DevOps Learning Notes

44 views . 1 likes . 1 shares

Edwiki Trainings
1 Followers
FOLLOW

Process & Performance Management

In this lecture you are learning below concepts:

1. User Configuration Files
2. Process Management
3. Performance Management

User Configuration Files:

When user account is created in Linux servers it by default create group also with the same name as user and the configurations are updated to the configuration file. Below are the important configuration files that will be updated.

1. **/etc/password:** Store User characterises like username, uid, primary group id and shell, home directory etc..
2. **/etc/shadow**: Store user password controls like, encrypted password, password controls like when the password changes, password expiry etc.
3. **/etc/group:** Store user group details and the users list that are associated to the groups.

Process Management:

In Linux when we run a command it will load the command into real memory and that creates a process.

Process: A Linux process is a command or program running in the Linus operating system. It is also **Daemon:** A service that run continually since the system start up time to system shutdown that process called as daemon.

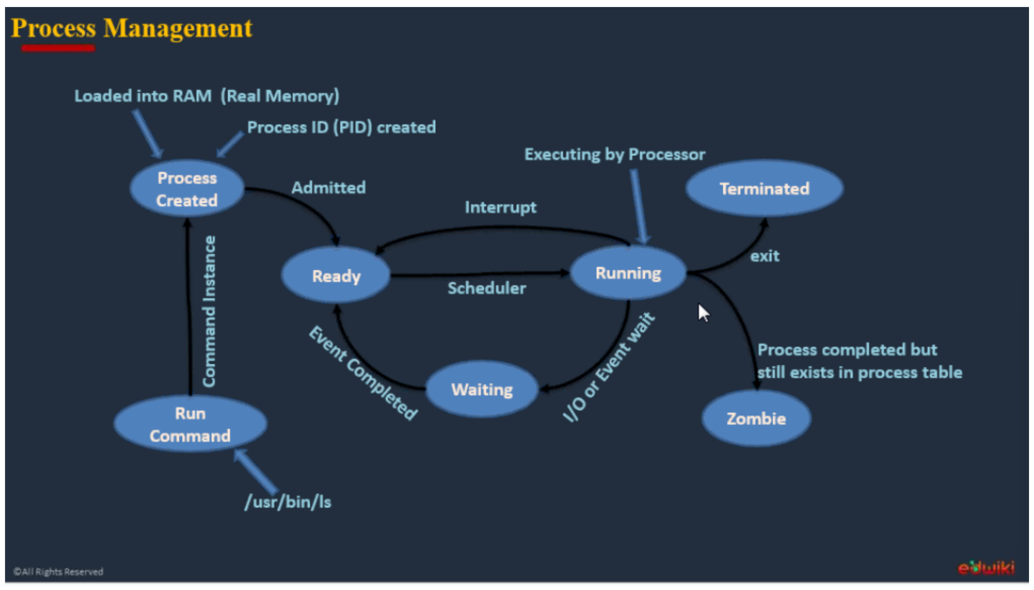
Notes:

1. Every command will run a process in the Linux system.
2. When you run command "ls" it will load into RAM (Real Memory) and every running process will be assigned a unique identifier to every process. It is known as the PID or process ID

Parent Process: These are processes that create other processes during run-time.

Child processes – These processes are created by other processes during run-time. You can see the parent process id (PPID) in the process table.

Process States: Once process created it's ready to schedule for processor and below are the different states that process.



Running: It is either running (it is current process in the system) or it's ready to run.

X

Below are the commands that you can use to work with linux process.

ps: To display currently active process.

Example:

```
[root@ip-172-31-53-144 ~]# ps
  PID TTY          TIME CMD
 3303 pts/0    00:00:00 sudo
 3304 pts/0    00:00:00 su
 3305 pts/0    00:00:00 bash
 3366 pts/0    00:00:00 ps
[root@ip-172-31-53-144 ~]#
```

ps -ef: To check currently running process in Linux.

Example:

```
[root@ip-172-31-53-144 ~]# ps –ef
UID        PID  PPID  C STIME TTY          TIME CMD
root         2     0  0 23:55 ?        00:00:00 [kthreadd]
root         3     2  0 23:55 ?        00:00:00 [rcu_gp]
root         4     2  0 23:55 ?        00:00:00 [rcu_par_gp]
root         5     2  0 23:55 ?        00:00:00 [kworker/0:0-cgr]
```

ps aux: To see a detailed list of process running the server.

a – All users
u – Shows the user/owner
x – List all processes when used together with the a option

Example:

```
[root@ip-172-31-53-144 ~]# ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         2  0.0  0.0      0     0 ?        S    23:55   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        I<   23:55   0:00 [rcu_gp]
root         4  0.0  0.0      0     0 ?        I<   23:55   0:00 [rcu_par_gp]
root         5  0.0  0.0      0     0 ?        I    23:55   0:00 [kworker/0:0-cgr]
root         6  0.0  0.0      0     0 ?        I<   23:55   0:00 [kworker/0:0H-ev]
```

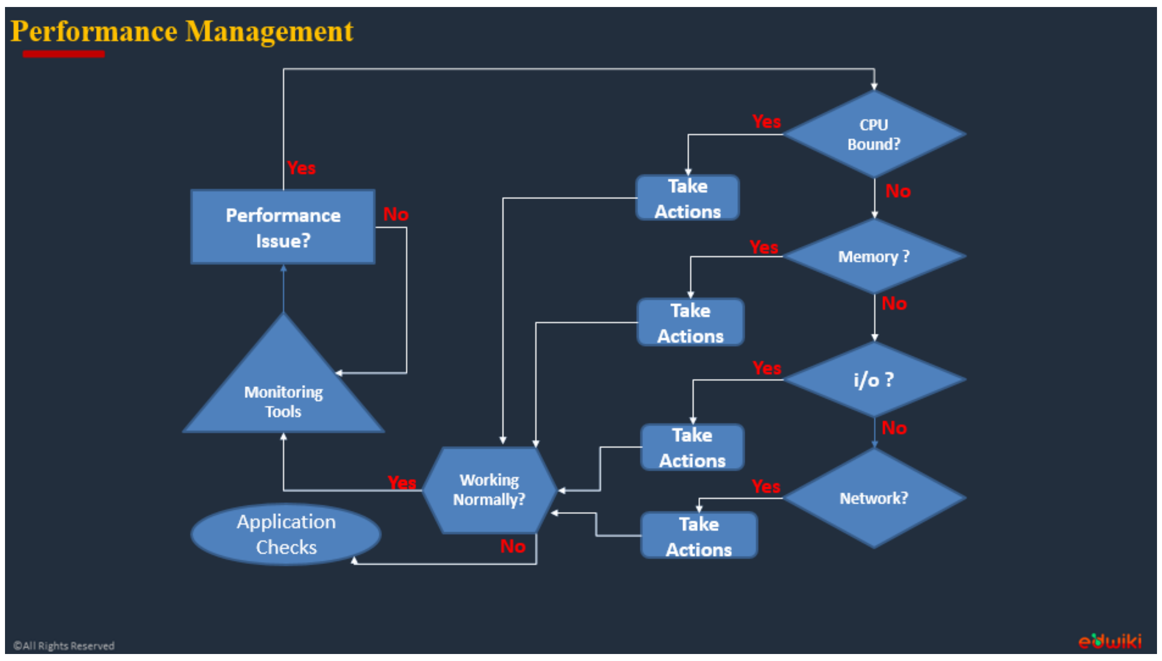
Explanation:

- **USER** The name of the user whose identity is used to run the process.
- **PID** The process identification number, which is a unique number that is needed to manage processes.
- **%CPU** The percentage of CPU cycles used by a process.
- **%MEM** The percentage of memory used by a process.
- **VSZ** The virtual memory size. This is the total amount of memory that is claimed by a process. It is common for processes to claim much more memory than they actually need. This is referred to as memory over allocation.
- RSS The resident memory size. This is the total amount of memory that a process is actually using.
- **TTY** If the process is started from a terminal, the device name of the terminal is mentioned in this column.
- **STAT** The current status of the process. The top three most common status indicators are S for sleeping, R for running, or Z for a process that has entered the zombie state.
- START The time that the process started.
- TIME The real time in seconds that a process has used CPU cycles since it was started.
- **COMMAND** The name of the command file that was used to start a process. If the name of this file is between brackets, it is a kernel process.

X

Performance Management:

If the system is running slow then you need to check the compute, network, io resources utilization as shown in the below diagram.



On a very high level, following are the four subsystems that needs to be monitored.

1. CPU
2. Memory
3. I/O
4. Network

CPU:

You should understand the four critical performance metrics for CPU:

1. Run queue
2. Cpu utilization,
3. Load average

Run Queue:

1. Run queue indicates the total number of active processes in the current queue for CPU. When CPU is ready to execute a process, it picks it up from the run queue based on the priority of the process.
2. Please note that processes that are in sleep state, or I/O wait state are not in the run queue.
3. So, a higher number of processes in the run queue can cause performance issues.

CPU Utilization:

1. This indicates how much of the CPU is currently getting used.
2. This is fairly straight forward, and you can view the CPU utilization from the top command.
3. 100% CPU utilization means the system is fully loaded.
4. So, a higher %age of CPU utilization will cause performance issues.

Load Average:

This indicates the average CPU load over a specific time period.

On Linux, load average is displayed for the last 1 minute, 5 minutes, and 15 minutes. This is helpful to see whether the overall load on the system is going up or down.

For example, a load average of “0.75 1.70 2.10” indicates that the load on the system is coming down. 0.75 is the load average in the last 1 minute. 1.70 is the load average in the last 5 minutes. 2.10 is the load average in the last 15 minutes.

Network :

1. For network interfaces, you should monitor total number of packets (and bytes) received/sent through the interface, number of packets dropped, etc.,

I/O :

X

1. **Reproduce the Problem:** You have to reproduce the problem, or at least simulate a situation that you think closely resembles the problem. This will later help you to validate the solution you come up to fix the performance issue.
2. **Monitor and collect data**: After defining the problem clearly, monitor the system and try to collect as much data as possible on various subsystems. Based on this data, come up list of potential issues or processed causing the issue.
3. **Eliminate and narrow down issues**: After having a list of potential issues, narrow it down further to see whether it is an application issue, or an infrastructure issue.
4. **One change at a time**: Once you've narrowed down to a small list of potential issues, don't try to make multiple changes at one time. If you make multiple changes, you wouldn't know which one fixed the original issue. Multiple changes at one time might also cause new issues, which you'll be chasing after instead of fixing the original issue. So, make one change at a time, and see if it fixes the original problem.

Performance Commands:

Top : To check linux server performance.

Example:

```
top - 00:18:38 up 23 min,  1 user,  load average: 0.06, 0.08, 0.03
Tasks:  97 total,    1 running,  54 sleeping,    0 stopped,    0 zombie
%Cpu(s):  6.2 us,   0.0 sy,  0.0 ni, 93.8 id,  0.0 wa,   0.0 hi,  0.0 si,  0.0 st
KiB Mem :   987924 total,   261176 free,    92996 used,   633752 buff/cache
KiB Swap:        0 total,        0 free,        0 used.   748088 avail Mem
  PID USER      PR  NI    VIRT     RES    SHR S %CPU %MEM     TIME+ COMMAND
    1 root      20   0  123596    5496   3944 S  0.0   0.6   0:02.01 systemd
    2 root      20   0       0       0      0 S  0.0   0.0   0:00.00 kthreadd
    3 root       0 -20       0       0      0 I  0.0  0.0   0:00.00 rcu_gp
    4 root       0 -20       0       0      0 I  0.0  0.0   0:00.00 rcu_par_gp
    6 root       0 -20       0       0      0 I  0.0  0.0   0:00.00 kworker/0:0H-ev
    8 root       0 -20       0       0      0 I  0.0  0.0   0:00.00 mm_percpu_wq
```

Explanation:

1. **load average:** This gives the load average of the last minute, the last 5 minutes, and the last 15 minutes.
2. **Running:** The number of active processes in the last polling loop.
3. **sleeping** The number of processes currently loaded in memory, which haven't issued any activity in the last polling loop.
4. **Stopped:** The number of processes that have been sent a stop signal but haven't yet freed all of the resources they were using.
5. **Zombie**: The number of processes that are in a zombie state. This is an unmanageable process state because the parent of the zombie process has disappeared and the child still exists but cannot no longer be managed because the parent is needed to manage that process.
6. **Id**: The amount of time the CPU has been idle.
7. **%sys**: How much percentage of CPU used for system processes
8. **%user**: How much percentage of CPU used for user running processes
9. **wa** :The amount of time the CPU has been waiting for I/O requests. This is a very common indicator of performance problems. If you see an evaluated value here, you can make your system faster by optimizing disk performance.
10. **Mem** : The total amount of memory that is available to the Linux kernel
11. **Used**: The total amount of memory that currently is used.
12. **Free:** The total amount of memory that is available for starting new processes.
13. **Buffers**: The amount of memory that is used for buffers. In buffers,

X

```
[root@ip-172-31-53-144 ~]#
```

Cat /proc/cpuinfo: To check how many core cpu's are attached to server and processor attributes.

Example:

```
[root@ip-172-31-53-144 ~]# cat /proc/cpuinfo
processor      : 0
vendor_id      : GenuineIntel
cpu family     : 6
model          : 63
model name     : Intel(R) Xeon(R) CPU E5-2676 v3 @ 2.40GHz
stepping       : 2
microcode      : 0x49
cpu MHz        : 2400.053
cache size     : 30720 KB
```

vmstat: To check more details about memory usage.

Example:

```
[root@ip-172-31-53-144 ~]# vmstat
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 0  0      0 261404   2088 631868    0    0   106   193   46  176  1  1 97  1  0
[root@ip-172-31-53-144 ~]#
```

netstat: To command to check network statistics.

Example:

```
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0    304 ip-172-31-53-144.ec:ssh broadband.actcorp:14515 ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  3      [ ]         DGRAM                    11536    /run/systemd/notify
unix  2      [ ]         DGRAM                    11537    /run/systemd/cgroups-agent
unix  5      [ ]         DGRAM                    11543    /run/syst
```

iostat: To check for input-output statistics and performance of device.

Example:

```
[root@ip-172-31-53-144 ~]# iostat
Linux 5.10.162-141.675.amzn2.x86_64 (ip-172-31-53-144.ec2.internal)     02/11/2023
_x86_64_        (1 CPU)
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.88    0.18    0.52    0.52    0.31   97.59
Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
xvda              7.80        99.95       181.52     231294     420060
[root@ip-172-31-53-144 ~]#
```

Previous Next

Comments(0)