Rechercher

Réseau ➡ Ping
Dern. mise à jour: 22-09-2010

WWW.OPENMANIAK.COM
World Wide Made

If you like our tutorials, don't hesitate to support us and visit our sponsors!
Si vous aimez nos tutoriaux, n'hésitez pas à nous supporter et visiter nos sponsors!

TOTAL
Depuis Dec 2006
1'942'871 Visiteurs
4'218'042 Pages

➡ **Stats Nov 2010**
82'909 Visiteurs
146'476 Pages
196 pays
Statistiques completes

➡ Aidez-nous à traduire nos tutoriaux!

➡ REJOINGNEZ l'équipe OpenManiak.

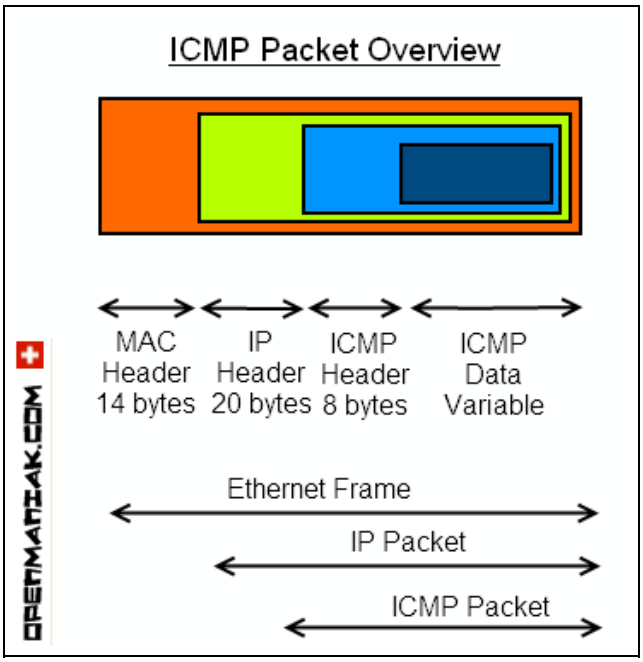
Ping - Table des matières

INTRODUCTION

Ping est un outil bien connu pour vérifier la connectivité réseau entre deux hôtes IP. Il a été créé en 1983 par Mike Muus qui a écrit un article à propos de son outil The Story of the PING Program" (L'histoire du programme Ping) peu de temps avant de mourir en 2000 dans un accident de voiture.

Ping est installé par défaut sur les systèmes d'exploitation Windows, Apple et Linux/Unix.
Il utilise le protocole ICMP qui a été créé pour vérifier la connectivité et obtenir des informations à propose de machines dans un réseau IP.
ICMP est encapsulé dans un paquet IP, mais est considéré comme faisant partie de la couche IP ou Internet.

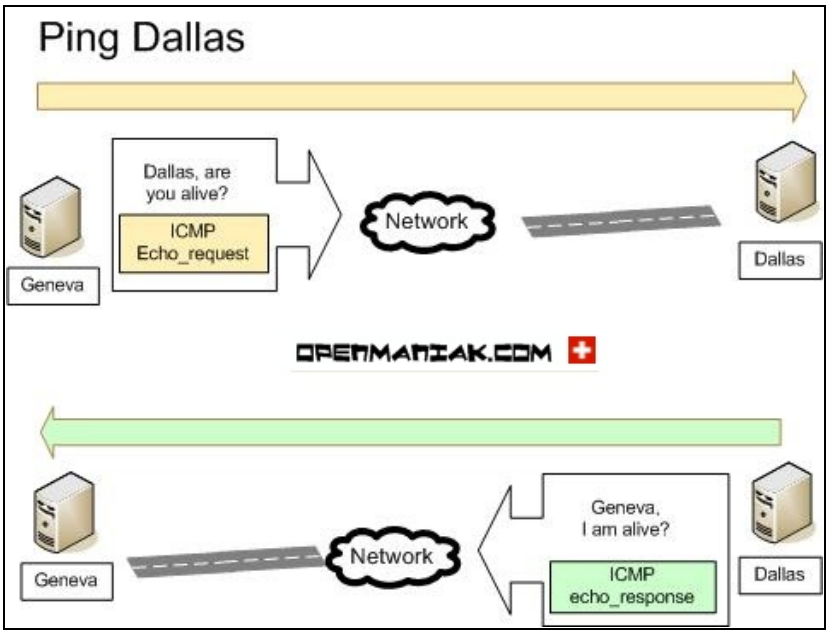
➡ Vue d'ensemble d'un paquet ICMP



➡ Fonctionnement de Ping

Ping envoie des très petits paquets vers un hôte IP qui va répondre en envoyant des paquets en retour. Les paquets ICMP envoyé à l'hôte sont appelés echo_request et les paquets envoyés en retour echo_response.

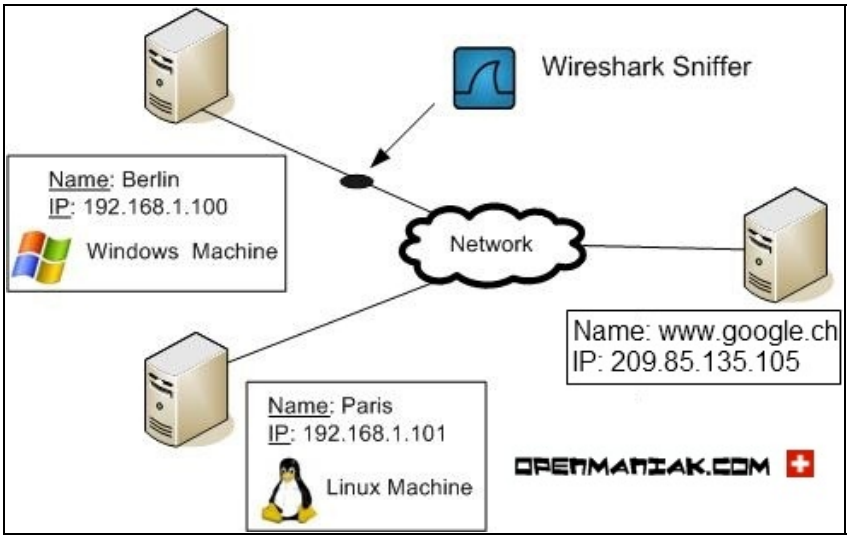
Il y a aussi d'autres types de paquets ICMP qui sont détaillé plus bas dans cette page.



Haut de la Page

ÉTUDE DE CAS

Examinons une étude de cas où deux machines appelées Paris et Berlin "pingent" une machine appelée "www.google.ch".
Le sniffeur Wireshark est situé en chemin entre Berlin et "www.google.ch". Il sera utilisé pour capturer le contenu des paquets.
Paris est une machine Linux Ubuntu et Berlin une machine Microsoft XP.



Ping depuis la machine Linux (Paris) vers "www.google.ch":

Paris: **ping www.google.ch**

```
PING www.google.ch (209.85.135.105) 56(84) bytes of data.
64 bytes from www.google.ch(209.85.135.105): icmp_seq=1 ttl=255 time=1.19 ms
64 bytes from www.google.ch (209.85.135.105): icmp_seq=2 ttl=255 time=1.25 ms
64 bytes from www.google.ch (209.85.135.105): icmp_seq=3 ttl=255 time=1.26 ms
64 bytes from www.google.ch (209.85.135.105): icmp_seq=4 ttl=255 time=1.29 ms

--- www.google.ch ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
rtt min/avg/max/mdev = 1.192/1.250/1.290/0.044 ms
```

Ping from the Windows machine (Berlin) to "www.google.ch":

Berlin: **ping www.google.ch**

```
Envoi d'une requête 'ping' sur www.google.ch [209.85.135.105] avec 32 octets de données :

Réponse de 209.85.135.105: octets=32 temps=18 ms TTL=250
Réponse de 209.85.135.105: octets=32 temps=21 ms TTL=250
```

```
Réponse de 209.85.135.105: octets=32 temps=20 ms TTL=250
Réponse de 209.85.135.105: octets=32 temps=33 ms TTL=250

Statistiques Ping pour 209.85.135.105:
    Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
Durée approximative des boucles en millisecondes :
    Minimum = 18ms, Maximum = 33ms, Moyenne = 23ms
```

Quelles informations peuvent être tirées des résultats ci-dessus?

- Est-ce que l'hôte distant est actif? => Connectivité de l'hôte
- Est-ce que la vitesse du réseau est bonne? => Congestion réseau
- Est-ce que l'hôte distant est loin? => longueur du trajet

La section suivante "ANALYSE DES RÉSULTATS" fournit des détails complets sur l'interprétation des résultats de la commande Ping.

Nous pouvons aussi dire que l'outil Ping nous fournit les mêmes informations indépendamment du système d'exploitation sur lequel il est installé. Toutefois, il y a quelques différence au niveau des arguments et paramètres par défaut de la commande ping. Voir la section "ARGUMENTS PING" pour plus de détails.

Regardons un résumé des données capturées par Wireshark. Détails complets à la section "ANALYSE DETAILLÉE".

No.	Time	Source	Destination	Protocol	length	Info
1	0.000000	192.168.1.100	209.85.135.105	ICMP	74	Echo (ping) request
2	0.028303	209.85.135.105	192.168.1.100	ICMP	74	Echo (ping) reply
3	1.006063	192.168.1.100	209.85.135.105	ICMP	74	Echo (ping) request
4	1.032258	209.85.135.105	192.168.1.100	ICMP	74	Echo (ping) reply
5	2.005990	192.168.1.100	209.85.135.105	ICMP	74	Echo (ping) request
6	2.044394	209.85.135.105	192.168.1.100	ICMP	74	Echo (ping) reply
7	3.005980	192.168.1.100	209.85.135.105	ICMP	74	Echo (ping) request
8	3.046121	209.85.135.105	192.168.1.100	ICMP	74	Echo (ping) reply

Cette capture Wireshark nous donne les informations suivantes:

- Les paquets ont été reconnus comme des paquets ICMP. *colonne protocol*
- Quatre paquets ICMP ont été envoyés (echo_request) et quatre paquets ICMP ont été reçus (echo_reply). *colonne info*
- Une valeur de paquet de 74 bytes qui est composée par les en-têtes (headers, 42 bytes) et la partie ICMP donnée (ICMP data, 32 bytes par défaut sur Windows). *colonne length (longueur)*

Haut de la Page

ANALYSE DES RÉSULTATS

Comme indiqué dans la section précédente, Ping nous donne trois informations majeures:

Connectivité d'hôte

Les résultats du Ping de la section précédente montrent que quatre paquets ICMP ont été envoyés et quatre reçus. Ce résultat nous montre que l'hôte de destination est bien "vivant" au niveau ICMP mais ne nous donne aucune autre indication comme par exemple un serveur web est actif ou pas.

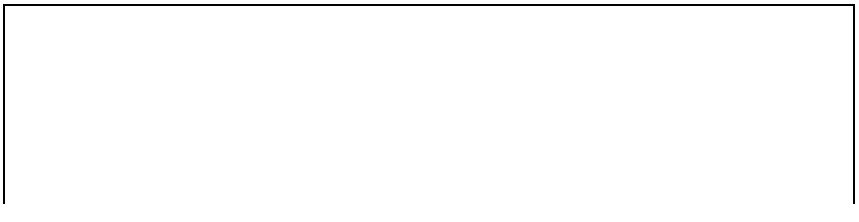
Que signifie un résultat négatif?
Regardons un exemple:

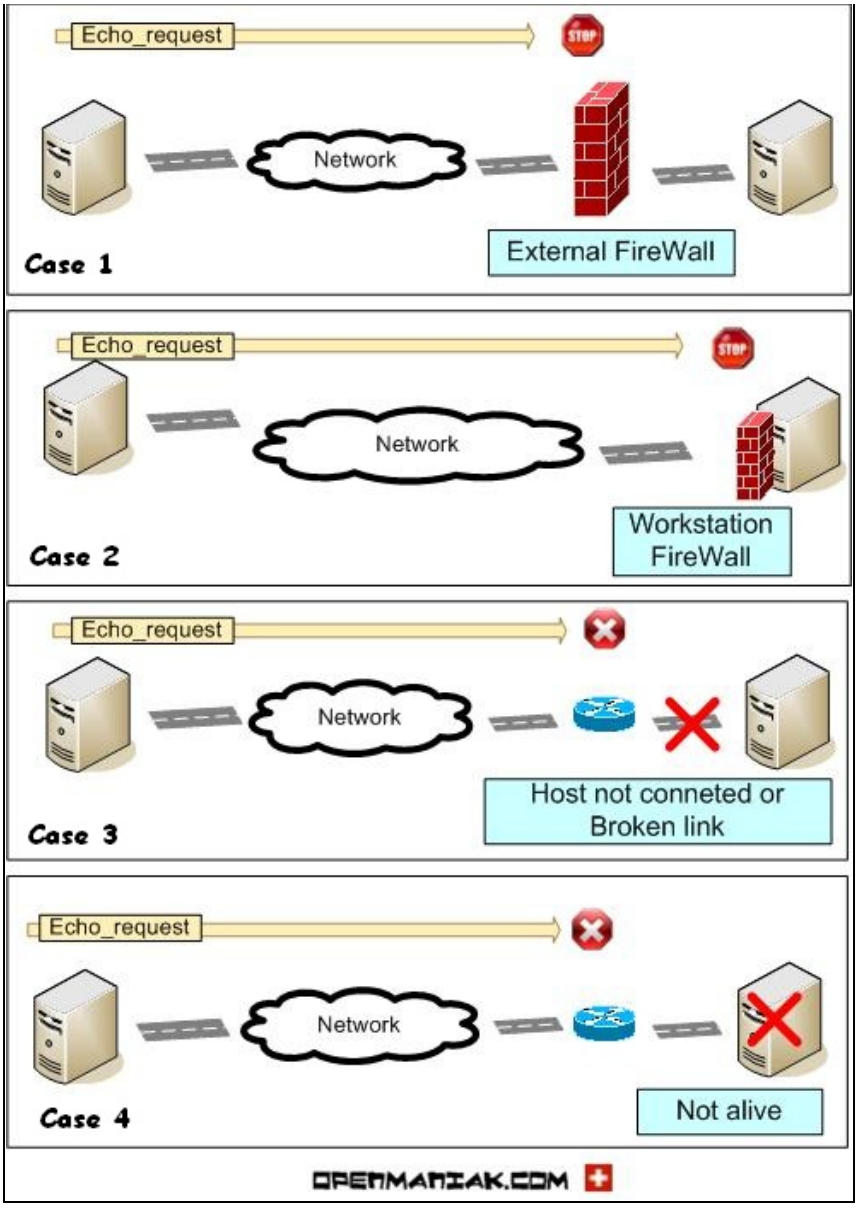
C:\>ping www.openmaniak.com

```
Envoi d'une requête 'ping' sur openmaniak.com [84.16.88.15] avec 32 octets de données :

Délai d'attente de la demande dépassé.
Délai d'attente de la demande dépassé.
Délai d'attente de la demande dépassé.
Délai d'attente de la demande dépassé.

Statistiques Ping pour 84.16.88.15:
    Paquets : envoyés = 4, reçus = 0, perdus = 4 (perte 100%),
```

L'hôte www.openmaniak.com qui a l'adresse IP 84.16.88.15 n'a pas répondu au Ping. Il est intéressant de dire que la réponse négative ne signifie pas toujours que le destinataire n'est pas vivant (dans notre exemple, il est vivant et fait tourner un serveur web), le résultat signifie juste qu'il ne répond pas à la requête ICMP.

Quelles peuvent être les raisons de cette "non-réponse"?
De nombreuses raisons peuvent en être la cause. Le dessin ci-dessous résume les quatre raison les plus fréquentes.





Dans le **premier cas**, un pare-feu (firewall) externe bloque les requêtes ICMP. ICMP peut être utilisé comme premier pas en vue d'une attaque parce qu'il peut indiquer les hôtes actifs sur le réseau. Dans ce cas, le réseau derrière le pare-feu est caché du monde extérieur même s'il ne répond pas au Ping. Bloquer les messages ICMP est une des recommandations de base pour protéger un réseau. Le pare-feu externe est plus souvent utilisé pour sécuriser les réseaux professionnels parce qu'il coûte cher et requière des connaissances avancées pour le configurer correctement.

Dans le **deuxième cas**, la station de travail possède un pare-feu (firewall) personnel qui bloque les messages ICMP. Un pare-feu personnel est recommandé pour les ordinateurs de maison pour les même raisons que celles décrites plus haut.

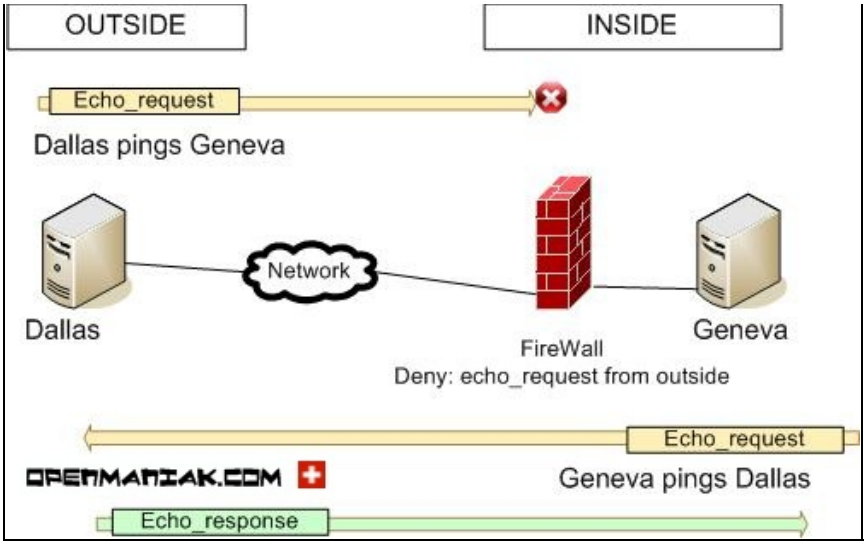
Dans le **troisième cas**, la machine "pingée" n'est pas connectée au réseau IP, par exemple parce que le câble réseau est débranché.
Le message echo_request sera détruit sur le dernier routeur ou équipement de niveau OSI 3 avant l'hôte distant.

Dans le **quatrième cas**, l'hôte est éteint ou a sa carte réseau désactivée. Comme dans le cas précédent, le message echo_request est détruit sur le dernier routeur ou équipement de niveau OSI 3 avant l'hôte distant.

Si un équipement appelé Geneva peut pinger un autre appelé Dallas, est-ce que cela signifie que l'inverse est aussi vrai? En d'autres termes, est-ce que Dallas peut aussi pinger Geneva?
La réponse est non. Comme vous pouvez le voir ci-dessous, un pare-feu (firewall) peut empêcher les paquets echo_request depuis uniquement un sens.

Quand Dallas ping Geneva, les paquets echo_request sont bloqués sur le pare-feu et donc Dallas ne reçoit aucune réponse de Geneva.

Quand Geneva ping Dallas, les echo_request atteignent Dallas et Geneva reçoit en retour les echo_response. Le ping s'est donc fait avec succès. Dans ce cas, les paquets ICMP ne sont pas bloqués parce que le pare-feu arrête seulement les echo_request venant de la zone outside. Ici nous avons des echo_request venant d'inside et des echo_response venant de outside.



Haut de la Page

Congestion réseau

Une autre information précieuse fournie par la commande Ping est le temps pris par un paquet pour atteindre la destination et revenir. Cette mesure est appelée le RTT (Round Trip Time) ou temps de réponse (response time) et est affichée en millisecondes.
Le temps de réponse ne doit pas être confondu avec la latence ou le délai qui sont le temps parcouru pas un paquet entre deux hôtes mais dans un sens.

Le temps de réponse va évidemment affecter la performance des applications réseau. Des temps de réponses élevés vont menés à des performances faibles.
Quand une application réseau est lente, un premier test basique pour dépanner est d'obtenir le temps de réponse entre le client et le serveur pour savoir si le réseau peut être la raison de la lenteur observée.

Il est important de ne pas oublier une autre information liée au temps de réponse: la perte de paquet (packet loss).
Un paquet est déclaré comme perdu si le message ICMP est détruit en chemin ou s'il retourne à l'expéditeur après le temps d'attente maximum (timeout) qui est de deux secondes par défaut.
Les pertes de paquets vont mener à un taux élevé de retransmission TCP et donc à des applications réseau lentes ou interrompues.
Dans un environnement local (LAN), il ne devrait pas y avoir la moindre perte de paquet.

Quels sont les facteurs pouvant affecter le temps de réponse et la perte de paquet?

- LES CABLES RÉSEAUX
La qualité et le type des câbles va grandement affecter le temps de réponse. Par exemple, les câbles en fibre optique sont plus efficaces que les vieux câbles en cuivre. De plus, des câbles de basse qualité ou endommagés mèneront à des pertes de paquets. Des tuyaux larges vont également aider à obtenir des temps de réponse bas parce qu'en cas de trafic élevé, les paquets ne vont pas ralentir comme pourront l'être des voitures sur un embouteillage.

- LES ÉQUIPEMENTS RÉSEAUX
Les équipements réseaux qui traitent les paquets le long du chemin peuvent légèrement accroitre le temps de réponse s'ils fonctionnent lentement. Par exemple, les paquets traversant un routeur surchargé vont subir des ralentissements parce que le routeur n'aura pas assez de ressources disponibles pour gérer les paquets rapidement.

- L'ÉLOIGNEMENT PHYSIQUE
Un équipement éloigné de soi aura un temps de réponse plus haut même avec des câbles et routeurs excellents. Ceci est du à l'éloignement physique et le nombre de routeurs à traverser pour joindre la destination.
Pour donner des valeurs très approximatives, le temps de réponse dans un environnement local (LAN) est d'environ 1 à2 millisecondes tandis que dans un équipement VPN internet éloigné environ 300 millisecondes.
Voici un exemple:
Un Ping de Genève/Suisse vers Google Suisse donne un temps de réponse d'environ 30 millisecondes et un Ping vers Google Japon 130 millisecondes.

- ÉQUIPEMENTS SOURCE ET DESTINATION
De nombreuses personnes pensent qu'un temps de réponse lent ou des pertes de paquets viennent de problèmes réseaux. Ce n'est très souvent pas le cas parce que les problèmes peuvent venir des équipements source et destination qui sont surchargées ou qui ont une carte réseau défectueuses.
Le résultat de la commande Ping donne les valeurs de temps de réponse minimum, moyenne et maximum qui aident à savoir si le temps de réponse reste constant ou varie.

Haut de la Page

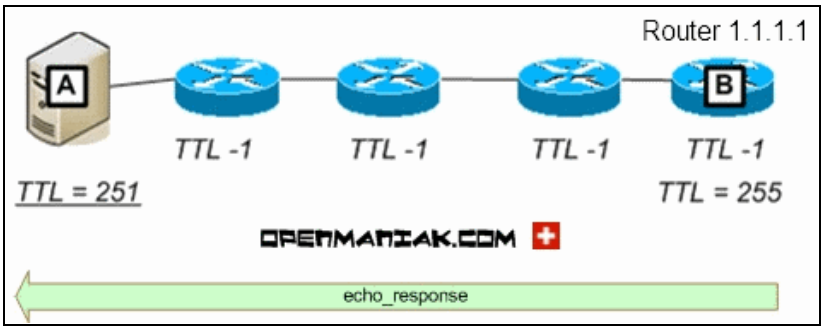
Time-to-Live (Temps-de-Vie)

Le TTL ou Time-To-Live (Temps-de-Vie) nous donne une indication sur le nombre de routeurs entre la source et la destination.
Le TTL est utilisé pour empêcher qu'un paquet IP ne tourne en boucle indéfiniment dans un réseau IP et ne mène à son effondrement.
La valeur initiale du TTL d'un paquet IP est de 255 et est ensuite est décrémentée de 1 chaque fois que le paquet traverse un routeur. Quand la valeur la valeur de 1 est atteinte, le paquet est détruit par un routeur. La valeur TTL est contenue dans chaque paquet IP incluant les paquets ICMP. Le TTL donné par la commande Ping est en fait la valeur TTL d'un paquet de type echo_response.
Par défaut, Windows va décroitre le TTL de 128 et Ubuntu Linux de 192.

Regardons trois scénarois où A ping B. B est un routeur (cas 1), une machine Microsoft Windows (cas 2) et une machine Ubuntu Linux (cas 3).
Le TTL est initialement de 255 et est diminué comme décrit ci-dessous.

La section "ANALYSE DETAILLÉE" fournit des informations à propos de la localisation du TTL dans un paquet IP.

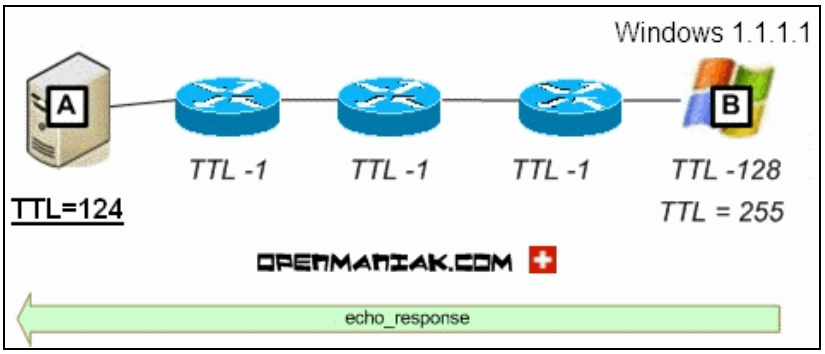
Cas 1:
Qaund A ping B, il reçoit un TTL de 251 parce que les paquets traversent 4 routeurs (-4).
TTL=255-4=251.



ping B

```
Envoi d'une requête 'ping' sur B [1.1.1.1] avec 32 octets de données :

Réponse de 1.1.1.1: octets=32 temps=18 ms TTL=251
Réponse de 1.1.1.1: octets=32 temps=21 ms TTL=251
Réponse de 1.1.1.1: octets=32 temps=20 ms TTL=251
Réponse de 1.1.1.1: octets=32 temps=33 ms TTL=251

Ping statistics for 1.1.1.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 18ms, Maximum = 33ms, Average = 23ms
```

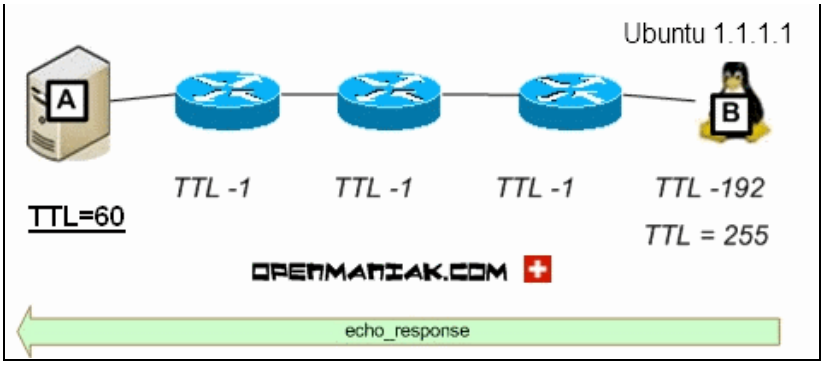
Cas 2:
Quand A ping B, il reçoit un TTL de 124 parce que les paquets traversent 3 routeurs (-3) et une machine Windows (-128).
TTL=255-3-128=124.



ping B

```
Envoi d'une requête 'ping' sur B [1.1.1.1] avec 32 octets de données :

Réponse de 1.1.1.1: octets=32 temps=18 ms TTL=125
Réponse de 1.1.1.1: octets=32 temps=21 ms TTL=125
Réponse de 1.1.1.1: octets=32 temps=20 ms TTL=125
Réponse de 1.1.1.1: octets=32 temps=33 ms TTL=125

Ping statistics for 1.1.1.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 18ms, Maximum = 33ms, Average = 23ms
```

Cas 3:
quand A ping B, il reçoit un TTL de 62 parce que les paquets traversent 3 routeurs (-3) et une machine Ubuntu (-192).
TTL=255-3-192=60.



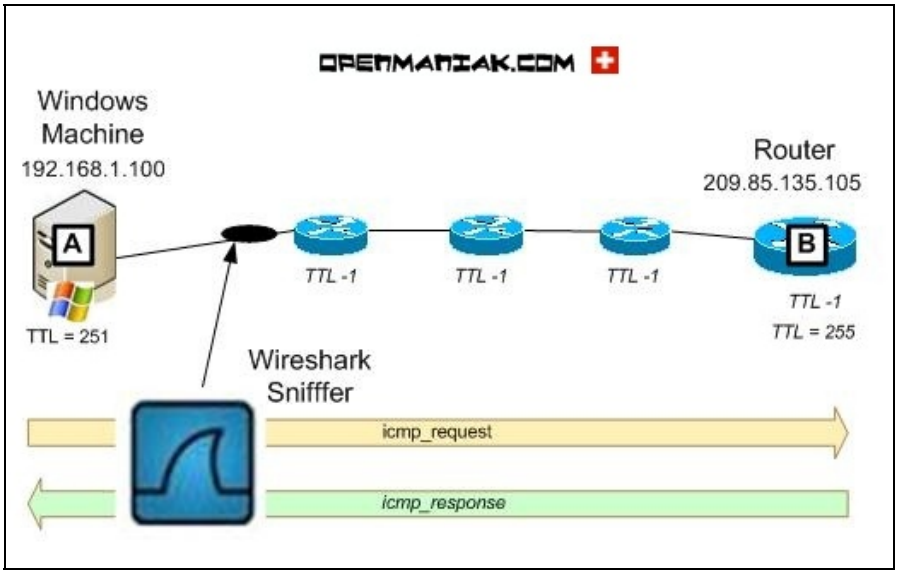
ping B

```
Pinging B [1.1.1.1] avec 32 octets de données :

Réponse de 1.1.1.1: octets=32 temps=18 ms TTL=60
Réponse de 1.1.1.1: octets=32 temps=21 ms TTL=60
Réponse de 1.1.1.1: octets=32 temps=20 ms TTL=60
Réponse de 1.1.1.1: octets=32 temps=33 ms TTL=60

Ping statistics for 1.1.1.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 18ms, Maximum = 33ms, Average = 23ms
```

Haut de la Page

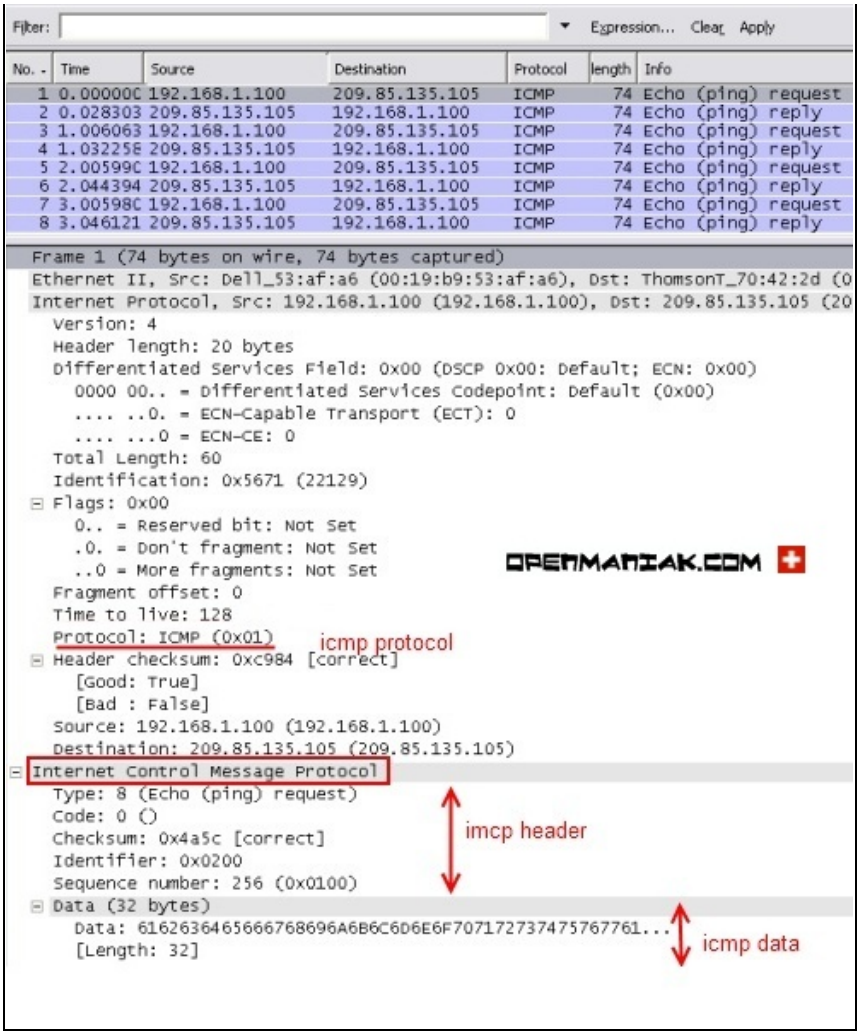
ANALYSE DETAILLÉE D'UN PAQUET ICMP

Cette section fournit une analyse détaillée d'un paquet echo_request et d'un paquet echo_response.



Echo_request

Le premier tableau est une copie d'écran d'une capture Wireshark et le second tableau une sortie texte de la même capture.



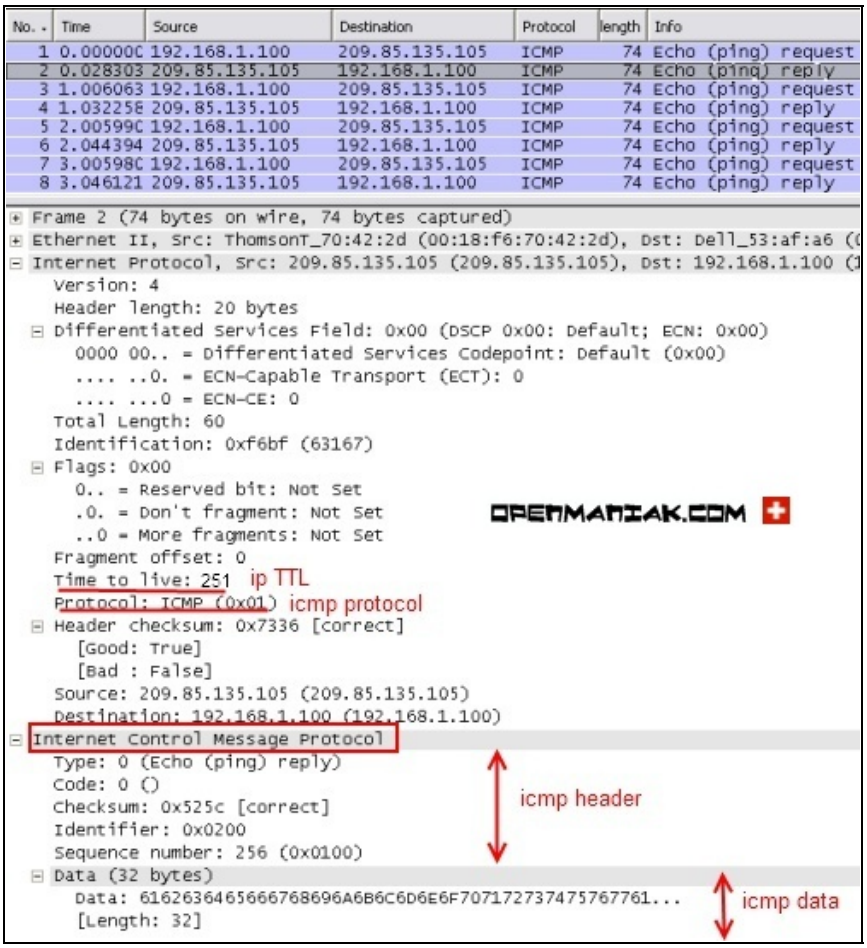

```
No.  Time        Source          Destination     Protocol length Info
1    0.000000    192.168.1.100   209.85.135.105  ICMP     74     Echo (ping) request

Frame 1 (74 bytes on wire, 74 bytes captured)
Ethernet II, Src: Dell_11:11:11 (00:19:b9:11:11:11), Dst: ThomsonT_99:99:99 (00:18:f6:99:99:99)
Internet Protocol, Src: 192.168.1.100 (192.168.1.100), Dst: 209.85.135.105 (209.85.135.105)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
            000000..        = Differentiated Services Codepoint: Default (0x00)
            ....  ..0.      = ECN-Capable Transport (ECT): 0
            ....  ...0      = ECN-CE:  0
    Total Length: 60
    Identification: 0x5671 (22129)
    Flags: 0x00
            0..   = Reserved bit: Not Set
            .0.   = Don't fragment: Not Set
            ..0   = More fragments: Not Set
    Fragment offset: 0
    Time to live: 128
    Protocol: ICMP (0x01)
    Header checksum: 0xc984 [correct]
            [Good: True]
            [Bad : False]
    Source: 192.168.1.100 (192.168.1.100)
    Destination: 209.85.135.105 (209.85.135.105)
Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0 ()
    Checksum: 0x4a5c [correct]
    Identifier: 0x0200
    Sequence number: 256 (0x0100)
    Data (32 bytes)

0000        61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f 70                 abcdefghijklmnop
0010        71 72 73 74 75 76 77 61 62 63 64 65 66 67 68 69                 qrstuvwabcdefghi
            Data: 6162636465666768696A6B6C6D6E6F707172737475767761...
            [Length: 32]
```

- La capture Wireshark (premier tableau en gris) montre les différentes couches OSI (Ethernet - IP - IMCP). A voir également, la section "Vue d'ensemble d'un paquet ICMP".
- Le paquet a été reconnu comme ICMP dans la couche IP.

- La couche ICMP est composée par une section en-tête (header) et une section donnée (data).
- La section en-tête ICMP est composée par: le type, code, checksum, identifiant (identifier) et numéro de séquence (sequence number).
- Le type ICMP a une valeur de 8 ce qui signifie que le paquet est un paquet echo_request.
- La valeur du numéro de séquence (sequence number), ici 256, est utilisée pour faire correspondre les echo_requests aux réponses associées (echo_response).

➡ Echo response

Le premier tableau est la capture d'écran Wireshark et le second est la capture Wireshark au format texte.




```
No.Time        Source          Destination                    Protocol lengthInfo
2  0.028303    209.85.135.105192.168.1.100                    ICMP     74    Echo (ping) reply

Frame 2 (74 bytes on wire, 74 bytes captured)
Ethernet II, Src: ThomsonT_70:42:2d (00:18:f6:99:99:99), Dst: Dell_53:af:a6 (00:19:b9:11:11:11)
Internet Protocol, Src: 209.85.135.105 (209.85.135.105), Dst: 192.168.1.100 (192.168.1.100)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
            000000..        = Differentiated Services Codepoint: Default (0x00)
            ....  ..0.      = ECN-Capable Transport (ECT): 0
            ....  ...0      = ECN-CE:  0
    Total Length: 60
    Identification: 0xf6bf (63167)
    Flags: 0x00
            0..   = Reserved bit: Not Set
            .0.   = Don't fragment: Not Set
            ..0   = More fragments: Not Set
    Fragment offset: 0
    Time to live: 251
    Protocol: ICMP (0x01)
    Header checksum: 0x7336 [correct]
            [Good: True]
            [Bad : False]
    Source: 209.85.135.105 (209.85.135.105)
    Destination: 192.168.1.100 (192.168.1.100)
Internet Control Message Protocol
    Type: 0 (Echo (ping) reply)
    Code: 0 ()
    Checksum: 0x525c [correct]
    Identifier: 0x0200
    Sequence number: 256 (0x0100)
    Data (32 bytes)
```

```
0000    61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f 70    abcdefghijklmnop
0010    71 72 73 74 75 76 77 61 62 63 64 65 66 67 68 69    qrstuvwabcdefghi
        Data: 6162636465666768696A6B6C6D6E6F707172737475767761...
        [Length: 32]
```

- La capture Wireshark (premier tableau en gris) montre les différentes couches OSI (Ethernet - IP - IMCP). A voir également, la section "Vue d'ensemble d'un paquet ICMP".
- The TTL value of the echo_response packet is 251, this is also the TTL value displayed in the Ping tool.
- Le paquet a été reconnu comme ICMP dans la couche IP.
- La couche ICMP est composée par une section en-tête (header) et une section donnée (data).
- La section en-tête ICMP est composée par: le type, code, checksum, identifiant (identifier) et numéro de séquence (sequence number).
- Le type ICMP a une valeur de 0 ce qui signifie que le paquet est un paquet echo_response.
- La valeur du numéro de séquence (sequence number), ici 256, est utilisée pour faire correspondre les echo_requests aux réponses associées (echo_response).

Haut de la Page

ARGUMENTS PING

Le fonctionnement de Ping est le même sous Windows ou Linux. Toutefois, il existe quelques petites différences dans l'utilisation et la présentation de la commande Ping.

Utilisation par défaut:

- Windows envoie quatre requêtes ICMP quand Linux envoie ces mêmes paquets indéfiniment. La commande "Cltr + C" interrompt l'envoie de paquet echo_request.
- Le champ data (donnée) d'un paquet ICMP est de 56 bytes sous Linux et 32 sous Windows. Cela signifie que la longueur totale du paquet ICMP, incluant les en-têtes de 42 bytes, est de 98 bytes sous Linux et 74 sous Windows.

Arguments:

Certains arguments peuvent être différents entre Linux et Windows. Par exemple, l'option "-l" est utilisée pour configurer la taille de paquet sous Windows tandis que sous Linux l'argument "-s" est utilisé pour la même fonction.
Voir ci-dessous pour des détails complets à propos des arguments Windows et Linux.

Résultats:

Linux affiche une autre valeur dans le résultat de la commande Ping qui est la déviation moyenne (mdev). Celle-ci est calculée avec les valeurs des temps de réponse. La déviation moyenne donne une indication à propos de la constance du temps de réponse. Autrement dit, une déviation moyenne basse signifiera que les valeurs des temps de réponse fournies par le Ping sont très similaires.

 Arguments Ping sous Windows

c:/ping -h

```
Usage:                  ping [-t] [-a] [-n count] [-l size] [-f] [-i TTL] [-v TOS]
                          [-r count] [-s count] [[-j host-list] | [-k host-list]]
                        [-w timeout] [-R] [-S srcaddr] [-4] [-6] target_name
Options:
-t                      Ping the specified host until stopped.
                        To see statistics and continue - type Control-Break;
                        To stop - type Control-C.
-a                      Resolve addresses to hostnames
-n count                Number of echo requests to send.
-l size                 Send buffer size.
-f                      Set Don't Fragment flag in packet (IPv4-only).
-i TTL                  Time To Live.
-v TOS                  Type Of Service (IPv4-only).
-r count                Record route for count hops (IPv4-only).
-s count                Timestamp for count hops (IPv4-only).
-j host-list            Loose source route along host-list (IPv4-only).
-k host-list            Strict source route along host-list (IPv4-only).
-w timeout              Timeout in milliseconds to wait for each reply.
-R                      Trace round-trip path (IPv6-only).
-S srcaddr              Source address to use (IPv6-only).
-4                      Force using IPv4.
-6                      Force using IPv6.
```

 Arguments Ping sous Linux

#man ping

```
NAME
        ping, ping6 - send ICMP ECHO_REQUEST to network hosts

SYNOPSIS
        ping [-LRUbdfnqrvVaAB] [-c count] [-i interval] [-l preload] [-p pattern] [-s packetsize] [-t
        ttl] [-w deadline] [-F flowlabel] [-I interface] [-M hint] [-Q tos] [-S sndbuf] [-T timestamp
        option] [-W timeout] [hop ...] destination

DESCRIPTION
        ping uses the ICMP protocolâs mandatory ECHO_REQUEST datagram to elicit an ICMP
        ECHO_RESPONSE from a host or gateway.  ECHO_REQUEST datagrams (âpingsâ) have
        an  IP and ICMP header, followed by a struct timeval and then an arbitrary number of
        âpadâ bytes used to fill out the packet

OPTIONS
```

-a Audible ping.

-A Adaptive ping. Interpacket interval adapts to round-trip time, so that effectively not more than one (or more, if preload is set) unanswered probes present in the network. Minimal interval is 200msec for not super-user. On networks with low rtt this mode is essentially equivalent to flood mode.

-b Allow pinging a broadcast address.

-B Do not allow ping to change source address of probes. The address is bound to one selected when ping starts.

-c count
Stop after sending count ECHO_REQUEST packets. With deadline option, ping waits for count ECHO_REPLY packets, until the timeout expires.

-d Set the SO_DEBUG option on the socket being used. Essentially, this socket option is not used by Linux kernel.

-F flow label
Allocate and set 20 bit flow label on echo request packets. (Only ping6). If value is zero, kernel allocates random flow label.

-f Flood ping. For every ECHO_REQUEST sent a period ââ.ââ is printed, while for ever ECHO_REPLY received a backspace is printed. This provides a rapid display of how many packets are being dropped. If interval is not given, it sets interval to zero and outputs packets as fast as they come back or one hundred times per second, whichever is more. Only the super-user may use this option with zero interval.

-i interval
Wait interval seconds between sending each packet. The default is to wait for one second between each packet normally, or not to wait in flood mode. Only super-user may set interval to values less 0.2 seconds.

-I interface address
Set source address to specified interface address. Argument may be numeric IP address or name of device. When pinging IPv6 link-local address this option is required.

-l preload
If preload is specified, ping sends that many packets not waiting for reply. Only the super-user may select preload more than 3.

-L
Suppress loopback of multicast packets. This flag only applies if the ping destination is a multicast address.

-n
Numeric output only. No attempt will be made to lookup symbolic names for host addresses.

-p pattern
You may specify up to 16 ââpadââ bytes to fill out the packet you send. This is useful for diagnosing data-dependent problems in a network. For example, -p ff will cause the sent packet to be filled with all ones.

-Q tos
Set Quality of Service -related bits in ICMP datagrams. tos can be either decimal or hex number. Traditionally (RFC1349), these have been interpreted as: 0 for reserved (currently being redefined as congestion control), 1-4 for Type of Service and 5-7 for Precedence. Possible settings for Type of Service are: minimal cost: 0x02, reliability: 0x04, throughput: 0x08, low delay: 0x10. Multiple TOS bits should not be set simultaneously. Possible settings for special Precedence range from priority (0x20) to net control (0xe0). You must be root (CAP_NET_ADMIN capability) to use Critical or higher precedence value. You cannot set bit 0x01 (reserved) unless ECN has been enabled in the kernel. In RFC2474, these fields has been redefined as 8-bit Differentiated Services (DS), consisting of: bits 0-1 of separate data (ECN will be used, here), and bits 2-7 of Differentiated Services Codepoint (DSCP).

-q
Quiet output. Nothing is displayed except the summary lines at startup time and when finished.

-R
Record route. Includes the RECORD_ROUTE option in the ECHO_REQUEST packet and displays the route buffer on returned packets. Note that the IP header is only large enough for nine such routes. Many hosts ignore or discard this option.

-r
Bypass the normal routing tables and send directly to a host on an attached interface. If the host is not on a directly-attached network, an error is returned. This option can be used to ping a local host through an interface that has no route through it provided the option -I is also used.

-s packetsize
Specifies the number of data bytes to be sent. The default is 56, which translates into 64 ICMP data bytes when combined with the 8 bytes of ICMP header data.

-S sndbuf
Set socket sndbuf. If not specified, it is selected to buffer not more than one packet.

-t ttl
Set the IP Time to Live.

-T timestamp option
Set special IP timestamp options. timestamp option may be either tsonly (only timestamps), tsandaddr (timestamps and addresses) or tsprespec host1 [host2[host3

[host4]]] (timestamp prespecified hops).

-M hint
Select Path MTU Discovery strategy. hint may be either do (prohibit fragmentation, even local one), want (do PMTU discovery, fragment locally when packet size is large), or dont (do not set DF flag).

-U Print full user-to-user latency (the old behaviour). Normally ping prints network round trip time, which can be different f.e. due to DNS failures.

-v Verbose output.

-V Show version and exit.

-w deadline
Specify a timeout, in seconds, before ping exits regardless of how many packets have been sent or received. In this case ping does not stop after count packet are sent, it waits either for deadline expire or until count probes are answered or for some error notification from network.

-W timeout
Time to wait for a response, in seconds. The option affects only timeout in absense of any responses, otherwise ping waits for two RTTs.

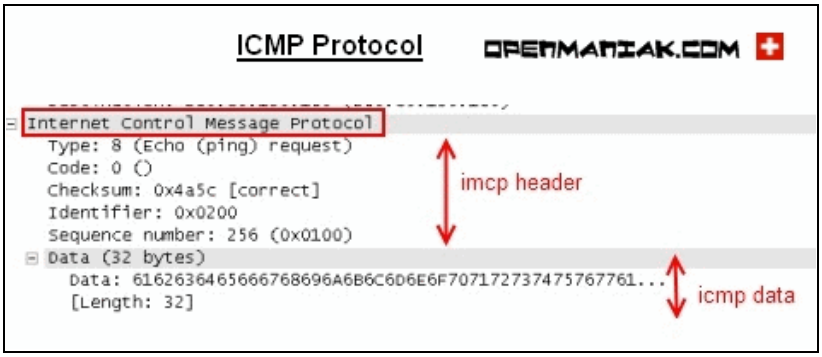
Haut de la Page

EN-TÊTE ICMP (HEADER)

L'en-tête ICMP est composé par:

- Le "Type"
- Le "Code"
- Le "Header checksum"
- L' "ID"
- La "Sequence"

Ci-dessous, la structure du protocole ICMP dans un paquet.



Ci-dessous, quelques types de types ICMP.

Type	Description
0	Echo reply
3	Destination unreachable
4	Source quench
5	Redirect
8	Echo request
9	Router advertisement
10	Router solicitation
11	Time exceeded
12	Parameter problem
13	Timestamp request
14	Timestamp reply
15	Information request
16	Information reply
17	Address mask request
18	Address mask reply
30	Traceroute

Haut de la Page

ADRESSE IP and NOM D'HÔTE

Ping est également très fréquemment utilisé pour connaître l'adresse IP correspondant à un nom d'hôte et l'inverse.

➡ Pour connaître l'adresse IP qui correspond au site web d'openmaniak:

ping www.openmaniak.com

```
Envoi d'une requête 'ping' sur www.openmaniak.com [84.16.88.15] avec 32 octets de données :

Délai d'attente de la demande dépassé.
```

```
Délai d'attente de la demande dépassé.
Délai d'attente de la demande dépassé.
Délai d'attente de la demande dépassé.

Ping statistics for 84.16.88.15:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

L'hôte ne répond pas mais dans ce cas le résultat qui nous intéresse est l'adresse IP affichée dans le résultat du Ping.

➡ Si vous avez une adresse IP et vous voulez voir si elle a un nom associé:

ping -a 84.16.88.15

```
Pinging imu138.infomaniak.ch [84.16.88.15] avec 32 octets de données :

Délai d'attente de la demande dépassé.
Délai d'attente de la demande dépassé.
Délai d'attente de la demande dépassé.
Délai d'attente de la demande dépassé.

Ping statistics for 84.16.88.15:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Le nom d'hôte associé à l'adresse IP 84.16.88.15 est imu138.infomaniak.ch.

Haut de la Page
