Troubleshoot Memory Issues in IOS XE via Callsite

Contents

Introduction

This document describes how to troubleshoot memory issues in Cisco IOS® XE based devices like routers and switches for a leaking callsite.

Prerequisites

Knowledge in memory management in Cisco IOS XE software based devices.

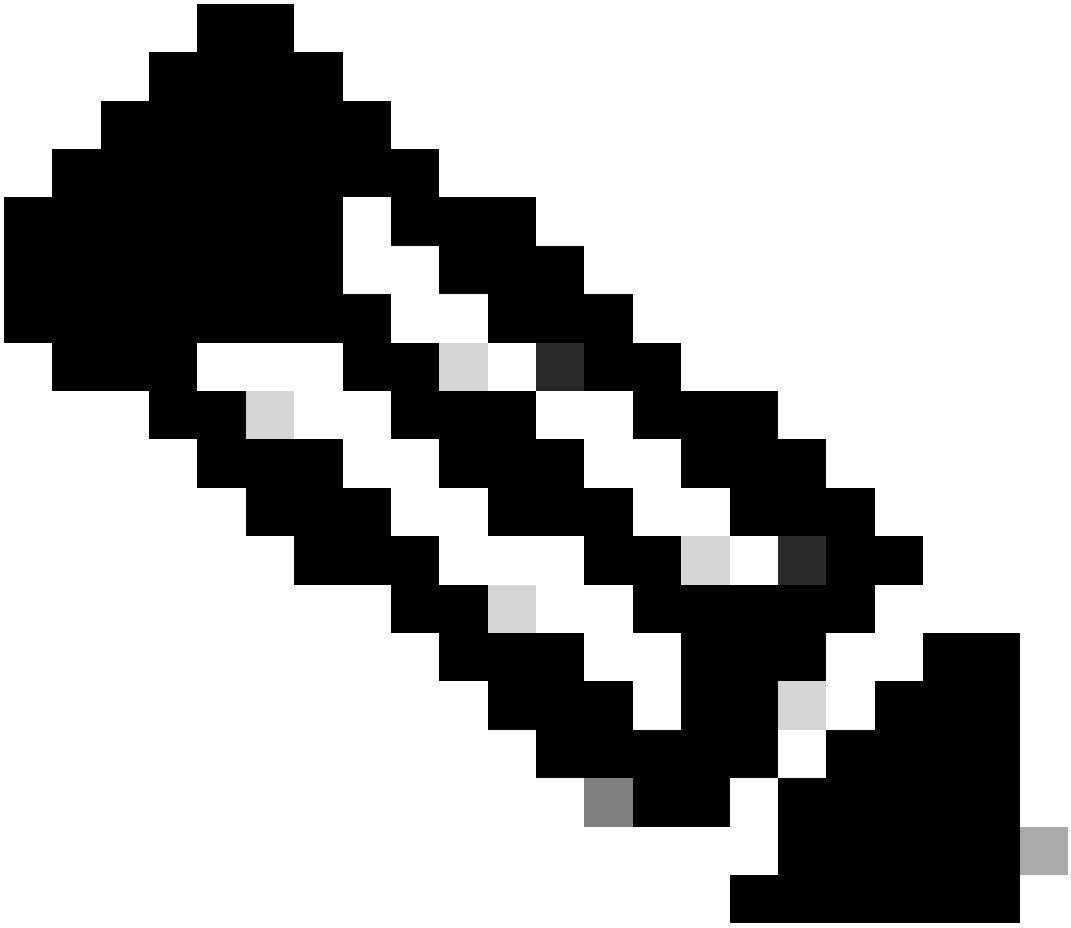
Components Used

This document is not restricted to specific software and hardware versions. It applies for routing and switching Cisco IOS XE software based platforms.

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, ensure that you understand the potential impact of any command.

Background Information

Monitoring production memory usage of the device for delta increments and confirming if it is expected is time-consuming. This document explains what a callsite is and how it helps troubleshoot memory issues quickly.



Note: This document is mainly focused on routing processor Dynamic Random-Access Memory (DRAM) memory usage troubleshoot.

What is the Callsite?

The callsite is a tag that is used by Cisco Technical Assistance Center (TAC) to verify and track which functions of source code are being called during memory allocations made by Cisco IOS-XE related processes.

Customers can provide this tag ahead of opening a TAC case for faster resolution and also customers can help in debugging it by the commands presented later on in this article.

Diff Calls and Diff Bytes

Diff calls monitor the disparity between the number of calls for memory allocations and deallocations. Typically, a high volume of diff calls can signify a memory-related problem. This occurs when there are excessive amounts of diffs, indicating that the system is not releasing memory and allocations are accumulating.

Both, diff calls and diff bytes can be seen with command**show processes memory platform accounting**:

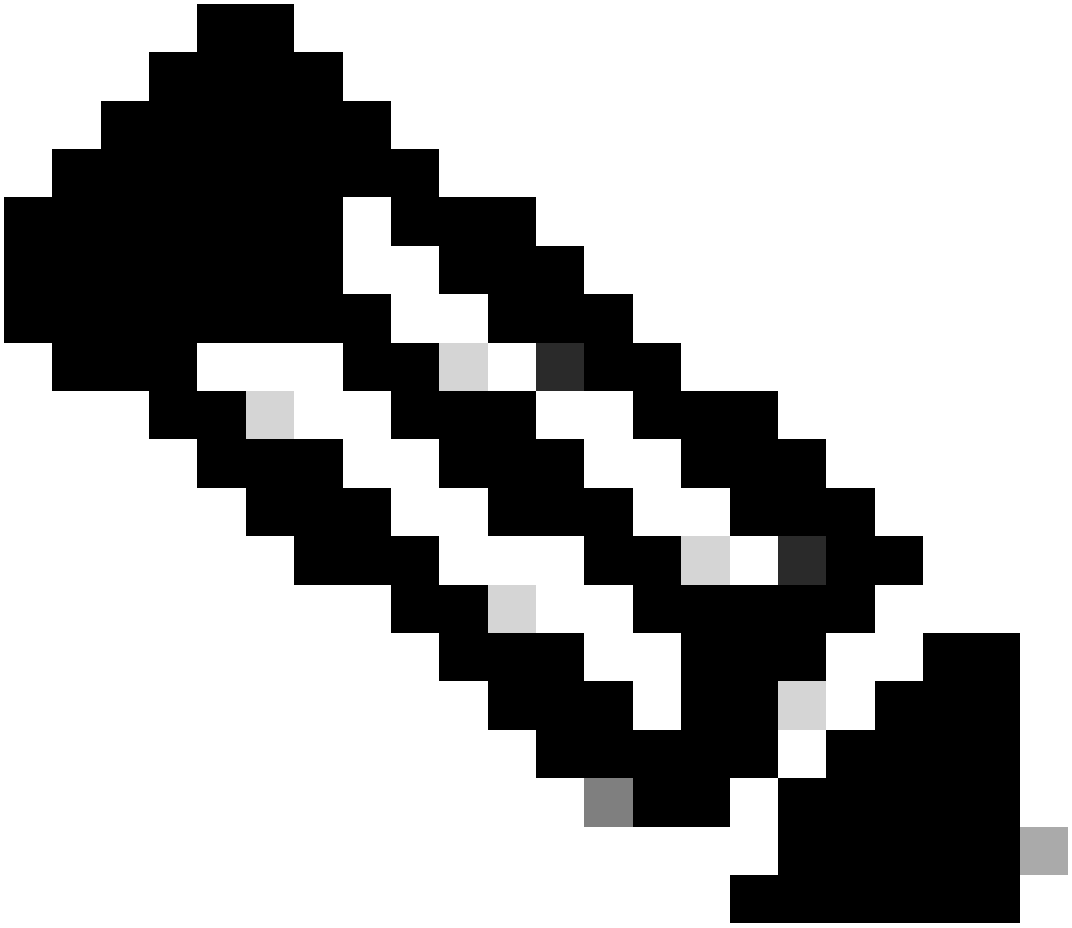
```
test1#show processes memory platform accounting
  Hourly Stats

  process                   callsite_ID(bytes)          max_diff_bytes    callsite_ID(calls)   max_diff
  ------------------------------------------------------------------------------------------------------
  sessmgrd_rp_0               F8E78C86E08C8003                 1579683      E6A19D3ED0064000
  cli_agent_rp_0              A5E99693AA3B8004                 1268440      5D11C89CA87A8003
  smand_rp_0                  3DFF8F3C424F400A                  918144      C34A609190E3C001
  <snip>
```

The system has internal memory usage thresholds that trigger memory warnings and critical-level syslogs. The percentage of memory usage based on these thresholds can be viewed using the command show platform resources.

```
test1#show platform resources
**State Acronym: H - Healthy, W - Warning, C - Critical
Resource                Usage                Max             Warning         Critical        State
-------------------------------------------------------------------------------------------------
RP0 (ok, active)                                                                             H
 Control Processor      1.17%                100%            80%             90%             H
  DRAM                  2639MB(34%)          7753MB          88%             93%             H
  bootflash             856MB(13%)           6338MB          88%             93%             H
  harddisk              0MB(0%)              0MB             88%             93%             H
ESP0(ok, active)                                                                             H
 QFP                                                                                         H
  TCAM                  10cells(0%)          131072cells     65%             85%             H
  DRAM                  89761KB(2%)          3670016KB       85%             95%             H
  IRAM                  13525KB(10%)         131072KB        85%             95%             H
  CPU Utilization       1.00%                100%            90%             95%             H
  Crypto Utilization    3.00%                100%            90%             95%             H
  Pkt Buf Mem (0)       67KB(0%)             524288KB        85%             95%             H

test1#
```



Note: File a TAC case to determine if the diff calls or diff bytes is concerning for a particular process. Generally, if there is low free system memory as visible with the command **show processes memory platform sorted**,it is worth checking further.

Symptoms for Memory Consumption or Leak Issues

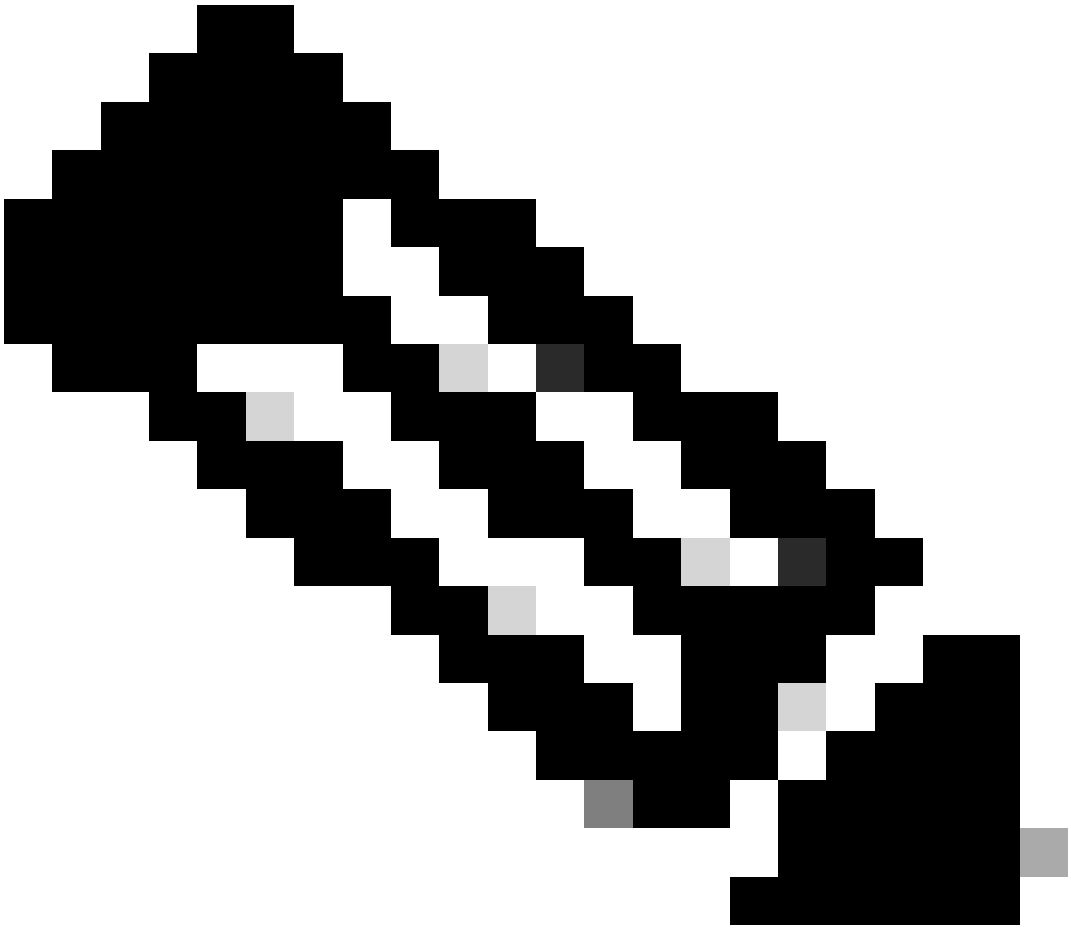
When there is a memory consumption or leak issue in Cisco IOS XE side, usually a warning or critical alarm is generated, as example:

```
Nov 22 11:37:16.770: %PLATFORM-4-ELEMENT_WARNING: RO/0: smand: RP/0: Used Memory value 89% exceeds warn
```

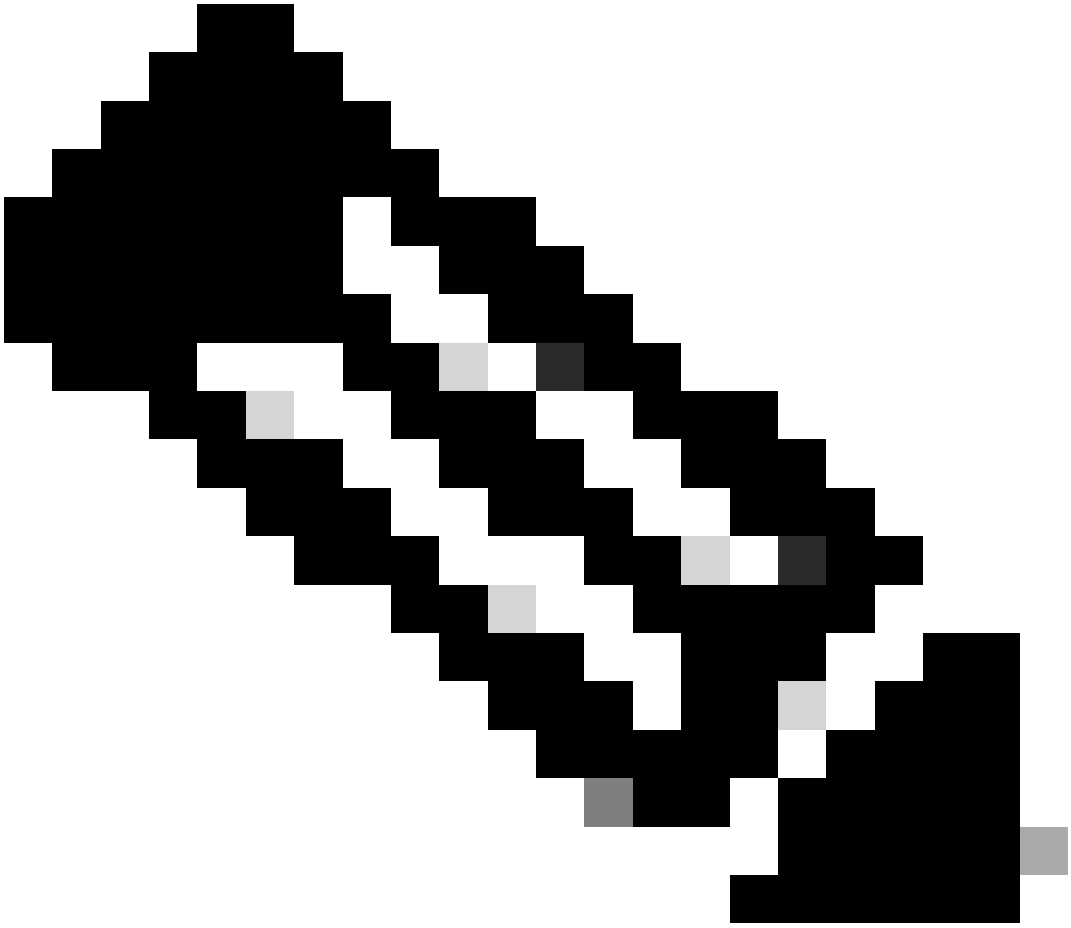
This type of alarm highlights valuable information as starting point for the troubleshooting:

- Date and time of the alarm received
- Percentage usage
- Component impacted

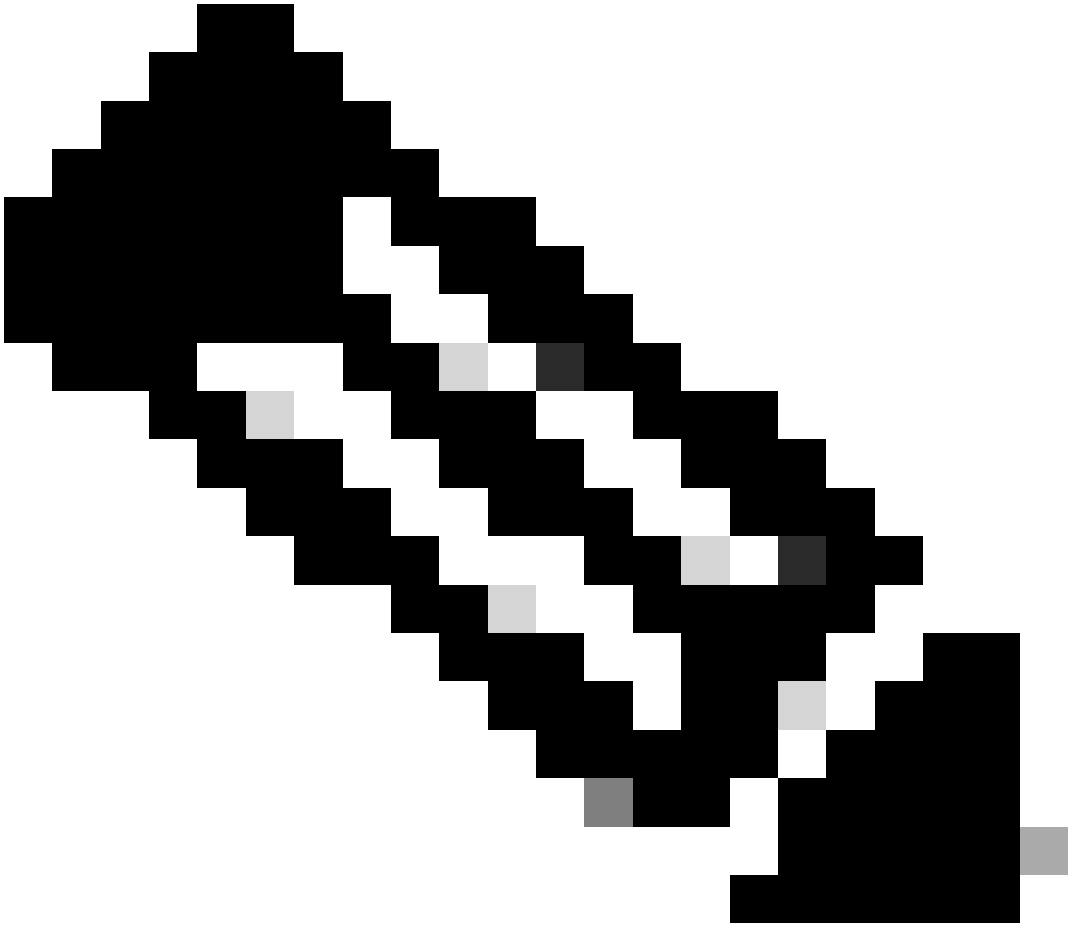
- Top Cisco IOS XE processes that system detected as top consumers based on diff calls.



Note: The %PLATFORM-4-ELEMENT_WARNING alarm is not necessarily a conclusive data point to get the Root Cause Analysis (RCA) of a memory consumption issue.



Note: There are other type of symptoms and memory usage alarms associated to different components like Temporal File System (TMPFS), Quantum Flow Processor (QFP) and Cisco IOS daemon (IOSd), however these are out of scope of this document.

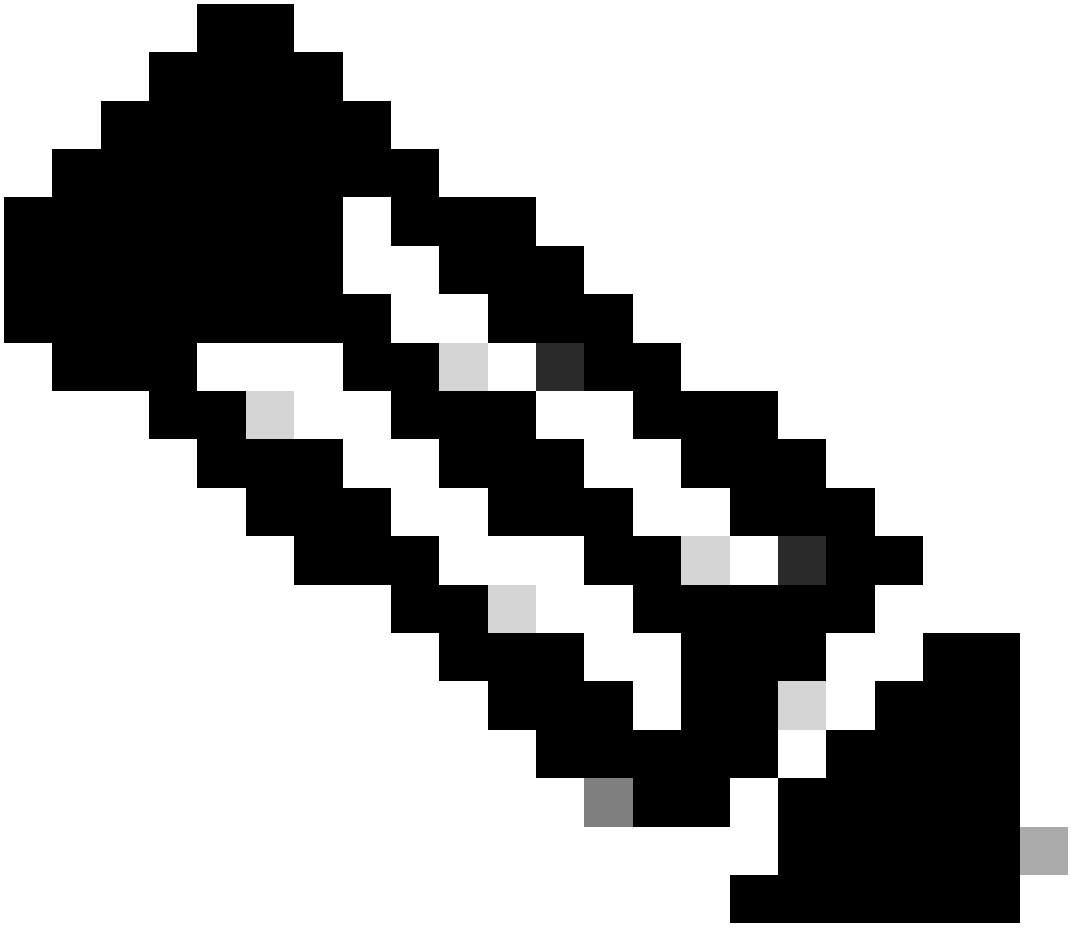


Note: This document does not cover the troubleshoot of SYS-2-MALLOCFAIL syslog messages that indicates memory issue under Cisco IOS daemon (IOSd).

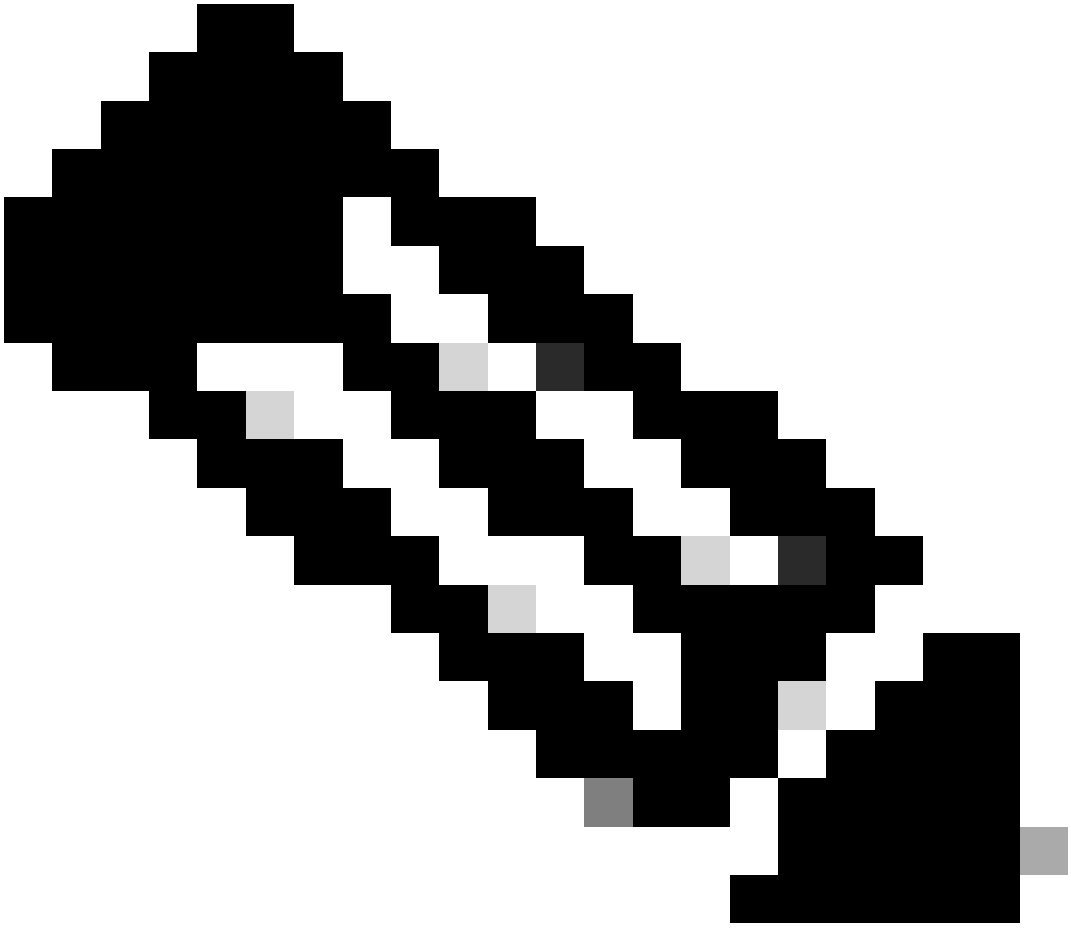
Troubleshooting Scenarios

Device has Crashed due to Out of Memory

When device crashes due to out of memory resources, it is important to verify the last logs before the crash in order to confirm and see if the syslog message **%PLATFORM-4-ELEMENT_WARNING: R0/0: smand: RP/0: Used Memory value X% exceeds warning level Y% is present.**



Note: Note that syslogs from local DRAM buffer are erased after a crash due to out of memory, hence checking archive logs from the syslog server prior the crash event is needed. If syslog server is not setup yet, refer to How to configure logging in Cisco IOS.

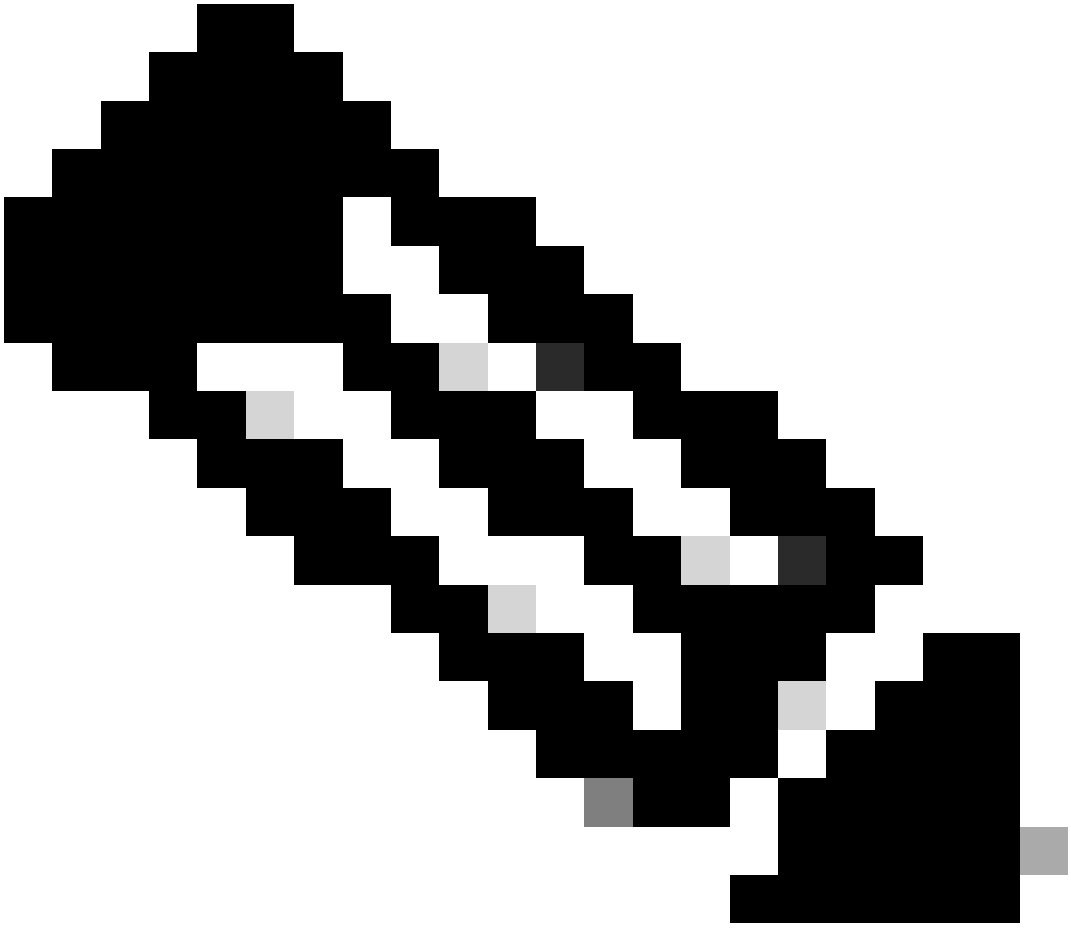


Note: The %PLATFORM-4-ELEMENT_WARNING: R0/0: smand: RP/0: Used Memory value X% exceeds warning level Y% alarm after a crash event can also be seen in the decoded Cisco IOS tracelogs. Refer to Collect and Manage Trace Logs with Unified Logging Enhancement for more information.

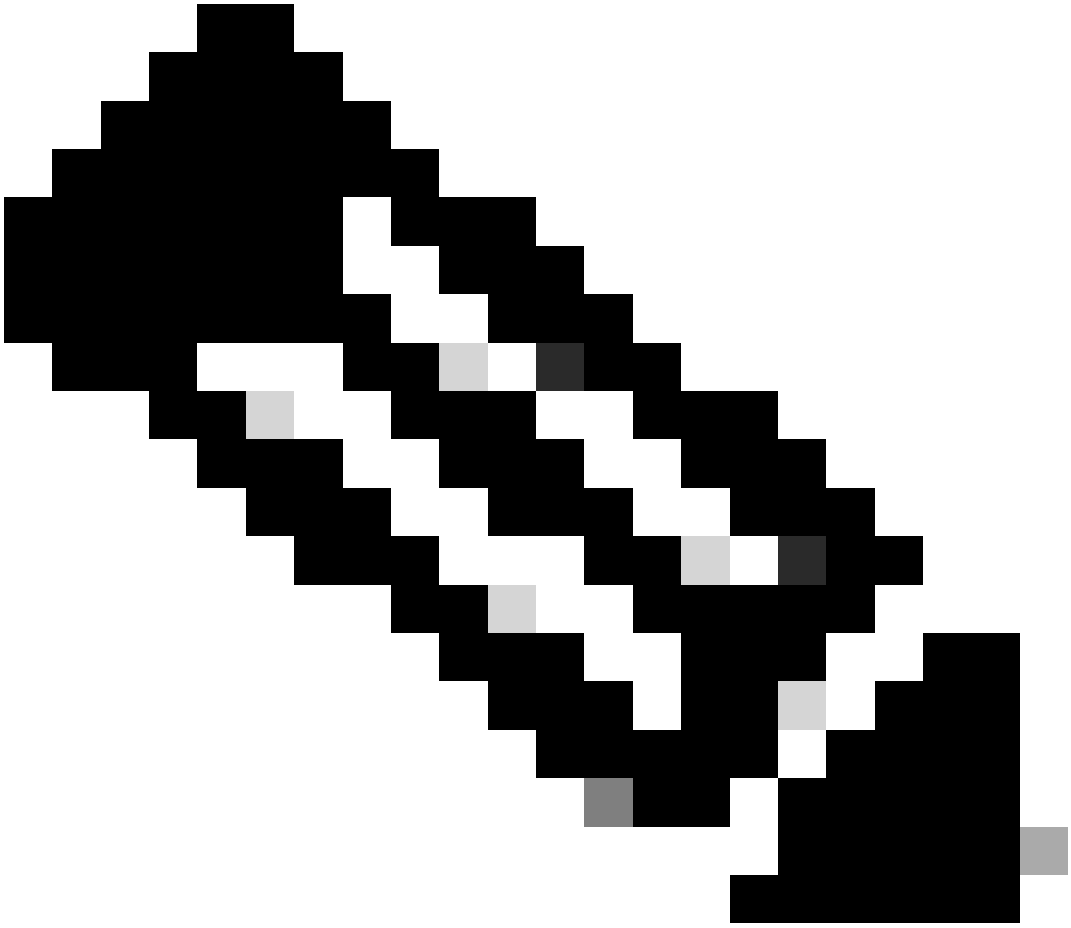
Due to insufficient memory, the system experienced a crash. Consequently, a system report is generated. This report is a .tar.gz file containing pertinent data that can be utilized to investigate the memory issue. Refer to Troubleshoot using system reports for more information.

When decompressed, the system report contains a directory called **maroon stats** within the **tmp** directory. The maroon stats is a serviceability facility implemented in code that tracks memory allocations and deallocations in diff calls and bytes for different Cisco IOS XE processes.

The maroon statistics snapshot contain within the system report, helps to identify a potential culprit callsite to determine the memory consumption or leak issue RCA or debug it further and understand it better.



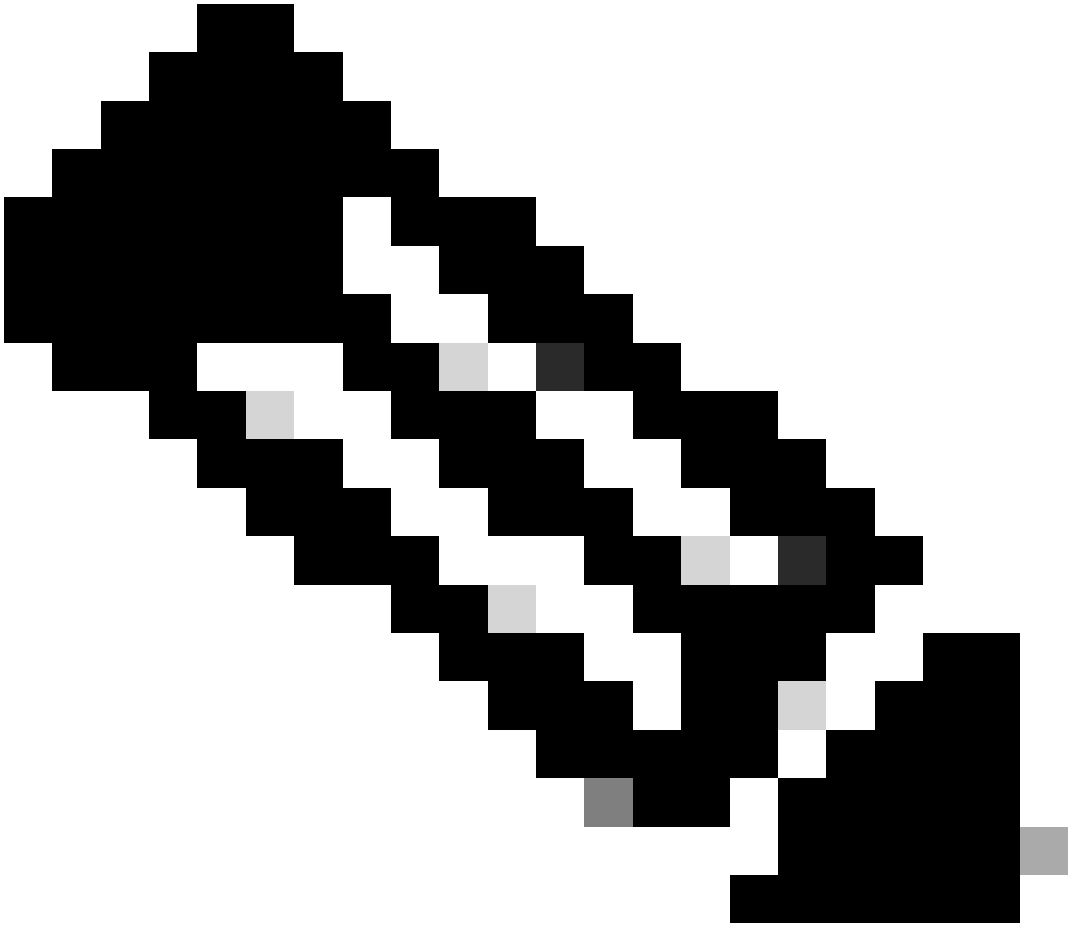
Note: Decoding the maroon stats directory from a system report can only be done by TAC as it contains internal and confidential functions of code that helps the TAC engineer to understand which functions of code are allocating the memory. Please file a TAC case and provide the system report.



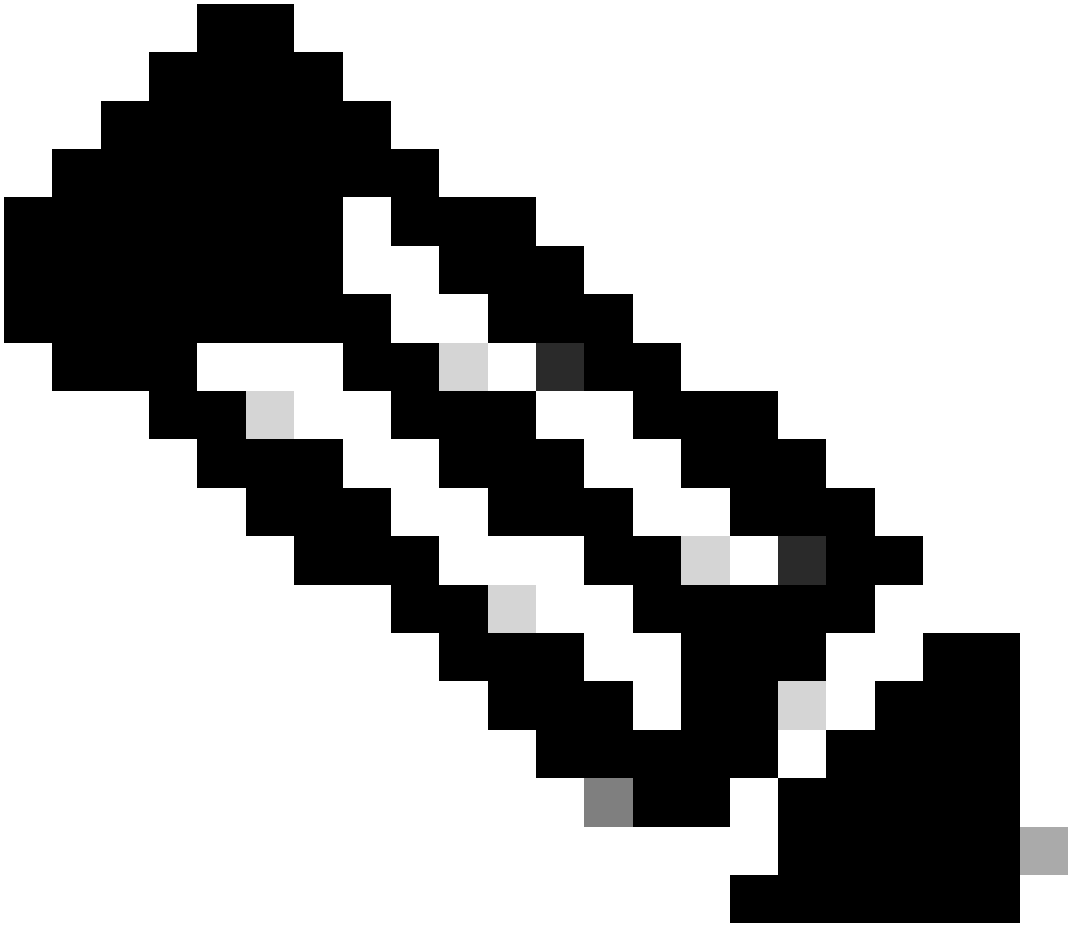
Note: Keep in mind that system report provides good amount of data to understand a memory crash due to out of memory, however, in some cases, further memory tracking, monitoring, debug and troubleshoot is needed.

Device has not Crashed yet but has Memory Usage Warnings

The command **show platform resources**, shows warning and critical memory usage thresholds.



Note: It is best practice to gather memory related output commands to debug further as depending on how fast the memory consumption or leak can happen, device can be at risk of crashing due to out of memory resources.



Note: When memory usage warnings are seen, you can file a TAC case and provide commands **show tech-support** and

show tech-support memory which helps the TAC engineer to initially triage the issue and potentially find a RCA promptly.

When device has not crashed yet and it is generating the memory alarms in the local syslog buffer or received from the syslog server via the monitoring tool, gather the output of **show processes memory platform sorted** to determine the bytes consumed by the offending process if any.

```
Router#show processes memory platform sorted
System memory: 4027884K total, 2580612K used, 1447272K free,
Lowest: 1447272K
   Pid    Text      Data   Stack   Dynamic       RSS     Total              Name
--------------------------------------------------------------------------------
 21240  263436    858000     136       308    858000   3632460   linux_iosd-imag
 27232   12877    195460     136     23592    195460   2231316     fman_fp_image
 26797      90    157260     136     22308    157260   1741996        cpp_cp_svr
 19194    7325    102756     136      2376    102756   1318608           fman_rp
 27179   18745    242708     136       448    242708   1160248    qfp-ucode-utah
```

```
<snip>
```

In this output, look at the **Resident Set Size (RSS)** column. This is an indicator of how many kilobytes each Cisco IOS XE process is consuming.

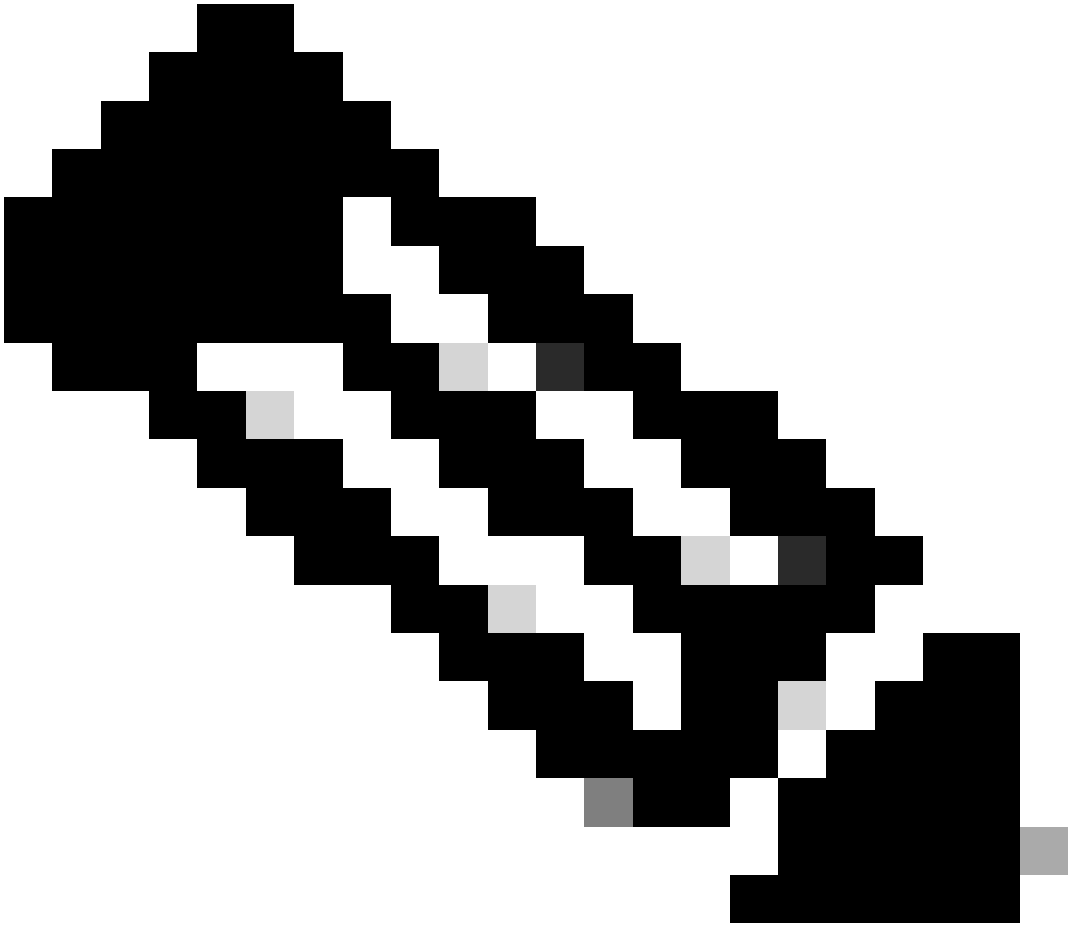
Identify the Callsite

Next, gather the output of **show processes memory platform accounting** which shows the diff calls and bytes values for different processes. Usually, we focus on the bigger values.

The diff call bytes is a good indicator to determine if there can be a potential memory leak as it shows how much bytes of memory are still hold by the system by a process without being released back to the system.

Based on this data, you can identify which is the callsite tag from the offending process which has the bigger diff calls and bytes values.

The **show process memory platform accounting** tracks these diff calls and bytes over time. In some cases, a backtrace is included at the bottom of this command output. This is important for TAC engineer as such backtrace is decoded using internal tools and helps in determining which functions of code can be causing a potential memory leak.



Note: Further debugging for a process is often needed if the command **show process memory platform accounting** does not provide enough information to troubleshoot a memory leak issue.

See also **Debug the Callsite** from this document for a secondary method to identify the callsite.

Process Memory Consumption in Modules, Databases and Messaging

Gathering these commands for a specific Cisco IOS XE process can be needed to further debug a Cisco IOS XE process memory leak:

```
# Allocations and frees per module
show platform software memory <process> <location>
show platform software memory <process> <location> bri

# Database diff and entries statistics
show platform software memory database <process> <location>  | ex diff:0
show platform software memory database <process> <location> bri | ex _0_

# Messaging diff and entries statistics
```

```
show platform software memory messaging <process> <location> | ex diff:0
show platform software memory messaging <process> <location> brief | ex _0_
```

These command outputs complement the investigation of a memory leak caused by a process and are often required if the initial basic triage commands do not provide enough information.

Debug the Callsite

A secondary method to identify the callsite is to debug it. These commands are required:

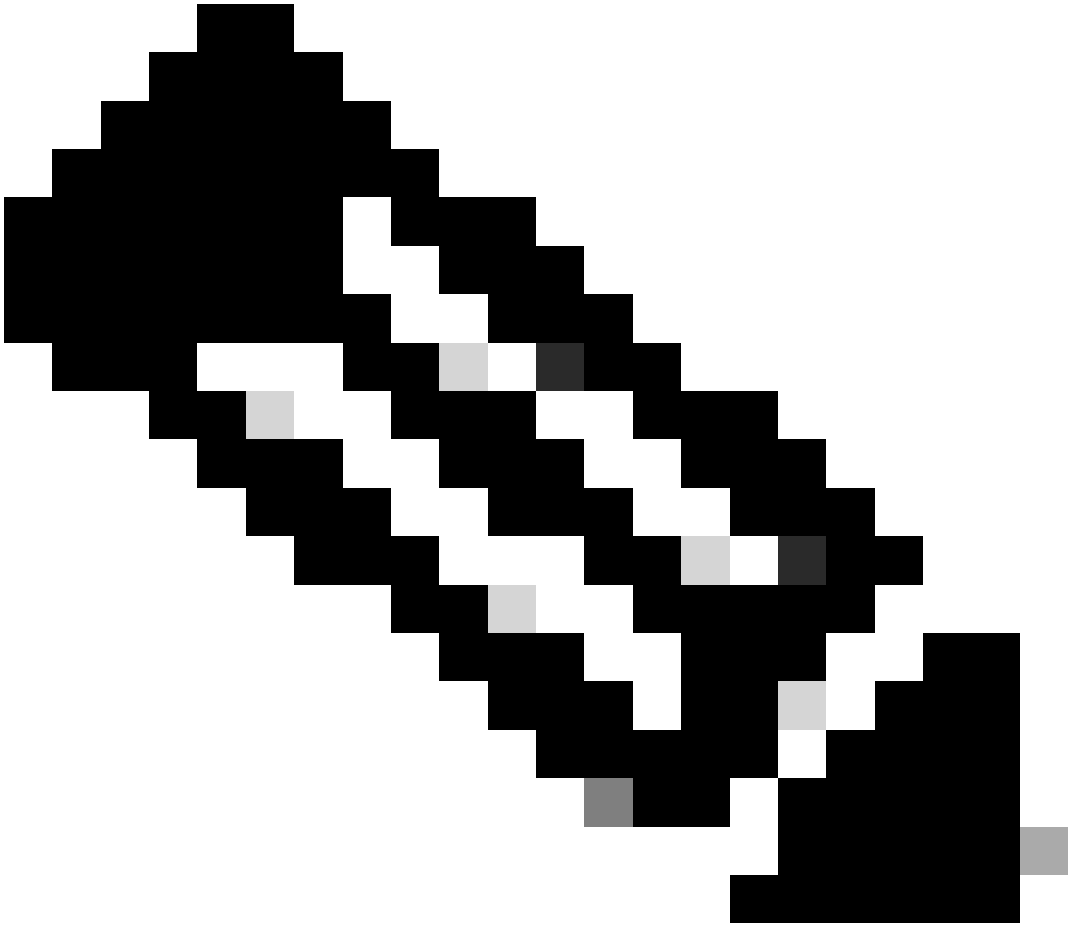
```
debug platform software memory <process> <location> alloc callsite start
show platform software memory <process> <location> alloc callsite brief
debug platform software memory <process> <location> alloc backtrace start <callsite> depth 10
show platform software memory <process> <location> alloc backtrace
```

The first command enables the debugging of allocations for the callsites of a process. In later versions, this command is enabled by default and it is not service impacting.

The **show platform software memory <process> <location> alloc callsite brief** command provides a table that shows the callsites for that process and the diff calls and bytes for each callsite. For example, here we show the output for the Cisco IOS process but it can be gathered for any other process:

```
test1# show platform software memory ios r0 alloc callsite brief
The current tracekey is   : 1#b1ba773f123f8d990fd84c82c1d0e1d3

  callsite            thread    diff_byte               diff_call
  ------------------------------------------------------------------
  3DFF8F3C424F4004    4115      57384                   1
  ABB2D8F932038000    4115      57360                   1
  3869885745FC8000    4115      16960                   1
  DF884D58A8EF0004    4115      8208                    1
  DF884D58A8EF0008    4115      8208                    1
  FAE69298A17B8000    4115      4243                    165
  FAE69298A17B8001    4115      2640                    165
  FAE69298A17B8002    4115      1958                    12
<snip>
```

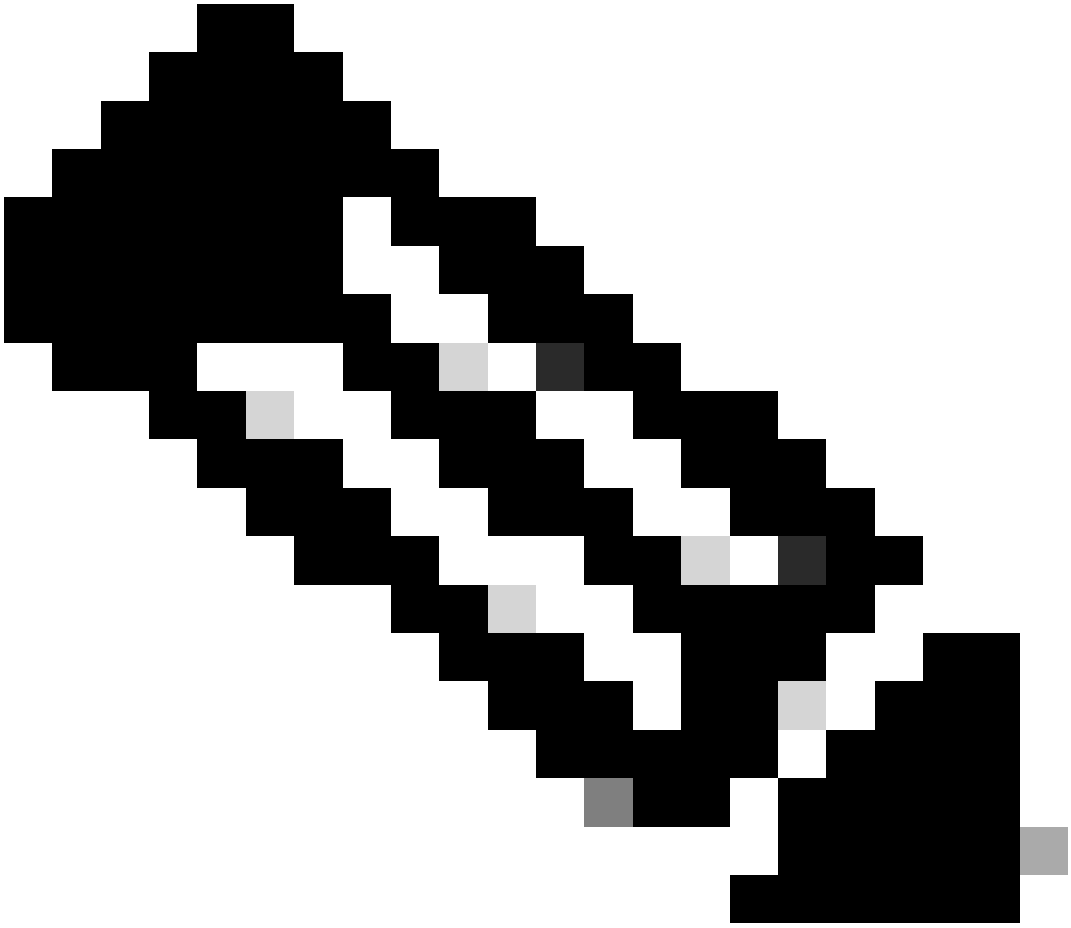


Note: The command **show plat soft memory <process> <location> alloc callsite bri** must be executed multiple times over time until finding the diff call or bytes column increasing as it would be an indicator that system is holding such memory without releasing it.

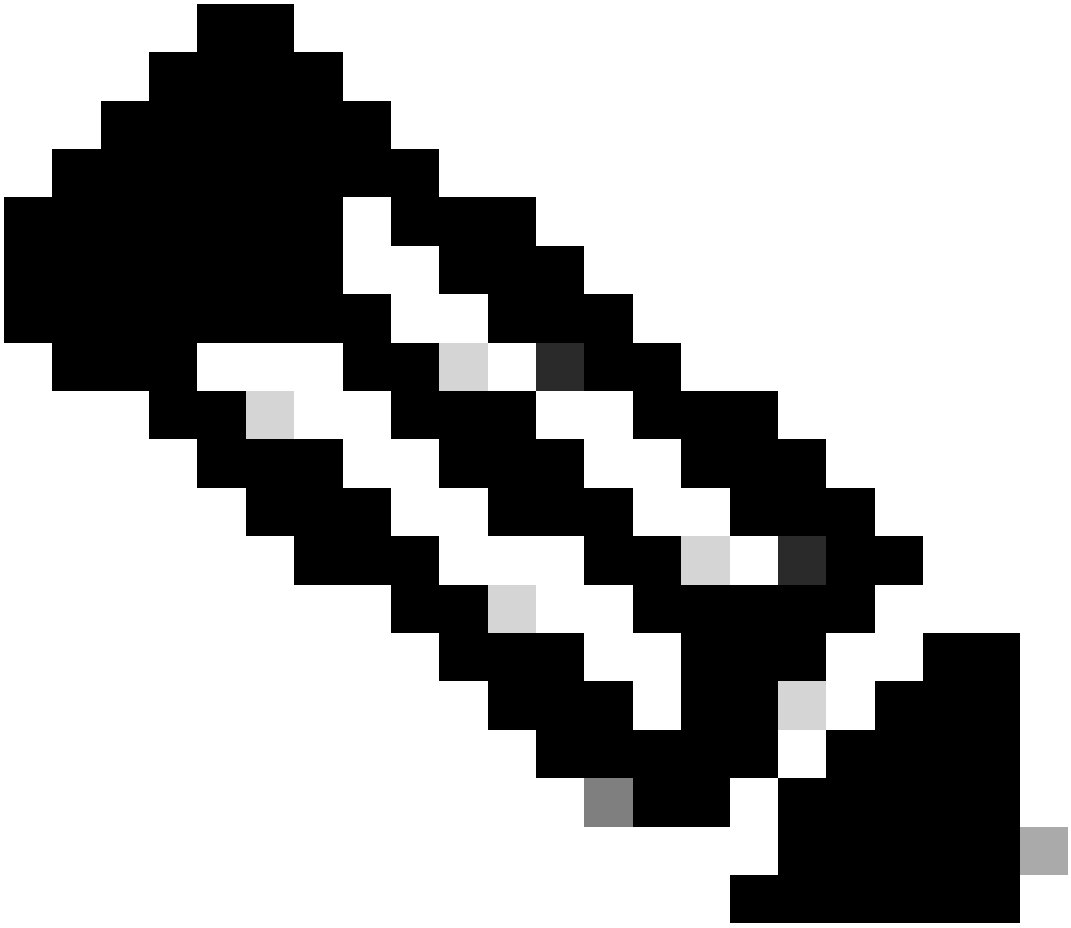
Once the callsite that has been identified to be leaking, the command **debug platform software memory <process> <location> alloc backtrace start <callsite> depth 10** must be executed for that callsite. This command can be left in place and is not service impacting.

Executing the command **show plat soft memory <process> <location> alloc callsite bri** again until seeing diff calls/bytes increases is still needed after enabling the debug of the identified callsite, this to track the functions of code allocating memory for that callsite. Later on, the backtrace can be gathered using **show platform software memory <process> <location> alloc backtrace**, for example:

```
show platform software memory install-manager switch active R0 alloc back
  backtrace: 1#83e58872a4792de086bf7191551098d7  maroon:7FCBACB87000+4642 maroon:7FCBACB87000+579C repm
  callsite: 7BD5593C00E30000, thread_id: 15556
  allocs: 70, frees: 0, call_diff: 70
```



Note: Provide this output to TAC for decoding the backtrace, then TAC engineer can verify the behavior in code, determine if there is an existing defect or better understand the behavior. If needed, TAC can reach out developer team.



Note: Ensure to have software up to date. In case a new software defect is found, TAC can work with developer team to further debug and investigate the condition.

Related Articles and Documentation

- ASR 1000 Series Router Memory Troubleshoot Guide
- Examine IOS-XE Memory Usage
- Report Problems of Memory Utilization via EEM
- Memory Troubleshooting Guide for Cisco 4000 Series ISRs