1. Retour d'expérience sur le greylisting du courrier électronique

Présentation faite par Fabrice Prigent, Université de Toulouse 1.
Présentation : http://www.ossir.org/resist/supports/cr/20050627/Greylisting.pdf

Fabrice Prigent a fait une première présentation sur le Greylisting en mars 2004 (voir http://www.ossir.org/resist/supports/cr/20040329/actualites.pdf). Cette seconde présentation correspond à un bilan après un peu plus d'une année d'utilisation de cette technique.

Pour rappel, le greylisting repose sur des triplets (adresse IP du serveur envoyant le message, adresse électronique de l'émetteur, adresse électronique du destinataire). Pour chaque message reçu, la plate-forme de réception examine ce triplet. S'il est inconnu, le message est rejeté temporairement (erreur 450) et le triplet ainsi que l'heure sont enregistrés. Si le triplet est connu, mais qu'il a été vu pour la première fois depuis moins d'un certain délai, le message est toujours rejeté temporairement. Enfin, si le triplet est connu et a été vu pour la première fois depuis plus d'un certain temps, le message est accepté.

Les délais sont typiquement :

- De 10 et 20 minutes pour qu'un triplet puisse être accepté (blanchi).
- Quelques heures pour qu'un triplet non blanchi soit effacé.
- Entre 30 et 40 jours pour qu'un triplet blanchi mais « inactif » soit effacé.

Le réglage des deux derniers délais est important. L'effacement des triplets non blanchis doit être en phase avec les délais de ré-émission des serveurs qui nous envoient des messages. Si, par exemple, on efface les triplets au bout d'une heure, un serveur ne ré-émettant ses messages que toutes les deux heures ne pourra jamais rien nous envoyer.

De même, le délai d'effacement d'un triplet valide mais inactif doit prendre en compte les listes de diffusion et autres outils, qui peuvent n'envoyer qu'un message par mois.

Quels sont les principaux enseignements que l'UT1 retire du greylisting ?

- Il est préférable de travailler avec la classe C d'une adresse plutôt que directement avec l'adresse du serveur émetteur. Cela permet d'éviter les problèmes avec des émetteurs comme Hotmail ou AOL, qui disposent de très nombreux serveurs d'émission. Le message pouvant être ré-émis par un serveur différent à chaque fois, le triplet de présentation change et le message risque de n'être jamais accepté.

- Dès lors que l'on utilise différents MX, il faut une base unique pour stocker les triplets. Cela permet d'accepter un message quel que soit le MX auquel il est soumis.
- Certains serveurs interprètent une erreur 450 (échec temporaire) comme une erreur 550 (échec définitif) et ne ré-émettent jamais les messages. Il faut donc pouvoir placer ces émetteurs en liste blanche, hors du champ d'application du greylisting.
- Il est vivement conseillé d'insérer, dans chaque message accepté, un en-tête spécifique indiquant le retard provoqué par le greylisting. Cela permet très facilement de calmer des utilisateurs qui auraient l'impression que leurs messages sont systématiquement retardés.

Certains outils de gestion du greylisting disposent d'un mécanisme de liste blanche automatique sur des adresses IP (jamais sur des classes d'adresses). Cette fonctionnalité permet d'accélérer la transmission des messages : si l'outil détecte, sur un certain intervalle de temps, qu'une adresse IP a validé plusieurs triplets (5 par défaut), cela signifie que le serveur est capable de ré-émettre ses messages. Il est de fait inutile de les ralentir, puisque le serveur les ré-enverra systématiquement peu de temps après.

Quels sont les résultats pour l'Université Toulouse 1 ? Les statistiques montrent que le greylisting a

- Provoqué une diminution de 80% du spam reçu,
- Provoqué une diminution de 75% des virus reçus.

A l'inverse, on observe que

- 42% des messages ne sont jamais ré-émis (virus ou spam),
- 45% des messages sont acceptés directement, sans rejet temporaire,
- 13% des messages sont ralentis par le greylisting.

Du point de vue des délais,

- 76% des messages sont transmis sans aucun délai,
- 6% sont transmis avec un délai inférieur à 15 minutes,
- 1% des messages sont transmis avec un délai supérieur à un jour.

Quelques conclusions qui peuvent être tirées de tout cela :

- Il est judicieux d'accélérer la ré-émission des messages sur nos serveurs. Un rejeu de la file d'attente toutes les 30 minutes semble une bonne valeur, qui

évite autant des délais d'attente trop importants que des rejets successifs des messages.
- Il est conseillé d'utiliser SPF, de nombreux sites n'activant le greylisting que si le domaine de l'émetteur n'a pas d'enregistrement SPF.
- Ne jamais mettre des réseaux entiers en liste blanche.

On notera que les émetteurs de spams s'adaptent au greylisting, et commencent à ré-émettre leurs messages. Toutefois, leurs coûts d'émission augmentent en parallèle, et donc le bénéfice de l'opération diminue d'autant. Et, même si tous les spammeurs acceptent ces surcoûts, le délai de rejet des messages permet à des outils comme Razor ou les RBLs de s'activer, offrant une seconde ligne de barrage au spam.

2. La prévention d'intrusion

Présentation faite par Denis Ducamp, HSC.
Présentation : http://www.hsc.fr/ressources/presentations/csm05-ips/cnet05_ips.pdf

Le délai d'exploitation d'une vulnérabilité est en chute libre. Il devient donc important d'empêcher ces exploitations, et non plus seulement de les détecter. Les IPS, s'ils ne peuvent en aucun cas remplacer les systèmes de sécurité existants, peuvent constituer une ligne de défense supplémentaire.

On notera les différences terminologiques :

- Un IDS (Intrusion Detection System) est un outil passif. Il peut fonctionner sur un système (Host IDS, HIDS) ou sur le réseau (Network IDS, NIDS). C'est un outil comme Snort, par exemple.
- Un IPS (Intrusion Prevention System) est un outil actif qui, de même, peut fonctionner sur un système (Host IPS, HIPS) ou sur le réseau (Network IPS, NIPS). Snort-inline est de ce type.

Les HIPS existent surtout dans l'univers Windows. Ils mettent en œuvre une détection comportementale (lecture/écriture de fichiers, comportement de l'application, accès à des ports d'E/S ou à des zones de la base de registres, détection de shellcode ou de débordement de pile, etc.).

Leurs avantages :

- Peu de faux positifs
- Gestion de comportements applicatifs, pas uniquement du trafic sur le réseau.

Leurs inconvénients :

- Les coûts d'exploitation sont élevés,
- L'interopérabilité est problématique,
- Les mises à jour peuvent être délicates.

Comment contourner un HIPS :

- Par des liens symboliques, du type lecteur X: associé à C:\Windows\System32
- Par des liens dans la base de registres,
- Par l'utilisation de fonctions peu connues ou mal documentées,
- Par l'injection de DLL dans des applications de confiance.

Les NIPS fonctionnent toujours en coupure de réseau, afin de pouvoir bloquer immédiatement un trafic suspect. Ils mettent en place une analyse comportementale et/ou une détection d'anomalies.

L'analyse comportementale correspond à la détection de comportements pathologiques (scans de ports) ou des situations anomalies protocolaires. Elle repose typiquement sur des bases de signatures d'attaques.

La détection d'anomalies repose plutôt sur des réseaux neuronaux, des règles explicites ou des systèmes bayésiens.

Leurs avantages :

- Mise en place d'une protection active sur le réseau

Leurs inconvénients :

- Constituent un élément névralgique du réseau. Une attaque en déni de service peut se révéler dévastateur.
- Les faux positifs deviennent très gênants (l'échange est immédiatement bloqué).
- Ces systèmes sont coûteux.
- Ils sont complexes, induisant des coûts d'exploitation élevés.

Comment contourner un NIPS :

- Par obfuscation (ce que font whisker et nikto),
- En utilisant des sessions successives pour le NIPS,
- En jouant sur l'insertion de flux et la fragmentation (fragroute),
- Grâce à du shellcode polymorphique,

- En bloquant le NIPS (déni de service, grâce à des outils appropriés comme snot).

Quels sont les critères de choix d'un IPS ?

- Il faut estimer le temps nécessaire pour le paramétrage, et le temps dont on dispose pour ce faire.
- Comme pour tout système de ce type, le nombre de signatures d'attaques et la fréquence des mises à jour est un paramètre très important.
- La possibilité de fabriquer ses propres signatures peut permettre d'attendre une mise à jour officielle, quitte à bloquer un peu trop de choses dans l'intervalle.
- Les modes de réponse ou d'action aux attaques sont à mettre en parallèle avec le pourcentage de faux positifs.
- Le temps de latence induit par l'outil doit être maîtrisé et validé.
- Pour un NIPS, qui est en coupure de réseau, son débit et le nombre de sessions en parallèle qu'il peut gérer doivent être connus. Il peut être nécessaire d'envisager des équipements redondants pour assurer une haute disponibilité de l'ensemble.
- En cas de délégation de certains aspects de la gestion (vers des responsables d'agences par exemple), l'interface d'utilisation/administration doit être appropriée.
- Il est intéressant de pouvoir faire des corrélations entre événements. La qualité des rapports produits est aussi un élément à ne pas ignorer.
- Si plusieurs équipements doivent être déployés, leur administration centralisée est souhaitable.
- La possibilité de recherches forensiques est un plus à ne pas négliger.

3. Présentation de mod_security (Apache)

Présentation faite par Frédéric Laplagne, Akersia.
Présentation : http://www.ossir.org/resist/supports/cr/20050627/mod_security.pdf

Mod_security permet de mettre en place un filtrage des URLs qui sont transmises à un serveur Web. Ce module est surtout intéressant en tant que relais inverse, permettant de protéger plusieurs serveurs, plutôt qu'intégré à chaque serveur à protéger. En couplant un ensemble de modules (mod_security, mod_proxy, mod_rewrite, mod_dosevasive, mod_headers et quelques autres), on peut ainsi construire un véritable système de protection d'un ou plusieurs serveurs Web.

Il existe deux versions de mod_security, l'une écrite en Java, l'autre sous la forme d'un module Apache écrit en langage C.

Mod_security peut être couplé avec des anti-virus, afin d'analyser le corps d'une requête (et notamment les fichiers téléchargés vers le serveur). Enfin, il peut appeler des programmes externes afin d'exécuter d'autres contrôles.

Les règles de filtrage mises en place par mod_security peuvent s'appliquer de manière générale (tout le serveur), pour un serveur virtuel, pour un chemin (logique ou physique). Les règles sont cumulatives, des plus générales aux plus particulières, sauf demande explicite d'oublier les niveaux « précédents ».

Les deux principales directives de filtrage sont SecFilter et SecFilterSelective :

- Une règle de type SecFilter s'applique à l'ensemble des informations envoyées par le navigateur.
- Alors qu'une règle SecFilterSelective ne s'applique qu'à certaines parties de la requête (REMOTE_ADDR, QUERY_STRING, COOKIES_VALUES, etc.).

Par défaut, lorsqu'une règle s'applique, l'action définie à l'aide de la directive SecFilterDefaultAction sera exécutée. Il est possible de passer outre cette action par défaut en précisant, en même temps que la règle, l'action à exécuter.

Les principales actions sont : allow (transmettre l'URL au serveur), chain (exécuter la règle suivante), deny (bloquer la requête), pass (ignorer la règle et aller à la suivante). D'autres actions facilitent la mise au point ou le suivi du filtrage : id:N (affecte l'identifiant N à la règle ; cet identifiant sera écrit dans le journal Apache si la règle est activée), log (l'activation de la règle est tracée dans le journal Apache), nolog (pas de trace d'activation de la règle). Enfin, quelques actions permettent de gérer des besoins divers : redirect:url (redirection du navigateur vers une autre URL), execute:programme (exécution du programme si la règle est activée), skipnext:N (ignorer les N prochaines règles), status:N (renvoyer le code d'état N au navigateur).

Les configurations de mod_security incorporent souvent des règles standardisées minimales, afin de gérer :

- Le protocole utilisé (http 0.9, 1.0, 1.1),
- Les requêtes acceptées (GET, HEAD, POST),
- La présence obligatoire de certains champs (USER_AGENT, HOST).

Au-delà de ces règles minimales, les configurations peuvent être très évoluées. Plusieurs exemples sont présentés, pour bloquer des injections SQL, de l'injection de code (XSS), un accès au système d'exploitation ou à certains fichiers, pour valider les paramètres d'un script (liste des paramètres et contenus), etc.

L'activation de mod_security n'amène, dans la majorité des situations, aucune dégradation de performances. Dans certains cas (analyse dynamique des requêtes), cela peut toutefois aller jusqu'à 10% des performances du serveur.