PW SKILLS | DevOps and Cloud Computing

System Performance Monitoring and Optimization in Linux

Objective

- Monitor CPU and memory usage using tools like top, htop, and free.
- Understand system load, context switching, CPU usage, and IO wait times.
- Identify and manage resource-intensive processes.
- Adjust process priorities using nice and renice for performance optimization.

PW SKILLS

‘top’ for real-time CPU and memory usage monitoring.

‘top’ command

The top command displays a real-time summary of system processes, allowing users to monitor CPU and memory usage, among other metrics. To start using it, simply open a terminal and type:

```
[Abhi@localhost ~]$ top
top - 10:45:05 up 6 min,  2 users,  load average: 0.00, 0.05, 0.04
Tasks: 124 total,   1 running, 123 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.3%us,  0.3%sy,  0.0%ni, 99.0%id,  0.0%wa,  0.0%hi,  0.3%si,  0.0%st
Mem:   1906908k total,   357536k used,  1549372k free,    21380k buffers
Swap:  4095992k total,        0k used,  4095992k free,   188828k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 1847 root      20   0  243m 4544 3508 S  0.7  0.2   0:03.52 vmtoolsd
 2587 Abhi      20   0 15028 1224  936 R  0.7  0.1   0:00.29 top
    4 root      20   0     0    0    0 S  0.3  0.0   0:00.19 ksoftirqd/0
    7 root      20   0     0    0    0 S  0.3  0.0   0:00.96 events/0
  247 root      20   0     0    0    0 S  0.3  0.0   0:00.17 mpt_poll_0
    1 root      20   0 19356 1572 1252 S  0.0  0.1   0:02.18 init
    2 root      20   0     0    0    0 S  0.0  0.0   0:00.01 kthreadd
    3 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
    5 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
```

‘top’ command

Common Options and Commands:
The ‘top’ command supports various options and interactive commands to customize its output:

Basic Commands:

- **Sort by CPU Usage:** Press Shift + P.
- **Sort by Memory Usage:** Press Shift + M.
- **Sort by Running Time**: Press Shift + T.
- **Kill a Process:** Press k, then enter the PID (Process ID).
- **Highlight Running Processes:** Press z to colorize the output for better visibility.
- **Change Refresh Interval:** Press d to set a new delay time between updates.

‘top’ command

Batch Mode:

To send output from the ‘top’ command to a file or another program, you can use:

```
top -b -n 1 > output.txt
```

This command runs ‘top’ in batch mode for one iteration and saves the output to a file named ‘output.txt’.

Pop Quiz

Q. What does the load average in the top command represent?

A

The average number of processes waiting to run

B

The average CPU usage over time

Pop Quiz

Q. What does the load average in the top command represent?

A

The average number of processes waiting to run

B

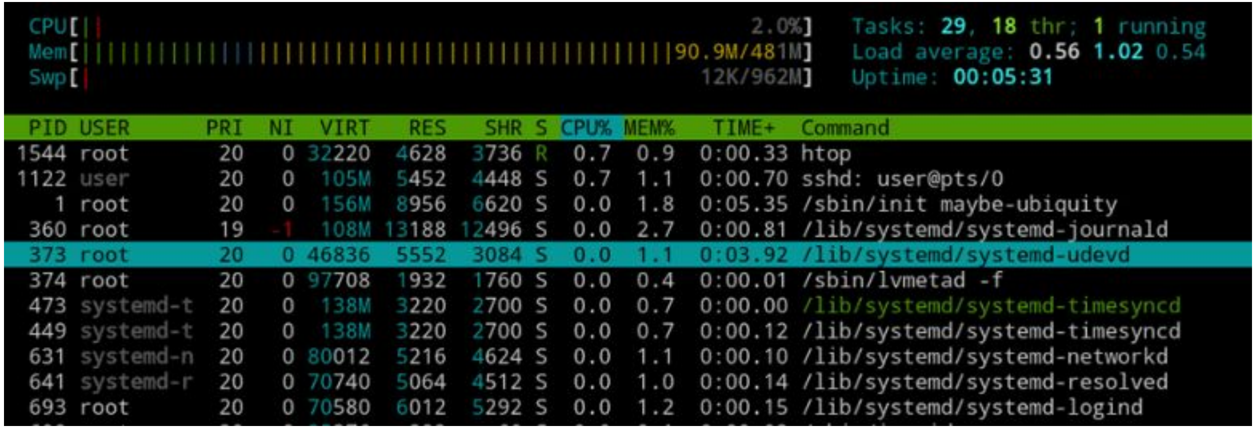
The average CPU usage over time

PW SKILLS

‘htop’ for enhanced process visualization and user-friendly controls

‘htop’ command

‘htop’ is an advanced, interactive process viewer for Linux that enhances the functionality of the traditional top command. It provides a more user-friendly interface, allowing users to monitor system processes in real-time with greater ease and clarity.



‘htop’ Command

Key Features of htop:

- Enhanced Visualization
- Interactive Controls
- Comprehensive Process List
- Process Management
- Customizable Layout

Pop Quiz

Q. What is htop primarily used for?

A

File management

B

Process management and system monitoring

Pop Quiz

Q. What is htop primarily used for?

A

File management

B

Process management and system monitoring

PW SKILLS

‘free’ command to check memory availability and swap usage

‘free’ Command

The ‘free’ command in Linux is a straightforward tool used to check memory availability and swap usage. It provides a quick overview of the system's memory status, including total, used, and free memory, as well as details about swap space.

How to Use the free Command:

1. Basic Command:
To display memory and swap usage, simply enter:

```
free
```

‘free’ Command

2. Human-Readable Format:
For easier interpretation, you can use the -h option to display the output in a human-readable format (e.g., MB, GB):

```
free -h
```

3. Display in Megabytes or Gigabytes:
You can specify the unit of measurement using options like ‘-m’ for megabytes or ‘-g’ for gigabytes:

```
free -m
```

```
free -g
```

‘free’ Command

4. Continuous Monitoring:
To monitor memory usage continuously at specified intervals, use the -s option followed by the number of seconds:

```
free -s 5
```

5. Total Summary:
To include a total summary of physical memory and swap space, you can add the -t option:

```
free -t
```

‘free’ Command

Example Output:

Running ‘free –h’ might yield an output like this:

```
               total        used        free      shared
buff/cache   available
 Mem:            15G        7.2G       4.1G       440M
4.0G       7.6G
 Swap:          2.0G       126M       1.9G
```

Pop Quiz

Q. What command is used to check memory availability and swap usage in Linux?

A

free

B

top

Pop Quiz

Q. What command is used to check memory availability and swap usage in Linux?

A

free

B

top

PW SKILLS

Hands-on examples for interpreting output from these tools.

Hands-On Examples for interpreting output

1. Using the ‘top’ Command

To start monitoring processes, type the following command in your terminal: ‘top’

Interpreting the Output:

```
[Abhi@localhost ~]$ top
top - 10:45:05 up 6 min,  2 users,  load average: 0.00, 0.05, 0.04
Tasks: 124 total,   1 running, 123 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.3%us,  0.3%sy,  0.0%ni, 99.0%id,  0.0%wa,  0.0%hi,  0.3%si,  0.0%st
Mem:   1906908k total,   357536k used,  1549372k free,    21380k buffers
Swap:  4095992k total,        0k used,  4095992k free,   188828k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 1847 root      20   0  243m 4544 3508 S  0.7  0.2   0:03.52 vmtoolsd
 2587 Abhi      20   0 15028 1224  936 R  0.7  0.1   0:00.29 top
    4 root      20   0     0    0    0 S  0.3  0.0   0:00.19 ksoftirqd/0
    7 root      20   0     0    0    0 S  0.3  0.0   0:00.96 events/0
  247 root      20   0     0    0    0 S  0.3  0.0   0:00.17 mpt_poll_0
    1 root      20   0 19356 1572 1252 S  0.0  0.1   0:02.18 init
    2 root      20   0     0    0    0 S  0.0  0.0   0:00.01 kthreadd
    3 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
    5 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
```

Hands-On Examples for interpreting output

2. Using the 'htop' Command

The 'htop' interface displays a colorful representation of system resources:

Interpreting the Output:

Hands-On Examples for interpreting output

3. Using the ‘free’ Command

To check memory availability and swap usage:

```
free -h
```

Interpreting the Output:

```
               total        used        free      shared
buff/cache     available
 Mem:           15Gi        10Gi        2Gi         <1Gi
2Gi          4Gi
 Swap:          2Gi         <1Gi        <1Gi
```

PW SKILLS

system load average and explain its significance in Linux.

System load average

System Load Average is a metric used in Linux to measure the average number of processes that are either actively being executed by the CPU or are waiting to be executed over a specific period of time.

- It is typically represented as three values corresponding to the last 1, 5, and 15 minutes, providing insight into the system's performance and workload trends.

System load average

Significance of Load Average in Linux:

1. Performance Monitoring
2. Resource Management
3. Troubleshooting
4. Capacity Planning

PW SKILLS

Context switching and how excessive switches impact performance.

Context Switching

Context Switching is the process by which an operating system saves the state of a currently running process or thread, allowing it to resume execution later, while switching to another process or thread.

- This mechanism is crucial in multitasking environments where multiple processes share the same CPU.

Significance of Context Switching:

- Multitasking
- Resource Management
- Handling Interrupts

Context Switching

Impact of Excessive Context Switching on Performance:

While context switching is vital for multitasking, excessive switches can negatively impact system performance due to:

- Overhead costs
- Cache Trashing
- Decreased Throughput

PW SKILLS

CPU usage breakdown (user, system, idle, etc.) using practical examples.

CPU usage breakdown

CPU Usage Breakdown refers to the distribution of CPU time across various categories of activity, typically including user processes, system processes, idle time, and I/O wait time.

Key components:

1. User CPU time
2. System CPU time
3. Idle time
4. I/O wait time

CPU usage breakdown

Practical Example:

Consider a scenario where you run the ‘top’ command on your Linux system:

```
%Cpu(s):  20.0 us,  5.0 sy,  0.0 ni, 75.0 id,  0.0 wa
```

- **20.0 us (User):** The CPU spends 20% of its time executing user processes.
- **5.0 sy (System):** The CPU spends 5% of its time executing kernel processes.
- **75.0 id (Idle):** The CPU is idle 75% of the time, indicating low overall usage.
- **0.0 wa (I/O Wait):** There are no delays waiting for I/O operations.

PW SKILLS

I/O wait and how it indicates disk or network bottlenecks

I/O wait

I/O Wait (Input/Output Wait) refers to the percentage of time that the CPU is idle while waiting for I/O operations, such as disk reads or writes, to complete.

Significance of I/O wait:

1. Indicates bottlenecks
2. Performance impact
3. Monitoring & Troubleshooting

I/O wait

Practical Example:
If you run the top command and see an output like this:

```
%Cpu(s):  10.0 us,  5.0 sy,  0.0 ni, 85.0 id,  0.0 wa
```

- The wa value (I/O wait) is at 0.0%, indicating that the CPU is not waiting for any I/O operations and is mostly idle.
- If the wa value were significantly higher (e.g., 30%), it would imply that a considerable amount of CPU time is spent waiting for I/O, suggesting potential performance issues related to disk or network operations.

Take A 5-Minute Break!

- Stretch and relax
- Hydrate
- Clear your mind
- Be back in 5 minutes

PW SKILLS

Identifying resource-heavy processes using top and htop.

Top & htop

To identify resource-heavy processes using top and htop, follow these practical steps:
Using the top Command:

1. **Launch ‘top’:**

```
top
```

2. Sort by CPU Usage:

By default, top sorts processes by CPU usage. The %CPU column shows how much CPU each process is using. Look for processes with high values (e.g., above 50%).

Top & htop

3. Sort by Memory Usage:
To sort processes by memory usage, press:

```
Shift + M
```

This will rearrange the list to show processes consuming the most memory at the top, indicated in the '%MEM' column.

4. Identify Resource-Heavy Processes:
Look for processes that consistently appear at the top of the list in either sorting. For example, if you see a process like 'mysqld' using 80% CPU, it may need further investigation.

Top & htop

5. Kill a Process:
If you identify a resource-heavy process that needs to be terminated, note its PID (Process ID), press k, enter the PID, and hit Enter to kill it.

Using the ‘htop’ Command:
1. Launch ‘htop’:

```
htop
```

2. Sort by CPU or Memory Usage:

- To sort by CPU usage, press: ‘F6’ then select ‘%CPU’
- To sort by memory usage, press: ‘F6’ then select ‘%MEM’

Top & htop

3. Visual Indicators:
The interface displays colored bars representing CPU and memory usage, making it easier to spot resource-heavy processes at a glance.

4. Filter Processes:
You can filter processes by pressing ‘F3’, allowing you to search for specific applications or services consuming resources.

5. Terminate a Process:
To kill a process directly from htop, navigate to the process using arrow keys, press ‘F9’, select the signal (e.g., 15 for TERM), and confirm.

PW SKILLS

How to terminate processes with kill or interactively using htop.

How to terminate processes with kill or interactively using htop.

Using the kill Command:

1. Identify the Process ID (PID):

- First, you need to find the PID of the process you want to terminate. You can use commands like ps, top, or pgrep to list processes.

2. Terminate the Process:

- Use the kill command followed by the PID. The basic syntax is:

```
kill [signal] PID
```

How to terminate processes with kill or interactively using htop.

To gracefully terminate a process (default signal is SIGTERM):

```
kill 1234
```

To forcefully terminate a process (using SIGKILL):

```
kill -9 1234
```

Killing Multiple Processes:

- You can also kill multiple processes at once by specifying multiple PIDs:

```
kill 1234 5678 91011
```

How to terminate processes with kill or interactively using htop.

Using htop:

1. **Launch htop:**

```
htop
```

2. Navigate to the Process:

Use the arrow keys to scroll through the list of processes and highlight the one you want to terminate.

3. Kill the Process:

1. Press F9 to bring up the kill menu.
2. Select a signal (e.g., SIGTERM or SIGKILL) using the arrow keys.
3. Press Enter to confirm and terminate the selected process.

Pop Quiz

Q. Which signal is sent by default when using the kill command without specifying a signal?

A

SIGTERM

B

SIGKILL

Pop Quiz

Q. Which signal is sent by default when using the kill command without specifying a signal?

A

SIGTERM

B

SIGKILL

PW SKILLS

Introducing nice and renice to adjust process priorities

Introduction to Nice and Renice

Nice Command: This command is used to start a new process with a specified priority level, known as the "niceness" value. The niceness value ranges from -20 (highest priority) to 19 (lowest priority).

- A lower niceness value means higher priority, allowing the process to receive more CPU time.

- **Example:** To start a new process with a higher priority:

```
nice -n -10 command_name
```

Introduction to Nice and Renice

Renice Command: Unlike nice, which sets the priority for new processes, renice modifies the priority of already running processes. This flexibility allows system administrators to adjust priorities dynamically based on current system load.

Example: To change the priority of an existing process with PID 1234:

```
sudo renice -n 5 -p 1234
```

Introduction to Nice and Renice

Impact of Priority Changes on Process Execution:

- Resource Allocation
- System Performance
- User Experience

Hands-On Examples for Modifying Priorities:

1. Starting a Process with Nice:

To start a new instance of ‘gnome-terminal’ with a lower priority:

```
nice -n 10 gnome-terminal
```

This sets the niceness value to 10, lowering its execution priority.

Introduction to Nice and Renice

2. Checking Current Nice Value:

To check the current nice value of a running process, use:

```
ps -o pid,nice,cmd -p <PID>
```

Replace <PID> with the actual process ID.

Introduction to Nice and Renice

3. Changing Priority of a Running Process with Renice:
First, find the PID of the process you want to modify (e.g., using ‘ps’ or ‘top’). Then run:

```
sudo renice -n -5 -p <PID>
```

This command raises the priority of the specified process by changing its niceness value to -5.

Introduction to Nice and Renice

4. Changing Priority for All Processes of a User:
To change the priority of all processes owned by a specific user (e.g., user "john"):

```
sudo renice -n 15 -u john
```

This sets all processes owned by "john" to a lower priority.

Pop Quiz

Q. What is the primary purpose of the ‘nice’ command in Linux?

A

To adjust the priority of a running process

B

To start a process with a specified priority

Pop Quiz

Q. What is the primary purpose of the ‘nice’ command in Linux?

A

To adjust the priority of a running process

B

To start a process with a specified priority

PW SKILLS

Real-world scenario where system performance is degraded due to a resource hog

Real-world scenarios

A real-world scenario illustrating degraded system performance due to a resource hog can be observed in a web hosting company, WebServe, which experienced significant slowdowns during peak traffic hours.

Scenario: WebServe's Performance Degradation

Context

WebServe is a web hosting provider that hosts numerous websites for small to medium-sized businesses. As their customer base grew, they noticed that during peak hours, their servers were becoming increasingly sluggish, leading to slow website loading times and customer complaints.

Real-world scenarios

Identification of the Resource Hog:

- Upon investigation, the system administrators used monitoring tools like ‘top’ and ‘htop’ to identify processes consuming excessive CPU and memory resources.

- They discovered that a single instance of a content management system (CMS) application was using over 90% of the CPU during peak traffic, significantly impacting the performance of other hosted sites.

Real-world scenarios

Impact of the Resource Hog:

1. Decreased Response Times
2. Increased Load Times
3. Customer Complaints
4. Resource Starvation

Resolution:

To resolve the issue, WebServe took several steps:

1. Process Optimization
2. Load Balancing
3. Resource Allocation

Time for case study!

Important

- Complete the post-class assessment
- Complete assignments (if any)
- Practice the concepts and techniques taught in this session
- Review your lecture notes
- Note down questions and queries regarding this session and consult the teaching assistants

Thanks !