Présentation des commandes ping et traceroute

Contenu

Introduction

Ce document montre l'utilisation des commandes **ping et traceroute**. À l'aide de quelques commandes de **débogage**, ce document saisit une vue plus détaillée de la façon dont ces commandes fonctionnent.

Remarque: Activer l´une des commandes de **débogage** sur un routeur de production peut poser des sérieux problèmes. Nous vous recommandons de lire soigneusement la section Utilisation de la commande de débogage avant d´émettre des commandes de **débogage**.

Conditions préalables

Conditions requises

Aucune spécification déterminée n'est requise pour ce document.

Composants utilisés

Ce document n'est pas limité à des versions de matériel et de logiciel spécifiques.

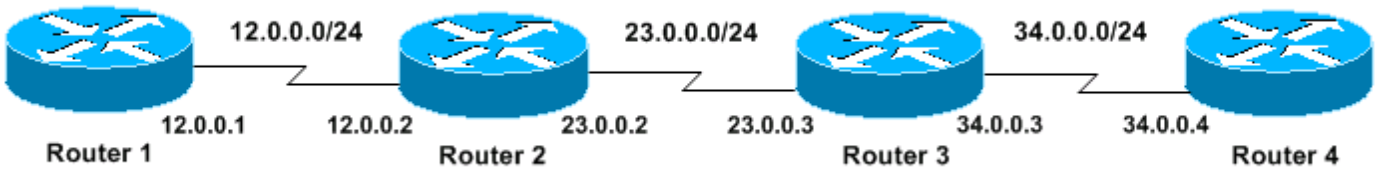
Les informations contenues dans ce document ont été créées à partir des périphériques d'un environnement de laboratoire spécifique. Tous les périphériques utilisés dans ce document ont démarré avec une configuration effacée (par défaut). Si votre réseau est opérationnel, assurez-vous que vous comprenez l'effet potentiel de toute commande.

Conventions

Pour plus d'informations sur les conventions de documents, reportez-vous à Conventions relatives aux conseils techniques Cisco.

Informations générales

Dans ce document, nous utilisons la configuration de base montrée ci-dessous comme base de nos exemples :



La commande ping

La commande **ping** est une méthode très classique pour dépanner l'accessibilité des périphériques. Elle emploie une série de messages d'écho d'Internet Control Message Protocol (ICMP) pour déterminer :

- Si un hôte distant est actif ou inactif.
- Le délai d'aller-retour en communiquant avec l´hôte.
- La perte de paquets.

La commande **Ping** envoie d'abord un paquet de demande d'écho à l'adresse, puis attend une réponse. Le ping est réussi seulement si :

- la demande d'écho arrive à la destination, et
- la destination peut renvoyer une réponse d'écho à la source dans un délai prédéterminé intitulé un délai d´attente. La valeur par défaut de ce délai d´attente est de deux secondes sur les routeurs Cisco.

Pour toutes les options relatives à cette commande, voir « Ping » sous Commandes de dépannage.

La valeur de TTL d'un **paquet ping** ne peut pas être changée.

Voici un exemple de résultat montrant la commande **ping** après activation de la commande **debug ip packet detail** :

Avertissement : L´utilisation de la commande **debug ip packet detail** sur un routeur de production

peut entraîner une utilisation élevée du CPU. Ceci peut avoir comme conséquence une grave dégradation des performances ou une panne du réseau. Nous vous recommandons de lire soigneusement Utilisation de la commande de débogage avant d´émettre des commandes de **débogage**.

```
Router1#debug ip packet detail
IP packet debugging is on (detailed)

Router1#ping 12.0.0.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 12.0.0.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/6/8 ms

Router1#
Jan 20 15:54:47.487: IP: s=12.0.0.1 (local), d=12.0.0.2 (Serial0), len 100,
  sending
Jan 20 15:54:47.491: ICMP type=8, code=0
!--- This is the ICMP packet 12.0.0.1 sent to 12.0.0.2. !--- ICMP type=8 corresponds to the echo
message. Jan 20 15:54:47.523: IP: s=12.0.0.2 (Serial0), d=12.0.0.1 (Serial0), len 100, rcvd 3
Jan 20 15:54:47.527: ICMP type=0, code=0
!--- This is the answer we get from 12.0.0.2. !--- ICMP type=0 corresponds to the echo reply
message. !--- By default, the repeat count is five times, so there will be five !--- echo
requests, and five echo replies.
```

Le tableau ci-dessous liste les valeurs possibles de type ICMP.

Type ICMP	Littéral
0	réponse d'écho
3	le code inaccessible 0 de destination = le net 1 inaccessible = l'hôte 2 inaccessibles = le protocole 3 inaccessibles = le port 4 inaccessibles = fragmentation a eu besoin et le DF placer 5 = artère de source a manqué
4	source-éteignez
5	réorientez le code 0 = réorientent des datagrammes pour le réseau 1 = réorientent des datagrammes pour l'hôte 2 = réorientent des datagrammes pour le type de service et le réseau 3 = réorientent des datagrammes pour le type du service et d'hôte
6	adresse alternative
8	écho
9	routeur-annonce
10	routeur-sollicitation
11	le code temps-dépassé 0 = Time to Live a dépassé en transit 1 = temps de réassemblage de fragments dépassé
1	Problème de paramètre

2	
13	demande d´horodatage
14	réponse d'horodatage
15	demande d'informations
16	réponse à la demande d'informations
17	demande de masque
18	réponse à la demande de masque
31	erreur de conversion
32	redirection de mobile

Le tableau ci-dessous liste les caractères de sortie possibles du service Ping :

Caractère	Description
!	Chaque point d'exclamation indique la réception d'une réponse.
.	Chaque période indique le serveur réseau qui a dépassé le délai d´attente en attendant une réponse.
U	Une erreur PDU de destination inaccessible a été reçue.
Q	Épuisement de la source (destination trop occupée).
M	N'a pas pu fragmenter.
?	Type de paquet inconnu.
&	Durée de vie du paquet dépassée.

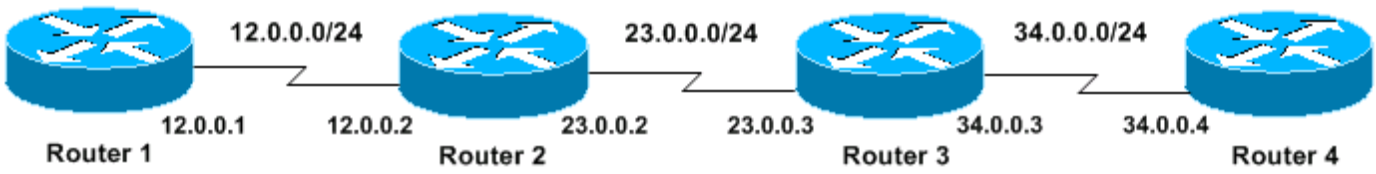
Pourquoi est-il impossible d´envoyer un ping ?

Si vous ne pouvez pas envoyer un ping avec succès à l'adresse, considérez ces causes :

Problème de routage

Voici des exemples de tentatives infructueuses de ping, de détermination du problème et ce qu´il faut faire pour résoudre le problème.

Ce scénario est expliqué à l´aide du diagramme de la topologie du réseau ci-dessous :



Router1#

```
!
!
interface Serial0
ip address 12.0.0.1 255.255.255.0
no fair-queue
clockrate 64000
!
!
```

Router2#

```
!
!
interface Serial0
ip address 23.0.0.2 255.255.255.0
no fair-queue
clockrate 64000
!
interface Serial1
ip address 12.0.0.2 255.255.255.0
!
!
```

Router3#

```
!
!
interface Serial0
ip address 34.0.0.3 255.255.255.0
no fair-queue
!
interface Serial1
ip address 23.0.0.3 255.255.255.0
!
!
```

Router4#

```
!
!
interface Serial0
ip address 34.0.0.4 255.255.255.0
no fair-queue
clockrate 64000
!
!
```

Essayons d´envoyer un ping de Router1 à Router4 :

```
Router1#ping 34.0.0.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 34.0.0.4, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
```

Allons voir plus de plus près ce qui s'est produit :

```
Router1#debug ip packet
IP packet debugging is on
```

Avertissement : L´utilisation de la commande **debug ip packet** sur un routeur de production peut entraîner une utilisation élevée du CPU. Ceci peut avoir comme conséquence une grave dégradation des performances ou une panne du réseau. Nous vous recommandons de lire soigneusement Utilisation de la commande de débogage avant d´émettre des commandes de **débogage**.

```
Router1#ping 34.0.0.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 34.0.0.4, timeout is 2 seconds:

Jan 20 16:00:25.603: IP: s=12.0.0.1 (local), d=34.0.0.4, len 100, unroutable.
Jan 20 16:00:27.599: IP: s=12.0.0.1 (local), d=34.0.0.4, len 100, unroutable.
Jan 20 16:00:29.599: IP: s=12.0.0.1 (local), d=34.0.0.4, len 100, unroutable.
Jan 20 16:00:31.599: IP: s=12.0.0.1 (local), d=34.0.0.4, len 100, unroutable.
Jan 20 16:00:33.599: IP: s=12.0.0.1 (local), d=34.0.0.4, len 100, unroutable.
Success rate is 0 percent (0/5)
```

Puisqu´aucun protocole de routage ne fonctionne sur Router1, il ne sait pas où envoyer son paquet et nous recevons un message « unroutable ».

Ajoutons maintenant une route statique à Router1 :

```
Router1#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router1(config)#ip route 0.0.0.0 0.0.0.0 Serial0
```

Nous avons maintenant :

```
Router1#debug ip packet detail
IP packet debugging is on (detailed)

Router1#ping 34.0.0.4
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 34.0.0.4, timeout is 2 seconds:
U.U.U
Success rate is 0 percent (0/5)

Jan 20 16:05:30.659: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 100,
  sending
Jan 20 16:05:30.663:     ICMP type=8, code=0
Jan 20 16:05:30.691: IP: s=12.0.0.2 (Serial0), d=12.0.0.1 (Serial0), len 56,
  rcvd 3
Jan 20 16:05:30.695:     ICMP type=3, code=1
Jan 20 16:05:30.699: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 100,
  sending
Jan 20 16:05:30.703:     ICMP type=8, code=0
Jan 20 16:05:32.699: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 100,
  sending
Jan 20 16:05:32.703:     ICMP type=8, code=0
Jan 20 16:05:32.731: IP: s=12.0.0.2 (Serial0), d=12.0.0.1 (Serial0), len 56,
  rcvd 3
Jan 20 16:05:32.735:     ICMP type=3, code=1
Jan 20 16:05:32.739: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 100,
  sending
Jan 20 16:05:32.743:     ICMP type=8, code=0
```

Examinons maintenant ce qui est erroné sur Router2 :

```
Router2#debug ip packet detail
IP packet debugging is on (detailed)

Router2#
Jan 20 16:10:41.907: IP: s=12.0.0.1 (Serial1), d=34.0.0.4, len 100, unroutable
Jan 20 16:10:41.911:     ICMP type=8, code=0
Jan 20 16:10:41.915: IP: s=12.0.0.2 (local), d=12.0.0.1 (Serial1), len 56, sending
Jan 20 16:10:41.919:     ICMP type=3, code=1
Jan 20 16:10:41.947: IP: s=12.0.0.1 (Serial1), d=34.0.0.4, len 100, unroutable
Jan 20 16:10:41.951:     ICMP type=8, code=0
Jan 20 16:10:43.943: IP: s=12.0.0.1 (Serial1), d=34.0.0.4, len 100, unroutable
Jan 20 16:10:43.947:     ICMP type=8, code=0
Jan 20 16:10:43.951: IP: s=12.0.0.2 (local), d=12.0.0.1 (Serial1), len 56, sending
Jan 20 16:10:43.955:     ICMP type=3, code=1
Jan 20 16:10:43.983: IP: s=12.0.0.1 (Serial1), d=34.0.0.4, len 100, unroutable
Jan 20 16:10:43.987:     ICMP type=8, code=0
Jan 20 16:10:45.979: IP: s=12.0.0.1 (Serial1), d=34.0.0.4, len 100, unroutable
Jan 20 16:10:45.983:     ICMP type=8, code=0
Jan 20 16:10:45.987: IP: s=12.0.0.2 (local), d=12.0.0.1 (Serial1), len 56, sending
Jan 20 16:10:45.991:     ICMP type=3, code=1
```

Router1 envoie correctement ses packets à Router2, mais Router2 ne sait pas comment accéder à l´adresse 34.0.0.4. Router2 renvoie un message « ICMP inaccessible » à Router1.

Activons maintenant le Protocole d'informations de routage (RIP) sur Router2 et Router3 :

```
Router2#debug ip packet detail
IP packet debugging is on (detailed)

Router2#
Jan 20 16:10:41.907: IP: s=12.0.0.1 (Serial1), d=34.0.0.4, len 100, unroutable
Jan 20 16:10:41.911:     ICMP type=8, code=0
Jan 20 16:10:41.915: IP: s=12.0.0.2 (local), d=12.0.0.1 (Serial1), len 56, sending
Jan 20 16:10:41.919:     ICMP type=3, code=1
Jan 20 16:10:41.947: IP: s=12.0.0.1 (Serial1), d=34.0.0.4, len 100, unroutable
Jan 20 16:10:41.951:     ICMP type=8, code=0
Jan 20 16:10:43.943: IP: s=12.0.0.1 (Serial1), d=34.0.0.4, len 100, unroutable
Jan 20 16:10:43.947:     ICMP type=8, code=0
Jan 20 16:10:43.951: IP: s=12.0.0.2 (local), d=12.0.0.1 (Serial1), len 56, sending
Jan 20 16:10:43.955:     ICMP type=3, code=1
Jan 20 16:10:43.983: IP: s=12.0.0.1 (Serial1), d=34.0.0.4, len 100, unroutable
Jan 20 16:10:43.987:     ICMP type=8, code=0
Jan 20 16:10:45.979: IP: s=12.0.0.1 (Serial1), d=34.0.0.4, len 100, unroutable
Jan 20 16:10:45.983:     ICMP type=8, code=0
Jan 20 16:10:45.987: IP: s=12.0.0.2 (local), d=12.0.0.1 (Serial1), len 56, sending
Jan 20 16:10:45.991:     ICMP type=3, code=1
```

Nous avons maintenant :

```
Router1#debug ip packet
IP packet debugging is on

Router1#ping 34.0.0.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 34.0.0.4, timeout is 2 seconds:

Jan 20 16:16:13.367: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 100,
 sending.
Jan 20 16:16:15.363: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 100,
 sending.
Jan 20 16:16:17.363: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 100,
 sending.
Jan 20 16:16:19.363: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 100,
```

```
 sending.
Jan 20 16:16:21.363: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 100,
 sending.
Success rate is 0 percent (0/5)
```

C'est légèrement meilleur. Router1 envoie des packets à Router4, mais il ne reçoit aucune réponse de Router4.

Voyons quel peut être le problème sur Router4 :

```
Router4#debug ip packet
IP packet debugging is on

Router4#
Jan 20 16:18:45.903: IP: s=12.0.0.1 (Serial0), d=34.0.0.4 (Serial0), len 100,
  rcvd 3
Jan 20 16:18:45.911: IP: s=34.0.0.4 (local), d=12.0.0.1, len 100, unroutable
Jan 20 16:18:47.903: IP: s=12.0.0.1 (Serial0), d=34.0.0.4 (Serial0), len 100,
  rcvd 3
Jan 20 16:18:47.907: IP: s=34.0.0.4 (local), d=12.0.0.1, len 100, unroutable
Jan 20 16:18:49.903: IP: s=12.0.0.1 (Serial0), d=34.0.0.4 (Serial0), len 100,
  rcvd 3
Jan 20 16:18:49.907: IP: s=34.0.0.4 (local), d=12.0.0.1, len 100, unroutable
Jan 20 16:18:51.903: IP: s=12.0.0.1 (Serial0), d=34.0.0.4 (Serial0), len 100,
  rcvd 3
Jan 20 16:18:51.907: IP: s=34.0.0.4 (local), d=12.0.0.1, len 100, unroutable
Jan 20 16:18:53.903: IP: s=12.0.0.1 (Serial0), d=34.0.0.4 (Serial0), len 100,
  rcvd 3
Jan 20 16:18:53.907: IP: s=34.0.0.4 (local), d=12.0.0.1, len 100, unroutable
```

Router4 reçoit les paquets ICMP, et essaye de répondre à 12.0.0.1, mais parce qu'il n'a pas de route à ce réseau, il échoue simplement.

Ajoutons une route statique à Router4 :

```
Router4(config)#ip route 0.0.0.0 0.0.0.0 Serial0
```

Tout fonctionne maintenant parfaitement, et les deux côtés peuvent s'accéder l´un à l´autre :

```
Router1#ping 34.0.0.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 34.0.0.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/35/36 ms
```

Interface inactive

C'est une situation où l´interface cesse de fonctionner. Dans l'example ci-dessous, nous essayons d´envoyer un ping de Router1 à Router4 :

```
Router1#ping 34.0.0.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 34.0.0.4, timeout is 2 seconds:
U.U.U
Success rate is 0 percent (0/5)
```

Puisque le routage est bon, nous ferons le dépannage point par point. D'abord, essayons d´envoyer un ping à Router2 :

```
Router1#ping 12.0.0.2
```

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 12.0.0.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/4 ms
```

D'après ce qui précède, nous voyons que le problème se trouve entre Router2 et Router3. Une possibilité est que l'interface série sur Router3 a été arrêtée :

```
Router3#show ip interface brief
Serial0   34.0.0.3    YES manual up                    up
Serial1   23.0.0.3    YES manual administratively down   down
```

C'est tout à fait simple à résoudre :

```
Router3#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router3(config)#interface s1
Router3(config-if)#no shutdown
Router3(config-if)#
Jan 20 16:20:53.900: %LINK-3-UPDOWN: Interface Serial1, changed state to up
Jan 20 16:20:53.910: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1,
  changed state to up
```

Commande Access-list

Dans ce scénario, nous voulons laisser seulement le trafic telnet entrer dans Router4 par l´interface Serial0.

```
Router4(config)# access-list 100 permit tcp any any eq telnet
Router4(config)#interface s0
Router4(config-if)#ip access-group 100 in
```

```
Router1#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router1(config)#access-list 100 permit ip host 12.0.0.1 host 34.0.0.4
Router1(config)#access-list 100 permit ip host 34.0.0.4 host 12.0.0.1
Router1(config)#end
Router1#debug ip packet 100
IP packet debugging is on
Router1#debug ip icmp
ICMP packet debugging is on
```

Référez-vous à la section Utilisation de la commande de débogage pour l'usage des listes d'accès avec les commandes **debug**.

Quand nous essayons maintenant d´envoyer un ping à Router4, nous avons ce qui suit :

```
Router1#ping 34.0.0.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 34.0.0.4, timeout is 2 seconds:
U.U.U
Success rate is 0 percent (0/5)

Jan 20 16:34:49.207: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 100,
 sending
Jan 20 16:34:49.287: IP: s=34.0.0.4 (Serial0), d=12.0.0.1 (Serial0), len 56,
 rcvd 3
Jan 20 16:34:49.291: ICMP: dst (12.0.0.1) administratively prohibited unreachable
 rcv from 34.0.0.4
Jan 20 16:34:49.295: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 100,
 sending
Jan 20 16:34:51.295: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 100,
```

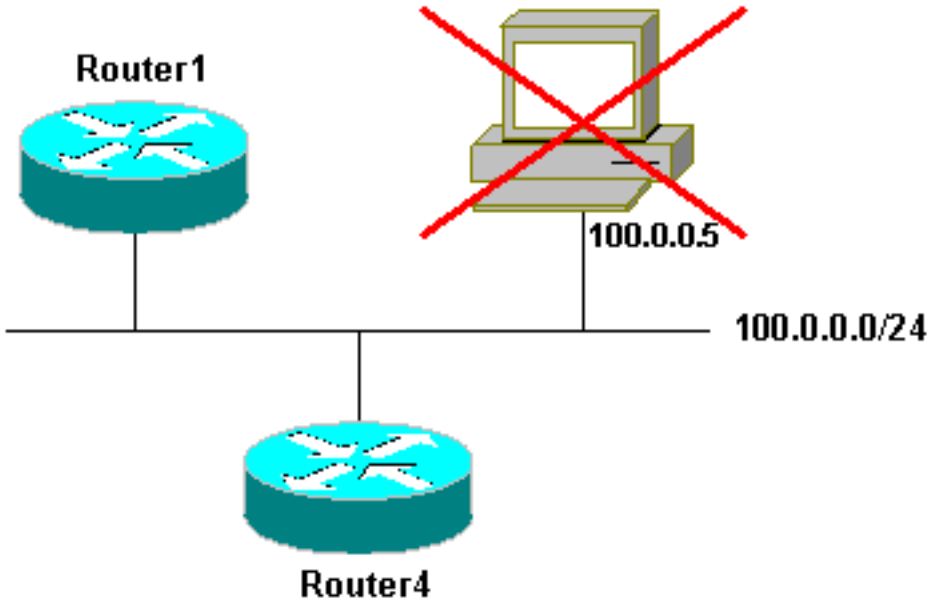
```
 sending
Jan 20 16:34:51.367: IP: s=34.0.0.4 (Serial0), d=12.0.0.1 (Serial0), len 56,
 rcvd 3
Jan 20 16:34:51.371: ICMP: dst (12.0.0.1) administratively prohibited unreachable
 rcv from 34.0.0.4
Jan 20 16:34:51.379: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 100,
 sending
```

At la fin d'une commande **access-list**, nous avons toujours un « refus tout » implicite. Ceci signifie que les paquets ICMP qui entrent dans l´interface Serial 0 sur Router4 sont refusés, et Router4 envoie un message ICMP « inaccessible, administrativement interdit » à la source du paquet original suivant les indications du message de **débogage** . La solution est d'ajouter la ligne suivante à la commande **access-list** :

```
Router4(config)#access-list 100 permit icmp any any
```

Problème de Protocole de résolution d'adresse (ARP)

Voici un scénario avec une connexion Ethernet :




```
Router4#ping 100.0.0.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 100.0.0.5, timeout is 2 seconds:

Jan 20 17:04:05.167: IP: s=100.0.0.4 (local), d=100.0.0.5 (Ethernet0), len 100,
  sending
Jan 20 17:04:05.171: IP: s=100.0.0.4 (local), d=100.0.0.5 (Ethernet0), len 100,
  encapsulation failed.
Jan 20 17:04:07.167: IP: s=100.0.0.4 (local), d=100.0.0.5 (Ethernet0), len 100,
  sending
Jan 20 17:04:07.171: IP: s=100.0.0.4 (local), d=100.0.0.5 (Ethernet0), len 100,
  encapsulation failed.
Jan 20 17:04:09.175: IP: s=100.0.0.4 (local), d=100.0.0.5 (Ethernet0), len 100,
  sending
Jan 20 17:04:09.183: IP: s=100.0.0.4 (local), d=100.0.0.5 (Ethernet0), len 100,
  encapsulation failed.
Jan 20 17:04:11.175: IP: s=100.0.0.4 (local), d=100.0.0.5 (Ethernet0), len 100,
```

```
  sending
Jan 20 17:04:11.179: IP: s=100.0.0.4 (local), d=100.0.0.5 (Ethernet0), len 100,
  encapsulation failed.
Jan 20 17:04:13.175: IP: s=100.0.0.4 (local), d=100.0.0.5 (Ethernet0), len 100,
  sending
Jan 20 17:04:13.179: IP: s=100.0.0.4 (local), d=100.0.0.5 (Ethernet0), len 100,
  encapsulation failed.
Success rate is 0 percent (0/5)
Router4#
```

Dans cet exemple, le ping ne fonctionne pas dû à « encapsulation à échoué ». Ceci signifie que le routeur sait sur quelle interface il doit envoyer le paquet, mais il ne sait pas comment le faire. Dans ce cas, vous devez comprendre comment le Protocole de résolution d'adresse (ARP) fonctionne. Voyez Configuration des méthodes de résolution d'adresse pour une explication détaillée.

Fondamentalement, ARP est un protocole utilisé pour tracer l´adresse de couche 2 (adresse MAC) à une adresse de couche 3 (adresse IP). Vous pouvez contrôler ce mappage au moyen de la commande **show arp** :

```
Router4#show arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  100.0.0.4               -   0000.0c5d.7a0d  ARPA   Ethernet0
Internet  100.0.0.1              10   0060.5cf4.a955  ARPA   Ethernet0
```

Revenez au problème « encapsulation a échoué ». Nous aurons une meilleure idée du problème en utilisant la commande **debug** :

```
Router4#debug arp
ARP packet debugging is on

Router4#ping 100.0.0.5

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 100.0.0.5, timeout is 2 seconds:

Jan 20 17:19:43.843: IP ARP: creating incomplete entry for IP address: 100.0.0.5
  interface Ethernet0
Jan 20 17:19:43.847: IP ARP: sent req src 100.0.0.4 0000.0c5d.7a0d,
                 dst 100.0.0.5 0000.0000.0000 Ethernet0.
Jan 20 17:19:45.843: IP ARP: sent req src 100.0.0.4 0000.0c5d.7a0d,
                 dst 100.0.0.5 0000.0000.0000 Ethernet0.
Jan 20 17:19:47.843: IP ARP: sent req src 100.0.0.4 0000.0c5d.7a0d,
                 dst 100.0.0.5 0000.0000.0000 Ethernet0.
Jan 20 17:19:49.843: IP ARP: sent req src 100.0.0.4 0000.0c5d.7a0d,
                 dst 100.0.0.5 0000.0000.0000 Ethernet0.
Jan 20 17:19:51.843: IP ARP: sent req src 100.0.0.4 0000.0c5d.7a0d,
                 dst 100.0.0.5 0000.0000.0000 Ethernet0.
Success rate is 0 percent (0/5)
```

La sortie ci-dessus montre que Router4 diffuse les packets en les envoyant à l'adresse de diffusion Ethernet FFFF.FFFF.FFFF. Ici, 0000.0000.0000 signifie que Router4 recherche l'adresse MAC de la destination 100.0.0.5. Puisqu'il ne connaît pas l´adresse MAC pendant la requête ARP de cet exemple, il utilise 0000.0000.000 comme espace réservé dans les trames de diffusion envoyées des interfaces Ethernet 0, demandant quelle adresse MAC correspond à 100.0.0.5. Si nous n'obtenons pas de réponse, l'adresse correspondante dans les outputis de show arp **marqués** en tant qu'inachevé :

```
Router4#show arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  100.0.0.4               -   0000.0c5d.7a0d  ARPA   Ethernet0
Internet  100.0.0.5               0   Incomplete      ARPA
```

```
Internet  100.0.0.1              2   0060.5cf4.a955  ARPA   Ethernet0
```

Après une période prédéterminée, cette entrée incomplète est purgée du tableau ARP. Tant que l´adresse MAC correspondante n'est pas dans le tableau ARP, le ping échoue en raison de « encapsulation a échoué ».

Retard

Par défaut, si vous ne recevez pas de réponse du dispositif distant dans deux secondes, le ping échoue :

```
Router1#ping 12.0.0.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 12.0.0.2, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
```

Sur des réseaux avec une liaison lente ou un long délai d'attente, deux secondes ne sont pas suffisantes. Vous pouvez changer ce défaut en utilisant un ping étendu :

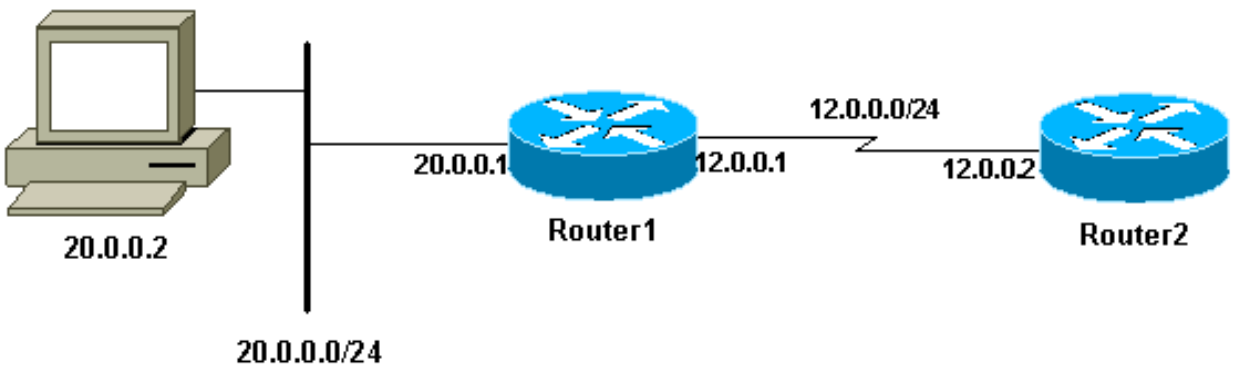
```
Router1#ping
Protocol [ip]:
Target IP address:  12.0.0.2
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]: 30
Extended commands [n]:
Sweep range of sizes [n]:

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 12.0.0.2, timeout is 30 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1458/2390/6066 ms
```

Dans l´exemple ci-dessus, l'augmentation du délai d'attente a permis un ping réussi.

Remarque: Le temps moyen de l´aller-retour est plus que deux secondes.

Adresse source correcte

Voici un exemple d'une situation typique :



Nous ajoutons une interface de LAN sur Router1 :

```
Router1(config)#interface e0
```

```
Router1(config-if)#ip address
Router1(config-if)#ip address 20.0.0.1 255.255.255.0
```

Depuis une station sur le LAN, vous pouvez envoyez un ping au Router1. Depuis Router1 vous pouvez envoyez un ping au Router2. Mais d'une station sur le LAN, vous ne pouvez pas envoyez un ping au Router2.

Du Router1, vous pouvez envoyez un ping au Router2 parce que, par défaut, vous utilisez l'adresse IP de l´interface sortante comme adresse source dans votre paquet ICMP. Router2 n'a aucune information sur ce nouveau LAN. S'il doit répondre à un paquet venant de ce réseau, il ne sait pas le traiter.

```
Router1#debug ip packet
IP packet debugging is on
```

Avertissement : L´utilisation de la commande **debug ip packet** sur un routeur de production peut entraîner une utilisation élevée du CPU. Ceci peut avoir comme conséquence une grave dégradation des performances ou une panne du réseau. Nous vous recommandons de lire soigneusement Utilisation de la commande de débogage avant d´émettre des commandes de **débogage**.

```
Router1#ping 12.0.0.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 12.0.0.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/7/9 ms
Router1#

Jan 20 16:35:54.227: IP: s=12.0.0.1 (local), d=12.0.0.2 (Serial0), len 100, sending
Jan 20 16:35:54.259: IP: s=12.0.0.2 (Serial0), d=12.0.0.1 (Serial0), len 100, rcvd 3
```

L´exemple de sortie ci-dessus fonctionne parce que l´adresse de la source du paquet que nous envoyons est s = 12.0.0.1. Si nous voulons simuler un paquet venant du LAN, nous devons utiliser un ping étendu :

```
Router1#ping
Protocol [ip]:
Target IP address: 12.0.0.2
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface: 20.0.0.1
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 12.0.0.2, timeout is 2 seconds:

Jan 20 16:40:18.303: IP: s=20.0.0.1 (local), d=12.0.0.2 (Serial0), len 100,
 sending.
Jan 20 16:40:20.303: IP: s=20.0.0.1 (local), d=12.0.0.2 (Serial0), len 100,
 sending.
Jan 20 16:40:22.303: IP: s=20.0.0.1 (local), d=12.0.0.2 (Serial0), len 100,
 sending.
Jan 20 16:40:24.303: IP: s=20.0.0.1 (local), d=12.0.0.2 (Serial0), len 100,
 sending
Jan 20 16:40:26.303: IP: s=20.0.0.1 (local), d=12.0.0.2 (Serial0), len 100,
```

```
 sending.
Success rate is 0 percent (0/5)
```

Cette fois-ci, l´adresse de la source est 20.0.0.1, et cela ne fonctionne pas ! Nous envoyons nos paquets, mais nous ne recevons rien. Pour résoudre ce problème, nous devons simplement ajouter une route à 20.0.0.0 dans Router2.

La règle de base est que le périphérique recevant le ping doit également savoir comment envoyer la réponse à la source du ping.

Pertes de file d'attente d'entrée élevées

Quand un paquet entre dans le routeur, celui-ci essaye de le transférer au niveau de priorité d'interruption. Si une correspondance ne peut pas être trouvée dans un tableau de cache approprié, le paquet est aligné dans la file d'attente de l'interface entrante à traiter. Certains packets sont toujours traités, mais avec la configuration appropriée et dans des réseaux stables, le débit des paquets traités ne doit jamais congestionner la file d'attente d'entrée. Si la file d'attente d'entrée est pleine, le paquet est perdu.

Bien que l´interface soit active et que vous puissiez ne pas envoyer un ping au périphérique dû aux pertes élevées de la file d'attente d'entrée. Vous pouvez vérifier les pertes d'entrée à l´aide de la commande **show interface**.

```
Router1#show interface Serial0/0/0

Serial0/0/0 is up, line protocol is up

  MTU 1500 bytes, BW 1984 Kbit, DLY 20000 usec,
     reliability 255/255, txload 69/255, rxload 43/255
  Encapsulation HDLC, loopback not set
  Keepalive set (10 sec)
  Last input 00:00:02, output 00:00:00, output hang never
  Last clearing of "show interface" counters 01:28:49
  Input queue: 76/75/5553/0 (size/max/drops/flushes);
     Total output drops: 1760
  Queueing strategy: Class-based queueing
  Output queue: 29/1000/64/1760 (size/max total/threshold/drops)
     Conversations  7/129/256 (active/max active/max total)
     Reserved Conversations 4/4 (allocated/max allocated)
     Available Bandwidth 1289 kilobits/sec

!--- Output supressed
```

Comme vu pour la sortie, la perte de la file d'attente d'entrée est élevée. Référez-vous au Dépannage des pertes de file d'attente d´entrée et des pertes de file d'attente de sortie afin de dépanner les pertes de file d'attente d'entrée/sortie.

La commande traceroute

La commande **traceroute** est utilisée pour découvrir les routes qui prennent réellement les paquets en voyageant à leur destination. Le périphérique (par exemple, un routeur ou un PC) envoie une séquence de datagrammes du Protocole de datagramme utilisateur (UDP) à une adresse de port non valide à l'hôte distant.

Trois datagrammes sont envoyés, chacun avec une valeur du champ Time to Live (TTL) égale à un. La valeur de 1 du TTL provoque un « délai expiré » du datagramme dès qu'il atteint le premier

routeur dans le chemin ; ce routeur répond alors avec un message de délai expiré (TEM) de l´ICMP indiquant que le datagramme a expiré.

Trois autres messages UDP sont maintenant envoyés, chacun avec la valeur de TTL mise à 2, ce qui cause le deuxième routeur de renvoyer des TEM de l´ICMP. Ce processus continue jusqu'à ce que les paquets atteignent réellement l'autre destination. Puisque ces datagrammes essayent d'accéder à un port incorrect à l´hôte de destination, les messages ICMP d'inaccessibilité du port sont renvoyés, indiquant un port inaccessible ; cet événement signale au programme Traceroute qu'il est terminé.

Le but derrière ceci est d'enregistrer la source de chaque message ICMP de temps dépassé afin de fournir une trace du chemin que le paquet a pris pour atteindre la destination. Pour toutes les options au sujet de cette commande, voyez **Trace (privilégiée)**.

```
Router1#traceroute 34.0.0.4

Type escape sequence to abort.
Tracing the route to 34.0.0.4

  1 12.0.0.2 4 msec 4 msec 4 msec
  2 23.0.0.3 20 msec 16 msec 16 msec
  3 34.0.0.4 16 msec *  16 msec

Jan 20 16:42:48.611: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 28,
 sending
Jan 20 16:42:48.615:     UDP src=39911, dst=33434
Jan 20 16:42:48.635: IP: s=12.0.0.2 (Serial0), d=12.0.0.1 (Serial0), len 56,
 rcvd 3
Jan 20 16:42:48.639:     ICMP type=11, code=0
!--- ICMP Time Exceeded Message from Router2. Jan 20 16:42:48.643: IP: s=12.0.0.1 (local),
d=34.0.0.4 (Serial0), len 28, sending Jan 20 16:42:48.647: UDP src=34237, dst=33435 Jan 20
16:42:48.667: IP: s=12.0.0.2 (Serial0), d=12.0.0.1 (Serial0), len 56, rcvd 3 Jan 20
16:42:48.671: ICMP type=11, code=0 Jan 20 16:42:48.675: IP: s=12.0.0.1 (local), d=34.0.0.4
(Serial0), len 28, sending Jan 20 16:42:48.679: UDP src=33420, dst=33436 Jan 20 16:42:48.699:
IP: s=12.0.0.2 (Serial0), d=12.0.0.1 (Serial0), len 56, rcvd 3 Jan 20 16:42:48.703: ICMP
type=11, code=0
```

C'est la première séquence de paquets que nous envoyons avec un TTL=1. Le premier routeur, dans ce cas Router2 (12.0.0.2), perd le paquet, et renvoie à la source (12.0.0.1) un message ICMP de type=11. Ceci correspond au message de temps dépassé.

```
Jan 20 16:42:48.707: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 28,
 sending
Jan 20 16:42:48.711:     UDP src=35734, dst=33437
Jan 20 16:42:48.743: IP: s=23.0.0.3 (Serial0), d=12.0.0.1 (Serial0), len 56,
 rcvd 3
Jan 20 16:42:48.747:     ICMP type=11, code=0
!--- ICMP Time Exceeded Message from Router3. Jan 20 16:42:48.751: IP: s=12.0.0.1 (local),
d=34.0.0.4 (Serial0), len 28, sending Jan 20 16:42:48.755: UDP src=36753, dst=33438 Jan 20
16:42:48.787: IP: s=23.0.0.3 (Serial0), d=12.0.0.1 (Serial0), len 56, rcvd 3 Jan 20
16:42:48.791: ICMP type=11, code=0 Jan 20 16:42:48.795: IP: s=12.0.0.1 (local), d=34.0.0.4
(Serial0), len 28, sending Jan 20 16:42:48.799: UDP src=36561, dst=33439 Jan 20 16:42:48.827:
IP: s=23.0.0.3 (Serial0), d=12.0.0.1 (Serial0), len 56, rcvd 3 Jan 20 16:42:48.831: ICMP
type=11, code=0
```

Le même processus se produit pour Router3 (23.0.0.3) avec un TTL=2 :

```
Jan 20 16:42:48.839: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 28,
 sending
Jan 20 16:42:48.843:     UDP src=34327, dst=33440
Jan 20 16:42:48.887: IP: s=34.0.0.4 (Serial0), d=12.0.0.1 (Serial0), len 56,
```

```
 rcvd 3
Jan 20 16:42:48.891:     ICMP type=3, code=3
!--- Port Unreachable message from Router4. Jan 20 16:42:48.895: IP: s=12.0.0.1 (local),
d=34.0.0.4 (Serial0), len 28, sending Jan 20 16:42:48.899: UDP src=37534, dst=33441 Jan 20
16:42:51.895: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 28, sending Jan 20 16:42:51.899:
UDP src=37181, dst=33442 Jan 20 16:42:51.943: IP: s=34.0.0.4 (Serial0), d=12.0.0.1 (Serial0),
len 56, rcvd 3 Jan 20 16:42:51.947: ICMP type=3, code=3
```

Avec un TTL=3, nous atteignons finalement Router4. Cette fois, puisque le port est incorrect, Router4 renvoie à Router1 un message ICMP avec type=3, un message d'inaccessibilité de destination, et le code=3 qui signifie port inaccessible.

Le tableau ci-dessous liste les caractères qui peuvent apparaître dans la sortie de la commande **traceroute**.

IP traceroute caractères des textes

Caractère	Description
nn millisecondes	Pour chaque noeud, le temps d´aller-retour en millisecondes pour le nombre spécifié de sondes
*	Le temps de la sonde est dépassé
A	Administrativement interdit (exemple, liste d'accès)
Q	Épuisement de la source (destination trop occupée)
I	Test interrompu par l´utilisateur
U	Port inaccessible
H	Inaccessible d'hôte
N	Réseau inaccessible
P	Protocol inaccessible
T	Délai d'attente
?	Type de paquet inconnu.

Représentation

Au moyen des commandes **ping et traceroute**, nous obtenons le temps d´aller-retour (RTT). C'est le temps requis pour envoyer un paquet d'écho, et recevoir une réponse. Ceci peut être utile pour avoir une idée approximative des délais sur la liaison. Cependant, ces chiffres ne sont pas assez précis pour être utilisés pour l'évaluation des performances.

Quand la destination d´un paquet est le routeur lui-même, ce paquet doit être commuté par processus. Le processeur doit traiter les informations de ce paquet, et renvoyer une réponse. Ce n'est pas l'objectif principal d'un routeur. Par définition, un routeur est construit pour router des paquets. Répondre à un ping est offert en tant que service minimal.

Pour illustrer ceci, voici l´exemple d'un ping de Router1 à Router2 :

```
Router1#ping 12.0.0.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 12.0.0.2, timeout is 2 seconds:
!!!!!
```

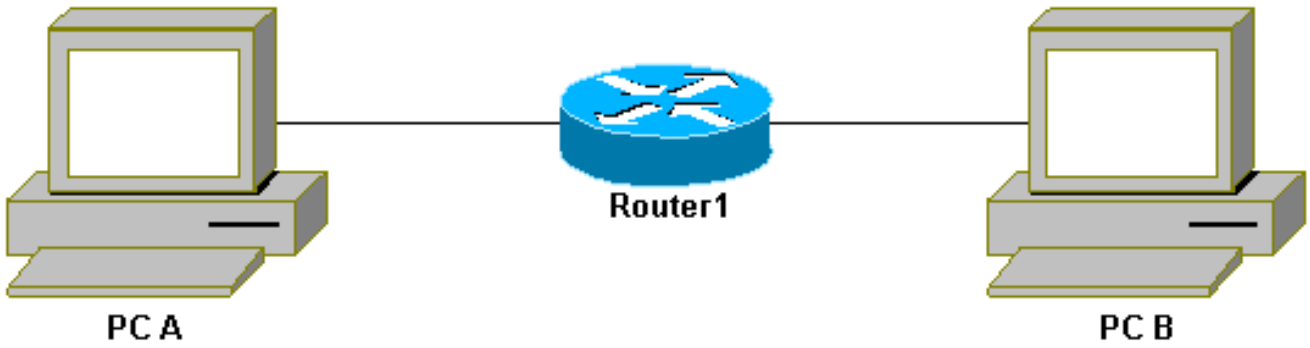
```
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/4 ms
```

Le RTT est approximativement quatre millisecondes. Après avoir activé certaines fonctionnalités à processus intensifs sur Router2, essayez d´envoyer un ping de Router1 à Router2.

```
Router1#ping 12.0.0.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 12.0.0.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 24/25/28 ms
```

Le RTT a considérablement augmenté ici. Router2 est passablement occupé, et la réponse au ping n'est pas sa principale priorité.

Une meilleure façon de tester les performances du routeur est avec le trafic passant **à travers** le routeur :



Le trafic est alors à commutation rapide, et il est traité par le routeur avec une priorité plus élevée. Pour illustrer ceci, retournons à notre réseau de base :

Envoyons un ping de Router1 à Router3 :

```
Router1#ping 23.0.0.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 23.0.0.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/32/32 ms
```

Le trafic passe par le Router2, et il est maintenant à commutation rapide.

Activons-maintenant la fonctionnalité à processus intensifs sur le Router2 :

```
Router1#ping 23.0.0.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 23.0.0.3, timeout is 2 seconds:
```

```
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/32/36 ms
```

Il n'y a presque aucune différence. C'est parce que, sur Router2, les packets sont maintenant traités au niveau de priorité d'interruption.

Utilisez la commande Debug

Avant d'exécuter les commandes **debug**, référez-vous à la section **Informations importantes sur les commandes Debug**.

Les différentes commandes **debug** que nous les avons utilisées jusqu'ici nous donne une vue de ce qui se produit quand nous utilisons une commande **ping ou traceroute**. Elles peuvent également être utiles pour le dépannage. Cependant, dans un environnement de production, les débogages devraient être utilisés avec prudence. Si votre CPU n'est pas puissant, ou si vous avez beaucoup de paquets à commutation par processus, elles peuvent facilement caler votre périphérique. Il y a quelques façons de réduire au minimum l'incidence de la commande **Debug** sur le routeur. Une façon est d'employer les listes d'accès pour se concentrer sur le trafic spécifique que vous voulez surveiller. Voici un exemple :

```
Router4#debug ip packet ?
  <1-199>      Access list
  <1300-2699>  Access list (expanded range)
  detail       Print more debugging detail

Router4#configure terminal
Router4(config)#access-list 150 permit ip host 12.0.0.1 host 34.0.0.4
Router4(config)#^Z

Router4#debug ip packet 150
IP packet debugging is on for access list 150

Router4#show debug
Generic IP:
  IP packet debugging is on for access list 150

Router4#show access-list
Extended IP access list 150
    permit ip host 12.0.0.1 host 34.0.0.4 (5 matches)
```

Avec cette configuration, Router4 imprime seulement le message de débogage qui apparie la liste d'accès 150. Un ping venant de Router1 cause le message suivant d'être affiché :

```
Router4#debug ip packet ?
  <1-199>      Access list
  <1300-2699>  Access list (expanded range)
  detail       Print more debugging detail

Router4#configure terminal
Router4(config)#access-list 150 permit ip host 12.0.0.1 host 34.0.0.4
Router4(config)#^Z

Router4#debug ip packet 150
IP packet debugging is on for access list 150

Router4#show debug
Generic IP:
  IP packet debugging is on for access list 150

Router4#show access-list
```

```
Extended IP access list 150
    permit ip host 12.0.0.1 host 34.0.0.4 (5 matches)
```

Nous ne voyons plus la réponse de Router4, parce que ces paquets ne correspondent pas à la liste d´accès. Pour les voir, nous devrions ajouter ce qui suit :

```
Router4(config)#access-list 150 permit ip host 12.0.0.1 host 34.0.0.4
Router4(config)#access-list 150 permit ip host 34.0.0.4 host 12.0.0.1
```

Nous avons alors :

```
Router4(config)#access-list 150 permit ip host 12.0.0.1 host 34.0.0.4
Router4(config)#access-list 150 permit ip host 34.0.0.4 host 12.0.0.1
```

Une autre façon de réduire au minimum l'incidence de la commande **Debug** est de mettre en mémoire tampon les messages de débogage et de les afficher au moyen de la commande **show log** une fois que le débogage a été arrêté :

```
Router4#configure terminal
Router4(config)#no logging console
Router4(config)#logging buffered 5000
Router4(config)#^Z

Router4#debug ip packet
IP packet debugging is on
Router4#ping 12.0.0.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 12.0.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 36/36/37 ms

Router4#undebug all
All possible debugging has been turned off

Router4#show log
Syslog logging: enabled (0 messages dropped, 0 flushes, 0 overruns)
    Console logging: disabled
    Monitor logging: level debugging, 0 messages logged
    Buffer logging: level debugging, 61 messages logged
    Trap logging: level informational, 59 message lines logged

Log Buffer (5000 bytes):

Jan 20 16:55:46.587: IP: s=34.0.0.4 (local), d=12.0.0.1 (Serial0), len 100,
 sending
Jan 20 16:55:46.679: IP: s=12.0.0.1 (Serial0), d=34.0.0.4 (Serial0), len 100,
 rcvd 3
```

Comme vous pouvez voir, les commandes **ping et traceroute** sont des utilitaires très utiles que vous pouvez employer pour dépanner les problèmes d'accès au réseau. Elles sont également très faciles à utiliser. Ces deux commandes étant les commandes les plus utilisées par les ingénieurs réseau, leur compréhension est très cruciale pour le dépannage de la connectivité réseau.

Informations connexes

- Utilisation des commandes ping et traceroute étendues
- Support technique - Cisco Systems