Lecture 28

28. Kernel-Tunning

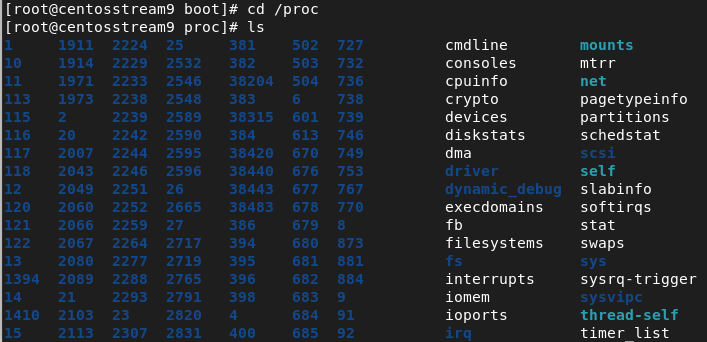
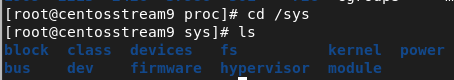
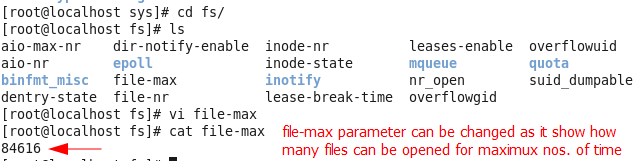
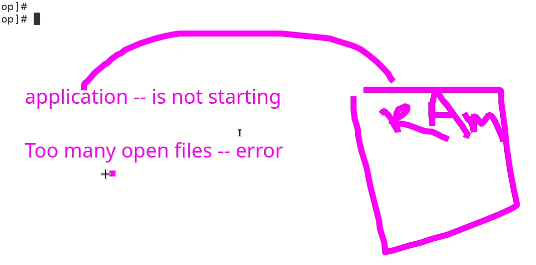
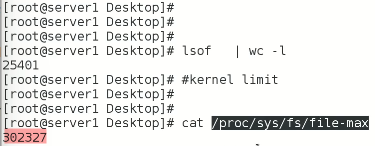
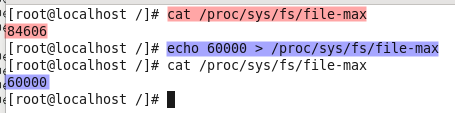
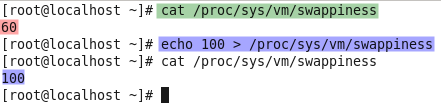
Kernel Tuning (interview topic)

* (It means) to change few parameters of Kernel. /to change values of running Kernel
* /proc 🡪 everything related to RAM
* Kernel is also loaded into RAM while running. 🡪 Kernel tuning is also done in /proc 🡪 it will be temporary.
* There r many types of Kernels
  + Monolethic Kernel / called modular user in Linux
    - Drivers can be loaded in real time 🡪 $ modprobe
  + Micro Kernel – used in soleris /AIX/ HP -UniX
    - Drivers can’t be loaded in real time.
  + .

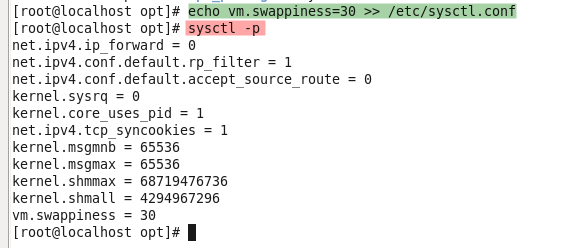
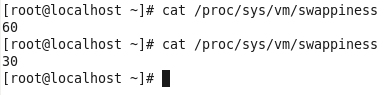
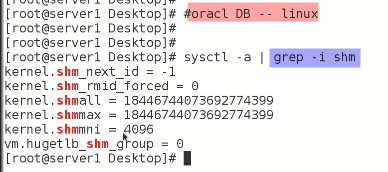
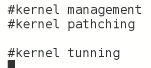
In the context of Linux, a kernel refers to the core component of the operating system that manages system resources and provides a platform for running applications. Here are some types of Linux kernels:

1. Monolithic Kernel: A monolithic kernel is a type of kernel where all the core operating system functions are included in a single binary file. This type of kernel is relatively simple and efficient, but it can be difficult to modify and customize.
2. Microkernel: A microkernel is a type of kernel that only includes the most essential operating system functions, such as memory management and process scheduling. Additional functionality, such as device drivers and file systems, are implemented as separate modules that run in user space. This type of kernel is more modular and customizable than a monolithic kernel, but it can be less efficient due to the overhead of inter-process communication.
3. Hybrid Kernel: A hybrid kernel is a type of kernel that combines elements of both monolithic and microkernel designs. Like a monolithic kernel, a hybrid kernel includes many operating system functions in the kernel itself. However, it also allows some functionality, such as device drivers, to run in user space. This type of kernel aims to strike a balance between simplicity and flexibility.
4. Real-time Kernel: A real-time kernel is a type of kernel that is designed for systems that require predictable and consistent response times, such as industrial control systems and embedded devices. Real-time kernels typically prioritize certain types of tasks over others to ensure that critical operations are executed quickly and reliably.

These are some of the main types of kernels used in Linux. Each type has its own advantages and disadvantages, and the choice of kernel depends on the specific requirements of the system.

* For Kernel tuning 🡪 /proc
* 
* These values can’t be changed inside /proct 🡪 it can be done in “sys”
* $ /proc/sys 🡪 files values and parameters can be changed.
* 
* Go into “fs” directory.
* 
* Another command is used $ sysctl -a 🡪 show which parameters or values can be changed.
*  🡪 719 values can be tuned or changed.
* 1st Issue
* Real world example, 🡪 a common error in industry.
* 
* If u heck RAM it will have enough space. $ free -g or $ free -m (commands to check RAM)
* It is because Kernel is restricting that application to open. 🡪 it is Kernel limitation.
* It can be solved by Kernel tuning,
* To check how many files are opened in the system
  + $ lsof 🡪 stands for “list open files”
  + 
  +  🡪 this nos of files are the maximum no which Kernel allows to open but if we are going to start a service which must open 50000 files, “to many files open error will pop up”
  + This value in file-max needed to change and it is called “Kernel tuning”
  + **But 🡪 this file is in RAM and it will not allow to write in it while its open.**
  + So, the solution is,
    - $ echo 60000 > /proc/sys/fs/file-max
    - It will write in the file on run time.
  + now check the file again, its value will be updated to 60000.
  + 
* 2nd issue
* RAM is full but “swap” is not being used
* There is a value in Krenel 🡪 Kernel Swapiness
* By default, it is 60
* It needed to be set at 100
* In this case Kernel Tuning is the solution,
* $cat /proc/sys/vm/swapiness
* 
* The value is changed,
* 
* The swap requirement depends upon application and can be noted in manual or documentation of that specific application.

The professional way to do Kernel Tuning,

* + $ sysctl -a 🡪 shows all the process with tag and value pairs form.
  + 
  + For Kernel Tuning,
  + $ sysctl -w <tag>=<value>
  + 
  + To persist the tuning,
  + Add the value in /etc/sysctl.conf
  + 
  + I changed the swappiness value to 30 by this method,
  + 
  + Then 🡪 $ sysctl -p 🡪 to reload or reread this “sysctl.conf” file
    - As the system boots and loads the Kernel into the RAM, the default value of swappiness will be set at “60”but as the Kernel will read “sysctl.cconf” file which holds the values of swappiness to 30 it will overwrite its default value with 30.
  + This will apply the changes without requiring a reboot. However, some parameters may require a system restart to take effect. It is recommended to test any kernel tuning changes in a non-production environment before implementing them on a production system.
  + Value is changed
  + 
  + As System Administrator,
  + 
  + These values will be updated or tuned according to documentation of oracle DB.
  + 🡪 it is the end of the topic.

The Kernel tuning in CentOS 9, (in lecture Sir Kazim used CentOS 6)

Kernel tuning in CentOS 9 involves modifying kernel parameters to optimize the performance and behavior of the operating system. Here are some steps you can take to tune the kernel in CentOS 9:

1. Check the current kernel parameters: Before making any changes, it's a good idea to check the current kernel parameters using the **sysctl** command. This command displays the current kernel parameters and their values.
2. Modify kernel parameters: You can modify kernel parameters by editing the **/etc/sysctl.conf** file. This file contains a list of kernel parameters and their values. To modify a parameter, you can add or modify a line in the file that specifies the parameter and its new value.
3. Apply the changes: After modifying the kernel parameters, you need to apply the changes using the **sysctl** command. You can either reboot the system to apply the changes at boot time or run the **sysctl -p** command to apply the changes immediately.
4. Monitor system performance: After applying the kernel changes, it's important to monitor system performance to ensure that the changes are having the desired effect. You can use system monitoring tools such as **top**, **vmstat**, and **sar** to monitor system resource usage and identify any performance issues.

Some common kernel parameters that you might want to modify include:

* swappiness: This parameter controls the degree to which the kernel swaps out processes to disk. A lower value means that the kernel will swap out processes less aggressively, which can improve performance for memory-intensive applications.
* file-max: This parameter controls the maximum number of open file descriptors that the system can handle. Increasing this parameter can improve the performance of applications that require a large number of open file descriptors.
* tcp\_tw\_reuse: This parameter controls whether the kernel can reuse TCP connections that are in the TIME\_WAIT state. Enabling this parameter can improve the performance of applications that use a large number of short-lived connections.

These are just a few examples of the many kernel parameters that you can modify to optimize the performance of CentOS 9. It's important to carefully evaluate the impact of any changes and monitor system performance to ensure that the changes are having the desired effect.