Compromettre son réseau en l'auditant ?

Renaud Deraison**

Directeur de la Recherche
Tenable Network Security

1 Introduction

Devant la recrudescence de failles et la crainte de vers de toutes sortes, ainsi que devant la croissance des réseaux d'entreprises, de plus en plus de professionels font appels à des *scanners de vulnérabilités*, des programmes balayant un réseau à la recherche de machines et interagissant avec chacune d'entre elles pour déterminer quel est le niveau de mise à jour et quelles sont leur failles de configuration.

Cependant, si une machine sur le réseau a été compromise, un pirate peut-il utiliser les habitudes de l'équipe de sécurité pour augmenter ses privilèges sur le réseau ou pour prendre le controle du poste d'un auditeur utilisant un scanner de sécurité pour faire l'inventaire du réseau ?

C'est en pensant constemment à cette problématique dans le cadre du développement du logiciel Nessus (`http://www.nessus.org`) que nous avons mis en place une procédure d'évaluation et de mitigation des risques associés à l'usage de ce dernier.

Le but de ce document n'est pas de dresser la liste des failles affectant les autres scanners (aucun autre scanner que Nessus ne sera cité) mais de documenter les mesures que nous avons mis en place.

Enfin, il n'est pas nécéssaire d'avoir un réseau compromis par un pirate pour être affecté par les attaques décrites dans ce document. La recrudescence de spywares et autres pouriciels peuvent avoir les mêmes effets et ainsi empecher l'audit d'un réseau.

2 Qu'est-ce que Nessus

Nessus est un scanner de vulnérabilité disponible gratuitement à `http://www.nessus.org` et développé depuis 1998. Il contient plus de 7.500 tests de sécurité, à la fois distants et locaux.

Plus de 70.000 organisations l'utilisent, ce qui en fait de l'un des scanners de vulnérabilités le plus populaire du marché.

Comme tous les scanners de vulnérabilités, Nessus balaye une plage d'adresse donnée à la recherche de machines, et pour chaque machine allumée, il procède

** Renaud Deraison est l'auteur du locigiel *Nessus*.

à un scan de port, suivi d'une énumération des failles présentes sur la machine distante. Plus l'utilisateur donnera de privilèges à Nessus, et plus le rapport sera complet. Par exemple, il est possible de donner à Nessus un login et un mot de passe SSH ou SMB (Windows) pour qu'il se loggue dans la machine distante et dresse la liste de patches manquants. Donner des privilèges à un scanner permet de réduire les faux positifs, accélérer le scan et obtenir un rapport plus complet.

3 Organisation de ce document

Ce document détaille quatres grands risques permettant à un pirate d'arriver à ses fins. Dans chacun des cas, nous verrons des exemples d'attaque qu'un pirate peut mener, puis les contre-mesures qui ont été prises du coté de Nessus pour empecher celles-ci.

En termes d'hypothèse de travail, nous considérerons que le réseau est switché et qu'une machine a été compromise par un pirate désirant gagner plus de privilèges sur le réseau. Le pirate sait que l'équipe de sécurité fait des scans réguliers et a décidé de se servir de ceux-ci pour empecher l'équipe de sécurité de faire son travail correctement (c'est à dire découvrir les failles du réseau) et pour gagner plus de privilèges grace aux audits automatisés de l'équipe de sécurité.

4 Liste des risques

Un pirate ayant le controle d'une machine scannée sur le réseau peut pratiquer l'une des quatres attaques suivantes sur le réseau :

- **Injection de vulnérabilités** : Un pirate peut faire croire au scanner que sa machine contient certaines vulnérabilités fait tourner un système d'exploitation qui n'est pas le sien, ou bien peut simuler certains services. Il est aussi possible d'envoyer de fausses informations sur l'état de machines tierces ou inexistantes
- **Empecher l'audit** : si le scanner contient certaines erreurs de design, il est possible de le « bloquer » et ainsi d'empêcher un audit
- **Récupération de logins et de mots de passe** : Afin de mieux faire son travail, la plupart des scanners de vulnérabilités permettent à l'utilisateur de leur donner certains moyens d'authentifications sur le réseau (login/mot de passe Windows, mot de passe SSH, etc...). Un pirate peut aisément attendre que le scanner se connecte à lui pour le forcer à fournir ces informations
- **Exécution de code** : si le scanner est mal écrit, alors il est possible pour un pirate d'envoyer des réponses mal formées à ce dernier et ainsi d'exploiter des dépassements de tampon ou autre chaines de formatage pour executer du code sur la machine faisant le scan.

5 Injection de vulnérabilités

Un pirate peut profiter d'un audit pour mentir sur l'état des vulnérabilités de sa machine et sur celle des autres. Le but de cette attaque peut tout simplement de faire perdre du temps à l'équipe de sécurité en remontant des faux positifs, ou bien de décridibiliser celle-ci auprès des administrateurs du réseau - si l'équipe de sécurité fournit régulièrement des rapports entièrements

5.1 Mentir sur l'état de sa machine

- **Comment** : A partir du moment où un pirate a le controle de la machine scannée, il devient aisé de mentir sur celle-ci. Pour ce faire, il suffit de modifier les bannières de la plupart des services (par exemple, remplacer la bannière de Postfix par celle de Sendmail, etc...). Le programme ippersonality [2] permet de faire passer aux yeux de Nmap une station linux pour autre chose (Windows, imprimante HP, etc...).
- **Solutions** : Il n'existe pas de solution fiable à ce problème. Il est possible de tenter de faire un fingerprint sur chaque service pour vérifier que le service correspond bien à ce qu'il affirme être (le plugin `www_fingerprinting_hmap.nasl` de Nessus signale qu'un serveur web n'est pas celui qu'il affirme être), mais devant un pirate déterminé, il est impossible de se prémunir contre ce type d'attaque.

5.2 Mentir sur l'existence de machines tierces

- **Comment** : Lorsqu'un scanner fait un audit, il va tout d'abord tenter de pinguer les machines du réseau, en envoyant à chacune un paquet ICMP (icmp-echo-request) ou bien un SYN TCP contre certains ports. Afin de transmettre le paquet, le routeur du réseau local de la machine cible va envoyer un paquet arp-request pour demander à la machine de s'identifier. Un pirate peut répondre à cette requète, grace à des outils comme LaBrea [3] ou Honeyd [4]. Utiliser ces programmes va non seulement surcharger de travail l'équipe de sécurité, en simulant l'existence de centaines, voire de milliers de machines, mais va aussi ralentir les scanners, transformant un scan qui durait une heure en une longue opération de plusieurs jours.
- **Solutions** : Il est possible de determiner si une réponse TCP provient de LaBrea [3] ou d'une autre machine - un plugin Nessus (`labrea.nasl`) a été écrit pour cela. En revanche, il n'existe pas encore de technique de fingerprint fiable pour Honeyd, et dans tous les cas un pirate talentueux pourra toujours écrire son propre outil passant au travers de mailles du filets.

5.3 Mentir sur l'état de la sécurité de machines existantes

- **Comment** : Si notre pirate travaille sur un réseau switché, alors il ne lui sera pas possible d'injecter des données par TCP pour simuler la présence

de service sur une machine tierce. En revanche, si le scanner fait un SYN scan, alors un pirate peut faire croire qu'une machine a de nombreux ports ouverts en envoyant tout simplement des paquets SYN—ACK spoofés, semblant provenir de la machine scannée. La plupart des scanners de port faisant du SYN scan ne font aucun test d'intégrité sur les paquets reçus et se contentent de croire aveuglément ce qu'ils reçoivent. De même, un pirate peut envoyer de fausses réponses par UDP spoofées (par exemple, en envoyant une réponse à une demande de version BIND au scanner de vulnérabilité).
Dans le cas d'un réseau non-switché (ou d'un réseau switché et d'un pirate très motivé), il est aisé pour un pirate d'injecter des données arbitraires dans un flux de données. Par exemple, un scanner se connecte à la base de registres d'un Windows du réseau afin de lire la clé `HKLM\SOFTWARE\Microsoft\Windows\Updates`, et un pirate injecte un paquet SMB contenant la réponse « pas de clé », ce qui fera croire au scanner que la machine n'a installé aucune mise à jour.

– **Solutions** : En ce qui concerne le scanner de ports, du coté de Nessus nous recommandons l'usage de notre TCP connect() scan, qui fait un handshake complet, vérifiant ainsi les numéros de séquences TCP et empèchant un pirate d'injecter des SYN—ACK à l'aveuglette. L'outil scanrand [5] utilise une méthode originale, en utilisant le numéro de séquence des paquets TCP comme d'un hash cryptographique - lorsque scanrand reçoit un paquet, il vérifie que le numéro de séquence est bien le hash d'un secret donné, ce qui permet d'éviter de garder la trace de tous les paquets envoyés.
En ce qui concerne l'injection de paquets UDP, le scanner doit s'assurer que les paquets UDP reçus sont bien envoyés en direction du port source utilisé (en utilisant un recvfrom() stricte, ou en utilisant la méthode connect() sur la socket UDP).
Enfin, dans le cas de l'injection de paquets TCP, le pirate doit être en mesure d'envoyer en temps réel des paquets spoofés contenant une fausse réponse à la question posée par le scanner. Dans le cas de protocoles comme SSH ou SMB, il convient d'activer et de négocier la signature de paquets pour empecher ce type d'attaque, mais d'autres protocoles plus simples (par ex., FTP) n'ont pas de solution « propre » pour gérer ce type de situation.

5.4 Conclusion

Un pirate ayant le controle d'une machine pourra toujours faire mentir celle-ci quand à ses vulnérabilités - il suffit d'analyser le fonctionnement du scanner utilisé, et d'envoyer les réponses que ce dernier attend pour lui faire croire ce que l'on veut. Le réel interet d'une telle attaque est plus que discutable, car finallement le pirate attirera l'attention sur sa machine.
Mentir sur la présence d'autres machines peut permettre à un pirate de ralentir considérablement l'audit de l'équipe de sécurité. C'est un problème plus sérieux et pour lequel il n'existe aucune solution miracle. En utilisant une attaque de

type LaBrea [3], il est possible de ralentir un audit de plusieurs heures, si ce n'est de plusieurs journées.

Enfin, il est relativement difficile dans la plupart de cas d'insérer des données arbitraires concernant des machines tierces mais c'est néanmoins possible, en particulier si le pirate controle une machine située dans le même réseau physique que la machine pour laquelle il veut mentir.

6 Audit impossible

Un pirate ayant le controle d'une machine sur un réseau peut décider d'empecher l'équipe de sécurité de faire un audit complet de ce dernier. L'interet est immédiat : en empechant l'équipe de sécurité de connaitre l'état de la sécurité du réseau alors un pirate peut bénéficier de plusieurs jours supplémentaires pour découvrir ce dernier à sa guise, tout en forçant l'équipe de sécurité à se focaliser sur un problème en particulier - le scanner qui semble ne pas fonctionner correctement - plutot que sa cause.

6.1 Ralentir l'audit

- **Comment** : Comme nous l'avons vu précédemment il est facile de ralentir un audit - il suffit d'utiliser un outil du meme type que LaBrea [3] pour faire croire au scanner qu'une machine qui n'existe pas est là : celui-ci va lancer une batterie de tests qui prendront chacun plusieurs secondes avant de se terminer (aucune réaction de la machine distante), et le scan sera ainsi grandement ralenti mais pas entièrement stoppé.
- **Solutions** : Comme nous l'avons déjà évoqué, il n'existe pas de solution parfaite pour ce problème. Il est possible de reconnaitre certains outils connus mais de manière générale, un pirate connaissant la plage d'adresses qui sera scannée trouvera toujours le moyen de faire passer une machine pour étant présente alors qu'elle ne l'est pas. Fort heuresement, le scan ne s'en trouve que ralenti.

6.2 Bloquer le scanner

- **Comment** : Certains scanners contiennent certains bugs qui permettent à un pirate de les bloquer complètement. Un bug que nous avons remarqué consiste à mettre en écoute le serveur discard [1] le port 80 d'une machine. Certains scanners du marché se connectent à ce port pour auditer le serveur web de la machine, et attendent une réponse indéfiniment.
  De manière plus générale, il suffit d'identifier dans le scanner un appel à la fonction `recv()` (lecture de paquets au dessus d'un flux TCP) pour
- **Solutions** : Dans le cas de Nessus, notre architecture de plugins permet de factoriser tous les appels à `recv()` en une seule fonction, qui a une durée maximum. De plus, chaque plugin executé a une durée de vie de 5 minutes

maximum, après lesquelles le scanner stoppera le plugin et passera à la suite automatiquement. Les scanners basés sur des plugins en C ou bien dont le moteur contient directement les tests de sécurité sont plus difficiles à débugguer dans ce cas là mais la logique est la même : il faut s'assurer que toute opération réseau est limitée dans le temps.

6.3 Faire planter le scanner

- **Comment** : En écrivant un serveur envoyant des réponses mal formées, un pirate peut aisément faire planter un scanner écrit en C ou C++. Un scanner de vulnérabilité va, de par sa nature, émuler un grand nombre de clients différents - FTP, SNMP, SMB, HTTP, finger, et plusieurs autres centaines de protocoles. Très souvent, le temps de dévelopement d'un « faux client » doit être extremement court, car les scanners doivent être mis à jour aussi rapidement que possible. Par conséquent, il est plus que probable que le scanner soit affecté par des « failles clientes » des plus triviales.
  Par exemple, en émulant un serveur web répondant avec un faux champ 'Content-Length' à chaque requète :
  ```
  HTTP/1.1 200 OK
  Date: Tue,  Apr 19 2005 16:10:29 GMT
  Server: Apache/1.3.33
  Connection: Close
  Content-Length: 2147483647 <-----
  ```
  Dans le cas précédent, un scanner n'ayant pas mis en place la moindre mesure de sécurité contre un serveur pirate va tenter d'allouer 2Go de mémoire vive pour lire la réponse. Il est même parfois possible de monter à 4Go en envoyant utilisant la valeur '-1' comme longueur de champ.
  Dans le même ordre d'idée, on peut imaginer écrire un tel serveur pour un protocole plus compliqué (RPC, MSRPC, etc...). Que se passe-t-il par exemple si un faux serveur SMB affirme envoyer un paquet de 2 Go ? (Windows va lire les 2 Go), ou bien si un serveur NFS affirme envoyer une liste de 8 milliards de partages ? etc...
- **Solutions** : Nous avons opté pour écrire tous les tests de sécurité dans un langage interprété : les scripts Nessus les plus mal écrits ne sont pas affectés par des débordement de tampon, des chaines de formatage, etc... Chaque script est executé dans sa propre machine virtuelle et est tué par le système d'exploitation en cas de tentative d'allocation de mémoire trop gourmande (au delà de 40 Mo, le système d'exploitation tue le processus du test en cours).
  Un scanner écrit en C ou C++ et basé sur un modèle de thread est beaucoup plus difficile à sécuriser dans cas là, car si un simple test de vulnérabilité vient à planter (ou à consommer trop de mémoire) alors c'est le scanner entier qui va planter. Une grande rigueur de codage est donc nécéssaire dans ce cas là, mais n'offrira jamais la même sécurité qu'un langage interpreté.

6.4 Conclusion

Il est aisé pour un pirate de géner l'équipe de sécurité dans son travail - en simulant la présence de machine n'existant pas ou bien en mettant en place des attaques plus complexes, un pirate va pouvoir empecher le scanner de compléter sa tache et donc de permettre à l'équipe de sécurité d'avoir un tableau de bord complet de la visibilité du réseau.

7 Récupération de logins et mots de passe

De plus en plus de scanners deviennent des outils de supervision du réseau : en leur donnant les comptes appropriés, ils sont ainsi capable d'extraire la liste des patches manquant sur un système, de détecter les failles locales, etc...
En revanche, une fois les mots de passes entrés dans le scanner, ce dernier risque de les diffuser à toutes les machines du réseau.

7.1 SMB

En configurant un faux serveur SMB, un pirate peut aisément récupérer soit le hash LMv1 (aisément crackable) du mot de passe administrateur, soit même le mot de passe en texte clair (si le scanner est basé sur Samba).

Pour se prémunir contre de telles attaques, l'équipe de sécurité devra sécuriser la machine Windows faisant tourner le scanner afin de n'utiliser que LMv2 ou plus récent. Cependant, en utilisant un tel paramètre, il deviendra impossible de se logguer dans des versions plus anciennes de Windows NT (pré-SP3).
Pour les scanners utilisant Samba, il n'y a pas de réelle solution.
Dans le cas de Nessus, nous tentons de nous connecter par Kerberos, et le cas échéant par LMv2. Il est possible de spécifier l'option « Only use NTLMv2 » pour s'assurer qu'aucun hash faible ne soit envoyé sur le réseau.

7.2 SSH

Lorsqu'un utilisateur s'authentifie par SSH, alors il envoye son mot de passe à travers un canal crypté. Si la machine distante n'est pas le serveur SSH attendu, alors elle peut récupérer le login et le mot de passe.
Pour se prémunir contre cette attaque, SSH stocke la clé publique de chaque machine et refuse de se connecter à ces dernières si leur clé publique a changé. De même, lors d'une première connection le fingerprint de la clé publique est affiché et c'est à l'utilisateur de confirmer qu'il souhaite se connecter à la machine en question.
Dans le cas d'un scanner, garder la clé publique de chaque machine est rigoureusement inutile : il est impensable de demander à l'utilisateur du scanner de vérifier le fingerprint de chaque machine testée.
La seule solution que nous avons identifié (et adopté) dans le cas de Nessus est de préconiser l'usage d'une authentification à clé publique ou bien en passant par kerberos.

7.3 Telnet, IMAP, POP, HTTP, etc...

Ces protocoles envoyent des mots de passe en texte clair sur le réseau. Il faut bien évidemment éviter de s'en servir.

7.4 Conclusion

Un pirate ayant pris le controle d'une machine peut aisément la modifier pour qu'elle enregistre les tentatives de logins ainsi que les mots de passes employés. Dans le cas de SMB, il est facile de mentir sur les protocoles supportés afin d'obtenir des mots de passes faciles à craquer.
Il faut donc configurer la machine faisant le scan de telle sorte que cette dernière soit stricte quand aux échanges de mots de passe utilisés, et éviter l'usage de mots de passes pour l'authentification. L'usage de kerberos (utilisé dans les Active Directory de Windows) est fortemment recommandé.

8 Execution de code

En examinant un peu plus loin la section 2 du paragraphe « Ralentir l'audit », il est évident qu'il est possible d'utiliser des bugs permettant de faire planter un scanner pour lui faire executer du code arbitraire.
Un scanner de vulnérabilité est une bonne cible pour une telle attaque, car il tourne en tant que root ou administrateur du domaine.

8.1 Etude d'un cas particulier : MS05-011

- **Le bug** : L'avis MS05-011 [6] de Microsoft parle d'une faille dans SMB qui permet d'executer du code à distance. L'analyse de ce bug montre que pour exploiter cette faille, un pirate a besoin de convaincre la machine cible de se connecter à un serveur SMB envoyant des paquets mal formés.
- **Comment l'exploiter** : De manière générale, l'exploitation de ce bug n'est pas aisée : il faut que la machine cible se connecte sur un serveur modifié, et que cette dernière demande le contenu d'un dossier. Le serveur modifié envoye des entrées dont le nom est trop long, ce qui cause un dépassement de buffer du coté du client.
  Exploiter ce bug au niveau d'un scanner est en revanche trivial, car on sait que le scanner va se connecter à certains partages (`C$` et `ADMIN$` en particulier). Comme les scanners de vulnérabilités tournant sous Windows utilisent l'API de Microsoft pour se connecter à un autre Windows, l'exploitation de ce bug devient facile pour le pirate - il suffit d'attendre le scan puis d'exploiter le scanner. Quelle que soit la qualité de codage de ce dernier, même s'il utilise un langage de script pour les tests eux-meme, il est vulnérable à cet overflow.
- **Solutions** : Nous recommandons l'implémentation de chaque protocole dans le langage de script du scanner afin d'éviter de tels dépassements

de tampon. En forçant le scanner à encoder et décoder lui-même tous les paquets du réseau (et non pas en passant par une librairie externe), alors seulement les auteurs de scanners de prémunissent contre de telles attaques. La seule librairie externe que Nessus utilise pour du traffic réseau est la librairie OpenSSL - tous les autres protocoles (SSH, SMB, SNMP, HTTP, etc...) sont implémentés en NASL afin d'éviter de telles attaques.

9 Conclusion

Utiliser un scanner de vulnérabilité est un acte de controle et de supervision de la sécurité du réseau qui est plus que nécéssaire. Cependant, un scanner n'est pas un programme anodin et son usage comporte de nombreux risques pour l'intégrité du réseau elle-même. En profitant d'un scanner mal codé, un pirate peut gagner plus de privilèges sur le réseau, transformant ainsi une opération d'audit en un cauchemar sécuritaire.

Certains vendeurs de scanners font très attention à ce type d'attaque, alors que d'autres sont plus négligeants tant dans le design de leur produit que dans leur implémentation, aussi il ne faut pas hésiter à demander à ceux-ci les gages de sécurité relatif à leur cycle de dévelopement.

Références

1. Discard est un faux service TCP qui se contente de garder une connection ouverte et d'ignorer tout le traffic qu'il reçoit. `http://descriptions.securescout.com/tc/15040`
2. G. Roualland et J.-M. Saffroy, *IPPersonality*, `http://ippersonality.sourceforge.net`
3. T. Liston, *LaBrea* (programme destiné à ralentir la propagation des vers en répondant à la place des adresses IP inutilisées du réseau. S'il voit passer un SYN, il renvoye un SYN—ACK et ne renvoye plus rien, ce qui ralentit énormément la machine infectée par un ver, car la connection est - de son point de vue - établie). `http://labrea.sourceforge.net`
4. N. Provos, *Honeyd* (daemon simulant la présence d'une machine (ainsi que ses services). Il est habituellement utilisé dans le cadre de tests de honeypots, mais fait très bien l'affaire lorsqu'il s'agit de surcharger de travail une équipe de sécurité), `http://www.citi.umich.edu/u/provos/honeyd/`
5. D. Kaminsky, *ScanRand* (scanner de ports privilégiant la vitesse sur la perte de paquets), `http://www.doxpara.com/paketto/paketto-1.10.tar.gz`
6. MS05-011 : Une faille client dans SMB découverte par la société eEye. `http://www.microsoft.com/technet/security/bulletin/ms05-011.mspx`