

# Troubleshooting Service Failures


- Identifying Memory Allocations for Processes, on page 1
- Identifying CPU Utilization for Processes, on page 2
- Monitoring Process Core Files, on page 3
- Processing the Crash Core Files, on page 3
- Clearing the Core, on page 3
- Enabling Auto-Copy for Core Files, on page 4


# Identifying Memory Allocations for Processes

You can identify the allocation, limit, memory allocation, and usage for each process in the memory. The following is a sample output from the **show processes memory** command. This output has been abbreviated to make the example more concise.

```
switch# show processes memory
 PID  MemAlloc MemLimit    MemUsed     StackBase/Ptr      Process
 ---- -------- --------    -------     -----------------  -----------
   1    159744  0           2027520     ff808d30/ffffffff  init
   2         0  0                 0                 0/0  kthreadd
   3         0  0                 0                 0/0  migration/0
   4         0  0                 0                 0/0  ksoftirqd/0
   5         0  0                 0                 0/0  watchdog/0
   6         0  0                 0                 0/0  migration/1
   7         0  0                 0                 0/0  ksoftirqd/1
   8         0  0                 0                 0/0  watchdog/1
   9         0  0                 0                 0/0  migration/2
  10         0  0                 0                 0/0  ksoftirqd/2
  11         0  0                 0                 0/0  watchdog/2
  12         0  0                 0                 0/0  migration/3
  13         0  0                 0                 0/0  ksoftirqd/3
  14         0  0                 0                 0/0  watchdog/3
  15         0  0                 0                 0/0  migration/4
  16         0  0                 0                 0/0  ksoftirqd/4
  17         0  0                 0                 0/0  watchdog/4
  18         0  0                 0                 0/0  migration/5
  19         0  0                 0                 0/0  ksoftirqd/5
  20         0  0                 0                 0/0  watchdog/5
  21         0  0                 0                 0/0  migration/6
  22         0  0                 0                 0/0  ksoftirqd/6
  23         0  0                 0                 0/0  watchdog/6
  24         0  0                 0                 0/0  migration/7
  25         0  0                 0                 0/0  ksoftirqd/7
  26         0  0                 0                 0/0  watchdog/7
```

```
27          0   0               0                 0/0   events/0
28          0   0               0                 0/0   events/1
29          0   0               0                 0/0   events/2
30          0   0               0                 0/0   events/3
31          0   0               0                 0/0   events/4
32          0   0               0                 0/0   events/5
33          0   0               0                 0/0   events/6
34          0   0               0                 0/0   events/7
35          0   0               0                 0/0   khelper
36          0   0               0                 0/0   netns
37          0   0               0                 0/0   kblockd/0
```

The **show processes memory** command includes the following keywords:

| Keyword | Description |
|---|---|
| > | Redirects the output to a file. |
| >> | Adds the output to an existing file. |
| shared | Displays shared memory information. |

# Identifying CPU Utilization for Processes

You can identify the CPU utilization for running process in the memory. The following is a sample output from the **show processes cpu** command. This output has been abbreviated to make the example more concise.

```
switch# show processes cpu

CPU utilization for five seconds: 0%/0%; one minute: 1%; five minutes: 2%

   PID    Runtime(ms) Invoked   uSecs 5Sec    1Min   5Min    TTY  Process
   ---    ----------------   ----- -----    -----  -----    ---  -----------
   1        28660    405831     70   0.00%   0.00%  0.00%   -    init
   2           21      1185     18   0.00%   0.00%  0.00%   -    kthreadd
   3          468     36439     12   0.00%   0.00%  0.00%   -    migration/0
   4        79725   8804385      9   0.00%   0.00%  0.00%   -    ksoftirqd/0
   5            0         4     65   0.00%   0.00%  0.00%   -    watchdog/0
   6          472     35942     13   0.00%   0.00%  0.00%   -    migration/1
   7        33967    953376     35   0.00%   0.00%  0.00%   -    ksoftirqd/1
   8            0        11      3   0.00%   0.00%  0.00%   -    watchdog/1
   9          424     35558     11   0.00%   0.00%  0.00%   -    migration/2
  10        58084   7683251      7   0.00%   0.00%  0.00%   -    ksoftirqd/2
  11            0         3      1   0.00%   0.00%  0.00%   -    watchdog/2
  12          381     29760     12   0.00%   0.00%  0.00%   -    migration/3
  13        17258    265884     64   0.00%   0.00%  0.00%   -    ksoftirqd/3
  14            0         2      0   0.00%   0.00%  0.00%   -    watchdog/3
  15        46558   1300598     35   0.00%   0.00%  0.00%   -    migration/4
  16      1332913   4354439    306   0.00%   0.00%  0.00%   -    ksoftirqd/4
  17            0         6      2   0.00%   0.00%  0.00%   -    watchdog/4
  18        45808   1283581     35   0.00%   0.00%  0.00%   -    migration/5
  19       981030   1973423    497   0.00%   0.00%  0.00%   -    ksoftirqd/5
  20            0        16      3   0.00%   0.00%  0.00%   -    watchdog/5
  21        48019   1334683     35   0.00%   0.00%  0.00%   -    migration/6
  22      1084448   2520990    430   0.00%   0.00%  0.00%   -    ksoftirqd/6
  23            0        31      3   0.00%   0.00%  0.00%   -    watchdog/6
  24        46490   1306203     35   0.00%   0.00%  0.00%   -    migration/7
```

```
  25      1187547   2867126     414   0.00%   0.00%  0.00%   -    ksoftirqd/7
  26            0        16       3   0.00%   0.00%  0.00%   -    watchdog/7
  27        21249   2024626      10   0.00%   0.00%  0.00%   -    events/0
  28         8503   1990090       4   0.00%   0.00%  0.00%   -    events/1
  29        11675   1993684       5   0.00%   0.00%  0.00%   -    events/2
  30         9090   1973913       4   0.00%   0.00%  0.00%   -    events/3
  31        74118   2956999      25   0.00%   0.00%  0.00%   -    events/4
  32        76281   2837641      26   0.00%   0.00%  0.00%   -    events/5
  33       129651   3874436      33   0.00%   0.00%  0.00%   -    events/6
  34         8864   2077714       4   0.00%   0.00%  0.00%   -    events/7
  35            0         8      23   0.00%   0.00%  0.00%   -    khelper
  36          234        34    6884   0.00%   0.00%  0.00%   -    netns
```

The **show processes cpu** command includes the following keywords:

| Keyword | Description |
|---|---|
| > | Redirects the output to a file. |
| >> | Adds the output to an existing file. |
| history | Displays information about the CPU utility. |
| sort | Sorts the list based on the memory usage. |

# Monitoring Process Core Files

You can monitor the process core files by using the **show cores** command.

```
switch# show cores
VDC  Module  Instance  Process-name     PID       Date(Year-Month-Day Time)
---  ------  --------  ---------------  --------  -------------------------
1    1       1         nginx            29017     2025-03-26 23:31:54
```

The output shows all cores that are presently available for upload from the active supervisor.

# Processing the Crash Core Files

You can process the crash core files by using the **show processes log** command.

```
switch# show process log
  Process            PID     Normal-exit  Stack-trace  Core     Log-create-time
  ----------------   ------  -----------  -----------  -------  ---------------
  ntp                919              N            N        N  Jun 27 04:08
  snsm               972              N            Y        N  Jun 24 20:50
```

# Clearing the Core

You can clear the core by using the **clear cores** command.

```
switch# clear cores
```

# Enabling Auto-Copy for Core Files

You can enter the **system cores** command to enable the automatic copy of core files to a TFTP server, the flash drive, or a file.

```
switch(config)# system cores tftp://10.1.1.1/cores
```