Troubleshooting High CPU Utilization

This document includes these sections:

CPU Utilization Overview

When the switch has completed the boot process, the CPU has two distinct functions. The first function is run the different processes under IOS to carry out the function for a switch operating in a network. The second is to send/receive packets to/from the switching hardware. The CPU is doing both of these functions simultaneously.

The CPU becomes too busy when either an IOS process consumes too much CPU time or the CPU receives too many packets from the switching hardware. When either of these two CPU consumers requires the CPU resource to the detriment of the other, then the CPU is too busy. For instance the CPU is receiving lots of packets from the hardware because there's a broadcast storm on the network. In this case the CPU is so busy processing all the received packets that the other IOS processes aren't given access to the CPU resource. This is just one example of a possible root cause for high CPU utilization. This debugging section will help you to identify other examples and describe how to take corrective action.

Under normal operating conditions, on a non-stackable switch at a minimum, the CPU will have a certain baseline utilization. Depending on the model and the type being used this can range from 5 to 40% If the switch is stacked then, at a minimum, the CPU will operate normally a few percent higher. The number of members in the stack makes a difference on overall CPU utilization. In the stacked switch, the CPU utilization is measured on the master switch only. If the CPU is busy 5% of the time then the CPU is idle the other 95% of the time. The switch will never report CPU utilization at 0%. There are multiple background IOS processes running on timers that execute multiple times a second. This is why even in the simplest of deployments, the switch never reports 0% CPU utilization.

Americas Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

One of the reasons that different ranges and models within those ranges will differ in the baseline utilization is differences in design. Where the earlier models of the switches with little usage of microcontrollers, the later ones do utilize these more. As more tasks are being offloaded to those microcontrollers there is an increase in communication between the CPU and the microcontrollers. The processes this will be reported under are the HULC led and the Redearth Tx an Rx processes.

To determine switch CPU utilization, enter the **show processes cpu sorted** privileged EXEC command. The output shows how busy the CPU has been in the past 5 seconds, the past 1 minute, and the past 5 minutes. The output also shows the utilization percentage that each system process has used in these periods.

```
Switch# show processes cpu sorted
CPU utilization for five seconds: 5%/0%; one minute: 6%; five minutes: 5%
 PID Runtime(ms)   Invoked      uSecs   5Sec   1Min   5Min TTY Process
   1        4539     89782         50  0.00%  0.00%  0.00%   0 Chunk Manager
   2        1042   1533829          0  0.00%  0.00%  0.00%   0 Load Meter
   3           0         1          0  0.00%  0.00%  0.00%   0 DiagCard3/-1
   4    14470573   1165502      12415  0.00%  0.13%  0.16%   0 Check heaps
   5        7596    212393         35  0.00%  0.00%  0.00%   0 Pool Manager
   6           0         2          0  0.00%  0.00%  0.00%   0 Timers
   7           0         1          0  0.00%  0.00%  0.00%   0 Image Licensing
   8           0         2          0  0.00%  0.00%  0.00%   0 License Client N
   9     1442263     25601      56336  0.00%  0.08%  0.02%   0 Licensing Auto U
  10           0         1          0  0.00%  0.00%  0.00%   0 Crash writer
  11      979720   2315501        423  0.00%  0.00%  0.00%   0 ARP Input
  12           0         1          0  0.00%  0.00%  0.00%   0 CEF MIB API
  <output truncated>
```

In this output, the CPU utilization for the last 5 seconds shows two numbers (`5%/0%`).

- The first number, `5%`, tells how busy the CPU was in the past 5 seconds. This number is the total CPU utilization for all the active system processes, including the percentage of time at the interrupt level.
- The second number, `0%`, shows the percentage of time at the interrupt level in the past 5 seconds. The interrupt percentage is the CPU time spent receiving packets from the switch hardware. The percentage of time at interrupt level is always less than or equal to the total CPU utilization.

Two other important numbers are shown on the same output line: the average utilization for the last 1 minute (6 percent in this example) and the average utilization for past 5 minutes (5 percent in this example). These values are typical for a nonstacked switch in a small and stable environment.

There can be hundreds of active system processes on the CPU at any time. This number can vary, based on the switch model, the Cisco IOS release, the feature set, and (if applicable) the number of switches in a switch stack. For example, on a stack of Catalyst 3750 switches running the IP base image, there are typically 475 active system processes. The Catalyst 2960 switch running the LAN base image has a smaller number of active processes than a stack of Catalyst 3750 switches. In general, the more features in the Cisco IOS image, the more system processes.

When High CPU Utilization Is a Problem

These sections tell how to identify high CPU utilization and determine if it is a problem:

- Normal Conditions with High CPU Utilization, page 4
- Network Symptoms Caused by High CPU Utilization, page 5
- Determining Interrupt Percentage, page 6

In some instances, high CPU utilization is normal and does not cause network problems. High CPU utilization becomes a problem when the switch fails to perform as expected.

Enter the **show processes cpu history** privileged EXEC command to see the CPU utilization for the last 60 seconds, 60 minutes, and 72 hours. The command output provides graphical views of how busy the CPU has been. You can see if the CPU has been constantly busy or if utilization has been spiking.

In this example, the CPU spiked to 100 percent 46 hours ago, shortly after the switch was rebooted. It spiked to 87 percent within the past hour.

```
Switch# show processes cpu history

    2222222222222222222222222222223333322222222222222222222222
    6666666666666666655555777777777711111999996666677777666666
100
 90
 80
 70
 60
 50
 40
 30 ************************************************************
 20 ************************************************************
 10 ************************************************************
   0....5....1....1....2....2....3....3....4....4....5....5....
             0    5    0    5    0    5    0    5    0    5
               CPU% per second (last 60 seconds)

    3222222222222222222223222222223233422222332222222228232222
    1989989888988888898908988889907008999971089999798977077 98
100
 90                                                        *
 80                                                        *
 70                                                        *
 60                                                        *
 50                                      *                 *
 40                                      *                 *
 30 ############################################################
 20 ############################################################
 10 ############################################################
   0....5....1....1....2....2....3....3....4....4....5....5....
             0    5    0    5    0    5    0    5    0    5
               CPU% per minute (last 60 minutes)
              * = maximum CPU%   # = average CPU%
                                                          1
    8544545455444554654455443445445445656645748 4808
    9664114557793371151750116923303433334881182 3006
100                                                       *
 90 *                                                     **
 80 *                                                  * ***
 70 *                                               *  * * ***
 60 **      **    * **  *                          * **  * * ***
 50 *** * ****** ** ** ***   * *  *  ********** ***
 40 ************************************************
 30 ################################################*
 20 ################################################*
 10 ################################################
   0....5....1....1....2....2....3....3....4....4....5....5....6....6....7.
             0    5    0    5    0    5    0    5    0    5    0    5    0
                  CPU% per hour (last 72 hours)
                 * = maximum CPU%   # = average CPU%
```

The CPU spiked to 87% 52 minutes ago.

The CPU spiked to 100% 46 hours ago.

205495

CPU utilization spikes caused by a known network event or activity are not problems. Even an 87-percent spike might be acceptable, depending on the cause. For example, an acceptable spike could be caused by the network administrator entering a **write memory** privileged EXEC command on the CLI. A spike is also a normal reaction to a topology change in a large Layer 2 network. See the "Normal Conditions with High CPU Utilization" section on page 4 for a list of events and activities that cause the

CPU utilization to spike.

Over time, the switch operates within a certain sustained CPU utilization range, which is considered the normal operations baseline. You can use the output of the **show processes cpu history** privileged EXEC command to determine normal operating baseline utilization for the previous 72 hours. CPU utilization exceeding the normal operating baseline for an unknown reason indicates that there is a problem.

In the previous example, the CPU utilization was between 40 and 56 percent for the past 37 hours. We consider anything below 50 percent CPU utilization to be acceptable. A sustained level of approximately 50 percent, such as this example, is also acceptable. A sustained CPU utilization of over 50 percent is potentially problematic. While the switch might appear to operate fine at this CPU utilization level, when the switch CPU sustains a 50 percent utilization, its ability to react to dynamic network events is compromised.

Frequent unexplained spikes to the established normal operating baseline or sudden utilization jumps with no explanations are causes for concern. See the "Determining the Root Cause" section on page 6 to identify the source of a high CPU utilization problem.

Normal Conditions with High CPU Utilization

In some network deployments, a busy CPU is normal. In general, the larger the Layer 2 or Layer 3 network, the greater the demand on the CPU to process network-related traffic. These are examples of operations that have the potential to cause high CPU utilization:

Spanning Tree

A Layer 2 spanning-tree instance runs for every VLAN configured on a Layer 2 switch by the per-VLAN spanning-tree (PVST) feature. The CPU time utilized by spanning tree varies depending upon the number of spanning-tree instances and the number of active interfaces. The more instances and the more active interfaces, the greater the CPU utilization.

IP Routing Table Updates

When a Layer 3 switch enabled for IP routing receives a large routing table, the switch must process the routing information updates. The CPU utilization spike during processing depends on these factors:

- The size of the routing information update
- The frequency of the updates
- The number of routing protocol processes receiving the updates
- The presence of any route maps or filters

Cisco IOS Commands

Some Cisco IOS commands cause a spike in the CPU utilization. These include:

- The **show tech-support** privileged EXEC command.
- The **write memory** privileged EXEC command (particularly if the switch is in a stack).
- The **show-running configuration** privileged EXEC command on a switch stack master.
- The **debug** privileged EXEC command to enable debugging of a feature. Printing debug messages to the console increases the CPU utilization as long as debugging is enabled.

Other Events Causing High CPU Utilization

- Frequent or large number of IGMP requests—the CPU processes IGMP messages .
- Large number of IP SLAs monitoring sessions—the CPU generates ICMP or traceroute packets.
- SNMP polling activities, particularly MIB walk—the Cisco IOS SNMP engine executes SNMP requests.
- Large number of simultaneous DHCP requests, such as when links are restored to numerous clients (when the switch is acting as DHCP server).
- ARP broadcast storms.
- Ethernet broadcast storms.

Network Symptoms Caused by High CPU Utilization

A CPU that is too busy affects the ability of the system processes to execute as expected. When the system processes do not execute, the switch (or the directly connected networking devices) react as if there was a network problem. For Layer 2 networks, a spanning-tree reconvergence could occur. In a Layer 3 network, a routing topology could change.

Known symptoms that can occur when the switch CPU is too busy:

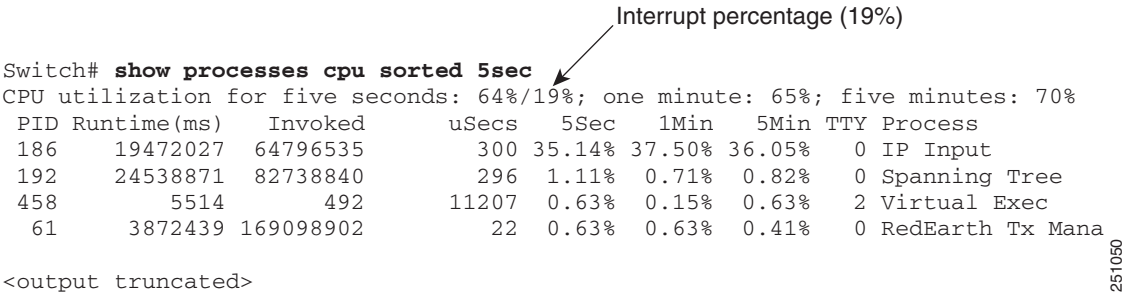
- Spanning-tree topology change—When a Layer 2 network device does not receive timely spanning-tree BPDUs on its root port, it considers the Layer 2 path to the root switch as down, and the device tries to find a new path. Spanning tree reconverges in the Layer 2 network.
- Routing topology change, such as BGP route flapping or OSPF route flapping.
- EtherChannel links bounce—When the network device at the other end of the EtherChannel does not receive the protocol packets required to maintain the EtherChannel link, this might bring down the link.
- The switch fails to respond to normal management requests:
  - ICMP ping requests.
  - SNMP timeouts
  - Telnet or SSH sessions that are slow or cannot be started
- UDLD flapping—The switch relies on keepalives from its peer in aggressive mode.
- IP SLAs failures due to SLAs responses beyond the acceptable threshold.
- DHCP or IEEE 802.1x failures if the switch cannot forward or respond to requests.
- Dropped packets or increased latency for those packets routed in software.
- HSRP flapping.

Determining Interrupt Percentage

The CPU utilization history shows only the total CPU utilization over time. It does not show the CPU time spent at interrupt. Knowing the time spent at interrupt is critical for determining the cause of CPU utilization. The CPU utilization history shows when the CPU is consistently receiving network packets, but it does not show the cause.

Enter the Cisco IOS **show processes cpu sorted 5sec** privileged EXEC command to show the current the CPU utilization and which IOS processes are using the most CPU time. In this example, the CPU is too busy because the sustained utilization is over the baseline of 50%.

The arrow is pointing to the interrupt percentage value. It is the second number in the 5 second utilization percentage.



Its normal for the interrupt percentage to be greater than 0 percent and less than 5 percent. It is acceptable for the interrupt percentage to be between 5 percent and 10 percent. An interrupt percentage over 10 percent should be investigated. See the "Analyzing Network Traffic" section on page 8 for investigation information.

Determining the Root Cause

When you suspect the CPU is too busy, first determine if it is busy because a system process is taking too much CPU time or because it is receiving too many network packets. The debugging techniques are different for these two root causes. These sections tell you how to identify and troubleshoot the cause:

- Identifying the Cause as System Process or Network Traffic, page 7
- Analyzing Network Traffic, page 8
- Debugging Active Processes, page 19

Note Always refer to the release notes for the specific platform and software release of your switch for any known Cisco IOS bugs. You can eliminate these issues from your troubleshooting steps.

When the switch CPU is busy, management tools such as Telnet or SSH are usually not very useful. We recommend that you use the switch console for debugging CPU utilization issues.

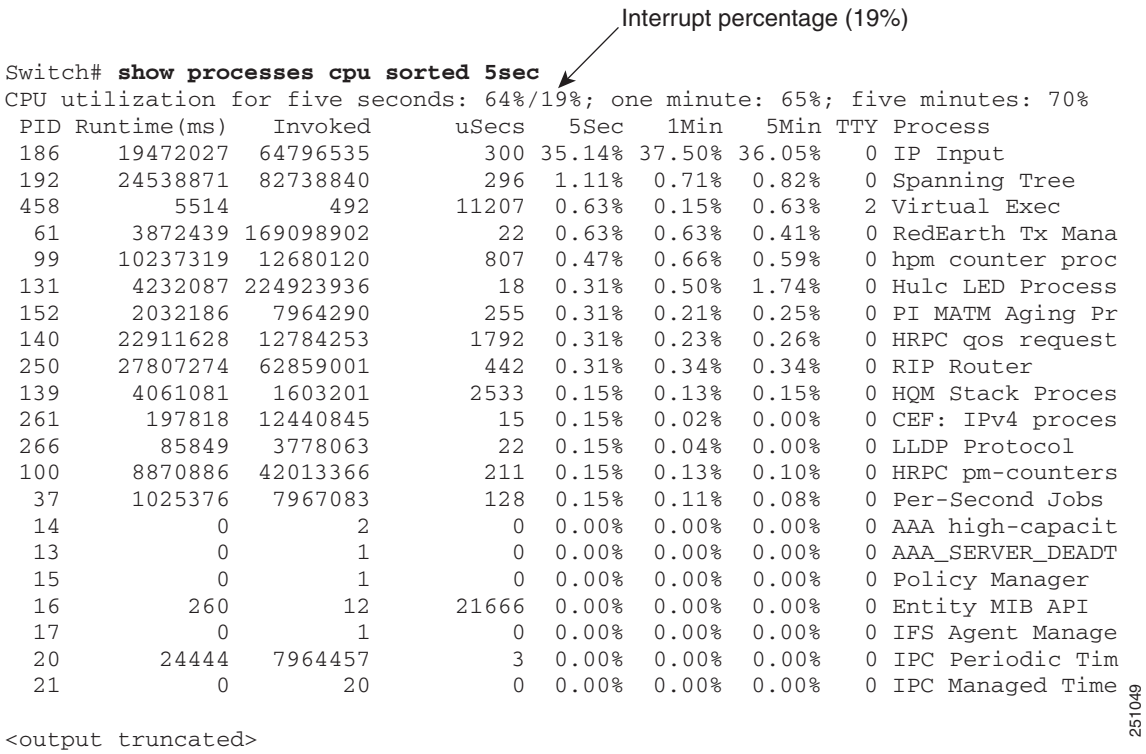
Identifying the Cause as System Process or Network Traffic

To determine how busy the CPU is and which operating system processes are using the most CPU time, enter the **show processes cpu sorted 5sec** privileged EXEC command. In the output, for `CPU utilization for five seconds,` the second number is the interrupt percentage. Use the interrupt percentage to determine if the problem is caused by a system process or high network traffic.

If the interrupt percentage is too high relative to the total CPU utilization percentage, the CPU utilization problem is caused by receiving too many packets from system hardware. See the "Determining Interrupt Percentage" section on page 6 for how to identify a high interrupt percentage.

- A high interrupt percentage indicates too much network traffic. This is the most common cause of high CPU utilization. To troubleshoot, see the "Analyzing Network Traffic" section on page 8.
- High CPU utilization percentage with a low interrupt percentage indicates a problem with an operating system process. To troubleshoot, see the "Debugging Active Processes" section on page 19.
- When both percentages are high or if you cannot determine whether or not the interrupt percentage is a significant contributor to CPU utilization, first see the "Analyzing Network Traffic" section on page 8. If the information provided in this section does not resolve the high CPU utilization problem, see the "Debugging Active Processes" section on page 19.

In this example, the CPU utilization is 64 percent, and the interrupt percentage is 19 percent, which is high.The utilization problem is caused by the CPU processing too many packets received from the network. In this case, see the "Analyzing Network Traffic" section on page 8.



In the next example, the interrupt percentage is low compared to the CPU utilization percentage (5 percent compared to 82 percent). A high CPU utilization and relatively low interrupt percentage indicates that one or more system processes is taking too much time. In this case, see the "Debugging Active Processes" section on page 19.

```
Switch# show processes cpu sorted 5sec
CPU utilization for five seconds: 82%/5%; one minute: 40%; five minutes: 34%
 PID Runtime(ms)   Invoked      uSecs   5Sec   1Min   5Min TTY Process
 217   135928429 493897689        275 45.68% 18.61% 16.78%   0 SNMP ENGINE
  47    61840574 480781517        128 23.80%  8.63%  7.43%   0 hrpc <-response
 158    58014186 265701225        218  1.11%  1.36%  1.35%   0 Spanning Tree
  46     1222030  67734870         18  0.47%  0.14%  0.08%   0 hrpc -> request
  75     1034724   8421764        122  0.15%  0.06%  0.02%   0 hpm counter proc
 223         125       157        796  0.15%  0.13%  0.03%   2 Virtual Exec
 213        2573       263       9783  0.15%  2.43%  0.71%   1 Virtual Exec
 150      578692   3251272        177  0.15%  0.02%  0.00%   0 CDP Protocol
 114     8436933   3227814       2613  0.15%  0.17%  0.16%   0 HRPC qos request
 105     1002819  96357752         10  0.15%  0.10%  0.06%   0 Hulc LED Process
  28      701287     68160      10288  0.15%  0.01%  0.00%   0 Per-minute Jobs
 215     9757808  42169987        231  0.15%  0.58%  0.56%   0 IP SNMP
  12           0         1          0  0.00%  0.00%  0.00%   0 IFS Agent Manage
  13           8     67388          0  0.00%  0.00%  0.00%   0 IPC

!<Output truncated>
```

Analyzing Network Traffic

When the interrupt percentage is high, the root cause of the problem is that the CPU is receiving too many packets. To resolve the problem, you need to find the source of the packets, and either stop the flow, or modify the switch configuration. See these sections:

- System Processes and Network Packets, page 8
- System Processes and Punted Packets, page 9
- Identifying Network Packets Received by the CPU, page 10
- Limiting Network Packets to the CPU, page 15
- Identifying Packets Punted from the Switch Hardware, page 16
- Identifying TCAM Utilization Issues, page 17
- Resolving TCAM Utilization Issues, page 18

System Processes and Network Packets

You can identify the type of network packets that are flooding the CPU by the system processes that the switch uses to manage the packets. When the CPU interrupt percentage is high compared to the overall CPU utilization percentage, enter the **show processes cpu sorted 5sec** privileged EXEC command to determine the most active system process. The CPU can receive multiple packet types and have multiple active system processes. The output lists the most active processes first. The most active system processes are probably responding to the receipt of network packets.

```
Switch# show processes cpu sorted 5sec
CPU utilization for five seconds: 64%/19%; one minute: 65%; five minutes: 70%
 PID Runtime(ms)   Invoked      uSecs   5Sec   1Min   5Min TTY Process
 186    19472027  64796535        300 35.14% 37.50% 36.05%   0 IP Input
 192    24538871  82738840        296  1.11%  0.71%  0.82%   0 Spanning Tree
 458        5514       492      11207  0.63%  0.15%  0.63%   2 Virtual Exec
  61     3872439 169098902         22  0.63%  0.63%  0.41%   0 RedEarth Tx Mana
  99    10237319  12680120        807  0.47%  0.66%  0.59%   0 hpm counter proc
 131     4232087 224923936         18  0.31%  0.50%  1.74%   0 Hulc LED Process
 152     2032186   7964290        255  0.31%  0.21%  0.25%   0 PI MATM Aging Pr
 140    22911628  12784253       1792  0.31%  0.23%  0.26%   0 HRPC qos request
 250    27807274  62859001        442  0.31%  0.34%  0.34%   0 RIP Router

!<Output truncated>
```

Table 1 lists some common system processes and the associated packet types. If one of the listed system processes is the most active process in the CPU, it is likely that the corresponding type of network packet is flooding the CPU.

Table 1 Processes Associated with Network Packet Processing

System Process Name	Packet Types
IP Input	IP packets (includes ICMP)
IGMPSN	IGMP snooping packets
ARP Input	IP ARP packets
SNMP Engine	SNMP packets

See the "Identifying Network Packets Received by the CPU" section on page 10 to find the source of the packets and how to troubleshoot.

System Processes and Punted Packets

On a Layer 3 switch, when the IP route is not known, the switch hardware *punts* (sends) IP packets to the CPU for IP routing. Punted packets are handled at the interrupt level and can cause the CPU to become too busy. If the interrupt percentage shown in the command output is high, but the most active processes are not those shown in Table 1, or no processes seem active enough to justify the CPU utilization, the high CPU utilization is most likely caused by *punted* packets, as shown in the example output.

```
Switch# show processes cpu sorted 5sec
CPU utilization for five seconds: 53%/28%; one minute: 48%; five minutes: 45%
 PID Runtime(ms)   Invoked      uSecs   5Sec   1Min   5Min TTY Process
  78      461805 220334990          2 11.82% 11.53% 10.37%   0 HLFM address lea
 309    99769798   1821129      54784  5.27%  1.53%  1.39%   0 RIP Timers
 192    19448090  72206697        269  1.11%  0.87%  0.81%   0 Spanning Tree
 250    25992246  58973371        440  0.63%  0.27%  0.29%   0 RIP Router
  99     6853074  11856895        577  0.31%  0.46%  0.44%   0 hpm counter proc
 131     3184794 210112491         15  0.31%  0.13%  0.12%   0 Hulc LED Process
 140    20821662  11950171       1742  0.31%  0.28%  0.26%   0 HRPC qos request
 139     3166446   1498429       2113  0.15%  0.11%  0.11%   0 HQM Stack Proces
  67     2809714  11642483        241  0.15%  0.03%  0.00%   0 hrpc <- response
 223      449344  16515401         27  0.15%  0.03%  0.00%   0 Marvell wk-a Pow
  10           0         1          0  0.00%  0.00%  0.00%   0 Crash writer
  11      227226    666257        341  0.00%  0.00%  0.00%   0 ARP Input

!<Output truncated>
```

Table 2 lists the most active system processes when the CPU is busy acting on punted IP packets. CPU processing of punted packets is not associated with a listed process.

Table 2 Processes Indicating Punted Packet Handling

System Process Name	Packet Types
HLFM address lea	IP forwarding manager process
Check heaps	Memory collection process
Virtual Exec	Cisco IOS CLI process
RedEarth Tx Mana	Microprocessor communication process
hpm counter proc	Statistics collection

See the "Identifying Packets Punted from the Switch Hardware" section on page 16 for troubleshooting procedures for punted packets.

Identifying Network Packets Received by the CPU

These techniques to determine the type of packets being sent to CPU can complement each other or can be used individually.

- The switch hardware counts the packets in each CPU receive queue. You can use this count to determine the type of packets being received. See the "Monitoring Packet Counts for CPU Receive Queues" section on page 10.
- You can use the **debug** privileged EXEC command to print to the console all CPU-received packets. The **debug** command can debug each receive queue separately. See the "Debugging Packets from the Switch CPU Receive Queues" section on page 12.
- The switch counts all IP packets that are sent and received. This information is useful to identify any counts that are particularly high and incrementing rapidly. See the "Monitoring IP Traffic Counts" section on page 14.

Monitoring Packet Counts for CPU Receive Queues

If the switch is being flooded by a specific packet type, the hardware places that packet type into the appropriate CPU queue and counts it. Enter the **show controllers cpu-interface** privileged EXEC command to see the packet count per queue.

```
Switch # show controllers cpu-interface
ASIC    Rxbiterr   Rxunder    Fwdctfix   Txbuflos   Rxbufloc   Rxbufdrain
------------------------------------------------------------------------
ASIC0      0          0          0          0          0          0
ASIC1      0          0          0          0          0          0

cpu-queue-frames  retrieved  dropped    invalid    hol-block  stray
----------------- ---------- ---------- ---------- ---------- ----------
rpc               726325     0          0          0          0
stp               16108      0          0          0          0
ipc               56771      0          0          0          0
routing protocol  3949       0          0          0          0
L2 protocol       827        0          0          0          0
remote console    58         0          0          0          0
sw forwarding     0          0          0          0          0
host              0          0          0          0          0
broadcast         382        0          0          0          0
```

```
cbt-to-spt        0          0          0          0          0
igmp snooping     3567       0          0          0          0
icmp              11256      0          0          0          0
logging           0          0          0          0          0
rpf-fail          0          0          0          0          0
dstats            0          0          0          0          0
cpu heartbeat     322409     0          0          0          0
<output truncated>
```

The switch also counts the CPU-bound packets that are discarded due to congestion. Each CPU receive queue has a packet-count maximum. When the receive queue maximum is reached, the switch hardware discards packets destined for the congested queue. The switch counts discarded packets for each queue. Increased discard counts for a particular CPU queue mean heavy usage for that queue.

Enter the **show platform port-asic stats drop** privileged EXEC command to see the CPU receive-queue discard counts and to identify the queue discarding packets. This command is not as useful as the **show controllers cpu-interface** command because the output shows numbers for the receive queues instead of names, and it shows only the discards. Because the switch hardware sees the CPU receive-queue dropped packets as sent to the supervisor, the dropped packets are called `Supervisor TxQueue Drop Statistics` in the command output.

```
Switch #show platform port-asic stats drop
Port-asic Port Drop Statistics - Summary
========================================
  RxQueue Drop Statistics Slice0
   RxQueue 0 Drop Stats Slice0: 0
   RxQueue 1 Drop Stats Slice0: 0
   RxQueue 2 Drop Stats Slice0: 0
   RxQueue 3 Drop Stats Slice0: 0
  RxQueue Drop Statistics Slice1
   RxQueue 0 Drop Stats Slice1: 0
   RxQueue 1 Drop Stats Slice1: 0
   RxQueue 2 Drop Stats Slice1: 0
   RxQueue 3 Drop Stats Slice1: 0

!<Output truncated>

  Port 27 TxQueue Drop Stats: 0

  Supervisor TxQueue Drop Statistics
    Queue  0: 0
    Queue  1: 0
    Queue  2: 0
    Queue  3: 0
    Queue  4: 0
    Queue  5: 0
    Queue  6: 0
    Queue  7: 0
    Queue  8: 0
    Queue  9: 0
    Queue 10: 0
    Queue 11: 0
    Queue 12: 0
    Queue 13: 0
    Queue 14: 0
    Queue 15: 0

! <Output truncated>
```

The queue numbers in this output for `Supervisor TxQueue Drop Statistics` are in the same order as the queue names in the **show controllers cpu-interface** command output. For example, `Queue 0` in this output corresponds to `rpc` in the previous output;. `Queue 15` corresponds to `cpu heartbeat`, and so on.

The statistics do not reset. Enter the command multiple times to review active queue discards. The command output also shows other drop statistics, some of which are truncated in the example.

See the "CPU Receive Queues" section on page 21 for more information about CPU queues.

Debugging Packets from the Switch CPU Receive Queues

The hardware inserts packets received from the network into 16 different queues for the CPU. You can identify packets sent to the CPU by enabling debugging for a queue. Use the output from the **show controllers cpu-interface** privileged EXEC command to learn which queues to start debugging first. See the "Monitoring Packet Counts for CPU Receive Queues" section on page 10.

If the output from the **show controllers cpu-interface** command does not indicate a good place to start, we recommend that you enable debugging on one queue at a time until the console is flooded with debug messages. You use a different **debug** command to turn debugging on and off for each CPU receive queue.

Before you start debugging the receive queues, use this procedure to prevent other applications from writing to the console, to increase the system log buffer (if it is at the default size), and to put a detailed timestamp on the debug messages. When the debugging session is over, the debug messages are in the system log.

Beginning in privileged EXEC mode, follow these steps to prepare for debugging the CPU queue:

	Command	Purpose
Step 1	**configure terminal**	Enter global configuration mode.
Step 2	**no logging console**	Disable logging to the console terminal.
Step 3	**logging buffered 128000**	Enable system message logging to a local buffer, and set the buffer size to 12800 bytes.
Step 4	**service timestamps debug datetime msecs localtime**	Configure the system to apply a timestamp to debugging messages or system logging messages.
Step 5	**exit**	Return to privileged EXEC mode.

Be ready to enter the **undebug all** privileged EXEC command to stop any packet flooding on the console. Even though you do not see a command prompt, the switch accepts the **undebug all** command at any time. After you enter the command, allow some time for the buffered debug messages to subside and for the debugging message buffer to empty.

The debug messages can help to determine the source of the packet flood. This is useful if the packets are all the same type or originating from the same source.

This is an example of turning on the CPU queues one at a time until console floods.

```
Switch# debug platform cpu-queues ?
  broadcast-q          Debug packets received by Broadcast Q
  cbt-to-spt-q         Debug packets received by cbt-to-spt Q
  cpuhub-q             Debug packets received by CPU heartbeat Q
  host-q               Debug packets received by host Q
  icmp-q               Debug packets received by ICMP Q
  igmp-snooping-q      Debug packets received by IGMP snooping Q
  layer2-protocol-q    Debug packets received by layer2 protocol Q
  logging-q            Debug packets received by logging Q
  remote-console-q     Debug packets received by remote console Q
  routing-protocol-q   Debug packets received by routing protocol Q
  rpffail-q            Debug packets received by RPF fail Q
  software-fwd-q       Debug packets received by software forward Q
  stp-q                Debug packets received by STP Q

Switch# debug platform cpu-queues broadcast-q
debug platform cpu-queue broadcast-q debugging is on
Switch#
Switch# debug platform cpu-queues cbt-to-spt-q
debug platform cpu-queue cbt-sbt-q debugging is on
Switch#
Switch# debug platform cpu-queues cpuhub-q
debug platform cpu-queue cpuhb debugging is on
Switch#
Switch# debug platform cpu-queues host-q
debug platform cpu-queue host-q debugging is on
Switch#
*Mar  2 22:48:06.227: L2B-Q:Dropped Null L3Hwidb: Local Port Fwding L3If:
L2If:GigabitEthernet4/0/23 DI:0x6F5, LT:1, Vlan:139   SrcGPN:185, SrcGID:185,
ACLLogIdx:0x4, MacDA:ffff.ffff.ffff, MacSA: 0009.9b00.edfe   ARP:
00010800_06040001_00099B00_EDFEAC14_8B850000_00000000_AC148B81
   TPFFD:E10000B9_008B008B_00800044-000406F5_1A1A0000_00000000

Switch# debug platform cpu-queues icmp-q
debug platform cpu-queue icmp-q debugging is on
Switch#
*Mar  2 22:48:16.947: ICMP-Q:Dropped Throttle timer not awake: Remote Port Blocked
L3If:Vlan200 L2If:GigabitEthernet1/0/3 DI:0xB4, LT:7, Vlan:200   SrcGPN:3, SrcGID:3,
ACLLogIdx:0x0, MacDA:001d.46be.7541, MacSA: 0000.0300.0101   IP_SA:10.10.200.1
IP_DA:10.10.200.5 IP_Proto:1
   TPFFD:ED000003_008B00C8_00B00222-000000B4_00060000_03090000

*Mar  2 22:48:16.947: ICMP-Q:Dropped Throttle timer not awake: Remote Port Blocked
L3If:Vlan200 L2If:GigabitEthernet1/0/3 DI:0xB4, LT:7, Vlan:200   SrcGPN:3, SrcGID:3,
ACLLogIdx:0x0, MacDA:001d.46be.7541, MacSA: 0000.0300.0101   IP_SA:10.10.200.1
IP_DA:10.10.200.5 IP_Proto:1
   TPFFD:ED000003_008B00C8_00B00222-000000B4_00050000_03090000

*Mar  2 22:48:16.947: ICMP-Q:Dropped Throttle timer not awake: Remote Port Blocked
L3If:Vlan200 L2If:GigabitEthernet1/0/3 DI:0xB4, LT:7, Vlan:200   SrcGPN:3, SrcGID:3,
ACLLogIdx:0x0, MacDA:001d.46be.7541, MacSA: 0000.0300.0101   IP_SA:10.10.200.1
IP_DA:10.10.200.5 IP_Proto:1
   TPFFD:ED000003_008B00C8_00B00222-000000B4_00040000_03090000
```

A single packet was received on the host queue. This is normal

205496

The sequence of actions in the examples:

- After the **debug platform cpu-queue host-q** command was entered, a single packet was received. This is normal.
- When the next command, **debug platform cpu-queue icmp-q**, was entered, the flood began. All packets received on icmp-q are the same. Only three packets are shown. Thus, the CPU is receiving an ICMP packet flood.

- Examine the output for information about the source, including the VLAN (200) and source MAC address (0000.0300.0101) of this packet (shown in bold text).

```
*Mar  2 22:48:16.947: ICMP-Q:Dropped Throttle timer not awake: Remote Port Blocked
L3If:Vlan200 L2If:GigabitEthernet1/0/3 DI:0xB4, LT:7, Vlan:200   SrcGPN:3, SrcGID:3,
ACLLogIdx:0x0, MacDA:001d.46be.7541, MacSA: 0000.0300.0101   IP_SA:10.10.200.1
IP_DA:10.10.200.5 IP_Proto:1
   TPFFD:ED000003_008B00C8_00B00222-000000B4_00040000_03090000
```

- Enter the **show mac address-table** privileged EXEC command for the VLAN to see the MAC address table and to find the interface where this MAC address was learned. The output shows the packets are being received on interface Gigabit Ethernet 1/0/3 (shown in bold text).

```
Switch# show mac address-table dynamic vlan 200
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
200    0000.0300.0101    DYNAMIC      Gi1/0/3

!<Output truncated>
```

You can use these steps for the different packet types when the CPU is being flooded by a single flow. Continue to enable the debugging of the different CPU queues until the console is flooded. See the "CPU Receive Queues" section on page 21 for details about the CPU queues.

Beginning in privileged EXEC mode, follow these steps to view the system log with the debug messages:

	Command	Purpose
Step 1	**terminal length 0**	Set the number of lines on the terminal screen for the current session to 0.
Step 2	**show logging**	Display the contents of the standard system logging buffer.
Step 3	**terminal length 30**	Set the terminal length to 30, or reset back to the original value.
Step 4	**exit**	Exit the CLI.

Note If you modified the configuration before debugging by increasing the system log buffer or adding a timestamp, consider returning these settings to the default configuration when debugging is complete.

Monitoring IP Traffic Counts

The switch counts all IP packet types received by the CPU. These packet counts do not include the IP packets switched or routed in hardware. Enter the **show ip traffic** privileged EXEC command to display the IP packet type counts. The output breaks down the IP packets into Layer 4 types (that is, ICMP, multicast, ICMP, or ARP). When a particular count is incrementing rapidly, the IP packet type is probably flooding the CPU.

```
Switch# show ip traffic
IP statistics:
  Rcvd:  12420483 total, 840467 local destination
         0 format errors, 0 checksum errors, 0 bad hop count
         0 unknown protocol, 222764 not a gateway
         0 security failures, 0 bad options, 0 with options
  Opts:  0 end, 0 nop, 0 basic security, 0 loose source route
         0 timestamp, 0 extended security, 0 record route
         0 stream ID, 0 strict source route, 0 alert, 0 cipso, 0 ump
         0 other
```

```
  Frags: 0 reassembled, 0 timeouts, 0 couldn't reassemble
         0 fragmented, 0 couldn't fragment
  Bcast: 0 received, 0 sent
  Mcast: 0 received, 0 sent
  Sent:  834640 generated, 928020828 forwarded
  Drop:  189206 encapsulation failed, 0 unresolved, 0 no adjacency
         0 no route, 0 unicast RPF, 0 forced drop
         0 options denied, 0 source IP address zero

ICMP statistics:
  Rcvd: 0 format errors, 0 checksum errors, 834640 redirects, 0 unreachable
        0 echo, 0 echo reply, 0 mask requests, 0 mask replies, 0 quench
        0 parameter, 0 timestamp, 0 info request, 0 other
        0 irdp solicitations, 0 irdp advertisements
  Sent: 834640 redirects, 0 unreachable, 0 echo, 0 echo reply
        0 mask requests, 0 mask replies, 0 quench, 0 timestamp
        0 info reply, 0 time exceeded, 0 parameter problem
        0 irdp solicitations, 0 irdp advertisements

TCP statistics:
  Rcvd: 5830 total, 0 checksum errors, 0 no port
  Sent: 0 total

UDP statistics:
  Rcvd: 0 total, 0 checksum errors, 0 no port
  Sent: 0 total, 0 forwarded broadcasts
PIMv2 statistics: Sent/Received
  Total: 0/0, 0 checksum errors, 0 format errors
  Registers: 0/0 (0 non-rp, 0 non-sm-group), Register Stops: 0/0,  Hellos: 0/0
  Join/Prunes: 0/0, Asserts: 0/0, grafts: 0/0
  Bootstraps: 0/0, Candidate_RP_Advertisements: 0/0
  State-Refresh: 0/0

IGMP statistics: Sent/Received
  Total: 0/0, Format errors: 0/0, Checksum errors: 0/0
  Host Queries: 0/0, Host Reports: 0/0, Host Leaves: 0/0
  DVMRP: 0/0, PIM: 0/0

EIGRP-IPv4 statistics:
  Rcvd: 0 total
  Sent: 0 total

ARP statistics:
  Rcvd: 0 requests, 0 replies, 0 reverse, 0 other
  Sent: 92 requests, 87 replies (0 proxy), 0 reverse
  Drop due to input queue full: 444087
```

Limiting Network Packets to the CPU

You can prevent problem network packets from impacting the CPU utilization by stopping them at the ingress interface:

- To limit Ethernet broadcast or multicast packet storms, use the **storm-control** {**broadcast** | **multicast** | **unicast**} **level** {*level* [*level-low*] | **bps** *bps* [*bps-low*] | **pps** *pps* [*pps-low*]} interface level configuration command. See the “Configuring Port-Based Traffic Control” chapter in the switch software configuration guide.
- If the root cause of the high CPU utilization is a Layer 2 loop, the spanning tree configuration could be the problem. See the “Configuring STP” chapter in the switch software configuration guide.
- Policing traffic can limit the number of packets that enter a switch. Policing can deny ingress traffic, limit it to a specific bits-per-second rate, or permit some traffic while limiting other traffic. You can police traffic on the MAC address, the IPv4 header, the IPv6 header (if IPv6 is supported on the

switch), or the Layer 4 port number. See the chapters on "Configuring Network Security with ACLs," "Configuring IPv6 ACLs" (if supported on the switch), and "Configuring QoS" in the switch software configuration guide.

- To prevent IP ARP packets from affecting the CPU utilization on Layer 3 switches, configure Dynamic ARP Inspection (DAI), and enter the **ip arp inspection limit** {**rate** *pps* [**burst interval** *seconds*] | **none**} interface configuration command to use the rate-limiting feature. See the chapter on "Configuring Dynamic ARP Inspection" in the switch software configuration guide.

Identifying Packets Punted from the Switch Hardware

As part of normal Layer 3 switch operation, when the IP route is not programmed into the switch hardware, the hardware punts IP packets to the CPU for IP routing. Punting occasional IP packets to the CPU is normal and expected, but if too many IP packets are punted, the CPU becomes too busy.

The switch counts every IP packet that the hardware punts to the CPU for IP routing. You can use the **show controllers cpu** privileged EXEC command to see the number of packets put into each CPU receive queue. When the switch hardware is punting packets, the row named `sw forwarding` is incrementing. Enter the command several times to see if the counts for `sw forwarding` are rapidly incrementing.

```
Switch# show controllers cpu-interface
ASIC    Rxbiterr   Rxunder    Fwdctfix   Txbuflos   Rxbufloc   Rxbufdrain
------------------------------------------------------------------------
ASIC0     0          0          0          0          0          0
ASIC1     0          0          0          0          0          0

cpu-queue-frames  retrieved  dropped    invalid    hol-block  stray
----------------- ---------- ---------- ---------- ---------- ----------
rpc               2811788    0          0          0          0
stp               944641     0          0          0          0
ipc               280645     0          0          0          0
routing protocol  813536     0          0          0          0
L2 protocol       8787       0          0          0          0
remote console    2808       0          0          0          0
sw forwarding     65614320   0          0          0          0
host              25         0          0          0          0
broadcast         794570     0          0          0          0
cbt-to-spt        0          0          0          0          0
igmp snooping     18941      0          0          0          0
icmp              0          0          0          0          0
logging           0          0          0          0          0
rpf-fail          0          0          0          0          0
dstats            0          0          0          0          0
cpu heartbeat     1717274    0          0          0          0
```

You can also use the **show platform ip unicast statistics** privileged EXEC to show the same information about punted packets. The punted IP packets are counted as `CPUAdj`, shown in bold in this example.

```
Switch# show platform ip unicast statistics
Global Stats:
        HWFwdLoc:0 HWFwdSec:0 UnRes:0 UnSup:0 NoAdj:0
        EncapFail:0 CPUAdj:1344291253 Null:0 Drop:0

Prev Global Stats:
        HWFwdLoc:0 HWFwdSec:0 UnRes:0 UnSup:0 NoAdj:0
        EncapFail:0 CPUAdj:1344291253 Null:0 Drop:0
```

These statistics are updated every 2 to 3 seconds. Enter the command multiple times to see the change in the `CPUAdj` counts. When the `CPUAdj` counts are rapidly incrementing, many IP packets are being forwarded to the CPU for IP routing.

Identifying TCAM Utilization Issues

On a Layer 3 switch, the hardware uses the TCAM to contain the IP routing database. TCAM space for Layer 3 routing information is limited. When this space is full, new routes learned by Cisco IOS cannot be programmed into the TCAM. If IP packets received by the switch hardware have a destination IP address that is not in the TCAM, the hardware *punts* the IP packets tothe CPU.

To see if the TCAM is full, enter the **show platform tcam utilization** privileged EXEC command.

```
Switch# show platform tcam utilization

CAM Utilization for ASIC# 0                     Max            Used
                                              Masks/Values    Masks/values
 Unicast mac addresses:                        6364/6364         31/31
 IPv4 IGMP groups + multicast routes:          1120/1120          1/1
 IPv4 unicast directly-connected routes:       6144/6144          4/4
 IPv4 unicast indirectly-connected routes:     2048/2048       2047/2047
 IPv4 policy based routing aces:                452/452          12/12
 IPv4 qos aces:                                 512/512          21/21
 IPv4 security aces:                            964/964          30/30

Note: Allocation of TCAM entries per feature uses
a complex algorithm. The above information is meant
to provide an abstract view of the current TCAM utilization
```

In the example, the `IP indirectly-connected routes` resource is full even though the output shows only 2047 of 2048 maximum as in use. If the switch TCAM is full, the hardware routes packets only for destination IP addresses that are in the TCAM. All other IP packets that had a TCAM miss are punted to the CPU. A full TCAM and increasing `sw forwarding` counts from the **show controllers cpu-interface** command output means that punted packets are causing high CPU utilization.

Cisco IOS learns about routes from routing protocols—such as BGP, RIP, OSPF, EIGRP, and IS-IS—and from statically configured routes. You can enter the **show platform ip unicast counts** privileged EXEC command to see how many of these routes were not properly programmed into the TCAM.

```
Switch# show platform ip unicast counts
# of HL3U fibs 2426
# of HL3U adjs 4
# of HL3U mpaths 0
# of HL3U covering-fibs 0
# of HL3U fibs with adj failures 0
Fibs of Prefix length 0, with TCAM fails: 0
Fibs of Prefix length 1, with TCAM fails: 0
Fibs of Prefix length 2, with TCAM fails: 0
Fibs of Prefix length 3, with TCAM fails: 0
Fibs of Prefix length 4, with TCAM fails: 0
Fibs of Prefix length 5, with TCAM fails: 0
Fibs of Prefix length 6, with TCAM fails: 0
Fibs of Prefix length 7, with TCAM fails: 0
Fibs of Prefix length 8, with TCAM fails: 0
Fibs of Prefix length 9, with TCAM fails: 0
Fibs of Prefix length 10, with TCAM fails: 0
Fibs of Prefix length 11, with TCAM fails: 0
Fibs of Prefix length 12, with TCAM fails: 0
Fibs of Prefix length 13, with TCAM fails: 0
Fibs of Prefix length 14, with TCAM fails: 0
Fibs of Prefix length 15, with TCAM fails: 0
Fibs of Prefix length 16, with TCAM fails: 0
Fibs of Prefix length 17, with TCAM fails: 0
Fibs of Prefix length 18, with TCAM fails: 0
Fibs of Prefix length 19, with TCAM fails: 0
Fibs of Prefix length 20, with TCAM fails: 0
Fibs of Prefix length 21, with TCAM fails: 0
```

```
Fibs of Prefix length 22, with TCAM fails: 0
Fibs of Prefix length 23, with TCAM fails: 0
Fibs of Prefix length 24, with TCAM fails: 0
Fibs of Prefix length 25, with TCAM fails: 0
Fibs of Prefix length 26, with TCAM fails: 0
Fibs of Prefix length 27, with TCAM fails: 0
Fibs of Prefix length 28, with TCAM fails: 0
Fibs of Prefix length 29, with TCAM fails: 0
Fibs of Prefix length 30, with TCAM fails: 0
Fibs of Prefix length 31, with TCAM fails: 0
Fibs of Prefix length 32, with TCAM fails: 693
Fibs of Prefix length 33, with TCAM fails: 0
```

This output shows 693 failures. You can use this statistic to determine how many additional TCAM resources are needed to hold the routes being advertised in the network at this time.

To view the number of the number of route entries used by each routing protocol, enter the **show ip route summary** privileged EXEC command.

```
Switch# show ip route summary
IP routing table name is Default-IP-Routing-Table(0)
IP routing table maximum-paths is 32
Route Source    Networks    Subnets     Overhead    Memory (bytes)
connected       5           0           320         760
static          0           0           0           0
rip             0           2390        152960      363280
internal        1                                   1172
Total           6           2390        153280      365212
Switch#
```

Resolving TCAM Utilization Issues

We recommend these solutions:

- Changing the SDM Template
- Optimizing IP Routes

Changing the SDM Template

The switch database management (SDM) template allocates the limited TCAM resources for different forwarding types. To resolve TCAM utilization issues, you should choose the appropriate SDM template for the switch application.

Enter the **show sdm prefer** privileged EXEC command to see the active SDM template on a switch:

```
Switch# show sdm prefer
 The current template is "desktop default" template.
 The selected template optimizes the resources in
 the switch to support this level of features for
 8 routed interfaces and 1024 VLANs.

  number of unicast mac addresses:                  6K
  number of IPv4 IGMP groups + multicast routes:    1K
  number of IPv4 unicast routes:                    8K
    number of directly-connected IPv4 hosts:        6K
    number of indirect IPv4 routes:                 2K
  number of IPv4 policy based routing aces:         0
  number of IPv4/MAC qos aces:                      0.5K
  number of IPv4/MAC security aces:                 1K
```

The default template on this switch allows only 2048 indirect Layer 3 routes in the TCAM. To allocate more TCAM resources to indirect Layer 3 routes, you must reduce some other TCAM resources. Use the **sdm prefer** *template-name* global configuration command to change to a template that reserves more resources for IP routing.

To see a list of available SDM templates for your switch, enter the **show sdm templates all** privileged EXEC command.

```
Switch# show sdm templates all
 Id  Type        Name
  0  desktop     desktop default
  1  desktop     desktop vlan
  2  desktop     desktop routing
  3  aggregator  aggregate default
  4  aggregator  aggregate vlan
  5  aggregator  aggregate routing
  6  desktop     desktop routing pbr
  8  desktop     desktop IPv4 and IPv6 default
  9  desktop     desktop IPv4 and IPv6 vlan
 10  aggregator  aggregate IPv4 and IPv6 default
 11  aggregator  aggregate IPv4 and IPv6 vlan
 12  desktop     desktop access IPv4
 13  aggregator  aggregator access IPv4
 14  desktop     desktop IPv4 and IPv6 routing
 15  aggregator  aggregator IPv4 and IPv6 routing
 16  desktop     desktop IPe
 17  aggregator  aggregator IPe
```

Note See the chapter on "Configuring SDM Templates" in the switch software configuration guide to see the templates available on your switch and the reserved TCAM resources for each template.

Optimizing IP Routes

When it is not possible or practical to change the SDM template on a Layer 3 switch, you can reduce the number of routes in the TCAM by using summary routes or by filtering routes.

Using summary routes reduces the routing table size. You enable summary routes on peer routers. Route summary is enabled by default for RIP and EIGRP and disabled by default for OSPF. To learn about route summary, see the "Configuring IP Unicast Routing" chapter in the software configuration guide (only Layer 3 switches).

You can use route filtering to prevent unwanted routes from being programmed into the TCAM. For information about OSPF route filtering, see the feature guide at this URL:
http://www.cisco.com/en/US/docs/ios/12_0s/feature/guide/routmap.html

Debugging Active Processes

If the CPU utilization percentage is high and the interrupt percentage is low, the high CPU utilization is caused by one or more system processes consuming the CPU resource. This is less common than high CPU utilization caused by the receipt of network packets. When a system process is consuming most of the CPU resources, an event usually triggered the process to become active. Review the syslog for any unusual events.

Table 3 lists some processes that consume CPU resources with normal and high percentages, possible root cause for the process, and suggested actions.

Table 3 System Processes that Consume CPU Resources

Process Name	Percentage of Active Resources		Possible Root Cause	Suggested Action
	Normal	Considered High		
Hulc LED Process	0 to 30%	24 ports or less: More than 20% 48 ports: More than 30%	Physical link flapping.	Review the syslog for a physical link losing and gaining connectivity.
Inline Power Twt	0	More than 5 %	Bad power supply.	Review the syslog for reports of a failed power controller.
HACL	0	More than 50 %	Too many ACLs configured on the switch in a short time period. This can occur when the ACLs are being applied in an automated fashion (from a script).	Consider changing the SDM template.
SNMP Engine	0	More than 40%	See the "SNMP Engine Process" section on page 20.	

SNMP Engine Process

The SNMP engine system process is only active when the switch is receiving SNMP queries. The required CPU time is directly proportional to the number of SNMP query packets received. Each received SNMP query packet is processed at the interrupt level before it is forwarded to the SNMP engine system process. When the SNMP engine process is busy, the interrupt percentage shown in the output of the **show processes cpu sorted** command also shows some non-zero interrupt percentages. The interrupt percentage is minor compared to the percentage of CPU utilized by the SNMP engine system process.

It is important to determine the switch baseline CPU utilization when evaluating CPU utilization for the SNMP engine system process. The switch typically receives SNMP queries at regular intervals, and the SNMP Engine system process consumes CPU resources handling the queries. The intervals are shown in the output for the **show processes cpu history** command. See the examples in the "When High CPU Utilization Is a Problem" section on page 2.

These scenarios can cause heavy CPU usage by the SNMP engine system process:

- Multiple servers simultaneously performing an SNMP query.
- SNMP query of the flash file system on the switch. Accessing the flash file is a CPU-intensive operation for SNMP Gets or SNMP GetNext operations.
- A complete or partial SNMP MIB walk.

Helpful Information

CPU Receive Queues

A packet sent to the CPU by switch hardware goes into one of 16 CPU queues, depending on the packet type. Each queue is given a priority, allowing the CPU to drain higher priority queues before lower priority queues. Each queue has some reserved memory in hardware to hold packets for the queue so that one queue or packet type cannot use all the available memory.

These are the CPU queues with their uses:

- rpc—Remote procedure call. Used by Cisco system processes to communicate across the stack.
- stp—Spanning Tree Protocol. A Layer 2 protocol with its own queue and used for protocol packets such as LACP and so on.
- ipc—Interprocess communication. Used by Cisco system processes to communicate across the stack.
- routing protocol—Used for routing protocol packets received by other network devices.
- L2 protocol—Used for protocol packets such as UDLD, and so on.
- remote console—Used for packets when you enter the **session** *switch-number* privileged EXEC command on a stack master switch to open the console on another switch member.
- sw forwarding—Used for packets punted by the hardware for the CPU to route.
- host—Used for packets with a destination IP address matching any switch IP address. Also IP broadcast packets.
- broadcast—Receives Layer2 broadcast packets.
- cbt-to-spt—Receives multicast packets for PIM_SM.
- igmp snooping—A queue for IGMP packets.
- icmp—A queue for ICMP redirect packets.
- logging—Used for receive packets generated by hardware for ACL logging.
- rpf fail—A queue for reverse path forwarding failures.
- dstats—drop stats. Not used during normal operation.
- cpu heartbeat—Used by the CPU receiving the packets it sends to itself.

Commands Used for CPU Troubleshooting

Table 4 show and debug Commands for CPU Troubleshooting

Command	Purpose	Usage
show controllers cpu-interface	Shows packet counts for all CPU receive queues.	Identify the type of packets that are flooding the CPU.
show ip route summary	Shows the number of route entries used by each protocol.	Use to help determine if additional TCAM resources are needed for IP routing.
show ip traffic	Shows a count of IP packet types received by the switch.	A rapidly incrementing packet type indicates that packet type is flooding the IP stack.

Table 4 show and debug Commands for CPU Troubleshooting

Command	Purpose	Usage
show platform ip unicast counts	Shows routes that are not programmed into the TCAM.	Use to determine how many additional TCAM resources are needed to hold the current network routes.
show platform ip unicast statistics	*CPUAdj* output identifies punted packets.	Enter the command several times. If the *CPUAdj* value increments rapidly, packets are being punted from switch hardware.
show platform port-asic stats drop	Shows CPU packets discarded due to congestion.	Identify CPU receive queues that are dropping packets due to flooding.
show platform tcam utilization	Shows TCAM maximum capacity and usage.	Determine if the TCAM is full. If the *IPv4 unicast indirectly-connected routes* output is at maximum, the IP routing database is full, and packets to other IP addresses are punted.
show processes cpu history	Shows a history of CPU utilization for 60 seconds, 60 minutes, and 72 hours.	Determine baseline CPU usage, and identify when spikes occur.
show processes cpu sorted [**5sec**]	Shows percentages for CPU utilization and CPU time spent on interrupts and lists the most active system processes in order of CPU utilization.	Determine if a CPU utilization problem is the result of too many network packets or of an active system process running on the switch.
show sdm templates all	Lists the SDM templates available on the switch.	Enter to determine if you can reallocate TCAM resources as needed.
debug platform cpu-queue *queue*	Debugs CPU queues.	Enter the command for each suspect CPU receive queue. The console floods at the problem queue.

Additional Documents

Another document on Cisco.com focuses on specific high utilization issues in the Catalyst 3750 switch, although the information also applies to other switches. See *Catalyst 3750 Series Switches High CPU Utilization Troubleshooting.*

Unresolved Issues

If the troubleshooting steps in this document do not help you to determine the root cause of high CPU utilization, you should contact the Technical Assistance Center (TAC) for Cisco. The technical assistance engineer will want to see the same information that you have gathered in the debugging efforts. Have this information ready when you contact Cisco technical support to reduce the time to resolve the problem.

Note See the next section for a link for submitting a service request.

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, submitting a service request, and gathering additional information, see the monthly *What's New in Cisco Product Documentation*, which also lists all new and revised Cisco technical documentation, at:

http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html

Subscribe to the *What's New in Cisco Product Documentation* as a Really Simple Syndication (RSS) feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service and Cisco currently supports RSS Version 2.0.