

# Comprensión de los comandos ping y traceroute

## Contenido

[Introducción](#)

[prerrequisitos](#)

[Requisitos](#)

[Componentes Utilizados](#)

[Convenciones](#)

[Antecedentes](#)

[El comando ping](#)

[¿Por qué no puedo hacer un ping?](#)

[Problema de ruteo](#)

[Interfaz inactiva](#)

[Comando de lista de acceso](#)

[Problema de protocolo de resolución de direcciones \(ARP\)](#)

[Demora](#)

[Dirección de fuente correcta](#)

[Descarte de cola de entradas elevadas](#)

[El comando traceroute](#)

[Rendimiento](#)

[Uso del Comando Debug](#)

[Información Relacionada](#)

## [Introducción](#)

Este documento ilustra el uso de los **comandos ping y traceroute** . Con la ayuda de algunos **comandos debug**, este documento ofrece una vista más detallada del funcionamiento de estos comandos.

**Nota:** La habilitación de cualquier **comando debug** en un router de producción puede causar serios problemas. Le recomendamos que lea cuidadosamente la sección [Uso del Comando Debug](#) antes de ejecutar los **comandos debug**.

## [prerrequisitos](#)

### [Requisitos](#)

No hay requisitos específicos para este documento.

### [Componentes Utilizados](#)

Este documento no tiene restricciones específicas en cuanto a versiones de software y de hardware.

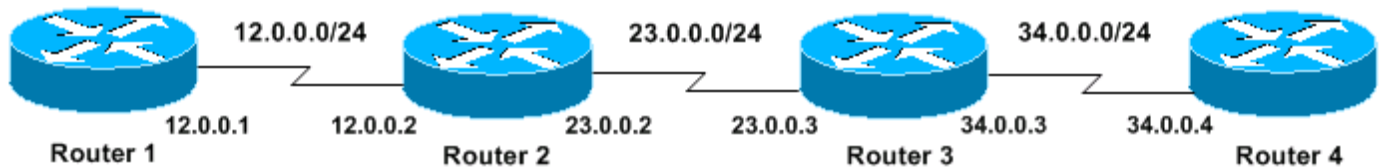
La información que contiene este documento se creó a partir de los dispositivos en un ambiente de laboratorio específico. Todos los dispositivos que se utilizan en este documento se pusieron en funcionamiento con una configuración verificada (predeterminada). Si la red está funcionando, asegúrese de haber comprendido el impacto que puede tener cualquier comando.

## [Convenciones](#)

Para obtener más información sobre las convenciones del documento, consulte las [Convenciones de Consejos Técnicos de Cisco](#).

## [Antecedentes](#)

En este documento, utilizamos la configuración básica que se muestra abajo como base para nuestros ejemplos:



## [El comando ping](#)

El **comando ping** es un método muy común para resolver problemas con la accesibilidad de dispositivos. Utiliza una serie de mensajes eco del protocolo Internet Control Message Protocol (ICMP) para determinar:

- Si un host remoto está activo o inactivo.
- El retraso de ida y vuelta al comunicarse con el host.
- Pérdida de paquetes.

En primer lugar, el comando ping envía un paquete de solicitud de eco a una dirección y después espera una respuesta. El ping es exitoso solo si:

- la petición de eco llega a destino y
- el destino puede obtener una respuesta de eco de regreso al origen dentro de un tiempo predeterminado llamado tiempo de espera agotado. El valor predeterminado de dicho tiempo de espera es de dos segundos en los routers de Cisco.

[Para ver todas las opciones relacionadas con este comando, consulte "Ping" en Comandos para la Resolución de Problemas.](#)

El valor TTL de un **paquete ping** no puede modificarse.

El siguiente ejemplo de resultado muestra el **comando ping** después de habilitar el **comando debug ip packet detail**:

**Advertencia:** Si usa el comando `debug ip packet detail` en un router de producción puede provocar una utilización de CPU elevada. Esto puede provocar una degradación grave del rendimiento o una interrupción de la red. Recomendamos que lea atentamente [Uso del Comando Debug](#) antes de ejecutar los comandos `debug`.

```
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.
```

La siguiente tabla enumera posibles valores de tipo ICMP.

Ti p o d e I C M P	Literal
0	respuesta de eco
3	no se puede acceder al código de destino 0 = no se puede acceder a la red 1 = imposible acceder al host 2 = no se puede acceder al protocolo 3 = no se puede acceder al puerto 4 = fragmentación necesaria y conjunto DF 5 = ruta de origen
4	desconexión del origen
5	redireccionar código 0 = redireccionar datagramas para la red 1 = redireccionar datagramas para el host 2 = redireccionar datagramas para el tipo de servicio y la red 3 = redireccionar datagramas para el tipo de servicio y el host
6	dirección alternativa
8	eco
9	anuncio del router
10	solicitud de router
11	código de tiempo excedido 0 = tiempo a producción excedido en tránsito 1 = tiempo de re ensamblado de fragmento excedido

1 2	problema de parámetro
1 3	timestamp-request
1 4	grupo fecha/hora-contestación
1 5	petición de información
1 6	información-contestación
1 7	petición de máscara
1 8	respuesta de máscara
3 1	conversion-error
3 2	mobile-redirect

La siguiente tabla enumera los posibles caracteres de resultados del recurso ping:

Carácter	Descripción
!	Cada signo de exclamación indica la recepción de una respuesta.
.	Cada período indica que finalizó el tiempo de espera del servidor de red mientras esperaba la respuesta.
U	Se recibió la PUD del error "No se puede acceder al destino".
Q	Desconexión del origen (destino demasiado ocupado).
M	No se pudo fragmentar.
¿?	Tipo desconocido de paquete.
y	Vida útil del paquete excedida.

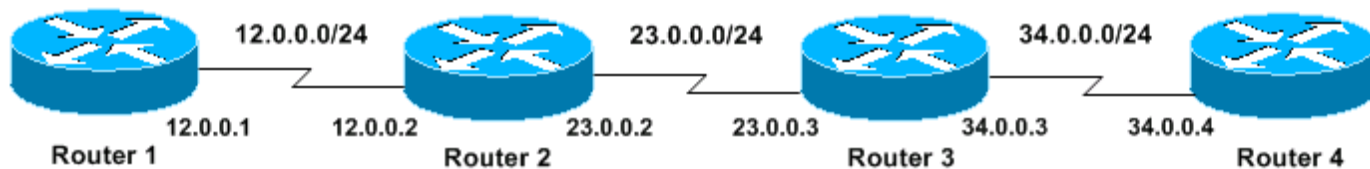
## ¿Por qué no puedo hacer un ping?

Si no puede hacer un ping con éxito, considere estas causas:

### Problema de ruteo

Los siguientes son ejemplos de intentos de ping fallidos, de la determinación del problema, y de qué hacer para resolver el problema.

Este escenario se explica utilizando la topología de red que aparece en el siguiente diagrama:



**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
!
!

```

Intentemos hacer ping en el Router 4 del Router 1:

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)

```

Analicemos con detalle lo que ha sucedido:

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

**Advertencia:** El uso del comando `debug ip packet` en un router de producción puede provocar una alta utilización de la cpu. Esto puede provocar una degradación grave del rendimiento o una interrupción de la red. Recomendamos que lea atentamente [Uso del Comando Debug](#) antes de ejecutar los comandos `debug`.

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

Debido a que no se ejecutan protocolos de ruteo en el Router1, no sabe adónde enviar el paquete y aparece el mensaje "unrouteable".

Agreguemos una ruta estática al Router 1:

```
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
```

Ahora tenemos:

```
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
```

Ahora examinemos cuál es el error en el Router 2:

```
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
```

El Router 1 envía sus paquetes al Router 2 correctamente, pero éste último no sabe cómo acceder a la dirección 34.0.0.4. El Router2 envía un mensaje hacia el Router1 donde informa sobre un "ICMP inalcanzable".

Ahora activemos el protocolo de información de ruteo (RIP) en Router2 y 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
```

Ahora tenemos:

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)
```

Esto es un poco mejor. El Router1 envía paquetes al Router4, pero no obtiene ninguna respuesta del Router4.

Veamos cuál podría ser el problema en el 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
```

El Router 4 recibe los paquetes ICMP e intenta responder a 12.0.0.1 pero, como no tiene una ruta a esta red, simplemente falla.

Agreguemos una ruta estática al Router4:

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

Ahora, funciona perfectamente y ambos lados son mutuamente accesibles:

```
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
```

## Interfaz inactiva

Esta es una situación donde la interfaz deja de funcionar. En el ejemplo a continuación, intentamos hacer ping al Router 4 desde el Router 1:

```
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)
```

Debido a que el ruteo está bien, procederemos a resolver el problema paso a paso. Primero, intentemos hacer ping en el Router 2:

```
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

A partir del resultado anterior, vemos que el problema radica entre Router2 y Router3. Una posibilidad es que la interfaz serial en el Router 3 se haya apagado:

```
Router3#show ip interface brief
```

```
Serial0  34.0.0.3    YES manual up          up
Serial1  23.0.0.3    YES manual administratively down  down
```

Este problema es fácil de resolver:

```
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
```

## Comando de lista de acceso

En este escenario, es conveniente permitir que sólo el tráfico telnet entre al Router 4 a través de la interfaz 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
```

Consulte la sección [Uso del Comando Debug](#) para obtener información sobre la utilización de listas de acceso con comandos debug.

Cuando intentamos hacer ping con el Router 4, obtenemos lo siguiente:

```
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,
```

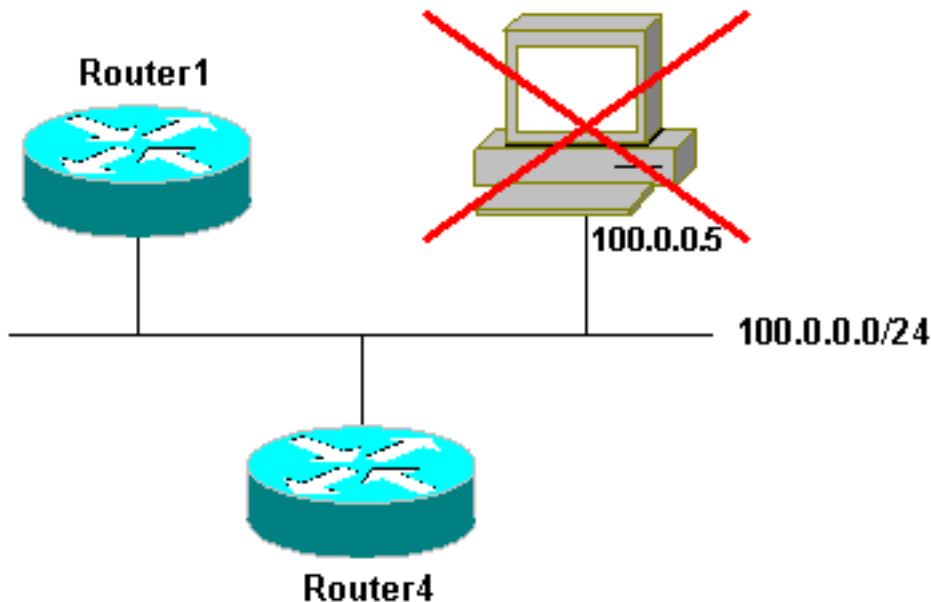
```
sending
Jan 20 16:34:51.367: IP: s=34.0.0.4 (Serial0), d=12.0.0.1 (Serial0), len 56,
rcvd 3
Jan 20 16:34:51.371: ICMP: dst (12.0.0.1) administratively prohibited unreachable
rcv from 34.0.0.4
Jan 20 16:34:51.379: IP: s=12.0.0.1 (local), d=34.0.0.4 (Serial0), len 100,
sending
```

Al final de un comando **access-list**, tendremos siempre implícito **deny all**. Esto significa que se deniegan los paquetes ICMP que ingresan en la interfaz serial 0 en el router 4, y el router 4 envía un mensaje ICMP de "administrativamente prohibido: no se puede alcanzar" al origen del paquete original, como se muestra en el mensaje **debug**. La solución es agregar la línea siguiente en el comando **access-list**:

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

## [Problema de protocolo de resolución de direcciones \(ARP\)](#)

El siguiente es un escenario con una conexión de 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#

```

En este ejemplo, el ping no funciona debido a la “encapsulación fallida”. Esto significa que el router sabe en qué interfaz debe mandar el paquete, pero no sabe cómo hacerlo. En este caso, debe entender cómo funciona el protocolo Address Resolution Protocol (ARP). Ver [Configurar Métodos de Resolución de Direcciones](#) para obtener una explicación detallada.

Básicamente, ARP es un protocolo usado para mapear la dirección de la Capa 2 (dirección MAC) en una dirección de la Capa 3 (dirección IP). Utilice el **comando show arp** para verificar el mapping mencionado anteriormente:

```

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

```

Regresa al problema "falla de encapsulación". Podemos tener una mejor idea del problema utilizando este **comando 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)

```

El resultado anterior muestra que el Router4 está transmitiendo paquetes y los envía a la dirección FFFF.FFFF.FFFF de difusión de Ethernet. Aquí, el 0000.0000.0000 significa que el Router 4 está buscando la dirección MAC del destino 100.0.0.5. Debido a que no conoce la dirección MAC durante la solicitud del ARP en este ejemplo, usa 0000.0000.0000 como un marcador de posición en las tramas de difusión enviadas de la interfaz Ethernet 0 preguntando cuál dirección MAC corresponde a 100.0.0.5. Si no conseguimos una respuesta, la dirección correspondiente en los outputis arp de la demostración **marcó** como incompleto:

```

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

```

Después de un período predeterminado, esta entrada incompleta se purga de la tabla ARP. Mientras la dirección MAC correspondiente no esté en la tabla ARP, el ping falla como resultado de la “encapsulación fallida”.

## Demora

De forma predeterminada, si no recibe una respuesta del extremo remoto en el plazo de dos segundos, el ping falla:

```
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)
```

En las redes con un link lento o una demora prolongada, dos segundos no son suficientes. Puede cambiar este valor predeterminado utilizando un ping extendido:

```
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
```

En el ejemplo anterior, el incrementar el tiempo de espera condujo a un ping exitoso.

**Nota:** El promedio del tiempo de viaje de ida y vuelta es más de dos segundos.

## Dirección de fuente correcta

El siguiente es un ejemplo de una situación típica:



Agregamos una interfaz LAN en el Router1:

```
Router1(config)#interface e0  
Router1(config-if)#ip address
```

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

Desde una estación de la LAN, puede enviar un ping al Router 1. Desde el Router1 puede realizar un ping al Router2. Pero desde una estación de la LAN, no puede enviar un ping al Router2.

Desde el Router1, puede hacer ping al Router2 porque, de manera predeterminada, Usted utiliza la dirección IP de la interfaz saliente como la dirección de origen en su paquete ICMP. El Router 2 no posee información sobre esta LAN nueva. Si tiene que responder a un paquete que viene de esta red, no sabe manejarla.

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

**Advertencia:** El uso del comando `debug ip packet` en un router de producción puede provocar una alta utilización de la cpu. Esto puede provocar una degradación grave del rendimiento o una interrupción de la red. Recomendamos que lea atentamente [Uso del Comando Debug](#) antes de ejecutar los comandos `debug`.

```
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
```

El ejemplo de resultado de arriba funciona porque la dirección de origen del paquete que estamos enviando es `s=12.0.0.1`. Si queremos simular un paquete proveniente de la LAN, tenemos que usar un ping extendido:

```
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)
```

Esta vez la dirección de origen es `20.0.0.1`, ¡y no funciona! Estamos enviando nuestros paquetes,

pero no estamos recibiendo nada. Para solucionar este problema, simplemente tenemos que agregar una ruta a 20.0.0.0 en el Router 2.

La regla básica es que el dispositivo al que se hizo ping también debería saber cómo enviar la respuesta al origen del ping.

## Descarte de cola de entradas elevadas

Cuando un paquete ingresa en el router, el router intenta reenviarlo en el nivel de interrupción. Si no se puede encontrar una coincidencia en una tabla de caché adecuada, el paquete se coloca en la cola de entrada de la interfaz entrante que se procesará. Algunos paquetes se procesan siempre, pero con la configuración apropiada y en las redes estables, el índice de paquetes procesados nunca debe congestionar la cola de entrada. Si la cola de entrada está completa, se perderá el paquete.

Sin embargo, la interfaz es ascendente y no puede hacer ping en el dispositivo debido a los descartes de cola de entrada elevada. Puede verificar los descartes de entrada con el comando **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 suppressed
```

Según lo observado en el resultado, el descarte de cola de entrada es elevado. Consulte [Resolución de Problemas con los Descartes de Cola de Entrada y los Descartes de Cola de Salida](#) para resolver problemas con los descartes de cola de entrada o salida.

## El comando traceroute

El comando **traceroute** se usa para descubrir las rutas que los paquetes toman realmente al desplazarse hacia su destino. El dispositivo (por ejemplo, un router o una PC) envía una secuencia de datagramas de Protocolo de datagrama de usuario (UDP) a una dirección de puerto no válida en el host remoto.

Se envían tres datagramas, cada uno con un valor de campo de Tiempo de vida (TTL) establecido en uno. El valor TTL de 1 hace que el datagrama entre en "tiempo de espera" tan pronto llegue al primer router en la trayectoria; este router luego responde con un Mensaje ICMP de tiempo excedido (TEM) que indica que el datagrama ha caducado.

Ahora se envían otros tres mensajes de UDP, cada uno de los cuales tiene el valor TTL configurado en 2. Esto hace que el segundo router devuelva TEM de ICMP. Este proceso continúa hasta que los paquetes alcancen realmente el otro destino. Puesto que estos datagramas intentan acceder a un puerto no válido en el host de destino, los mensajes inalcanzables del puerto ICMP se devuelven, lo que indica que no se puede acceder al puerto; este evento señala el programa Traceoute que ha finalizado.

El propósito detrás de esto es registrar la fuente de cada mensaje de tiempo excedido ICMP para proporcionar la traza de la trayectoria que el paquete tomó para alcanzar el destino. Para todas las opciones sobre este comando, consulte la [traza \(privilegiada\)](#).

```
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
```

Esta es la primera secuencia de paquetes enviada con un TTL=1. El primer router, en este caso el Router2 (12.0.0.2) descarta el paquete y envía un mensaje type=11 ICMP al origen(12.0.0.1). Esto corresponde al Mensaje de tiempo excedido.

```
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
```

El mismo proceso ocurre para el Router3 (23.0.0.3) con 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

Con un TTL=3, finalmente alcanzamos el Router 4. Esta vez, dado que el puerto no es válido, el Router4 devuelve al Router1 un mensaje ICMP tipo=3, un mensaje de destino inalcanzable, y un código=3 que significa que el puerto es inalcanzable.

La siguiente tabla incluye los caracteres que pueden aparecer en el resultado del comando **tracert**.

### Caracteres de Texto del Rastreador de Ruta del IP

Carácter	Descripción
nn msec	Para cada nodo, el Round-Trip Time en milisegundos para el número especificado de probes
*	Se agotó el tiempo de espera del probe
A	Administrativamente prohibido (por ejemplo, lista de acceso)
Q	Source quench (destino muy ocupado)
I	Prueba interrumpida del usuario
U	Puerto inalcanzable
H	Host fuera de alcance
N	Red inalcanzable
P	Protocolo inalcanzable
T	Tiempo de espera
¿?	Tipo desconocido de paquete

## Rendimiento

Mediante el uso de los comandos ping y tracert, obtenemos el tiempo de viaje de ida y vuelta (RTT). Este es el tiempo necesario para enviar un paquete de eco y obtener una respuesta. Esto puede resultar útil para tener una leve idea del retraso en el link. Sin embargo, estas cifras no son lo suficientemente exactas para ser utilizadas para la evaluación de rendimiento.

Cuando el destino de un paquete es el router en sí, se trata de un process-switched-packet. El procesador debe manejar la información de este paquete, y devuelve una respuesta. Este no es el objetivo principal de un router. Por definición, un router está diseñado para rutear paquetes. Se ofrece como el mejor servicio posible la respuesta a un ping.

Para ilustrarlo, aquí hay un ejemplo de un ping desde el Router1 al 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
```

El RTT es aproximadamente cuatro milisegundos. Después de que habilite algunas características de proceso intensivo en el Router 2, intente hacer ping en el Router 2 desde el



## Router 1.

```
Router1#ping 12.0.0.2
```

```
Type escape sequence to abort.
```

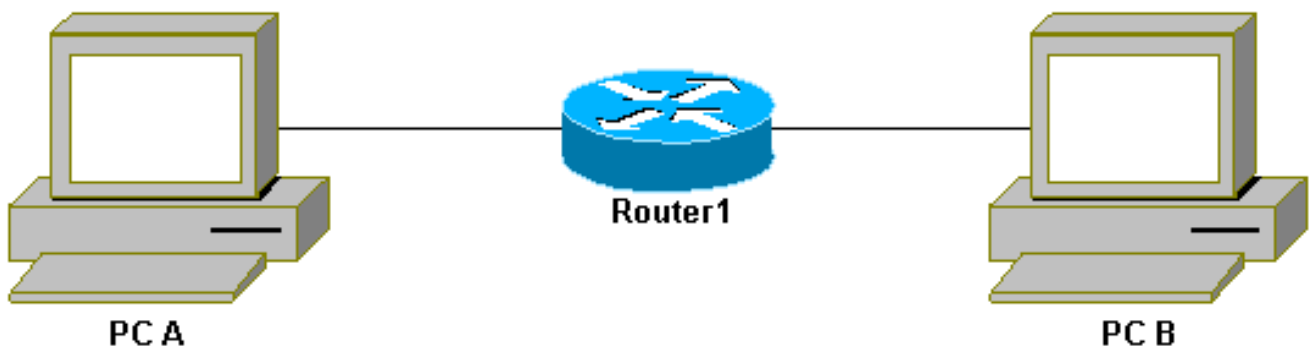
```
Sending 5, 100-byte ICMP Echos to 12.0.0.2, timeout is 2 seconds:
```

```
!!!!!
```

```
Success rate is 100 percent (5/5), round-trip min/avg/max = 24/25/28 ms
```

El RTT ha aumentado dramáticamente aquí. El Router2 está muy ocupado, con lo que dar respuesta al ping no es su prioridad básica.

Una mejor manera de probar el rendimiento del router es con tráfico **atravesando** el router.



El tráfico es entonces se conmuta de manera rápida, y es dirigido por el router con la prioridad más alta. Para ejemplificar esto, volvamos a la red básica:



Hagamos ping en el Router 3 desde el Router 1:

```
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
```

El tráfico está pasando por el Router2 y ahora se lo conmuta de manera rápida.

Ahora, habilitemos la característica de proceso intensivo en el Router 2:

```
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
```

No hay casi diferencia. Esto se debe a que, en el Router 2, los paquetes se manejan ahora en el

nivel de interrupción.

## Uso del Comando Debug

Antes de ejecutar un comando debug, consulte Información Importante sobre Comandos Debug.

Los diversos **comandos debug** que hemos utilizado hasta ahora nos proporciona una perspectiva sobre lo que sucede cuando utilizamos un **comando ping o traceroute**. También pueden ser útiles para resolver problemas. Sin embargo, en un entorno de producción, los debugs se deben utilizar con la precaución. Si su CPU no es potente, o si tiene muchos process-switched packets, pueden fácilmente estancar su dispositivo. Hay un par de maneras de minimizar el impacto del **comando debug** en el router. Una manera es usar listas de acceso para acotar el tráfico específico que desea monitorear. Aquí tiene un ejemplo:

```
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)
```

Con esta configuración, el Router4 sólo imprime el mensaje de depuración que coincide con la lista de acceso 150. Un ping que proviene del Router1 provoca el siguiente mensaje:

```
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)
```

Ya no vemos la respuesta del Router 4 debido a que estos paquetes no coinciden con la lista de acceso. Para verlos, debemos agregar lo siguiente:

```
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
```

Entonces tenemos:

```
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
```

Otra forma de minimizar el impacto del **comando debug** es almacenar en búfer los mensajes de depuración y mostrarlos mediante el **comando show log**, una vez completada la depuración:

```
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
```

Como puede ver, los **comandos ping y traceroute** son utilidades de gran ayuda que puede utilizar para resolver problemas de acceso a la red. También son muy fáciles de utilizar. Como estos dos comandos son los comandos utilizados más ampliamente de los ingenieros de red, comprenderlos es muy importante para localización de averías de la conectividad de red.

## [Información Relacionada](#)

- [Uso de los Comandos Ping Extendido y Traceroute Extendido](#)
- [Soporte Técnico - Cisco Systems](#)