour les opérations de ce type. Je suppose que l'utilisateur se sert d'un système basé sur un système UNIX (FreeBSD, OpenBSD, Linux, etc.), une partie des informations contenues dans l'article est toutefois universelle et peut se rapporter au système Windows.

Le point élémentaire d'une analyse après l'attaque consiste à ne pas chercher quelque chose de particulier. Tous ceux qui cherchent une réponse concrète dans le système ne trouveront rien en réalité. En effet, à quoi faut-il faire attention ? Il est difficile de définir au début les points concrets à analyser. Bien évidemment, nous pouvons nous demander qui, quand et comment est entré dans notre système d'exploitation et ce qu'il y a fait. Mais pour répondre à ces questions, il faut recueillir de nombreuses données, sans trouver sur la route aucune preuve claire et concrète. Nous aurons tout au plus des fragments d'informations que nous devons lier entre elles et en tirer des conclusions.

Quelles informations sont stockées dans notre système ? Dans la plupart de cas, il s'agit des « déchets » : des fichiers qui ne sont pratiquement jamais utilisés. Seule une partie minime des ensembles sur les serveurs UNIX est ouverte de manière systématique (ouverte et lue). La plupart d'entre eux ne sont pas nécessaires pendant longtemps, par exemple les fichiers de configuration ne sont lus qu'au moment du démarrage du programme et ensuite, ils restent sur le disque en attendant un prochain démarrage de l'ordinateur ou redémarrage, pour que leur contenu soit lu. Comme nous le savons, certains serveurs ne sont pas redémarrés pendant des années. Que faut-il en conclure ? Les ordinateurs modernes sont capables de remplir en quelques secondes les disques durs les plus grands. Les processus système se rapportent toutefois toujours aux męmes données, en utilisant les fichiers. Lorsque le système lit donc tout le temps et enregistre sur les męmes fichiers, il efface en réalité ses traces. Ses traces à lui et les traces d'un intrus potentiel : les indicateurs

de temps, les traces de toute modification indésirable dans les fichiers. Pour cette raison, les informations exceptionnelles, qui n'ont rien à voir avec les opérations ordinaires d'accès aux données de fichiers ou d'ouverture de dossiers non utilisés sont très précieuses dans une analyse après l'attaque.

Une autre question importante est l'ordre de modification d'informations (en anglais *Order Of Volatility*). Comme vous le savez, une partie d'informations est soumise à un effacement plus rapide et par conséquent à une suppression. Les informations se perdent le plus rapidement sur les supports électriques (tels que les registres du processeur, sa mémoire cache, la mémoire RAM) ou les récepteurs réseau, tels que les cartes réseau, qui contiennent leurs propres mémoires tampon ou les modules de mémoires. Ces informations sont en général inaccessibles pour nous car la durée de leur vie oscille dans les micro ou nanosecondes. Ensuite, nous avons des processus dont la durée de vie se situe entre plusieurs secondes à plusieurs heures (mais nous savons que les informations sur lesquelles ils reposent sont modifiées sans cesse, donc elles sont rapidement obsolètes). Les disques durs, en fonction du type d'informations, peuvent les stocker entre plusieurs secondes jusqu'à plusieurs mois, voire années. Tout est question d'utilisation des informations : les fichiers temporaires créés par certains programmes ne peuvent exister que quelques secondes alors qu'une énorme partie de données est stockée sur les disques sous la même forme pendant des mois. Les supports externes comme les disquettes, les mémoires de masse et les CD//DVD/BlueRay sont capables de garder les données y enregistrées pendant de longues années.

Que pouvons-nous en conclure ? Dans un premier temps, il faut protéger les données éphémères que nous pouvons perdre en un rien de temps. Nous devons ainsi protéger les informations et leurs supports dans un ordre adéquat : il faut commencer par les données éphémères et terminer par les données qui ne sont pas menacées.

Une autre chose dont il faut se rendre compte en effectuant une telle analyse est *l'illusion* créée par le système d'exploitation autour de nous. L'ensemble de système de fichiers est en effet une illusion. Les fichiers constituent en effet une suite de zéro et de un, une information électromagnétique enregistrée sur un disque dur. Les dossiers et les fichiers – tout est une illusion créée par le système d'exploitation, une sorte de facilité pour nous simplifier l'utilisation de l'ordinateur. Il ne faut pas l'oublier au moment de récupérer les fichiers perdus ou d'analyser les fragments des ensembles trouvés quelque part sur le disque ; nous pouvons le faire en omettant le système de fichiers, ce qui permettra d'augmenter considérablement la quantité d'informations.

Il faut aussi faire attention au niveau de confiance que nous pouvons avoir par rapport à une information donnée. Une seule information peut sembler peu crédible mais si elle se répète dans de nombreux endroits différents, elle commence à l'être davantage. Prenons cet exemple : nous avons trouvé une entrée sur la connexion d'un utilisateur dans le fichier contenant les logs du serveur. C'est une information individuelle, elle peut ne pas être crédible. Si nous regardons toutefois le fichier avec l'historique de commandes de cet utilisateur dans son shell, nous remarquerons qu'il avait saisi des commandes appropriées. Nous avons obtenu une confirmation supplémentaire de l'information relative à sa connexion. Si de plus, un système IDS ou autre renifleur fonctionnant dans le réseau en question confirme qu'une telle connexion a eu lieu depuis l'hôte dont l'adresse IP est `x.x.x.x`, nous pouvons être quasiment sûrs que l'information est vraie. Nous ne sommes pas toutefois toujours sûrs à 100 % car un attaquant expérimenté aurait pu préparer toutes les sources susmentionnées. Malgré tout cela, plus de sources confirment l'existence de l'information, plus nous pouvons y faire confiance.

Nous distinguons deux méthodes de recueillir des informations : dans le livre *Forensic Discovery* de Dan Farmer et Wierse Venem, elles s'appellent l'archéologie numérique et la géologie numérique. Il s'agit bien évidemment d'une analogie à ces domaines scientifiques et leur utilisation dans le monde réel, non virtuel. L'archéologie, comme son nom l'indique, consiste à analyser ce qui a été créé par homme. En le rapportant aux ordinateurs, nous en concluons qu'il faut analyser l'activité de l'utilisateur sur un ordinateur concret. Il faut donc analyser les fichiers qu'il avait utilisés, les processus lancés, tout ce qui avait été initié depuis le compte

Listing 1. *Fonction lstat() et la structure y liée*

```
#include <sys/stat.h>

int lstat(const char* path, struct stat* buf);

struct stat {

dev_t     st_dev;     /* ID de l'appareil contenant le fichier */
ino_t     st_ino;     /* numéro inode */
node_t    st_mode;    /* protection */
nlink_t   st_nlink;   /* nombre de liens matériels */
uid_t     st_uid;     /* ID du propriétaire du fichier */
gid_t     st_gid;     /* ID du gr. du propriétaire du fichier  */
dev_t     st_rdev;    /* ID de l'appareil (si fichier spécial */
off_t     st_size;    /* taille complète en octets */
blksize_t st_blksize; /* taille du bloc du système de fichiers */
blkcnt_t  st_blocks;  /* nombre de blocs alloués */
time_t    st_atime;   /* heure du dernier accès (access) */
time_t    st_mtime;   /* heure de la dernière modification (modification) */
time_t    st_ctime;   /* heure de la dernière modification (time) */
};
```

systém correspondant à l'identifiant du suspect. La géologie en revanche est un processus d'analyse de l'activité du système d'exploitation en tant que l'environnement parent de l'utilisateur, qu'il forme en quelque sorte. Simplement parlant, nous analysons tout ce que l'utilisateur n'avait pas lancés et ce qui fonctionne dans le système : l'accès aux fichiers de configuration, les opérations sur les disques, les informations stockées dans le système de fichiers qui n'avaient pas été créés par l'utilisateur.

Nous savons donc comment procéder à l'analyse, il faut maintenant préparer le système d'exploitation à cette analyse. Il est évident qu'il faut disposer d'un espace suffisant sur les disques durs pour copier et ensuite monter les images du système de fichiers du système compromis. Certains outils, permettant de réaliser des opérations sur le système de fichiers de l'ordinateur compromis, seront également indispensables : pour monter son image sur un autre ordinateur ou sur un ordinateur qui fait objet de nos recherches. Il ne faut pas oublier que les informations sont éphémères : il faut dans un premier temps collecter les informations dans la mémoire de l'ordinateur, les processus et tout ce que nous sommes incapables de copier physiquement sur un autre ordinateur.

The Coroner's Toolkit, et le projet qui devait le remplacer The Sleuth Kit, constitue l'ensemble d'outils nécessaires. TCT est un projet créé par les auteurs du livre susmentionné, disponible gratuitement sur Internet. Vous trouverez davantage d'informations sur ce sujet dans l'encadré Sur le Net. L'installation de deux ensembles d'outils est plutôt intuitive et tout utilisateur intermédiaire du système UNIX sera capable de la faire. Procédons à l'analyse.

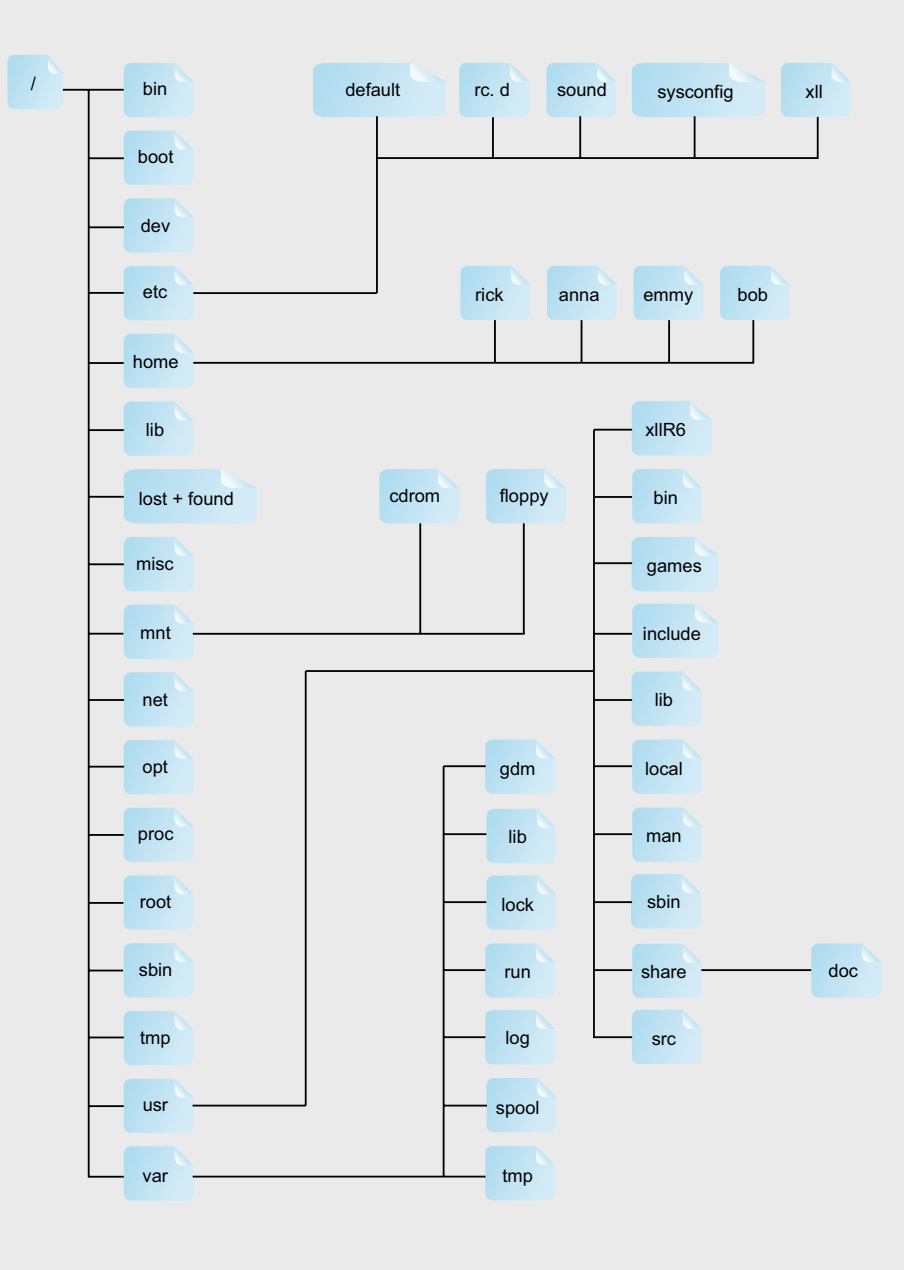

Figure 1. *XXX*

Le temps est l'argent – trouver des informations sur le temps

Dans la plupart de cas, l'objectif de l'analyse ne consiste pas à voir ce qui s'est passé. C'est plutôt évident : si quelqu'un a accédé à notre ordinateur, il a pu faire quasiment tout dont il avait envie. Nous ne pouvons rien y faire. L'information sur *la date et l'heure* de cet événement est beaucoup plus importante. Cette information nous permettra de nous rendre compte quelles données auraient pu fuir de notre ordinateur ou pendant combien de temps l'ordinateur a été exposé à l'intervention de l'extérieur. En vérifiant l'heure, nous apprendrons aussi tôt ou tard quelle était la raison de la corruption du système et ce qui *a pu* être fait dedans. Rappelons que cet article n'expliquera pas comment détecter que le système a été compromis, c'est un sujet d'un autre article. Notre article décrit ce qu'il faut faire, en disposant des informations relatives à l'attaque de l'intégralité de notre système d'exploitation.

Indicateurs MAC

MAC (en anglais *Modified, Accessed, Changed* – modifié, utilisé, changé) sont des attributs du système de fichiers, déterminant la durée de différentes opérations d'accès au fichier. Regardons le Listing 1. Il présente le prototype de la fonction `lstat()`. Sa tâche consiste à recueillir les informations sur le fichier et les enregistrer dans une structure spéciale. Cette structure correspond aux

paramètres de l'ensemble, stockés dans le système de fichiers.

La structure `stat` contient les variables correspondant aux paramètres du fichier. Nous nous concentrons sur trois dernières positions : il s'agit des indicateurs MAC. Cette structure nous aide à écrire un programme, chargé de parcourir le système de fichies à la recherche des modifications suspectes. Il peut nous servir par exemple à vérifier les fichiers système ou les fichiers de configuration récemment modifiés. Comme nous l'avons mentionné auparavant, ces fichiers sont ouverts très rarement. Si nous remarquons des modifications suspectes de temps, des modifications des fichiers système ou des fichiers de configuration, voire l'apparition de nouveaux programmes qui devaient ętre absents dans le système – c'est une trace. Nous parlerons davantage des indicateurs MAC dans la partie de l'article consacrée aux systèmes de fichiers UNIX. Nous y décrirons en détails qu'est ce qu'un système de fichiers et comment l'utiliser à nos fins, autrement dit, pour trouver les traces indispensables.

Système qui connecte le trafic réseau – source d'informations

De nombreux grands serveurs d'entreprise ou systèmes dans de petites entreprises dont l'infrastructure réseau n'est pas très développée, sont équipés des systèmes de connexion du trafic réseau. Ces programmes, dont `argus` est un exemple, permettent d'enregistrer tous les événements qui ont eu lieu sur le réseau. Bien évidemment, nous pouvons rencontrer un problème. Un serveur Web d'une taille moyenne est capable en effet de générer des dizaines (voire des centaines) de gigaoctets du trafic réseau. L'analyse de toutes les données recueillies par ce logiciel pourrait poser un souci : qui voudrait lire toutes ces informations ? Grâce aux programmes de connexion, il est par exemple possible de détailler uniquement les connexions sur un port donné ou depuis un hôte défini. Il est également possible de générer des logs d'après la date de l'événement dans lé réseau : c'est une question du bon choix de logiciel et de la spécification des filtres de recherches. Imaginons que l'analyse des indicateurs MAC nous a permis de trouver un nouveau programme dans le système ; appelons ce programme `telnetd`. Le programme prend la place du serveur telnet en écoutant bien évidemment sur un autre port pour masquer son existence. Grâce au logiciel qui analyse le trafic réseau, nous pouvons détecter qu'un programme écoute sur un port déterminé. Nous avons donc deux informations : un nouveau programme dans le système qui attend des connexions depuis Internet. Un hasard ? Probablement pas…

L'exemple ci-dessus démontre qu'il est recommandé d'installer un logiciel dont l'objectif consiste à surveiller le trafic réseau. Dans des situations comme celle décrite ci-dessus, il peut nous ętre d'un grand recours. Munissons donc à l'avance de ce type de l'outil.

Structure du système de fichiers de UNIX. Utilisation pratique de ces connaissances

Le système de fichiers dans les UNIX se diffère de manière considérable du système dans les systèmes d'exploitation Microsoft. La différence élémentaire se situe dans l'approche des fichiers et des répertoires ; un utilisateur débutant d'un UNIX entend sûrement souvent que « tout est fichier » dans le système UNIX. C'est en partie vrai car la plupart (sinon tous) d'appareils dans ce système ont leur représentation sous forme d'un fichier spécial. Cette démarche permet d'accéder directement à cet appareil, ce qui nous sera utile dans la suite de notre analyse

La hiérarchie du système de fichiers se diffère également du système Windows. Le produit de Microsoft propose des disques durs marqués par les lettres de l'alphabet latin. Pour accéder à une partition donnée du disque, il faut dans un premier temps choisir la lettre qui correspond au disque physique ou logique donné (donc par exemple une partition). Sous Linux, FreeBSD ou autres systèmes, nous ne ressentons aucunement le fait d'avoir accédé à un autre disque dur ou une autre partition. Nous ne nous rapportons en effet à aucun symbole permettant de choisir un disque donné.

Le répertoire / constitue le niveau le plus élevé dans la hiérarchie *niksa. C'est un endroit supérieur pour tous les autres fichiers et répertoires dans le système ; il est impossible de passer au répertoire dessus (cela correspond au répertoire X: dans le système Windows où X signifie la lettre du disque dur). Tout système UNIX contient un ensemble standard de sous-répertoires dans le répertoire principal : il s'agit notamment de /mnt/, /bin/, /usr/, etc. Ils correspondent au répertoire `C:\WINDOWS`, `C:\PROGRAM FILES` sous Windows. Si vous ętes attentifs, vous remarquerez tout de suite la différence dans la convention de séparation des répertoires : dans le système Windows, nous utilisons le caractère \, tandis que dans les systèmes Linux ou FreeBSD, il s'agit du caractère / (bien que dans les nouveaux systèmes Windows, le caractère / fonctionne également).

Le système de fichiers *niksa est sensible à la casse donc les fichiers XYZ et xyz sont des objets complètement différents. De plus, il ne faut pas oublier que la notion d'extension du fichier est ici absente : elle est optionnelle et ne sert qu'à nous faciliter le travail pour que nous puissions nous rendre compte à quoi nous avons affaire.

Il nous reste encore à analyser plusieurs notions importantes que nous mettrons en pratique dans un instant. La première d'entre elles est un lien au fichier, appelé un noeud intermédiaire (en anglais *inode*). Il s'agit d'un chiffre définissant le fichier donné, pointant à l'objet donné et permettant d'y accéder. Nous avons aussi deux types de liens : liens symboliques et liens matériels. Le lien matériel indique directement les données enregistrées sur le disque dur, il se rapporte tout simplement à un espace donné occupé par le fichier sur l'appareil. Il définit par exemple le bloc de la mémoire du disque dur où se trouve le fichier en question, son adresse physique. Le lien symbolique en revanche est une structure qui indique le nom du fichier dans le système et non directement

les données auxquelles ce fichier se rapporte. C'est autrement dit un raccourci courant au fichier. Imaginons que nous avons créé le fichier `/Monfichier`. Nous disposons donc d'un lien matériel pointant aux données stockées par ce fichier. Grâce à la commande `ln -s`, nous sommes capables de créer de nombreux liens symboliques qui seront *de facto* des raccourcis de ce fichier car ils se rapporteront à son nom dans le système de fichiers. Un seul lien matériel pointera toutefois aux données dans ce fichier.

Pourquoi avons-nous besoin de tout cela ? C'est indispensable pour comprendre le concept de suppression des fichiers par le système d'exploitation. Comme tout le monde en a sûrement entendu parler, il est possible de récupérer les données depuis un fichier supprimé. Et la suppression d'un fichier par le système d'exploitation n'est rien d'autre que la suppression des liens matériels et symboliques au fichier en question, de sorte qu'il ne soit pas possible de le lire depuis le niveau du système de fichiers. De plus, le bloc du disque donné où s'est trouvé le fichier, est « marqué » par le système d'exploitation pour être écrasé. Au moment opportun, le système d'exploitation enregistre ici d'autres données en détruisant ce qui y avait été stocké auparavant. Cette possibilité dépend de l'activité de l'ordinateur en question : des opérations de lecture/d'enregistrement y sont souvent effectuées, la probabilité de la suppression, qui rend impossible de récupérer le fichier, augmente considérablement.

Ce message est très utile dans l'analyse après l'attaque. Nous pouvons essayer de lire le contenu du disque dur en omettant le système de fichiers en pensant pouvoir lire des informations précieuses. De plus, nous pouvons essayer de récupérer les fichiers précieux supprimés par l'attaquant. Mais c'est un processus qui prend beaucoup de temps et d'effort et qui se termine souvent par un échec. Lorsque nous voulons récupérer des données précieuses, il est conseillé de confier cette tâche aux spécialistes qualifiés qui travaillent dans les entreprises, chargées des travaux de ce type au quotidien.

La dernière caractéristique (importante de notre point de vue) du système de fichiers d'un système d'exploitation moderne tel que Linux, FreeBSD, Microsoft Windows, est un journaling, autrement dit, une journalisation. Comme son nom l'indique, il s'agit d'une manière d'enregistrer des informations sur les événements survenus dans le système de fichiers : les opérations d'enregistrement d'un fichier, sa lecture, bref, un journal de tout ce que le système de fichiers a réalisé en une durée déterminée. Il en est ainsi dans la plupart de cas car il est également possible de configurer la journalisation de sorte qu'elle enregistre – en fonction de nos besoins – le fichier deux fois, ce qui permettra de récupérer les données incorrectement enregistrées. Tout cela est une question d'un certain compromis entre la performance (donc l'opération de lecture/d'enregistrement) et la sécurité et bien évidemment, de l'espace disponible sur le disque. Un double enregistrement occupe en effet deux fois plus d'espace qu'une opération standard d'enregistrement. Le mode le plus populaire est celui qui n'enregistre pas le fichier en entier mais seulement ses métadonnées (les données dont dispose le système de fichiers sur le fichier, illustrées à l'aide de la structure `stat` présentée sur le Listing 1).

Parcourir le système de fichiers

La première opération à faire consiste à créer une image du système de fichiers. Pour ce faire, nous disposons de la commande `dd`. Le Listing 2 présente son fonctionnement.

La commande `dd` permet de créer l'image du disque dur, appelons-la par exemple `image.hda`. Ensuite, il est possible soit de copier manuellement l'image en question sur un autre ordinateur soit d'utiliser le réseau pour le faire (comme le présente le Listing). Il faut toutefois prendre en considération le fait que le réseau peut ne pas être sécurisé et une partie d'informations peut alors être interceptée par des personnes non autorisées. Dans une telle situation, il faut penser à chiffrer le fichier transféré.

L'étape suivante consistera à monter le système de fichiers de la victime sur notre ordinateur. Pour ce faire, nous faisons la commande `mount` comme si nous montions un autre disque. Le commutateur `-t` servira à déterminer quel système de fichiers est contenu dans l'image. Ajoutons également les options `ro, noexec, nodev` (pour éviter d'écraser accidentellement l'image ou de démarrer les programmes). Maintenant, nous sommes prêts à agir.

Dans un premier temps, nous vérifions les indicateurs MAC du système de fichiers monté. Pour ce faire, nous disposons de la commande `mctime` du paquet d'outils TCT, que nous avions installé auparavant sur le système utilisé pour effectuer l'analyse. Cette commande affichera quels fichiers étaient utilisés et en effet, comme nous l'avons évoqué au début, toute utilisation d'un fichier non utilisé d'habitude doit attirer notre attention. Imaginons que nous avons découvert un fichier appelé `telnetd`, comme c'était le cas dans l'exemple analysé ci-dessus. Dans un premier temps, il faut s'assurer qu'il s'agit d'un « vrai » serveur telnet. La manière la plus simple consiste à générer la somme md5 pour ce fichier et à la comparer aux sommes md5 disponibles pour les fichiers de chaque distribution des systèmes, que vous trouverez sur Internet. La commande est la suivante : `md5sum telnetd`. Nous comparons la somme à la valeur appropriée en provenance de la base de données correspondant au système de la distribution en question. Si les sommes md5 sont différentes, il s'agit de deux programmes différents. Il est également possible que le fichier `telnetd` soit en réalité un autre fichier du système donné, par exemple `/bin/login`, ce qui permet à l'intrus de se connecter au système à distance. Il faut donc vérifier si l'une

Listing 2. *Créer une image de la partition et la copier via le réseau*

```
#!/bin/bash
dd if=/dev/hda1 bs=100k of=obraz.hda
nc -l -p 2345 > obraz.hda
```

des sommes md5 correspond au fichier analysé. L'heure de la création du fichier donné est aussi importante. Elle nous informe de la date probable où le système a été compromis.

L'étape suivante consiste à analyser les logs des interfaces réseau. Cette démarche nous permet souvent de déterminer quels hôtes se sont connectés à une date déterminée à l'ordinateur sur le port donné, par exemple sur le port où écoute le programme `telnetd`. Grâce à l'adresse IP de cet hôte, nous pouvons vérifier s'il est présent quelque part dans les logs. En général il se trouve à une date inférieure, ce qui permet de voir quel programme était à l'origine du système compromis. Simplement parlant, quelle application contenait des failles permettant à l'attaquant de l'exploiter à distance.

Lorsque nous connaissons l'origine du système compromis et la date de l'événement, nous pouvons passer à une analyse plus détaillée du programme trouvé. Ce n'est pas le sujet de notre article, nous ne nous limiterons donc qu'à un bref aperçu de possibilités disponibles.

L'analyse du programme suspect peut être divisée en statique et dynamique. L'analyse statique comprend tout ce que nous pouvons faire sans lancer le programme suspect. Nous pouvons faire la commande `strings` pour afficher toutes les suites de caractères dans le programme, vérifier les bibliothèques avec lesquelles il est lié de manière dynamique. La dernière étape la plus difficile consiste à désassembler le code du programme pour observer en détails son fonctionnement. C'est une tâche qui demande beaucoup de temps et d'effort. Les connaissances excellentes de l'assembleur sont absolument indispensables.

L'analyse dynamique comprend toutes les opérations que nous pouvons effectuer lors du fonctionnement du programme analysé. Elle comprend donc de telles opérations que le débogage du programme en temps réel, son suivi à l'aide de la fonction système `strace`. Cette démarche est toutefois liée avec un risque de détruire le système sur lequel nous travaillons. Les systèmes virtuels spéciaux ont été donc créés à des fins d'une telle analyse. Ils essaient d'émuler le système déterminé avec une plate-forme matérielle donnée pour tromper au maximum le programme suspect. Il est possible d'intercepter les appels des fonctions système, des interruptions matérielles, l'accès aux interfaces réseau, bref, tout ce dont ce programme a besoin lorsqu'il est lancé dans un environnement réel.

Connaître le fonctionnement du programme suspect est un élément important de l'analyse après l'attaque. Il permet de se rendre compte à quoi le système compromis a été utilisé. Une fois cette analyse effectuée, nous devons disposer de l'information sur la date de l'événement. Nous pouvons même connaître la date de la dernière connexion de l'intrus dans le système compromis. Cette information suffit pour créer un rapport d'une telle analyse.

Recherche d'informations dans des endroits atypiques

Le dernier point abordé dans notre article est une sorte de curiosité : recherche d'informations dans des endroits atypiques.

Dans un premier temps, parlons du journal du système de fichier. L'accès y est possible uniquement en appelant le journal et son noeud intermédiaire, sans les structures du système de fichiers. Il faut donc trouver, au moyen du programme `tune2fs` pour le système ext3, le numéro adéquat du noeud intermédiaire correspondant au journal du système de fichiers . Ensuite, à l'aide du programme icat (inode cat) du paquet TCK, nous pouvons copier le contenu de ce nouvel dans le fichier sur le disque dur. Il est alors possible de parcourir le journal à la recherche de quelque chose d'intéressant.

Une chose intéressante consiste aussi à rechercher directement dans la mémoire des systèmes UNIX. Bien évidemment, cette démarche est utile seulement si nous avons rapidement découvert l'attaque. Il est alors possible de vérifier ce qui se trouve actuellement dans la mémoire et filtrer les résultats à la recherche des preuves. Pour ce faire, nous pouvons utiliser un fichier-outil spécial dans le répertoire /dev – il s'agit de /dev/mem, donc la mémoire. À l'aide de la combinaison des commandes `cat /dev/mem | grep quelqueChoseDintéressant`, nous parcourons la mémoire du point de vue du contenu des données importantes pour nous. Le Listing 3 présente comment lire le journal du système de fichiers, la manière de le faire est assez atypique. C'est pour cette raison nous avons parlé de cette opération dans cette partie de l'article.

Conclusion

L'analyse après l'attaque est un outil indispensable dans la situation où nous sommes victime d'une cyberattaque. Elle permet de déterminer la date de l'événement et les opérations qui ont pu être effectuées dans le système compromis : pourquoi l'attaquant s'en est servi ?. Ces informations sont nécessaires si nous souhaitons nous protéger contre une nouvelle attaque de notre système. Nous espérons que les systèmes que vous administrez n'auront jamais besoin d'être soumis à une telle analyse après l'attaque.

Listing 3. *Lecture d'informations dans le journal du système de fichiers*

```
# tune2fs -l /dev/hda1 | grep -i journal
filesystem features:  has_journal filetype needs_recovery sparse_super
Journal UUID:         <none>
Journal inode:        8
Journal device:       0x0000
# icat /dev/hda1 8 > ~/fsJournal
```

Sur le Net :

- *http://www.porcupine.org/forensics/forensic-discovery* – une excellente publication relative à l'analyse après l'attaque,
- *http://www.porcupine.org/forensics/tct.html* – The Coroner's Toolkit,
- *http://fr.wikipedia.org/wiki/Ext3* – système de fichiers ext3.