Détection passive de systèmes d'exploitation

Julien Bordet

3 janvier 2002

Résumé

La détection de l'OS installé sur une machine cible est aujourd'hui une fonctionnalité fondamentale offerte par les scanners de ports comme *nmap*. Précise et efficace, elle permet d'obtenir des informations sensibles concernant la machine visée qui peuvent constituer un bon début pour un attaquant lors d'une tentative de piratage. Néanmoins, cette technique souffre de deux défauts majeurs : tout d'abord elle n'est pas discrète et ensuite elle peut être contournée. En effet, l'envoi de paquets formatés spécialement pour la détection de systèmes d'exploitation ne passe pas inaperçu et peut être détectée par un système de détection d'intrusion. De plus, il est possible à l'administrateur de tromper l'attaquant en modifiant les réponses du système, à l'aide d'outils tels que *ippersonality*[1],*grsecurity*[2] ou simplement en paramètrant son système.

La détection d'OS passive offre une alternative à cette première technique. Elle permet d'obtenir de relativement bonnes informations concernant l'hôte cible, **sans lui envoyer le moindre paquet** : le scanner se contente alors d'écouter les paquets qui circulent sur le cable, et en déduit avec une grande probabilité le système utilisé sur les machines du LAN. Elle souffre néamoins de plusieurs problèmes, dont le principal la nécessite d'obtenir, la plupart du temps, un accès sur le même réseau que la machine cible pour l'exploiter.

Ce document est une introduction à la détection passive de systèmes d'exploitation et les principes mis en jeu.

[1]http ://ippersonality.sourceforge.net/
[2]http ://www.grsecurity.net

Table des matières

1 Principes **3**

2 En-tête IP et TCP **3**
2.1 IP 3
2.2 TCP 4
2.2.1 En-tête TCP 4
2.2.2 Options TCP 5

3 Détection passive d'OS **5**
3.1 A quels paquets s'intéresser ? 5
3.2 Quels paramètres étudier ? 6
3.3 Quels avantages ? 7
3.4 Quels inconvénients ? 7

1 Principes

La détection d'OS passive consiste à *sniffer* le réseau pour scruter les paquets qui y passent, et notamment leur en-tête. En effet, chaque système d'exploitation a *sa* méthode pour, par exemple, construire l'en-tête IP ou l'en-tête TCP d'un paquet avant de l'envoyer[3].

Ainsi, certaines valeurs de l'en-tête TCP, telles que la taille la fenêtre, ou de l'en-tête IP, telles que le champ TTL, sont initialisées différemment selon les systèmes. En combinant un certain nombre de ces informations, il devient possible de tenter de deviner quel système d'exploitation est utilisé sur la machine cible.

2 En-tête IP et TCP

2.1 IP

Pour des rappels sur l'en-tête du protocole IP, voir la RFC 791.

Passons néanmoins en revue ses champs :

- **version** : la plupart du temps, ce champ à pour valeur *4*[4]
- **LET** : ce champ indique la taille de l'en-tête IP. Il peut se révéler important.
- **TOS** : Type Of Service
- **Longueur totale**
- **Identification**
- **FLAGS** : parmi ces drapeaux figure le champ *Don't Fragment*, dont la présence ou non peut permettre d'obtenir des informations. Ce champ permet de ne pas autoriser les routeurs de réseaux traversés à segmenter le paquet en plusieurs paquets de taille plus petite lorsque cela est requis. Dans ce cas, le comportement normal des routeurs consiste à envoyer un message ICMP à l'expéditeur signalant le problème, charge à ce dernier d'envoyer des paquets plus petits.
- **Fragment Offset**
- **TTL** : ce champ, décrémenté par chaque routeur par lequel transite le paquet permet de limiter la durée de vie d'un paquet lors d'une "boucle" éventuelle de routage, au sein de laquelle le paquet "tournerait" sans fin. Lorsque la valeur de ce champ atteint 0, le paquet est détruit.
- **Protocole** : indique le protocole de couche supérieure contenu dans le paquet.

[3] Le protocole UDP n'est bien entendu par concerné ici, car tellement simple qu'aucune différence n'existe entre les systèmes

[4] et avec l'avènement de IPv6, on pourra voir de plus en plus souvent la valeur 6. Néanmoins, cela ne nous informe que peu a propos du système d'exploitation. La seule certitude est que si la valeur 6 est trouvée, ce ne peut être qu'un système supportant - déjà - IPv6 ;-)

- **Somme de contrôle d'en-tête** : permet de contrôler l'intégrité de l'en-tête IP.
- **Adresse source**
- **Adresse destination**
- **Options**

2.2 TCP

Le protocole TCP est défini dans la RFC 793. S'y référer pour plus d'informations.

2.2.1 En-tête TCP

Les champs de l'en-tête TCP sont les suivants :
- **Ports source et destination**
- **Numéro de séquence** : le numéro de séquence permet d'identifier de manière unique le paquet dans le flux de communication entre deux machines, notamment pour pouvoir remettre les paquets dans l'ordre. Théoriquement, il pourrait être possible d'obtenir des informations sur les systèmes d'exploitations en observant la manière dont ces numéros de séquences évoluent. Néanmoins, les procédés aléatoires utilisés par certains d'entre eux ne permet de déduire de l'enchaînement des numéros de séquence les OS.
- **Accusé de réception** : ce champ permet de préciser à l'expéditeur quels sont les paquets que le destinataire a bien reçu. Il est donc utile dans le processus de contrôle de flux.
- **Data offset** : ce champ représente la taille de l'en-tête TCP. Le lien étant direct entre la taille de l'en-tête IP et celle de l'en-tête TCP, ce champ peut être informatif au même titre que le champ LET.
- **Bits de contrôle** : ces champs permettent de caractériser la place du paquet dans le flux (demande de connexion, réponse, demande de fermeture, ...). Ils sont fondamentaux car seuls certains de ces types de paquets sont intéressants.
- **Fenêtre** : nombre d'octet à partir de la position marquée dans l'accusé de réception que l'expéditeur est capable de recevoir à un instant donné. Ce champ permet le contrôle de flux et la modulation de la vitesse d'émission en fonction de la charge.
- **Somme de contrôle** : permet de contrôler l'intégrité du datagramme.
- **Pointeur de données urgentes**

2.2.2 Options TCP

De plus, l'en-tête TCP peut contenir un certain nombre d'options[5]. Les champs d'option sont situés après l'en-tête standard.

A l'aide de la première de ces options, il est possible à un hôte de préciser la taille maximale des paquets TCP qu'il pourra recevoir (ou **MSS** pour **Maximum Segment Size**). Elle peut être précisée au début de la connexion, uniquement dans les paquets dont l'en-tête contient le drapeau **SYN**.

L'option **Windows Scale** apporte une extension au champ **Fenêtre** déjà présent dans l'en-tête TCP. En effet, devant l'augmentation des débits sur Internet, les fenêtres maximums de 2^{16} bits, soit 64Kb, se sont avérées insuffisantes. L'option **Window Scale** permet donc de préciser des fenêtres de taille supérieure.

Devant l'augmentation de la taille des fenêtres évoquée ci-dessus, émettre un accusé de réception par fenêtre a commencé à devenir insuffisant. Il fallait donc pallier ce problème en autorisant des acquittements sélectifs au sein de la fenêtre : c'est le rôle de l'option **SACK** (pour Selective Acknowlegment).

De même, les *timeouts* relatifs aux acquittements TCP, mesuré à partir du round trip time[6], se devait d'être plus précis du fait de l'augmentation de la bande passante. L'option **TimeStamp** permet donc de mesurer ce temps, et d'agir sur les paramètres de la pile TCP/IP correspondant.

Enfin, l'option **NOP** ne fait rien.

3 Détection passive d'OS

Lors de l'analyse des flux, seul le protocole TCP sera utilisé. En effet, comme on l'a vu précédemment, la "complexité" de son en-tête permet une modulation des valeurs des champs en fonction des systèmes d'exploitation, et donc l'apport d'informations intéressantes.

3.1 A quels paquets s'intéresser ?

Les paquets TCP scrutés sont toujours ceux du début de la connexion (SYN et SYN/ACK) pour une raison simple : toutes les informations utiles peuvent être optenues à partir de ces types de paquets, et certaines informations ne sont présentes que dans ces paquets[7].

[5] définiées dans les RFCs 1323 et 2018

[6] ou temps nécessaire pour un allez-retour entre l'émetteur et le destinataire

[7] Notamment certaines options TCP sont "proposées" au destinataire lors de l'initialisation de la connexion, et peuvent ne pas être présentes par la suite : par exemple le **Selective Acknowledgment**

Les détecteurs d'OS passifs scrutent et enregistrent donc uniquement les paquets TCP SYN et SYN/ACK.

3.2 Quels paramètres étudier ?

Les champs ou options étudiés lors de l'analyse passive de système d'exploitation sont les suivants :

- la valeur du champ **Time To Live** IP : la valeur initiale de ce champ dépend du système utilisé. Néanmoins, comme elle peut avoir été décrémentée par le passage par un ou plusieurs routeurs, il est nécessaire de l'arrondir à la puissance de 2 supérieure[8].
- la présence du drapeau **Don't Fragment** IP : certains OSs utilisent cette option pour savoir quelle est la taille maximale de paquets (Path MTU Discovery).
- la valeur de la **Fenêtre** TCP : la valeur initiale de ce champ est dépend du système utilisé.
- la valeur du champ **Maximum Segment Size** TCP : *idem.*
- la valeur de l'option **Windows Scale** TCP : *idem.*
- la présence de l'option **Selective Acknowledgment** TCP : cette option peut être supportée ou non par les piles réseaux des deux hôtes. Par conséquent, il est nécessaire de se mettre d'accord au début de la connexion - et donc lors des paquets SYN et SYN/ACK - sur son utilisation ou non.
- la présence de l'option **NOP** TCP : certaines implémentations du protocole TCP ajoutent au sein des paquets SYN un certain nombre d'options **NOP**. Leur présence et leur nombre est une information précieuse.
- la présence de l'option **Timestamp** TCP : toutes les implémentations du protocole TCP n'utilisent pas cette option.

La présence ou non de ces champs ainsi que leur valeur éventuelle permettra de déterminer avec une précision relativement importante les systèmes d'exploitation utilisés par les machines du réseau local : l'association de chacun de ces paramètres permet de caractériser un ou plusieurs OSs.

[8] la valeur initiale affectée est soit une puissance de 2, soit la valeur maximale, 255

Ainsi, on pourra dire que

$$\left.\begin{array}{r} TTL = 64 \\ !DF \\ Fenetre = 19636 \\ MSS = 43013 \\ Window\_Scale = 00 \\ !SACK \\ NOP \\ Timestamp \end{array}\right\} \Rightarrow OpenBSD\ 2.9$$

L'établissement d'une base de données des associations paramètres $\leftrightarrow$ système d'exploitation permet donc d'opérer la détection d'OS passive.

Cette méthode permet de plus de déterminer quels sont les ports ouverts sur la machine cible : en effet, le port source des paquets TCP SYN/ACK est forcément un port ouvert sur la machine émettrice du paquet. En observant ces informations pendant un temps long, on peut en déduire la plupart des ports TCP ouverts.

3.3 Quels avantages ?

La technique de détection d'OS passive possède un certain nombre d'avantages :

- **l'analyse des hôtes présents sur le réseau** : bien entendu, l'avantage principal de cette technique est sa capacité à identifier de manière relativement précise les systèmes d'exploitation utilisés.
- **la discrétion** : le sniffer n'émet **aucun** paquet corresdpondant à sa recherche d'information.

3.4 Quels inconvénients ?

Néanmoins, cette méthode n'est pas parfaite :

- elle n'est pas un modèle de précision : en effet, une même signature correspond souvent à plusieurs OSs. De plus, la plupart des valeurs utilisées pour détecter l'OS utilisé par la cible peuvent être modifiée manuellement par l'administrateur.
- elle nécessite soit que l'actif réseau soit un hub pour pouvoir récupérer les paquets des autres machines, soit d'utiliser la technique de corruption de cache ARP, qui n'est pas passive[9] et qui peut donc être repérée[10].

[9] voir le logiciel **ettercap** (`http ://ettercapp.sourceforge.net`)

[10] Notons par ailleurs que le sniff sur un hub nécessite un passage en mode *promiscous*, qui peut être détecté

- elle est relativement lente : il est conseillé de laisser fonctionner les logiciels utilisant cette technique durant plusieurs jours pour obtenir des résultats intéressants et significatifs.
- certaines applications (hping, nmap) construisent leurs paquets elle-même : il n'est alors pas possible d'obtenir des informations pertinentes.
- la technique de scan de ports passive dépend du trafic : en effet, si aucun client ne se connecte à un port ouvert donné, le sniffer ne pourra savoir qu'il est ouvert.
- du point de vue de l'attaquant, il est nécessaire pour lui d'avoir un accès à une machine sur le réseau dont il veut obtenir des informations[11].

[11] Contrairement à la détection d'OS active qui est réalisable à distance

Bibliographie

- Jacobson V., Braden R., Borman D. - *RFC 1323 TCP Extensions for High Performance* - 1992
- Mathis M., Mahdavi J., Floyd S., Romanow A. - *RFC 2018 TCP Selective Acknowledgment Options* - 1996
- Ornaghi A., Valleri M. - *README du logiciel ettercap* - 2001
- Spitzner L., *Know Your Enemy : Passive Fingerprinting* - 2001
- University of Southern California, *RFC 791 INTERNET PROTOCOL* - 1981
- University of Southern California, *RFC 793 TRANSPORT CONTROL PROTOCOL* - 1981