

Cisco 12000, 10000, 7600, and 7500 Series Routers: Troubleshooting IPC-3-NOBUFF Messages

Document ID: 21165

Introduction

Prerequisites

- Requirements
- Components Used
- Conventions

Background Information

IPC Terminology Fundamentals

- IPC Address Format

What Information Needs to be Transmitted Through IPC?

How IPC Messages Are Transmitted

- Cisco 7500 Series
- Cisco 12000 Series

Steps to Troubleshoot Issues, Known Bugs and Improvements

- Step 1: Tune the IPC Cache
- Step 2: Tune the IPC Throughput
- List of IPC Enhancements
- Cisco 7600 Series

Collect Troubleshooting Information for the Cisco TAC

Related Information

Introduction

This document explains why your router reports IPC-related log messages and how to troubleshoot this problem. This document also includes a review of IPC terminology.

Prerequisites

Requirements

Readers of this document should have knowledge of these topics:

- Cisco Router administration
- IPC and its terminology

Components Used

The information in this document is based on these software and hardware versions:

- All Cisco IOS® software releases that support the Cisco 12000, 10000, 7600, and 7500 Series Routers.
- Cisco 12000, 10000, 7600, and 7500 Series Routers.

The information in this document was created from the devices in a specific lab environment. All of the

devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Conventions

Refer to the Cisco Technical Tips Conventions for more information on document conventions.

Background Information

The Cisco IOS software Inter-Process Communication (IPC) module provides a communication infrastructure by which processes in a distributed system can interact with each other. It also provides transparent communication across backplanes, networks and shared memory.

IPC services serve as the means by which line cards (LCs) and the central route processor (RP) in a distributed system communicate with each other through an exchange of IPC messages sent from the RP to the LCs, and also between active and standby RPs. These messages include configuration commands and responses to those commands, and also "events" that need to be reported by an LC to the RP.

The Cisco 12000 Series, Cisco 10000 Series, Cisco 7600 Series, and the Cisco 7500 Series use a distributed architecture based on IPC messages. Under rare conditions, these routers may report these IPC-related log messages:

- Cisco 12000 Series `%IPC-3-NOBUFF`: The main IPC message header cache has emptied
- Cisco 7500 Series `%IPC_RSP_CBUS-3-NOBUF`: No more IPC memd buffers to transmit IPC message

Note: IPC is also used on Cisco 6400 series and Cisco 7304 series.

IPC Terminology Fundamentals

The more common IPC terminologies are:

- **IPC** Inter-Process Communication.
- **IPC Address** A 32-bit word that is composed of a 16-bit seat ID and a 16-bit port ID.
- **IPC Client** A software module that uses IPC services.
- **IPC Port** A communication endpoint within IPC used as the source and destination of all communication.
- **IPC Seat** An IPC seat is a computational element, such as a processor, that can be communicated with the help of IPC. An IPC seat is where IPC clients and ports reside.
- **IPC Session** An IPC session is an active simplex communication channel between two IPC ports.

All communication that uses IPC happens between IPC ports. A port is a communication endpoint in IPC. Each IPC port is associated with a logical address called an IPC address. IPC uses the IPC address of an IPC port as a return address when it sends IPC messages, or a destination address when it receives IPC messages.

IPC Address Format

IPC addresses are assigned to IPC ports by the local IPC seat manager. A seat is the processor on which the IPC protocol is currently executing. A seat manager is a process that maintains a list of local IPC ports and a local name service, and also maintains open IPC communication sessions.

When an IPC port is created, the IPC client assigns a port name to the IPC port. Other IPC clients can then use a port name when they refer to the newly created IPC port. A port name is a string of characters that consists of a seat name and a port function or description.

Cisco IPC has three different levels of reliability on delivery to a port; this is defined when the port is opened.

- **Reliable:** The delivery of the message is guaranteed. Upon failure, delivery will be retried.
- **Unreliable:** The delivery is a best-effort attempt. There is no indication if the delivery fails.
- **Unreliable with Notification:** The delivery of the message is not guaranteed. However, the sender receives notification of the failure.

The **show ipc nodes** command displays the IPC seats present in a so-called IPC realm.

```
Router#show ipc nodes
      There are 3 nodes in this IPC realm.
      ID Type           Name                               Last Sent Last Heard
      10000 Local        IPC Master                          0       0
      1030000 RSP-CY      RSP IPC card slot 3                 7       7
      1000000 RSP-CY      RSP IPC card slot 0                 10      10
```

When a slave RP is present, the **show ipc nodes** command lists the slave RP address, as shown in this sample output from a Cisco 10000 Series Router:

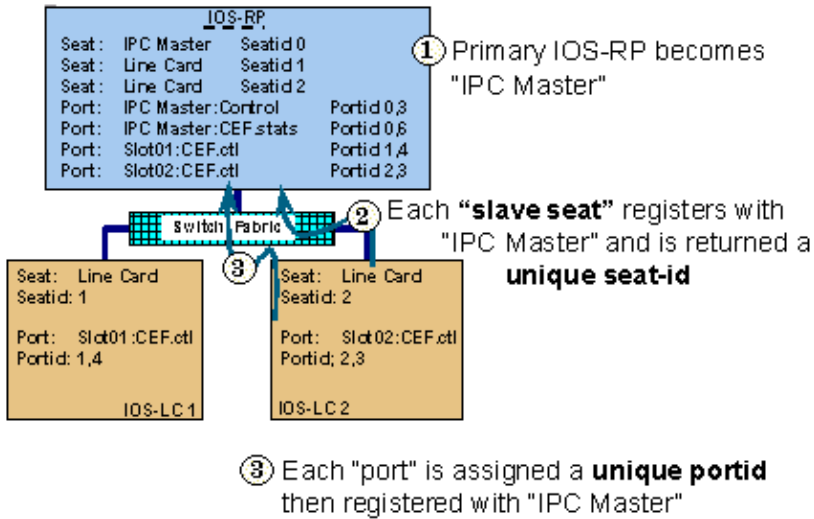
```
10k-2#show ipc nodes
      There are 5 nodes in this IPC realm.
      ID Type           Name                               Last Sent Last Heard
      10000 Local        IPC Master                          0       0
      20000 UDP           C10K Line Card slot 2/0             3       3
      30000 UDP           C10K Line Card slot 3/0             3       3
      40000 UDP           C10K Line Card slot 1/0             3       3
      50000 Ethernet Slave                            18      45
```

After it creates an IPC port, an IPC client may then register its port name with the global name service controlled by the IPC Master.

A collection of IPC seats, which communicate with each other, is called a zone. For each IPC zone, a single IPC seat is designated the IPC Zone Manager or Master, or IPC Master for short. Logically, all IPC seat connections in the IPC protocol are point-to-point connections. All IPC seat communication is typically between the active RP and a line card, or standby RP. Line card to line card communication is possible.

A device must create local ports and locate destination ports before it exchanges any IPC messages. Although a device creates local ports, these ports are not considered source ports because IPC communication is simplex. When the RP wants to communicate with an LC, it first opens a port on the LC (the LC needs to have created the port and registered it with the IPC Master – RP). When the open succeeds, normal IPC message traffic can start.

In the Cisco 12000 and 7500 Series, the route processor, either a Gigabit Route Processor (GRP) or a Route Switch Processor (RSP), and the intelligent line cards act as IPC endpoints. An "IPC Master" controls a group of processors. As the router initializes, the IPC Master discovers the IPC endpoints present on line cards in the system. To do so, the IPC Master scans all slots, identifies the controller type, and determines whether the controller has IPC capabilities.



Use the **show ipc ports** command to view these ports. On an IPC slave, this command lists the ports that have been created on that particular IPC seat. When issued on the IPC Master, this command displays the ports that have been created on the Master, and also the ports that have been registered by the IPC slaves (LCs). In addition, the **show ipc ports open** command lists the ports that have been opened from this IPC seat. Here is an example output:

```
router#show ipc ports
      There are 87 ports defined.
      Port ID Type Name
10000.1 unicast IPC Master:Zone
10000.2 unicast IPC Master:Echo
10000.3 unicast IPC Master:Control
10000.4 unicast IPC Master:Init
port_index = 0 seat_id = 0x1020000 last sent = 0 last heard = 1
port_index = 1 seat_id = 0x1010000 last sent = 0 last heard = 1
port_index = 2 seat_id = 0x1040000 last sent = 0 last heard = 1
port_index = 3 seat_id = 0x1050000 last sent = 0 last heard = 1
port_index = 4 seat_id = 0x1060000 last sent = 0 last heard = 1
port_index = 5 seat_id = 0x1070000 last sent = 0 last heard = 1
port_index = 6 seat_id = 0x1080000 last sent = 0 last heard = 1
port_index = 7 seat_id = 0x1090000 last sent = 0 last heard = 1
port_index = 8 seat_id = 0x10A0000 last sent = 0 last heard = 1
port_index = 9 seat_id = 0x10B0000 last sent = 0 last heard = 1
port_index = 10 seat_id = 0x1030000 last sent = 0 last heard = 1
10000.5 unicast Remote TTY Server Port
port_index = 0 seat_id = 0x1070000 last sent = 0 last heard = 2
port_index = 1 seat_id = 0x1010000 last sent = 0 last heard = 2
port_index = 3 seat_id = 0x1040000 last sent = 0 last heard = 2
port_index = 4 seat_id = 0x1050000 last sent = 0 last heard = 2
      Port ID Type Name
port_index = 5 seat_id = 0x1060000 last sent = 0 last heard = 3
port_index = 6 seat_id = 0x1080000 last sent = 0 last heard = 2
port_index = 7 seat_id = 0x1090000 last sent = 0 last heard = 2
port_index = 8 seat_id = 0x10A0000 last sent = 0 last heard = 2
port_index = 9 seat_id = 0x10B0000 last sent = 0 last heard = 2
      [output omitted]
```

The `port_index` field is the session ID used by the destination IPC when it processes incoming messages. When a slave RP is present, the **show ipc ports** command displays standby port information, as illustrated in this sample output:

```
10k-2#show ipc ports
      There are 16 ports defined.
      Port ID      Type      Name
```

```

10000.1      Unicast      IPC Master:Zone
10000.2      Unicast      IPC Master:Echo
10000.3      Unicast      IPC Master:Control
10000.4      Unicast      Microcode Server
10000.5      Unicast      RFS Server Port
10000.6      Unicast      Remote File System Server Port
10000.7      Unicast      Master : TTY Server Port
port_index = 0 seat_id = 0x50000 last sent = 0      last heard = 0
10000.8      Unicast      C10K Line Card API
port_index = 0 seat_id = 0x20000 last sent = 0      last heard = 58521
port_index = 1 seat_id = 0x30000 last sent = 0      last heard = 64235
port_index = 2 seat_id = 0x40000 last sent = 0      last heard = 13486
50000.3      Unicast      Slave IPC:Control
50000.9      Unicast      Secondary RFS Server Port
50000.A      Unicast      Secondary Old RFS Server Port
50000.8      Unicast      Slave : TTY Client Port
50000.7      Unicast      Secondary Services Port
50000.B      Unicast      IF-con server port
50000.C      Unicast      RF : Standby
50000.D      Unicast      CF : Standby

```

What Information Needs to be Transmitted Through IPC?

IPC messages are the basic unit of communication exchanged between IPC clients. During normal operation, the RP and the line cards interact frequently through IPC messages. A message includes a header, source and destination addressing information, and the message data.

In the IPC header, IPC defines several different message flags which alter the receive processing of an IPC message. Of the flags defined, four flags are related to the type of communication used (unreliable, unreliable with notification, reliable), another four are related to Remote Procedure Call (RPC) messaging, or internal control processing, and two are not used at all.

Here are a few IPC clients:

- Commands sent by the RP to query the line cards for information such as version, memory amounts, interface statistics, changes in interface status, and configuration data.
- Responses to the commands from the RP, which are sent from the line card to the RP. Examples of information contained in the IPC messages include timed statistics updates and windows messages that indicate how many more IPC messages the line card is able to queue.
- Events or messages generated asynchronously. Examples are reporting errors such as input errors, runts, and giants, as well as reporting statistics and other accounting information, such as byte and packet counts.
- Messages between an active and standby RP to checkpoint proper operation.
- Some Cisco IOS software processes need to exchange information between line cards and the route processor. These processes are considered IPC applications. Examples include Cisco Express Forwarding (CEF) and remote file systems to exchange images between Cisco 12000 Series route processors.

Table 1 lists the layers of the IPC protocol stack:

IPC Protocol Stack

IPC Applications

IPC Mechanism Itself

How IPC Messages Are Transmitted

Both the 7500 Series and the 12000 Series Routers allocate a special set of buffers to store IPC messages that are queued for transmission and are waiting for acknowledgment from the destination IPC port.

Cisco 7500 Series

The 7500 Series uses a special set of buffers in the system packet memory (MEMD). For more information on MEMD and the 7500 architecture, see [What Causes a "%RSP-3-RESTART: cbus complex"?](#) and [Understanding VIP CPU Running at 99% and Rx-Side Buffering](#).

On the 7500 Series, the IPC queues are in the processor memory. In some versions of Cisco IOS (see the sample output below), the aggregate IPC buffer space in processor memory can be tuned by the **ipc cache size** command. The MEMD holds some limited buffers that cannot be tuned. When an IPC message which is enqueued in processor memory is sent, and when there is some free space in MEMD, the IPC messages are "moved" from the processor memory to MEMD before they are sent to the LC.

Use the **show ipc queue** command to view the status of the IPC queues.

```
Router#show ipc queue
  There are 0 IPC messages waiting for acknowledgment in the transmit queue.
  There are 0 IPC messages waiting for a response.
  There are 0 IPC messages waiting for additional fragments.
  There are 0 IPC messages currently on the IPC inbound.
  There are 0 messages currently in use by the system.
```

Note: These queues are IPC-maintained software queues, and must not be confused with the QA-ASIC hardware queues of the 7500 Series.

Cisco 12000 Series

On the 12000 Series, the GRP sends IPC messages over the switch fabric. At boot up, the buffer-carving algorithm creates two sets of pools in the so-called tofab (receive side) and frfab (transmit side) memory. As shown in the sample output of the **show controller tofab queues** command (see below), the two sets are Non-IPC Free Queues and IPC Queues. For guidance on how to interpret the output, see [Cisco 12000 Series Internet Router: Frequently Asked Questions](#).

On the Cisco 12000 Series, the GRP allocates a certain number of message headers at initialization. There have been several modifications done to improve the memory allocation for these headers.

Cisco IOS Software Release 12.0(18)S/ST has increased the default number of message headers created at initialization from 1000 to 5000 on both the GRP and the LCs (see the output that follows). From release 12.0(23)S and later, the IPC header cache is allowed to grow dynamically. Hence, it no longer needs to be tuned manually.

The LCs maintain IPC message headers in Dynamic RAM (DRAM). In addition, the LCs set aside 100 buffers in the tofab and fromfab memory for IPC messages. With each transmitted IPC message, the LC must request an IPC message header from the cache, and then send a request to the frfab Buffer Management ASIC (BMA) for an IPC message buffer to be used to send the message to the GRP over the fabric.

```
LC-Slot1#show controllers tofab queues
Carve information for ToFab buffers
SDRAM size: 33554432 bytes, address: 30000000, carve base: 30029100
```

```

33386240 bytes carve size, 4 SDRAM bank(s), 8192 bytes SDRAM pagesize, 2 carve(s)
max buffer data size 9248 bytes, min buffer data size 80 bytes
40606/40606 buffers specified/carved
33249088/33249088 bytes sum buffer sizes specified/carved

```

```

Qnum      Head      Tail      #Qelem  LenThresh
-----

```

5 non-IPC free queues:

```

20254/20254 (buffers specified/carved), 49.87%, 80 byte data size
 1          17297   17296   20254   65535
12152/12152 (buffers specified/carved), 29.92%, 608 byte data size
 2          20548   20547   12152   65535
6076/6076 (buffers specified/carved), 14.96%, 1568 byte data size
 3          32507   38582   6076    65535
1215/1215 (buffers specified/carved), 2.99%, 4544 byte data size
 4          38583   39797   1215    65535
809/809 (buffers specified/carved), 1.99%, 9248 byte data size
 5          39798   40606   809     65535

```

IPC Queue:

```

100/100 (buffers specified/carved), 0.24%, 4112 byte data size
 30          72      71      100     65535

```

Raw Queue:

```

31          0      17302   0       65535

```

[output omitted]

Steps to Troubleshoot Issues, Known Bugs and Improvements

Step 1: Tune the IPC Cache

Note: See table 2 for a list of IOS versions that have the enhancements listed in this section.

Under rare conditions (for example, when a large amount of information needs to be exchanged between IPC clients), the IPC buffer cache can become depleted. Cisco IOS software uses these log messages to report this condition:

```

Oct 7 03:36:49: %RSP-3-RESTART: interface Serial0/0/4:1, not transmitting
Oct 7 03:39:51: %IPC_RSP_CBUS-3-NOBUF: No more IPC memd buffers to transmit
IPC message
Oct 7 03:40:09: %RSP-3-RESTART: interface Serial0/0/2:1, not transmitting
Oct 7 03:40:19: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/0,
changed state to down
Oct 7 03:40:19: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/1,
changed state to down
Oct 7 03:40:19: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/2,
changed state to down
Oct 7 03:40:19: %LINEPROTO-5-UPDOWN: Line protocol on InterfaceSerial0/1/3,
changed state to down
Oct 7 03:40:21: %IPC_RSP_CBUS-3-NOBUF: No more IPC memd buffers to transmit
IPC message
Oct 7 03:40:26: %FIB-3-FIBDISABLE: Fatal error, slot 0: IPC failure
Oct 7 03:40:26: %FIB-3-FIBDISABLE: Fatal error, slot 1: IPC failure
Oct 7 03:40:26: %FIB-3-FIBDISABLE: Fatal error, slot 4: IPC failure
Oct 7 03:40:26: %FIB-3-FIBDISABLE: Fatal error, slot 5: IPC failure
Oct 7 03:40:29: %LINEPROTO-5-UPDOWN: Line protocol on Interface

```

As the above output illustrates, the RP disables CEF on all line cards in this condition because it will no longer be able to update the CEF tables on the line cards with the help of IPC. Thus, FIBDISABLE messages are reported against all line cards.

To solve these kind of failures, the IPC cache on RP and IPC memory on line cards may need to be increased. Before you do so, use the **show ipc status** command to investigate whether the RP or the LC or both run out of IPC buffers. Take this output and examine it from both the RP and the LC.

Originally, the default number of buffers allocated for all systems with the help of IPC was 1000 cached message headers, which were shared among inbound and outbound messages. Based on the version of Cisco IOS software installed, the number of IPC-cached message headers is either static, dynamic, or can be tuned.

Here is the output of the **show ipc status** command from a router with the default 1000 message headers.

Note: Cisco IOS Software Release 12.2T and 12.2S introduce changes to the output of this command.

```
router#show ipc status
  IPC System Status:
  This processor is the IPC master server.
  1000 IPC message headers in cache
  4049362 messages in, 92615 out, 4048932 delivered to local port,
  352 acknowledgments received, 386 sent,
  0 NACKS received, 0 sent,
  15326 messages dropped on input, 154 messages dropped on output
  0 no local port, 110 destination unknown, 0 no transport
  0 missing callback or queue, 34 duplicate ACKs, 0 retries,
  0 message timeouts.
  0 ipc_output failures, 0 mtu failures,
  7707 msg alloc failed, 0 emer MSG alloc failed, 0 no origs for RPC replies
  0 pak alloc failed, 0 memd alloc failed
  0 no hwq, 0 failed opens, 0 hardware errors
```

The required amount of memory to be allocated depends on the type of card (RP or LC, RSP or VIP) on the platform, and the activity of the applications that need IPC (distributed CEF, for example).

From Cisco IOS Software Release 12.0(23)S, 12.2(18)S, and the new IOS trains 12.3 and 12.3T, the IPC message cache is managed dynamically rather than static allocation of the IPC cache. The proposed solution to the IPC message cache depletion problem due to bursty heavy IPC traffic has been to grow and shrink message cache dynamically. Upon initialization, the system allocates a platform specified default number of messages. When the number of free messages falls short of "minimum" buffers, it notifies the critical background process to grow the cache. This enables IPC to continue to grow the cache to meet the needs of its clients. If the recently allocated buffers are never used by IPC for a specified time frame, this process begins to shrink. The cache stops to shrink when it reaches the default size. This performance improvement was introduced in CSCdv57496. With the implementation of CSCdv57496, the **ipc cache <size>** command no longer works as it is done automatically. This is valid across all IPC platforms.

Important note: From Cisco IOS Software Release 12.3(5.5)T, the ability to manually tune the IPC cache has been removed. See CSCec17505 (registered customers only) for more information.

When you check the output of the **show ipc queue** command, here is what you must see:

```
c7500#show ipc queue

Message waiting for acknowledgement in Tx queue :      0
Maximum acknowledgement msg usage in Tx queue :      0

Message waiting for additional Fragments           :      0
Maximum message fragment usage                     :      0

There are 0 IPC messages waiting for a response.
There are 0 IPC messages currently on the IPC inboundQ.
```

```

Messages currently in use           :           0
Message cache size                 :          1000
Maximum message cache usage        :          1344
0 times message cache crossed 5000 [max]

Emergency messages currently in use :           0

Inbound message queue depth 0
Zone inbound message queue depth 0

```

If the router runs a Cisco IOS software version that does not include dynamically–managed IPC cache buffers, that is, images prior to 12.0(23)S, 12.2(18)S, 12.3 and 12.3T, the IPC cache on the RP and the IPC memory on line cards can be manually increased. Before you do so, use the **show ipc status** command to investigate whether the RP, LC, or both, are running out of IPC buffers. Take this output and examine it from both the RP and the LC.

If necessary, you can use these commands to tune the memories:

- The **ipc cache 5000** configuration command to increase the IPC header cache on the RP.
- The **ipc cache <size> [slot {slot_num / all}]** command to increase the cache on the Cisco 12000 LC.

Note: When you allocate more memory for IPC messages, less memory is available for other processes. The size of a single IPC message actually varies with different Cisco IOS software branches. Use the **show memory summary** command to check whether there is enough free memory in the Processor pool.

Step 2: Tune the IPC Throughput

Note: See table 2 for a list of IOS versions that have the enhancements listed in this section.

In some situations, you may want to also tune the IPC throughput between RP and LC. This is especially the case when the RP needs to upload a large CEF table to the LC. For example, this could happen while the router boots, when it receives a large amount of routing information from a BGP peer. You can configure extra IPC buffering on the LC with the **ip cef linecard ipc memory xxxxx** command to increase the IPC bandwidth. This command was introduced by CSCds89515 (registered customers only) . The value for this memory has been set to an acceptable default with CSCdu54205 (registered customers only) and CSCuk27162 (registered customers only) .

Here are the commands that indicate the result when you change this parameter:

```

Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#ip cef line ipc mem 20000
Router(config)#^Z
Router#show cef state
...
RP state:
  Expanded LC ipc memory:           20000 Kbytes
...
or, alternatively:
Router#show cef line
Slot      MsgSent   XDRSent  Window  LowQ   MedQ   HighQ  Flags
0         12515     21687    505     0      0      0      up
1         12515     21675    505     0      0      0      up
3         12515     21701    505     0      0      0      up
5         12515     21700    505     0      0      0      up
2         12518     22008    505     0      0      0      up
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#ip cef line ipc mem 20000

```

```

Router(config)#^Z
Router#show cef line
Slot      MsgSent    XDRSent   Window   LowQ    MedQ    HighQ  Flags
0         12538     22097     4966     0       0       0      up
1         12538     22081     4966     0       0       0      up
3         12538     22115     4966     0       0       0      up
5         12538     22114     4966     0       0       0      up
2         12541     22418     4966     0       0       0      up

```

List of IPC Enhancements

Table 2 provides an overview of enhancements implemented in Cisco IOS software to manually and dynamically tune IPC memory across different platforms.

Cisco Bug ID

Fixed In

Enhancement

CSCdk75315 (registered customers only)

12.0(5)S

12.0(5)

12.0(5)T

11.3(10)AA

Introduces an IPC cache size that can be configured with the help of the **ipc cache <size>** command.

CSCds89515 (registered customers only)

12.2(4)B

12.1(9)E

12.1(8a)E

12.2(3)T

12.2(2)S

12.1(9)

12.0(14)ST1

12.2(2)

12.2(1)T

12.0(15)S3

12.0(16)ST

12.0(16)S

On a Cisco 12000 Series Internet Router, distributed Cisco Express Forwarding (dCEF) can be disabled because of a low-memory condition during a large routing update (for example, while booting up).

As a workaround, reduce the maximum path in Border Gateway Protocol (BGP) to reduce the amount of information CEF propagates to the line cards. Alternatively, reduce the TCP window size to reduce the speed of incoming BGP updates. See *Achieve Optimal Routing and Reduce BGP Memory Consumption*.

Alternatively, you can also enter the **ip cef linecard ipc memory 0–128000** interface configuration command. The amount of line card processor or main memory is limited to 50 percent of the total memory. This command allows you to allocate a larger amount of line card processor memory to the queueing for CEF routing to update messages. It allows the RP to release CEF updates more quickly to free up memory, and it prevents the occurrence of low-memory condition on the RP. Based on the number of Versatile Interface Processors (VIPs), dCEF needs a large amount of temporary memory on the RSP to buffer IPC messages bound to the VIP, especially in cases when large BGP peers come up or when the Forwarding Information Base (FIB) gets propagated to the VIP after a CBUS complex or a VIP crash (or when the **clear cef line** command is issued).

CSCdu21591 (registered customers only)

12.0(17)ST4

12.0(18)ST

12.0(18)S

Increases the default IPC message header cache size from 1000 to 5000 on 12000 Series Routers. Earlier, the parser accepted any number between the hard-coded values of 1000 and 15000. Today, the parser only accepts numbers between the platform-defined minimum and maximum cache size.

In addition, originally, it was not possible to clear the **ipc cache** command from the configuration even if you executed the **no ipc cache** command in the configuration to remove a custom IPC cache value. Instead, it inserted an **ipc cache x** command, where *x* is the currently defined default cache size. Today, the **no ipc cache** command has the expected behavior. It completely removes the **ipc cache** command from the configuration.

CSCdu12540

12.0(19)ST

12.0(19)S

Only applicable to the Cisco 12000 Series:

Originally, the **ipc cache <size>** command worked only for the RP IPC cache. Now, the **ipc cache** command can be used on LCs as follows:

```
ipc cache <size> [slot {slot_num | all}]
```

The options `slot_num` and `all` are not mutually exclusive. For example, these commands are valid:

```
ipc cache 4000 slot all
```

ipc cache 3000 slot 5

These commands increase the cache size in slot 5 to 3000 and to 4000 for all other slots. If you want to use the `all` option to overwrite previous cache size configuration statements for LCs, ensure that you also use "NOPREFIX" to delete the previous commands in Nonvolatile RAM (NVRAM), and implement the correct results.

In noprefix mode, use the **no ipc cache slot** `{slot_num / all}` command to reset the cache size to its default value.

CSCdu54205

12.0(19)ST

12.0(19)S

Only applicable to the Cisco 12000 Series:

This enhancement changed the default value for line card CEF update memory allocation to 512 messages. It is no longer necessary to use the **ip cef linecard ipc memory** `xxxxx` command unless the problem is observed.

CSCuk27162 (registered customers only)

12.2(9)T

12.2(9)S

12.2(9)

12.0(21)ST

12.0(22)S

This software enhancement changes the default per-platform number of line card ipc buffers allocated at bootup.

It also increases the RSP per-platform default line card IPC memory from 25 to 128 IPC messages.

Workaround: Use the **ip cef linecard ipc memory** `xxxxx` global configuration command to increase the number of buffers on the line cards.

CSCdv57496

12.0(23)S

Manage IPC message cache dynamically instead of static allocation of the IPC cache.

With the implementation of CSCdv57496, the **ipc cache** `<size>` command is no longer valid as this is done automatically. This is valid across all IPC platforms.

CSCdz77490

12.2(19.7)S

12.0(26.2)S

12.3(1)B

12.3(1)

With the implementation of CSCdz77490, the **ipc cache <size>** command line interface is removed from Cisco IOS software trains 12.3 and 12.3T.

In the Cisco IOS 12.3 train, this command is hidden, but, if configured from the terminal, it prints a message to the user.

In the next major release 12.4, this command will be removed.

CSCec17505 (registered customers only)

TBD

Symptoms: The ipc cache size does not change when you use the **ipc cache <size>** CLI command to change the cache size.

Conditions: This condition occurs as a result of architectural changes with IPC.

Workaround: The IPC cache functionality is now done automatically, and cannot be changed by the user on the CLI. This enhancement removes the **ipc cache <size>** CLI command in the Cisco IOS software versions that no longer allow the user to manually change the IPC cache.

There should be no backward-compatibility issues as the CLI will still exist in versions where the user can manually change the IPC cache with the **ipc cache <size>** CLI command.

Cisco 7600 Series

When running Catalyst OS, the Catalyst 6000 / Cisco 7600 Series uses a Supervisor Engine with an optional router card known as the Multilayer Switch Feature Card (MSFC). The CPU on the Supervisor and the CPU on the MSFC communicate through IPC messages across an Ethernet out-of-band management bus. When running Cisco IOS System Software, the RP and the switch processor (SP) also communicate through IPC messages. Originally, 3000 buffers were created for IPC messages. In rare cases, the system runs out of IPC buffers and reports these error messages:

```
01:52:13: %ICC-2-NOMEM: No memory available for unregistering card Card2
02:42:08: %IPC-3-NOBUFF: The main IPC message header cache has emptied
-Traceback= 4026779C 40268350 4025F930 40223D34 40221C40 40221EA4 401EAB10
```

Note: ICC stands for InterCard Communications.

From Cisco IOS Software Releases 12.1(08a)E01 and 12.1(10)E, the Cisco 7600 Series now creates 6000 IPC message buffers by default. In addition, changes made in versions 12.1(08a)E and 12.1(09)EC help to avoid IPC header depletion that results from a large number of Virtual LAN (VLAN)-related updates. Each ICC message advertises a group of VLAN link-state changes, rather than one VLAN at a time.

Newer line cards for the Cisco 7600 Series support a distributed feature daughter card (DFC) for high-speed packet handling rates. The DFC-enabled line cards maintain local Cisco Express Forwarding and adjacency tables and communicate with the Supervisor using IPC messages.

Some IPC messages are greater than the maximum transmission unit (MTU) of the Catalyst 6000 switching bus (for example, IPC messages used to report SONET interface statistics in messages larger than 1500 bytes). Such messages need to be fragmented. Under rare conditions, the IPC fragment header cache is depleted, and the system reports this error message:

```
%IPC-DFC6-3-NOBUFF: The fragment IPC message header cache has emptied
```

Changes made in Cisco IOS Software Releases 12.1(08a)E and 12.1(09.05)EC increase the number of IPC fragment buffer headers from 32 to 128.

This message might appear in the debug output if duplicate acknowledgements are received by the IPC client.

IPC: Cannot find original message for ACK HDR:

Duplicate acknowledgements are most commonly due to media problems that cause the acknowledgement messages to get lost. In order to resolve this acknowledgement loss, reseal or replace of the line card in the slots correctly to avoid media problems.

Collect Troubleshooting Information for the Cisco TAC

If you still need assistance after you follow the troubleshooting steps above and want to create a service request with the Cisco TAC, be sure to include the following information for troubleshooting IPC-3-NOBUFF-related error messages:

- Troubleshooting performed before you opened the case.
- **show technical-support** output (if possible, in enable mode).
- **show log** output or console captures, if available.

Please attach the collected data to your case in non-zipped, plain text format (.txt). You can attach information to your case. To do so, upload it with the help of the Case Query Tool (registered customers only). If you cannot access the Case Query Tool, you can attach the relevant information to your case by sending it to attach@cisco.com with your case number in the subject line of your message.

Note: Please do not manually reload or power-cycle the router before collecting the above information unless required to troubleshoot an IPC-3-NOBUFF exception, as this can cause important information that is needed for determining the root cause of the problem to be lost.

Related Information

- [What Causes a "%RSP-3-RESTART: cbus complex"?](#)
- [Displaying CPU Hog Information for IPC Processes](#)
- [Cisco 12000 Series Internet Router: Frequently Asked Questions](#)
- [Achieve Optimal Routing and Reduce BGP Memory Consumption](#)
- [Cisco 12000 Series Internet Router Technical Support Page](#)
- [Cisco Routers Product Support Page](#)

• **Technical Support – Cisco Systems**

[Contacts & Feedback](#) | [Help](#) | [Site Map](#)

© 2008 – 2009 Cisco Systems, Inc. All rights reserved. [Terms & Conditions](#) | [Privacy Statement](#) | [Cookie Policy](#) | [Trademarks of Cisco Systems, Inc.](#)

Updated: Jun 24, 2008

Document ID: 21165
