Wichtige Informationen über Debugbefehle

Inhalt

Einleitung

Diese Seite enthält allgemeine Richtlinien zur Verwendung der auf Cisco IOS®-Plattformen verfügbaren Debugging sowie Beispiele für die ordnungsgemäße Verwendung der **debug ip packet** Befehls- und konditionelles Debuggen.

> **Anmerkung:** In diesem Dokument wird nicht erläutert, wie bestimmte Debugbefehle und Ausgaben verwendet und interpretiert werden. Weitere Informationen zu den jeweiligen Cisco Debugbefehlsverweisen finden Sie in der entsprechenden Dokumentation. **debug** Befehle.

Die Ausgabe von **debug** privilegierte EXEC-Befehle stellen Diagnoseinformationen bereit, die eine Reihe von Internetworking-Ereignissen enthalten, die sich auf den Protokollstatus und die Netzwerkaktivität im Allgemeinen beziehen.

Voraussetzungen

Anforderungen

Cisco empfiehlt, dass Sie über Kenntnisse in folgenden Bereichen verfügen:

- Herstellen einer Verbindung mit dem Router über die Konsolen-, Aux- und VTY-Ports
- Allgemeine Cisco IOS-Konfigurationsprobleme
- Interpretieren von Cisco IOS-Debug-Ausgaben

Verwendete Komponenten

Dieses Dokument ist nicht auf bestimmte Software- und Hardware-Versionen beschränkt.

Die Informationen in diesem Dokument beziehen sich auf Geräte in einer speziell eingerichteten Testumgebung. Alle Geräte, die in diesem Dokument benutzt wurden, begannen mit einer gelöschten (Nichterfüllungs) Konfiguration. Wenn Ihr Netz Live ist, überprüfen Sie, ob Sie die mögliche Auswirkung jedes möglichen Befehls verstehen.

Warnungen

Nutzung **debug** mit Vorsicht. Im Allgemeinen wird empfohlen, dass diese Befehle nur unter der Anleitung des technischen Supports für den Router verwendet werden, wenn spezifische Probleme behoben werden.

Wenn das Debuggen aktiviert wird, kann der Betrieb des Routers gestört werden, wenn in Internetworks hohe Auslastungsbedingungen auftreten. Wenn also die Protokollierung aktiviert ist, kann der Zugriffsserver periodisch einfrieren, sobald der Konsolenport mit Protokollmeldungen überlastet wird.

Bevor Sie beginnen **debug** -Befehls verwenden, sollten Sie immer die Ausgabe berücksichtigen, die dieser Befehl generiert, und die Zeit, die dies dauern kann. Wenn Sie beispielsweise einen Router mit einer BRI-Schnittstelle (Basic Rate Interface) haben, **debug isdn q931** wird das System wahrscheinlich nicht beschädigen. Dasselbe Debuggen auf einem AS5800 mit voller E1-Konfiguration kann jedoch möglicherweise so viele Eingaben generieren, dass es hängen bleibt und nicht mehr reagiert.

Überprüfen Sie vor dem Debuggen die CPU-Auslastung mit dem **show processes cpu** aus. Überprüfen Sie, ob genügend CPU verfügbar ist, bevor Sie mit dem Debuggen beginnen. Unter Fehlerbehebung bei hoher CPU-Auslastung auf Cisco Routern finden Sie weitere Informationen zum Umgang mit hoher CPU-Auslastung. Wenn Sie beispielsweise über einen Cisco 7200-Router verfügen, dessen ATM-Schnittstelle das Bridging vornimmt, kann der Router, abhängig von der Anzahl der konfigurierten Subschnittstellen, einen Großteil seiner CPU für das Neustarten verwenden. Der Grund hierfür ist, dass für jeden Virtual Circuit (VC) ein BPDU-Paket (Bridge Protocol Data Unit) generiert werden muss. Das Starten von Debugging während einer solch kritischen Zeit kann dazu führen, dass die CPU-Auslastung drastisch zunimmt und zu einem Ausfall der Netzwerkverbindung führt.

> **Anmerkung:** Beim Ausführen von Debuggen wird normalerweise keine Routeraufforderung angezeigt, insbesondere wenn das Debuggen intensiv ist. In den meisten Fällen können Sie jedoch alle Befehle ohne Debuggen oder Debuggen ausführen, um das Debuggen zu beenden. Weitere Informationen zur sicheren Verwendung von Debuggen finden Sie im Abschnitt Abrufen von Debugausgängen.

Konventionen

Weitere Informationen zu Dokumentkonventionen finden Sie unter Cisco Technical Tips Conventions (Technische Tipps von Cisco zu Konventionen).

Vor dem Debuggen

Stellen Sie zusätzlich zu den oben genannten Punkten sicher, dass Sie die Auswirkungen des

Debuggens auf die Stabilität der Plattform verstehen. Sie sollten auch überlegen, mit welcher Schnittstelle auf dem Router eine Verbindung hergestellt werden soll. Dieser Abschnitt enthält einige Richtlinien.

Abrufen von Debug-Ausgaben

Router können Debug-Ausgaben für verschiedene Schnittstellen anzeigen, einschließlich Konsolen-, AUX- und VTY-Ports. Router können auch Meldungen in einem internen Puffer auf einem externen Unix-Syslog-Server protokollieren. Anweisungen und Hinweise zu den einzelnen Methoden werden nachfolgend erläutert:

Konsolenport

Wenn Sie auf der Konsole verbunden sind, müssen Sie bei normalen Konfigurationen keine zusätzlichen Arbeiten durchführen. Die Debug-Ausgabe sollte automatisch angezeigt werden. Stellen Sie jedoch sicher, dass **logging console level** wird wie gewünscht festgelegt, und die Protokollierung wurde nicht deaktiviert mit dem **no logging console** aus.

> **Warnung:** Übermäßige Debug-Vorgänge am Konsolen-Port eines Routers können dazu führen, dass er hängen bleibt. Dies liegt daran, dass der Router die Konsolenausgabe automatisch gegenüber anderen Routerfunktionen priorisiert. Wenn der Router eine große Debug-Ausgabe an den Konsolenport verarbeitet, kann dies dazu führen, dass er hängen bleibt. Wenn die Debugausgabe übermäßig hoch ist, können Sie die vty-Ports (telnet) oder die Protokollpuffer verwenden, um das Debuggen abzurufen. Weitere Informationen finden Sie weiter unten.

> **Anmerkung:** Standardmäßig ist die Protokollierung auf dem Konsolenport aktiviert. Daher verarbeitet der Konsolenport immer die Debugausgabe, selbst wenn Sie tatsächlich einen anderen Port oder eine andere Methode (z. B. Aux, vty oder Puffer) verwenden, um die Ausgabe zu erfassen. Cisco empfiehlt daher, unter normalen Betriebsbedingungen jederzeit den Befehl no logging console zu aktivieren und andere Methoden zum Erfassen von Debugging zu verwenden. Wenn Sie die Konsole verwenden müssen, schalten Sie die Protokollierungskonsole vorübergehend wieder ein.

AUX-Port

Wenn Sie über einen externen Port verbunden sind, geben Sie die **terminal monitor** aus. Überprüfen Sie außerdem, ob **no logging on** wurde auf dem Router nicht aktiviert.

> **Anmerkung:** Wenn Sie den Aux-Port zur Überwachung des Routers verwenden, beachten Sie, dass der Aux-Port beim Neustart des Routers die Ausgabe der Startsequenz nicht anzeigt. Stellen Sie eine Verbindung zum Konsolenport her, um die Startsequenz anzuzeigen.

VTY-Ports

Wenn Sie über einen externen Port oder Telnet verbunden sind, geben Sie **terminal monitor** aus. Überprüfen Sie außerdem, ob **no logging on** wurde nicht verwendet.

Protokollieren von Nachrichten in einem internen Puffer

Das Standardprotokollierungsgerät ist die Konsole. alle Meldungen werden auf der Konsole angezeigt, sofern nichts anderes angegeben wird.

Um Meldungen in einem internen Puffer zu protokollieren, verwenden Sie die **logging buffered** Router-Konfigurationsbefehl. Dies ist die vollständige Syntax dieses Befehls:

```
logging buffered
no logging buffered
```

Die **logging buffered** kopiert Protokollmeldungen in einen internen Puffer, anstatt in die Konsole zu schreiben. Der Puffer ist kreisförmig, daher überschreiben neuere Nachrichten ältere Nachrichten. Um die im Puffer protokollierten Meldungen anzuzeigen, verwenden Sie den Befehl privilegierter EXEC **show logging**. Die erste angezeigte Meldung ist die älteste im Puffer. Sie können die Größe des Puffers sowie den Schweregrad der zu protokollierenden Meldungen angeben.

> **Tipp:** Vergewissern Sie sich, dass genügend Speicher im Feld vorhanden ist, bevor Sie die Puffergröße eingeben. Verwenden des Cisco IOS **show proc mem** um den verfügbaren Speicher anzuzeigen.

Die **no logging buffered** unterbricht die Verwendung des Puffers und schreibt Meldungen in die Konsole (der Standardwert).

Protokollieren von Nachrichten auf einem UNIX Syslog-Server

Verwenden Sie den Konfigurationsbefehl logging router, um Meldungen beim Host des Syslog-Servers zu protokollieren. Die vollständige Syntax dieses Befehls lautet wie folgt:

```
logging no logging
```

Die **logging** identifiziert einen Syslog-Server-Host, um Protokollmeldungen zu empfangen. Das < *ip-address* >-Argument ist die IP-Adresse des Hosts. Durch mehrmaliges Ausgeben dieses Befehls erstellen Sie eine Liste von Syslog-Servern, die Protokollmeldungen empfangen.

Die **no logging** Löscht den Syslog-Server mit der angegebenen Adresse aus der Liste der Syslogs.

Weitere Vordebugaufgaben

1. Richten Sie die Terminal-Emulator-Software (z. B. HyperTerminal) ein, damit die Debug-Ausgabe in eine Datei aufgezeichnet werden kann. Klicken Sie z. B. in HyperTerminal auf **Transfer**, und klicken Sie anschließend auf **Capture Text**, und wählen Sie die entsprechenden Optionen aus. Weitere Informationen finden Sie unter Erfassen der Textausgabe von Hyperterminal. Weitere Terminal-Emulator-Software finden Sie in der Softwaredokumentation.
2. Aktivieren von Millisekunde-Zeitstempeln (msec) mithilfe des **service timestamps** command:

```
router(config)#service timestamps debug datetime msec
     router(config)#service timestamps log datetime msec
```

Diese Befehle fügen Zeitstempel zum Debuggen im Format MMM DD HH:MM:SS hinzu, die das Datum und die Uhrzeit gemäß Systemuhr angeben. Wenn die Systemuhr nicht eingestellt wurde, werden dem Datum und der Uhrzeit ein Sternchen (*) vorangestellt, das angibt, dass Datum und Uhrzeit wahrscheinlich nicht korrekt sind.

Generell ist es ratsam, Zeitstempel in Millisekunden zu konfigurieren, da dies bei Debugausgängen eine hohe Klarheit gewährleistet. Millisekunde-Zeitstempel liefern eine bessere Anzeige des Zeitpunkts der verschiedenen Debugereignisse im Verhältnis zueinander. Beachten Sie jedoch, dass der Konsolen-Port, wenn er viele Meldungen ausgibt, möglicherweise nicht mit dem tatsächlichen Zeitpunkt des Ereignisses korreliert. Wenn Sie z. B. **debug x25** alle auf einem Gerät mit 200 VCs, und die Ausgabe wird im Puffer (mithilfe der **no logging console** und **logging buffered** -Befehlen), kann der in der Debug-Ausgabe angezeigte Timestamp (innerhalb des Puffers) nicht die genaue Zeit sein, zu der das Paket die Schnittstelle durchläuft. Verwenden Sie daher keine msec-Zeitstempel, um Leistungsprobleme zu beweisen, sondern um relative Informationen zu erhalten, wann Ereignisse auftreten.

So beenden Sie das Debuggen

Um einen Debugvorgang zu beenden, verwenden Sie die **no debug all** Oder **undebug all** Befehle. Stellen Sie sicher, dass die Debugging-Funktion mithilfe des Befehls deaktiviert wurde. **show debug**.

Beachten Sie, dass die Befehle **no logging console** und **terminal no monitor** verhindern nur, dass die Ausgabe auf der Konsole, Aux oder vty ausgegeben wird. Sie beendet das Debuggen nicht und verwendet daher Router-Ressourcen.

Verwenden des Befehls debug ip packet

Die **debug ip packet** erzeugt Informationen zu Paketen, die vom Router nicht schnell geswitcht werden. Da jedoch für jedes Paket eine Ausgabe generiert wird, kann die Ausgabe umfangreich sein und somit den Router hängen lassen. Aus diesem Grund darf nur **debug ip packet** unter strengsten Kontrollen gemäß diesem Abschnitt.

Die beste Möglichkeit, die Ausgabe von **debug ip packet** erstellt eine Zugriffsliste, die mit dem Debugger verknüpft ist. Nur Pakete, die die Kriterien der Zugriffsliste erfüllen, unterliegen **debug ip packet**. Diese Zugriffsliste muss nicht auf eine Schnittstelle angewendet werden, sondern wird auf den Debugvorgang angewendet.

Vor der Anwendung **debugging ip packet**, beachten Sie, dass der Router standardmäßig Fast Switching durchführt oder CEF-Switching durchführt, wenn dies konfiguriert ist. Dies bedeutet, dass das Paket nach der Implementierung dieser Techniken nicht an den Prozessor weitergeleitet wird, sodass beim Debuggen nichts angezeigt wird. Damit dies funktioniert, müssen Sie das schnelle Switching auf dem Router mit **no ip route-cache** (für Unicast-Pakete) oder **no ip mroute-cache** (für Multicast-Pakete). Dies sollte auf die Schnittstellen angewendet werden, an denen der Datenverkehr fließen soll. Überprüfen Sie dies mit dem **show ip route** aus.

Warnungen:

- Das Deaktivieren des Fast-Switching auf einem Router, der eine große Anzahl von Paketen

verarbeitet, kann dazu führen, dass die CPU-Auslastung steigt, sodass die Box hängen bleibt oder die Verbindung zu den Peers unterbrochen wird.
- Vermeiden Sie das schnelle Switching auf einem Router, auf dem MPLS (Multi Protocol Label Switching) ausgeführt wird. MPLS wird in Verbindung mit CEF verwendet. Aus diesem Grund kann die Deaktivierung des Fast-Switching auf der Schnittstelle verheerende Auswirkungen haben.

Betrachten wir ein Beispielszenario:

Die auf router_122 konfigurierte Zugriffsliste lautet:

```
access-list 105 permit icmp host 10.10.10.2 host 13.1.1.1
access-list 105 permit icmp host 13.1.1.1 host 10.10.10.2
```

Diese Zugriffsliste erlaubt jedes Internet Control Message Protocol (ICMP)-Paket vom Host-Router_121 (mit der IP-Adresse 10.10.10.2) zum Host-Router_123 (mit der IP-Adresse 13.1.1.1) sowie in die andere Richtung. Es ist wichtig, dass Sie die Pakete in beide Richtungen zulassen. Andernfalls kann der Router das zurückgegebene ICMP-Paket verwerfen.

Entfernen Sie Fast-Switching auf nur einer Schnittstelle auf Router_122. Dies bedeutet, dass Sie nur die Debugging für die Pakete sehen können, die für diese Schnittstelle bestimmt sind, wie aus der Sicht des IOS-Abfangs für das Paket ersichtlich ist. Beim Debuggen werden solche Pakete mit "d=" angezeigt. Da Sie das schnelle Switching auf der anderen Schnittstelle noch nicht deaktiviert haben, unterliegt das Rückgabepaket nicht der Richtlinie **debug ip packet**. Diese Ausgabe zeigt, wie Sie das schnelle Switching deaktivieren können:

```
router_122(config)#interface virtual-template 1
router_122(config-if)#no ip route-cache
router_122(config-if)#end
```

Sie müssen jetzt aktivieren **debug ip packet** mit der zuvor definierten Zugriffsliste (Zugriffsliste 105).

```
router_122#debug ip packet detail 105
IP packet debugging is on (detailed) for access list 105
router_122#
00:10:01: IP: s=13.1.1.1 (Serial3/0), d=10.10.10.2 (Virtual-Access1),
g=10.10.10.2, len 100, forward

00:10:01:     ICMP type=0, code=0
! -- ICMP packet from 13.1.1.1 to 10.10.10.2. ! -- This packet is displayed because it matches
the ! -- source and destination requirements in access list 105 00:10:01: IP: s=13.1.1.1
(Serial3/0), d=10.10.10.2 (Virtual-Access1), g=10.10.10.2, len 100, forward 00:10:01: ICMP
type=0, code=0 00:10:01: IP: s=13.1.1.1 (Serial3/0), d=10.10.10.2 (Virtual-Access1),
```

```
g=10.10.10.2, len 100, forward 00:10:01: ICMP type=0, code=0
```

Entfernen wir Fast-Switching auf der anderen Schnittstelle (auf Router_122). Dies bedeutet, dass alle Pakete über diese beiden Schnittstellen jetzt paketvermittelt werden (eine Anforderung für **debug ip packet**):

```
router_122(config)#interface serial 3/0
router_122(config-if)#no ip route-cache
router_122(config-if)#end

router_122#
00:11:57:  IP: s=10.10.10.2 (Virtual-Access1), d=13.1.1.1
(Serial3/0), g=172.16.1.6, len 100, forward
00:11:57:  ICMP type=8, code=0
! -- ICMP packet (echo) from 10.10.10.2 to 13.1.1.1 00:11:57: IP: s=13.1.1.1 (Serial3/0),
d=10.10.10.2 (Virtual-Access1),
g=10.10.10.2, len 100, forward
00:11:57:  ICMP type=0, code=0
! -- ICMP return packet (echo-reply) from 13.1.1.1 to 10.10.10.2 00:11:57: IP: s=10.10.10.2
(Virtual-Access1), d=13.1.1.1 (Serial3/0), g=172.16.1.6, len 100, forward 00:11:57: ICMP type=8,
code=0 00:11:57: IP: s=13.1.1.1 (Serial3/0), d=10.10.10.2 (Virtual-Access1), g=10.10.10.2, len
100, forward 00:11:57: ICMP type=0, code=0
```

Beachten Sie, dass die Debug-IP-Paketausgabe keine Pakete anzeigt, die nicht den Kriterien der Zugriffsliste entsprechen. Weitere Informationen zu diesem Verfahren finden Sie unter Understanding the Ping and Traceroute Commands.

Weitere Informationen zum Erstellen von Zugriffslisten finden Sie unter Standard IP Access List Logging (Standardprotokollierung für IP-Zugriffslisten).

Ausgelöste Debugger

Wenn die Funktion für das bedingte Auslösen des Debuggens aktiviert ist, generiert der Router Fehlerbehebungsmeldungen für Pakete, die in den Router eingehen oder ihn auf einer angegebenen Schnittstelle verlassen. Der Router generiert keine Debugausgabe für Pakete, die über eine andere Schnittstelle eingehen oder gehen.

Sehen Sie sich eine einfache Implementierung von bedingten Debuggen an. Betrachten Sie dieses Szenario: Der unten gezeigte Router (trabol) verfügt über zwei Schnittstellen (Serial 0 und Serial 3), auf denen beide HDLC-Kapselung ausgeführt wird.

Sie können die normale **debug serial interface** um die auf allen Schnittstellen empfangenen HDLC-Keepalives zu beobachten. Sie können die Keepalives auf beiden Schnittstellen beobachten.

```
traxbol#debug serial interface
Serial network interface debugging is on
traxbol#
*Mar  8 09:42:34.851: Serial0: HDLC myseq 28, mineseen 28*, yourseen 41, line up
! -- HDLC keeplaive on interface Serial 0 *Mar 8 09:42:34.855: Serial3: HDLC myseq 26, mineseen
26*, yourseen 27, line up
! -- HDLC keeplaive on interface Serial 3 *Mar 8 09:42:44.851: Serial0: HDLC myseq 29, mineseen
29*, yourseen 42, line up *Mar 8 09:42:44.855: Serial3: HDLC myseq 27, mineseen 27*, yourseen
28, line up
```

Aktivieren von bedingtem Debuggen für die Schnittstellenseriennummer 3. Das bedeutet, dass nur Debugger für die Schnittstellenseriennummer 3 angezeigt werden. Verwenden Sie **debug interface** ***<interface_type interface_number>*** aus.

```
traxbol#debug interface serial 3
Condition 1 set
```

Verwenden Sie **show debug condition** um zu überprüfen, ob das bedingte Debuggen aktiv ist. Beachten Sie, dass eine Bedingung für Schnittstellenseriennummer 3 aktiv ist.

```
traxbol#show debug condition

Condition 1: interface Se3 (1 flags triggered)
Flags: Se3
traxbol#
```

Beachten Sie, dass jetzt nur die Debug-Meldungen für die Schnittstellenseriennummer 3 angezeigt werden.

```
*Mar  8 09:43:04.855: Serial3: HDLC myseq 29, mineseen 29*, yourseen 30, line up
*Mar  8 09:43:14.855: Serial3: HDLC myseq 30, mineseen 30*, yourseen 31, line up
```

Verwenden Sie **undebug interface <*interface_type interface_number*>** um das bedingte Debuggen zu entfernen. Es wird empfohlen, die Debugger (z. B. unter Verwendung von "underbug all") zu deaktivieren, bevor Sie den bedingten Trigger entfernen. Dadurch wird eine Flut von Debug-Ausgaben vermieden, wenn die Bedingung entfernt wird.

```
traxbol#undebug interface serial 3
This condition is the last interface condition set.
Removing all conditions may cause a flood of debugging
messages to result, unless specific debugging flags
are first removed.
Proceed with removal? [yes/no]: y
Condition 1 has been removed
traxbol
```

Sie können jetzt beobachten, dass Debugging sowohl für die Schnittstelle seriell 0 als auch für seriell 3 angezeigt wird.

```
*Mar  8 09:43:34.927: Serial3: HDLC myseq 32, mineseen 32*, yourseen 33, line up
*Mar  8 09:43:44.923: Serial0: HDLC myseq 35, mineseen 35*, yourseen 48, line up
```

> **Warnung:** Einige Debugvorgänge sind von sich aus bedingt. Ein Beispiel ist ATM-Debuggen. Beim ATM-Debuggen sollten Sie explizit die Schnittstelle angeben, für die Debuggen aktiviert werden sollen, anstatt Debuggen auf allen ATM-Schnittstellen zu aktivieren und eine Bedingung anzugeben.

In diesem Abschnitt wird die richtige Methode veranschaulicht, um das Debuggen von ATM-Paketen auf eine Subschnittstelle zu beschränken:

```
arielle-nrp2#debug atm packet interface atm 0/0/0.1
!--- Note that you explicitly specify the sub-interface to be used for debugging ATM packets
debugging is on Displaying packets on interface ATM0/0/0.1 only
arielle-nrp2#
*Dec 21 10:16:51.891: ATM0/0/0.1(O):
VCD:0x1 VPI:0x1 VCI:0x21 DM:0x100 SAP:AAAA CTL:03 OUI:0080C2 TYPE:0007
Length:0x278
*Dec 21 10:16:51.891: 0000 FFFF FFFF FFFF 0010 7BB9 BDC4 0800 4500 025C 01FE
0000 FF11 61C8 0A30
```

```
*Dec 21 10:16:51.891: 4B9B FFFF FFFF 0044 0043 0248 0000 0101 0600 0015 23B7
0000 8000 0000 0000
*Dec 21 10:16:51.891: 0000 0000 0000 0000 0000 0000 0010 7BB9 BDC3 0000 0000
0000 0000 0000 0000
*Dec 21 10:16:51.891: 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000
*Dec 21 10:16:51.891: 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000
*Dec 21 10:16:51.891: 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000
*Dec 21 10:16:51.891: 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000
*Dec 21 10:16:51.895: 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000
*Dec 21 10:16:51.895:
arielle-nrp2#
```

Wenn Sie versuchen, **atm debugging** An allen Schnittstellen (mit angewendetem Zustand) kann der Router hängen, wenn er über eine große Anzahl von ATM-Subschnittstellen verfügt. Ein Beispiel für die falsche Methode für ATM-Debuggen wird angezeigt.

In diesem Fall können Sie sehen, dass eine Bedingung angewendet wird, aber Sie sehen auch, dass dies keine Wirkung hat. Sie können das Paket immer noch von der anderen Schnittstelle aus sehen. In diesem Übungsszenario gibt es nur zwei Schnittstellen und nur sehr wenig Datenverkehr. Wenn die Anzahl der Schnittstellen hoch ist, ist die Debug-Ausgabe für alle Schnittstellen extrem hoch und kann dazu führen, dass der Router hängen bleibt.

```
arielle-nrp2#show debugging condition
Condition 1: interface AT0/0/0.1 (1 flags triggered)
Flags: AT0/0/0.1
! -- A condition for a specific interface. arielle-nrp2#debug atm packet
ATM packets debugging is on
Displaying all ATM packets
arielle-nrp2#
*Dec 21 10:22:06.727: ATM0/0/0.2(O):
! -- You see debugs from interface ATM0/0/0/.2, even though the condition ! -- specified ONLY
AT0/0/0.1 VCD:0x2 VPI:0x5 VCI:0x37 DM:0x100 SAP:AAAA CTL:03 OUI:0080C2 TYPE:000E Length:0x2F
*Dec 21 10:22:06.727: 0000 0000 0180 0000 107B B9BD C400 0000 0080 0000 107B B9BD C480 0800 0014
*Dec 21 10:22:06.727: 0002 000F 0000 *Dec 21 10:22:06.727: un a *Dec 21 10:22:08.727:
ATM0/0/0.2(O): VCD:0x2 VPI:0x5 VCI:0x37 DM:0x100 SAP:AAAA CTL:03 OUI:0080C2 TYPE:000E
Length:0x2F *Dec 21 10:22:08.727: 0000 0000 0180 0000 107B B9BD C400 0000 0080 0000 107B B9BD
C480 0800 0014 *Dec 21 10:22:08.727: 0002 000F 0000 *Dec 21 10:22:08.727: ll *Dec 21
10:22:10.727: ATM0/0/0.2(O): VCD:0x2 VPI:0x5 VCI:0x37 DM:0x100 SAP:AAAA CTL:03 OUI:0080C2
TYPE:000E Length:0x2F *Dec 21 10:22:10.727: 0000 0000 0080 0000 107B B9BD C400 0000 0080 0000
107B B9BD C480 0800 0014 *Dec 21 10:22:10.727: 0002 000F 0000 *Dec 21 10:22:10.727: *Dec 21
10:22:12.727: ATM0/0/0.2(O): VCD:0x2 VPI:0x5 VCI:0x37 DM:0x100 SAP:AAAA CTL:03 OUI:0080C2
TYPE:000E Length:0x2F *Dec 21 10:22:12.727: 0000 0000 0080 0000 107B B9BD C400 0000 0080 0000
107B B9BD C480 0800 0014 *Dec 21 10:22:12.727: 0002 000F 0000 *Dec 21 10:22:12.727: *Dec 21
10:22:13.931: ATM0/0/0.1(O):
!--- You also see debugs for interface ATM0/0/0.1 as you wanted. VCD:0x1 VPI:0x1 VCI:0x21
DM:0x100 SAP:AAAA CTL:03 OUI:0080C2 TYPE:0007 Length:0x278 *Dec 21 10:22:13.931: 0000 FFFF FFFF
FFFF 0010 7BB9 BDC4 0800 4500 025C 027F 0000 FF11 6147 0A30 *Dec 21 10:22:13.931: 4B9B FFFF FFFF
0044 0043 0248 0000 0101 0600 001A 4481 0000 8000 0000 0000 *Dec 21 10:22:13.931: 0000 0000 0000
0000 0000 0000 0010 7BB9 BDC3 0000 0000 0000 0000 0000 0000 *Dec 21 10:22:13.931: 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 *Dec 21 10:22:13.931: 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 *Dec 21 10:22:13.931: 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 *Dec 21 10:22:13.931: 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 *Dec 21 10:22:13.935: 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
```

Zugehörige Informationen

- Unterstützung von DFÜ- und Zugriffstechnologie
- Technischer Support und Dokumentation für Cisco Systeme