Understanding VIP CPU Running at 99% and Rx−Side Buffering

Document ID: 12810

Contents

Introduction

This document explains why the Versatile Interface Processor (VIP) CPU runs at 99%, and what are Rx−side buffers.

Prerequisites

Requirements

There are no specific requirements for this document.

Components Used

This document is not restricted to specific software and hardware versions.

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

For more information on document conventions, refer to the Cisco Technical Tips Conventions.

Background Information

Rx−side buffering is the process that occurs when the outbound interface:

- is congested.
- uses the First in, First out (FIFO) queueing strategy.

The inbound Versatile Interface Processor (VIP) does not drop the packet immediately. Instead, it buffers the packet in its packet memory until buffers are available for the outgoing interface. Based on the type of VIP, the packet memory can be Static RAM (SRAM) or Synchronous Dynamic RAM (SDRAM).

Cisco 7500 Series Architecture Basics

Each interface processor (legacy IP or VIP) has one connection to a high−speed extended system bus called a CyBus. Route/Switch Processors (RSPs) are connected to two CyBuses (see Figure 1).

Figure 1   Cisco 7500 Series Architecture
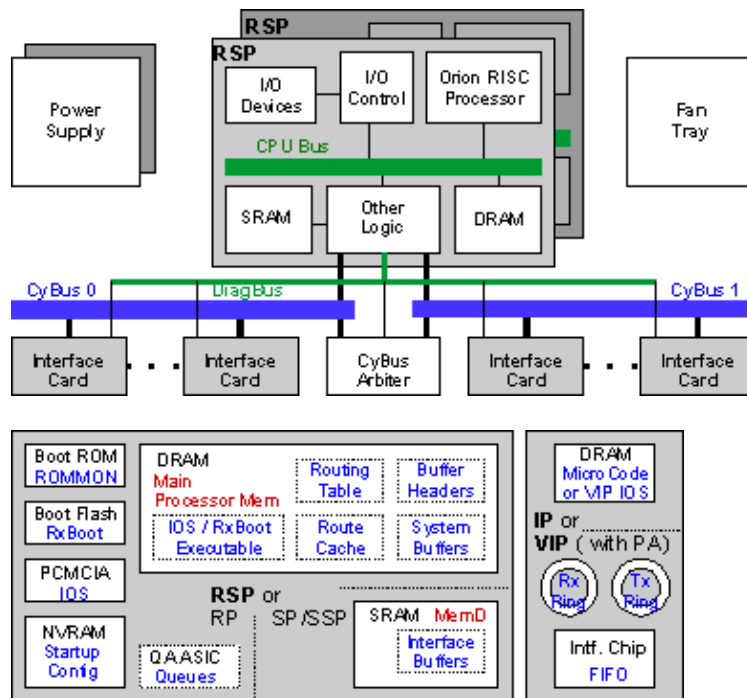
Types of Packet Buffers

This section describes the various types of packet buffers.

- **System buffers in processor memory on the RSP**

  These buffers are used for process−switched packets. You can see these buffers in the output of the **show interfaces** (input and output queues) and **show buffers** commands. A Cisco 7500 Series router must not do much process−switching. Therefore, if you have problems with system buffers, it means that too many packets are sent to the process level. This can be due to many factors, such as:

  - A broadcast storm
  - Instability in the network that causes routing updates
  - A "denial of service" (DoS) attack
  - A feature that is not supported in the fast−switching path (for example, X.25)
  - IP packets with options.

  For information on how to troubleshoot excessive process switching, refer to these documents:

  - Troubleshooting High CPU Utilization on Cisco Routers
  - Troubleshooting Input Queue Drops and Output Queue Drops
- **Packet memory on RSP (MEMD) buffers**

MEMD size is fixed at 2 MB on the RSP7000, RSP1, RSP2, and RSP4. On the RSP8 and RSP16, MEMD size is 8 MB. MEMD is distributed between all interfaces at the time of bootup, when there is an online insertion and removal (OIR), a microcode reload, a maximum transmission unit (MTU) change, or a cbus complex. For more information on cbus complex, refer to What Causes a "%RSP−3−RESTART: cbus complex"?. You can use the **show controllers cbus** command to check the status of MEMD buffers.

When MEMD is allocated, these structures are created:

- A **local free queue (lfreeq)** It is assigned to each interface, and is used for packets received on this interface.
- A **global free queue (gfreeq)** It is also allocated, and an interface can fall back to that queue within some limits.
- A **transmit queue (txqueue or txq)** It is assigned to each interface, and is used for packets that go out through this interface.
- The **transmit accumulator (txacc)** It represents the number of elements on the output interface transmit queue (txqueue). When the transmit accumulator (txacc) equals the transmit limit (txlimit), all buffers are freed. When the txacc is 0, the queue is full, and no more queueing is allowed.

- **Packet memory**

On a VIP, packet memory contains packet buffers (particles) used for packets received from or sent to a VIP interface. Figure 2 represents packet flow.

Figure 2   Packet Flow
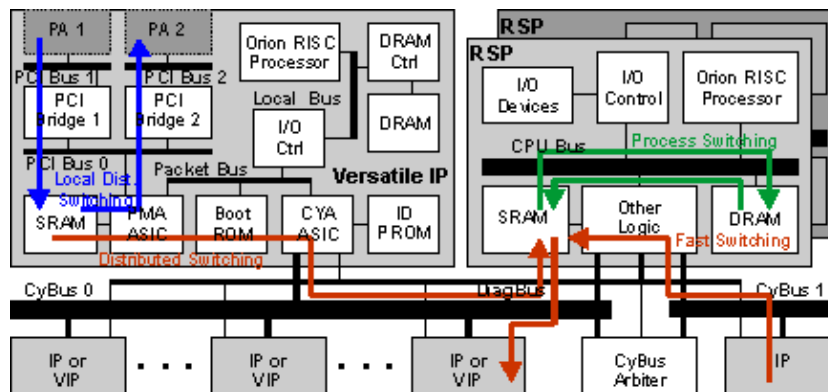
This section focusses on a VIP where distributed switching is enabled, because Rx−side buffering usually occurs when a packet follows this type of switching path. Different scenarios are possible, which are explained here:

Scenario 1: When there is no congestion on the outbound interface.

- A packet is received on a port adapter (PA) and moved into a packet buffer in the packet memory.
- If the VIP cannot distribute−switch the packet, it forwards the packet to the RSP, which makes the switching decision.
- If the VIP can make the switching decision and the outgoing interface is on the same VIP, the packet is sent out on the outbound interface. The packet is said to be "locally switched" on the VIP, because it does not cross the cbus.
- If the VIP can make the switching decision and the outgoing interface is in another slot, the VIP tries to copy the packet over the cbus into the txqueue (in MEMD) of the outbound interface.

♦ The packet is then copied to the outgoing (V)IP over the cbus and sent out on the line.
Scenario 2: When the outbound interface is congested.

There are two possibilities:

♦ If queueing is configured on the outgoing interface, the VIP transfers the packet into the txqueue in MEMD, and the packet is pulled immediately from the queue by the queueing code.

◊ If RSP−based queueing is configured, the packet is copied into the system buffers in processor memory on the RSP.
◊ If VIP−based queueing is used, the packet is copied into the packet memory of the outbound VIP.

♦ If the queueing strategy of the outbound interface is FIFO, the interface does not drop the packet immediately (this is what normally happens with FIFO when an outbound interface is congested). Instead, the inbound VIP buffers the packet in its packet memory until some buffers are again available for the outgoing interface. This is called Rx−side buffering.

Use the **show controllers vip accumulator** command to check the status of Rx−side buffering. The status indicates:

• The number of output interfaces present in the router.
• How many packets the VIP has Rx−buffered for these interfaces.
• Why the VIP has Rx−buffered.
• How many packets the VIP dropped, and why.

VIP Runs at 99% CPU Utilization

A consequence of Rx−side buffering is that the VIP runs at 99% CPU utilization. The VIP continuously monitors the status of the txqueue of the outbound interface and, as soon as there is a free buffer, it copies the packet over the cbus into the txqueue.

There is nothing alarming in itself when the VIP runs at 99% when Rx−buffering occurs. It does not mean that the VIP is overloaded. If the VIP receives something more important to do (for example, another packet to switch), this is not affected by the high CPU.

Here is a simple test you can do in the lab to illustrate this:

Serial 2/0/0 has a clock rate of 128 Kbps, and receives traffic at line rate. The traffic is switched to Serial 10/0 where the clock rate is 64 Kbps, and the queueing strategy is FIFO. The only option is to drop packets.

```
router#show controller cbus

MEMD at 40000000, 8388608 bytes (unused 697376, recarves 6, lost 0)

RawQ 48000100, ReturnQ 48000108, EventQ 48000110

BufhdrQ 48000130 (21 items), LovltrQ 48000148 (15 items, 2016 bytes)

IpcbufQ 48000158 (24 items, 4096 bytes)

IpcbufQ_classic 48000150 (8 items, 4096 bytes)

3570 buffer headers (48002000 − 4800FF10)

pool0: 8 buffers, 256 bytes, queue 48000138
```

```
pool1: 2940 buffers, 1536 bytes, queue 48000140

pool2: 550 buffers, 4512 bytes, queue 48000160

pool3: 4 buffers, 4544 bytes, queue 48000168

slot2: VIP2, hw 2.11, sw 22.20, ccb 5800FF40, cmdq 48000090, vps 8192

software loaded from system

IOS (tm) VIP Software (SVIP-DW-M), Version 12.0(21)S, EARLY DEPLOYMENT RELEASE
SOFTWARE (fc1)

ROM Monitor version 122.0

Mx Serial(4), HW Revision 0x3, FW Revision 1.45

Serial2/0/0, applique is V.35 DCE

received clockrate 2015232

gfreeq 48000140, lfreeq 480001D0 (1536 bytes)

rxlo 4, rxhi 336, rxcurr 16, maxrxcurr 293

txq 48001A00, txacc 48001A02 (value 294), txlimit 294

Serial2/0/1, applique is V.35 DTE

received clockrate 246

gfreeq 48000140, lfreeq 480001D8 (1536 bytes)

rxlo 4, rxhi 336, rxcurr 0, maxrxcurr 0

txq 48001A08, txacc 48001A0A (value 6), txlimit 6

Serial2/0/2, applique is Universal (cable unattached)

received clockrate 246

gfreeq 48000140, lfreeq 480001E0 (1536 bytes)

rxlo 4, rxhi 336, rxcurr 0, maxrxcurr 0

txq 48001A10, txacc 48001A12 (value 6), txlimit 6

Serial2/0/3, applique is Universal (cable unattached)

received clockrate 246

gfreeq 48000140, lfreeq 480001E8 (1536 bytes)

rxlo 4, rxhi 336, rxcurr 0, maxrxcurr 0

txq 48001A18, txacc 48001A1A (value 6), txlimit 6

slot10: FSIP, hw 1.12, sw 20.09, ccb 5800FFC0, cmdq 480000D0, vps 8192

software loaded from system
```

Serial10/0, applique is V.35 DTE

gfreeq 48000140, lfreeq 48000208 (1536 bytes)

rxlo 4, rxhi 336, rxcurr 1, maxrxcurr 1

```
txq 48000210, txacc 480000B2 (value 2), txlimit 294

Serial10/1, applique is Universal (cable unattached)

gfreeq 48000140, lfreeq 48000218 (1536 bytes)

rxlo 4, rxhi 336, rxcurr 0, maxrxcurr 0

txq 48000220, txacc 480000BA (value 6), txlimit 6

Serial10/2, applique is Universal (cable unattached)

gfreeq 48000140, lfreeq 48000228 (1536 bytes)

rxlo 4, rxhi 336, rxcurr 0, maxrxcurr 0

txq 48000230, txacc 480000C2 (value 6), txlimit 6

Serial10/3, applique is Universal (cable unattached)

gfreeq 48000140, lfreeq 48000238 (1536 bytes)

rxlo 4, rxhi 336, rxcurr 0, maxrxcurr 0

txq 48000240, txacc 480000CA (value 6), txlimit 6

Serial10/4, applique is Universal (cable unattached)

gfreeq 48000140, lfreeq 48000248 (1536 bytes)

rxlo 4, rxhi 336, rxcurr 0, maxrxcurr 0

txq 48000250, txacc 480000D2 (value 6), txlimit 6

Serial10/5, applique is Universal (cable unattached)

gfreeq 48000140, lfreeq 48000258 (1536 bytes)

rxlo 4, rxhi 336, rxcurr 0, maxrxcurr 0

txq 48000260, txacc 480000DA (value 6), txlimit 6

Serial10/6, applique is Universal (cable unattached)

gfreeq 48000140, lfreeq 48000268 (1536 bytes)

rxlo 4, rxhi 336, rxcurr 0, maxrxcurr 0

txq 48000270, txacc 480000E2 (value 6), txlimit 6

Serial10/7, applique is Universal (cable unattached)

gfreeq 48000140, lfreeq 48000278 (1536 bytes)

rxlo 4, rxhi 336, rxcurr 0, maxrxcurr 0

txq 48000280, txacc 480000EA (value 6), txlimit 6

router#
```

The **value 2** means that only two buffers are left. Rx–buffering does not queue packets in MEMD when the txacc is less than 4.

The **show controllers vip 2 tech–support** command from the VIP shows that it runs at 99% CPU:

```
router#show controllers vip 2 tech-support

show tech-support from Slot 2:

----------------- show version -----------------


Cisco Internetwork Operating System Software

IOS (tm) VIP Software (SVIP-DW-M), Version 12.0(21)S, EARLY DEPLOYMENT RELEASE
SOFTWARE (fc1)

Copyright (c) 1986-2000 by cisco Systems, Inc.

Compiled Tue 18-Jul-00 22:03 by htseng

Image text-base: 0x600108F0, data-base: 0x602E0000


ROM: System Bootstrap, Version 11.1(4934) [pgreenfi 122], INTERIM SOFTWARE


VIP-Slot2 uptime is 1 week, 23 hours, 27 minutes

System returned to ROM by power-on

Running default software



cisco VIP2 (R4700) processor (revision 0x02) with 32768K bytes of memory.

Processor board ID 00000000

R4700 CPU at 100Mhz, Implementation 33, Rev 1.0, 512KB L2 Cache

4 Serial network interface(s)

Configuration register is 0x0

...

----------------- show process cpu -----------------

CPU utilization for five seconds: 99%/97%; one minute: 70%; five minutes: 69%
```

The VIP runs at 99% CPU utilization even though it receives only 128 Kbps. This shows that CPU utilization is not linked to the number of packets per second. This is because a VIP 2 is able to switch many more packets than this. It is simply a sign of Rx−side buffering.

In order to check what Rx−side buffering does, execute these commands:

```
router#show controllers vip 2 accumulator

show vip accumulator from Slot 2:

Buffered RX packets by accumulator:
...
Serial10/0:
 MEMD txacc 0x00B2: 544980 in, 2644182 drops (126 paks, 378/376/376 bufs) 1544kbps
   No MEMD acc: 544980 in
      Limit drops  : 2644102 normal pak drops, 80 high prec pak drops
      Buffer drops : 0 normal pak drops, 0 high prec pak drops
   No MEMD buf: 0 in
      Limit drops  : 0 normal pak drops, 0 high prec pak drops
```

```
       Buffer drops : 0 normal pak drops, 0 high prec pak drops
...
Interface x:
MEMD txacc a: b in, c drops (d paks, e/f/g bufs) h kbps
No MEMD acc: i in
       Limit drops  : j normal pak drops, k high prec pak drops
       Buffer drops : l normal pak drops, m high prec pak drops
No MEMD buf: n in
       Limit drops  : o normal pak drops, p high prec pak drops
       Buffer drops : q normal pak drops, r high prec pak drops
```

Key	Description
a	Address of the txacc in MEMD. There is one Rx−side buffer queue for each txacc in the system (up to 4096).
b	Number of packets that are Rx−buffered.
c	Number of packets that the VIP dropped. If there are enough packet memory buffers, the VIP can Rx−buffer up to one second of traffic. However, if the interface is continuously congested, it is not possible to avoid drops.
d	Number of packets currently Rx−buffered.
e	Number of particles currently Rx−buffered. A packet can be made of multiple particles.
f	Soft limit, which is the maximum number of particles when the VIP memory is low.
g	Hard limit, which is the maximum number of particles that can be used at any time.
h	The speed of the outgoing interface in kbps.
i	The number of Rx−buffered packets because no txacc was available in MEMD. This means that the output queue was congested (no more free buffers are available in the tx−queue). The solution to this problem is to increase the output interface bandwidth (if possible).
j	The number of packets with IP precedence other than 6 or 7 that could not be sent because there is no MEMD acc, and are dropped because the soft or hard limit of particles has been reached.
k	Same as **j**, but for packets with IP precedence 6 or 7 (internetwork and network).
l	The number of packets with IP precedence other than 6 or 7 that the VIP wants to Rx−buffer, but drops due to a lack of free buffers in the packet memory. From Cisco IOS Software Releases 12.0(13)S and 12.1(4) onwards, you can also use the **show controller vip [***all / slot#***] packet−memory−drops** command to see the number of packets dropped. In this case, an upgrade of the packet memory helps.
m	Same as **l**, but for packets with IP precedence 6 or 7 (internetwork and network).
n	

	The number of packets the VIP tries to Rx−buffer because there is no MEMD buffer, but cannot do so due to a lack of packet memory buffers. Upgrade the packet memory in this case. From Cisco IOS Software Releases 12.0(13)S and 12.1(4) onwards, you can also use the **show controllers vip [***all / slot#***] packet−memory−drops** command to understand why the packets have been dropped.
o	The number of Rx−buffered packets with IP precedence other than 6 or 7 with no MEMD buffer that are dropped because the soft (**f**) or hard (**g**) limit has been reached. In this situation, an RSP16 helps because it has more MEMD memory (8 MB compared to 2 MB for the RSP1, RSP2, RSP4, and RSP7000). You can also reduce the MTU of some interfaces (such as ATM, POS, or FDDI) in this case. These interfaces usually have a 4470−byte MTU, and fewer MEMD buffers can be allocated because the buffers have to be larger.
p	Same as **o**, but for packets with IP precedence 6 or 7 (internetwork and network).
q	The number of packets with IP precedence other than 6 or 7 that the VIP tries to Rx−buffer because there is no MEMD buffer, but cannot do so due to a lack of packet memory buffers. An upgrade of the packet memory helps in this case. From Cisco IOS Software Releases 12.0(13)S and 12.1(4) onwards, you can also use the **show controllers vip [***all / slot#***] packet−memory−drops** command to better understand why the packets have been dropped.
r	Same as **q**, but for packets with IP precedence 6 or 7 (internetwork and network).

If the router runs a Cisco IOS software version earlier than 12.0(13)ST, 12.1(04)DB, 12.1(04)DC, 12.0(13)S, 12.1(4) 12.1(4)AA 12.1(4)T 012.0(13), or 12.0(13)SC, the output of **show controllers vip [***all / slot#***] accumulator** provides a simplified version of the above. It does not take into account the different IP precedences of the dropped packets because of Rx−side buffering.

The output looks like this:

```
Serial10/0:
MEMD txacc 0x00B2: 544980 in, 2644182 drops (126 paks, 378/376/376 bufs) 1544kbps
No MEMD acc: 544980 in, 2644182 limit drops, 0 no buffer
No MEMD buf: 0 in, 0 limit drops, 0 no buffer


Interface x:
MEMD txacc a: b in, c drops (d paks, e/f/g bufs) h kbps
No MEMD acc: i in, j+k limit drops, l+m no buffer
No MEMD buf: n in, o+p limit drops, q+r no buffer
```

Examples of Buffers on Rx−side

Example 1: The VIP in slot 2 receives traffic on a 128Kbps and routes it to serial 10/0 (64Kbps).

```
Serial10/0:
 MEMD txacc 0x00B2: 544980 in, 2644182 drops (126 paks, 378/376/376 bufs) 1544kbps
   No MEMD acc: 544980 in
      Limit drops : 2644102 normal pak drops, 80 high prec pak drops
      Buffer drops : 0 normal pak drops, 0 high prec pak drops
   No MEMD buf: 0 in
      Limit drops : 0 normal pak drops, 0 high prec pak drops
      Buffer drops : 0 normal pak drops, 0 high prec pak drops
```

- Here, 544980 packets are successfully Rx−buffered and 2644182 are dropped. 80 packets out of the 2644182 that are dropped had an IP precedence of 6 or 7.
- 126 packets are currently Rx−buffered and they use 378 particles.
- All the packets are Rx−buffered because of a lack of free buffer in the tx−queue in MEMD. This means that the output interface is congested. The drops occur because the maximum number of Rx−buffered packets is reached. The typical solution is to upgrade the outbound interface bandwidth, re−route some traffic so that the outbound interface is less congested, or enable some queueing to drop the less important traffic.

Example 2: Rx−side Buffers without Drops.

```
ATM1/0:
MEMD txacc 0x0082: 203504 in, 0 drops (0 paks, 0/81/37968 bufs) 155520kbps
No MEMD acc: 85709 in
      Limit drops  : 0 normal pak drops, 0 high prec pak drops
      Buffer drops : 0 normal pak drops, 0 high prec pak drops
No MEMD buf: 117795 in
      Limit drops  : 0 normal pak drops, 0 high prec pak drops
      Buffer drops : 0 normal pak drops, 0 high prec pak drops
```

- In this example, 85709 packets are Rx−buffered because ATM 1/0 is congested but no packets are dropped.
- 117795 packets are Rx−buffered because the VIP cannot get an MEMD buffer. No packets are dropped. A typical solution is to decrease some of the MTUs so that more MEMD buffers can be allocated. An RSP8 also helps.

Example 3: Local Switching.

```
SRP0/0/0:

 local txacc 0x1A02: 2529 in, 0 drops (29 paks, 32/322/151855 bufs) 622000kbps
```

Local txacc means that this output interface is on the same VIP as the interface where the packets are received. These packets are switched locally, but the outbound interface (in this case, srp 0/0/0) is congested. 2529 packets are Rx−buffered, and no packets are dropped.

Example 4: Forward Queues.

```
router#show controllers vip 2 accumulator
Buffered RX packets by accumulator:
 Forward queue 0 : 142041 in, 3 drops (0 paks, 0/24414/24414 bufs) 100000kbps
   No MEMD buf: 142041 in
      Limit drops  : 0 normal pak drops, 0 high prec pak drops
      Buffer drops : 3 normal pak drops, 0 high prec pak drops
 Forward queue 9 : 68 in, 0 drops (0 paks, 0/15/484 bufs) 1984kbps
   No MEMD buf: 68 in
```

```
        Limit drops  : 0 normal pak drops, 0 high prec pak drops
        Buffer drops : 0 normal pak drops, 0 high prec pak drops
 Forward queue 13: 414 in, 0 drops (0 paks, 0/14/468 bufs) 1920kbps
   No MEMD buf: 414 in
        Limit drops  : 0 normal pak drops, 0 high prec pak drops
        Buffer drops : 0 normal pak drops, 0 high prec pak drops
 Forward queue 14: 46 in, 0 drops (0 paks, 0/14/468 bufs) 1920kbps
   No MEMD buf: 46 in
        Limit drops  : 0 normal pak drops, 0 high prec pak drops
        Buffer drops : 0 normal pak drops, 0 high prec pak drops
```

Some packets cannot be distribute−switched. In this case, the VIP has to forward the packets to the raw queue of the RSP, which then makes the switching decision. When packets cannot be copied immediately into MEMD, the VIP Rx−buffers them and keeps track of how many packets are Rx−buffered per inbound interface.

Forward queues 0−7 are for the first port adapter (PA) and 8−15 for the second PA.

Forward Queue Number	...shows the number of Rx−buffered packets that are received on the...
0	first plughole of the first port adapter (PA)
8	first plughole of the second PA
9	second plughole of the second PA

Other Causes of High CPU Utilization on VIPs

When Rx−side buffering is found to be inactive, one of these factors can cause high CPU utilization on the VIP:

- **99% CPU utilization on VIP, caused by distributed Traffic Shaping**

  When distributed Traffic Shaping (dTS) is configured, the VIP CPU jumps to 99% as soon as one packet enters the dTS queue.

  This is the correct and expected behavior. When dTS is configured, VIP CPU spins to check whether the next time interval (Tc) arrives when the CPU is not busy (that is, when there is no traffic). Otherwise, the verification is piggybacked in the tx/rx interrupt routines. You spin the CPU only when it is not busy. Therefore, performance is not affected.

  To understand what the "next time interval" means, see What Is a Token Bucket?

  Note: Traffic shaping becomes active only when it has to enqueue a packet in the shaping queue. In other words, when the amount of traffic exceeds the shaping rate. This explains why the VIP CPU is not always 99% when dTS is configured. For more information on dTS, see:

    ♦ Distributed Traffic Shaping
    ♦ Configuring Distributed Traffic Shaping
- **High CPU utilization on VIP caused by spurious memory accesses and alignment errors**

  Alignment errors and spurious memory accesses are software failures that are corrected by the Cisco IOS software without the need to crash the VIP. If these errors appear frequently, it causes the operating system to make a lot of corrections which can lead to high CPU utilization.

For more information on alignment errors and spurious memory accesses, see Troubleshooting Spurious Accesses, Alignment Errors, and Spurious Interrupts.

To check for spurious memory accesses and alignment errors, use the **show alignment** command. An example of such an error looks like this:

```
VIP-Slot1#show alignment
No alignment data has been recorded.
No spurious memory references have been recorded.
```

Other causes of high CPU utilization can be the amount and extent of distributed features that are enabled. If you suspect that this could be the reason, or if you cannot identify any of the causes for high CPU utilization explained in this document, open a service request with the Cisco Technical Assistance Center (TAC).

Information to Collect if You Open a TAC Service Request

If you still need assistance after you follow the troubleshooting steps above and want to open a service request (registered customers only) with the Cisco TAC, be sure to include this information:
• Output from the **show controllers vip [***all / slot#***] accumulator** command • Output from the **show technical−support** command from the relevant RSP and VIP Please attach the collected data to your service request in non−zipped, plain text format (.txt). In order to attach information to your service request, upload it through the TAC Service Request Tool (registered customers only) . If you cannot access the Service Request Tool, you can attach the relevant information to your service request, and send it to attach@cisco.com with your service request number in the subject line of your message. **Note:** Please do not manually reload or power−cycle the router before you collect the above information (unless required to restore network operation), because this can cause important information to be lost that is needed to determine the root cause of the problem.

Related Information

- **Cisco Routers Product Support**
- **Troubleshoot and Alerts: Cisco 7500 Series Routers**
- **Technical Support & Documentation – Cisco Systems**