RFC 4443 : Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 17 octobre 2010

Date de publication du RFC : Mars 2006

https://www.bortzmeyer.org/4443.html

Il y a un protocole peu connu dans le monde IP, c'est ICMP. Situé dans les couches basses, il ne sert pas directement aux applications, à part à ping, avec qui il est souvent confondu (c'est ainsi que pas mal de blaireaux administrateurs d'un pare-feu bloquent tout ICMP parce qu'ils se disent qu'il n'ont pas besoin de ping). ICMP pour IPv4 était normalisé dans le RFC 792 [1] et la version (très proche) qui sert pour IPv6 était dans le RFC 2463, désormais remplacé par notre RFC 4443.

En quoi consiste ICMP ? Techniquement parlant, il est situé au dessus d'IP dans le modèle en couches, son champ "*Next Header*" portant le numéro 58. Mais, en fait, ce protocole est une partie intégrante d'IP. Il sert à transporter les messages liés au fonctionnement d'IP, par exemple à prévenir si un routeur ne peut plus acheminer les paquets car il n'a plus de route pour cette destination, ou bien à prévenir qu'un paquet est trop gros pour la MTU du lien suivant. ICMP v6 sert aussi à bâtir d'autres protocoles, comme NDP (RFC 4861), qui utilise des messages ICMP pour permettre la découverte d'une machine voisine sur le réseau local. ICMP faisant partie intégrante d'IP (cf. section 2), ce RFC 4443 fait donc partie des lectures indispensables pour qui veut comprendre la version 6 d'IP.

Le format des messages ICMP v6 est en section 2.1. Après l'en-tête IPv6 et zéro, un ou plus en-têtes d'extension (par exemple pour la fragmentation) arrive la partie purement ICMP, introduite par un champ "*Next Header*" valant 58. Les trois champs communs à tout paquet ICMP sont ensuite le **type** (huit bits), le **code** (huit bits) et une somme de contrôle (seize bits, définie en section 2.3). Voici comment Wireshark le représente en C :

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc792.txt

```
struct icmp6_hdr {
guint8 icmp6_type; /* type field */
guint8 icmp6_code; /* code field */
guint16 icmp6_cksum; /* checksum field */
union {
guint32 icmp6_un_data32[1]; /* type-specific field */
guint16 icmp6_un_data16[2]; /* type-specific field */
guint8 icmp6_un_data8[4];  /* type-specific field */
} icmp6_dataun;
};
```

Comme l'indiquent les commentaires dans le source de Wireshark, le format des données dépend du type. Il y a deux catégories de types, ceux qui indiquent une erreur (de 0 à 127) et les autres. Il existe de nombreux types possibles (stockés dans un registre IANA <https://www.iana.org/assignments/icmpv6-parameters>, cf. section 6). Parmi ceux définis directement par ce RFC, pour vous donner une idée :

— 1 est envoyé par un routeur lorsque la destination n'est pas joignable,
— 2 indique un paquet trop gros pour la MTU,
— 3 indique que le nombre de routeurs traversé a atteint une limite (il est donc utilisé par traceroute) : on notera que le RFC l'appelle « "*Time exceeded*" », comme en IPv4, alors que le champ correspondant du paquet n'est officiellement plus un TTL en IPv6 mais un nombre de routeurs ("*hop count*"),
— 100 et 101 sont réservés pour des expérimentations,
— 128 (qui n'est donc pas une erreur, puisque ¿ 127) est la demande d'un écho et 129 la réponse. Ce sont donc les deux types utilisés par ping.

Notez que les valeurs ne sont pas du tout les mêmes qu'en IPv4, même lorsque la sémantique est la même.

Le format détaillé figure en section 3. Ainsi, lorsque le type est 1 ("*Destination unrechable*"), le code indique la raison pour laquelle la dite destination était injoignable : 0 pour « aucune route disponible », 1 pour « interdit » (par un pare-feu), 3 pour « machine injoignable » (on est sur le bon réseau mais la machine de destination ne répond pas), etc. traceroute traduit ces codes en lettres précédées d'un ! donc respectivement !N, !X et !H.

Le cas du type 2, « paquet trop gros » (section 3.2) est intéressant car les routeurs IPv6, contrairement à leurs frères IPv4, n'ont pas le droit de fragmenter. La source doit donc prendre garde de ne pas générer des paquets plus gros que la MTU du chemin (cf. RFC 1981) et, pour cela, a besoin de ces paquets ICMP de type 2 (qui comprennent, après les trois champs génériques, un champ de 32 bits indiquant la MTU du prochain lien). Hélas, beaucoup de pare-feux sont gérés par des ignorants et bloquent **tout** l'ICMP. Ces paquets n'arrivent donc pas à la source et celle-ci ne peut plus ajuster la taille des paquets. C'est un des problèmes réseau les plus courants avec IPv6, que l'on voit dès que la MTU est inférieure aux traditionnels 1500 octets, par exemple parce qu'il y a un tunnel sur le trajet.

Les paquets liés à la fonction écho comprennent quant à eux un identificateur et un numéro de séquence (sections 4.1 et 4.2), de seize bits chacun. Copiés automatiquement dans la réponse, ils servent à mettre en correspondance une réponse avec une question (au cas où plusieurs utilisateurs d'une machine se servent de ping en même temps).

Quelle adresse IP source doit être utilisée lorsqu'un équipement IPv6 génère un paquet ICMP ? La section 2.2 est claire : si le paquet ICMP répond à un message qui était adressé personnellement à la machine (cas d'un ping), l'adresse source doit être celle à qui était adressé le message. Sinon (cas d'un routeur qui doit traiter un paquet qui ne lui est pas destiné), l'adresse source doit être une adresse "*unicast*" du routeur émetteur.

Reprenant les principes du RFC 1122, notre RFC spécifie, en section 2.4, les règles à suivre lors du traitement de paquets ICMP. Ainsi, les messages d'information de type inconnu doivent être ignorés (pour permettre leur introduction ultérieure sans rien casser). Comme en IPv4, un équipement IPv6 ne doit jamais répondre à un message ICMP d'erreur (pour éviter les boucles). Les messages d'erreur à propos d'un paquet doivent inclure autant d'octets que possible du paquet original sans toutefois dépasser les 1260 octets qui sont le minimum qu'un réseau doit accepter pour faire de l'IPv6 (cette règle est plus stricte que la règle IPv4 originale qui n'imposait pas de remplir le paquet au maximum). Autre point où ICMP v6 diffère, la limitation du nombre de paquets ICMP émis est obligatoire, pour éviter certaines attaques où le méchant convainc un routeur de générer des paquets ICMP en rafale. La méthode recommandée pour ce "*rate-limiting*" est le seau qui fuit. Les méthodes simplistes (du genre « un paquet émis toutes les N milli-secondes »), qui ne peuvent pas faire face à du trafic très variable (comme celui de traceroute) sont par contre déconseillées. Les paramètres du seau devraient être configurables (Linux permet apparemment cette configuration mais je n'ai pas regardé la documentation du paramètre sysctl `net.ipv6.icmp.ratelimit`, qui vaut 1000 par défaut, mais 1000 quoi ?).

Contrairement au RFC 792 sur ICMP v4, qui ne disait pas **un mot** sur la sécurité, notre RFC 4443, écrit à une époque plus sensible, consacre une section, la 5, à ces questions. Elle rappelle qu'il n'y a aucune authentification des paquets ICMP et qu'il ne faut donc pas agir aveuglément sur la base d'un paquet ICMP reçu. On peut toujours, prétend le RFC, utiliser IPsec pour les authentifier mais très peu de gens font cela. À part IPsec, solution peu réaliste aujourd'hui, notre RFC recommande des tests de validité (ou, plus exactement, de vraisemblance) pratiqués par les protocoles de transport, comme ceux du RFC 5927.

Depuis la précédente version, le RFC 2463, ICMP v6 a subi plusieurs changements, résumés dans l'annexe A. Parmi eux, le champ de 32 bits dans les messages d'erreur, après les trois champs génériques, même lorsqu'il est inutilisé (cas le plus fréquent), afin de permettre de le sauter facilement, même avec les types inconnus, pour accéder à la copie du paquet original. On note aussi la réservation de types pour des expérimentations.