**1. Advanced File System Management and Troubleshooting**

**Disk Partitioning and LVM**

* **fdisk**  
  **Example:**
* sudo fdisk /dev/sda

**Explanation:**

* + No extra options are used here. Running fdisk on /dev/sda launches an interactive session for creating, deleting, or modifying partitions on that disk.
* **pvcreate**  
  **Example:**
* sudo pvcreate /dev/sda1

**Explanation:**

* + No options are used; the command simply marks /dev/sda1 as a physical volume for use with LVM.
* **vgcreate**  
  **Example:**
* sudo vgcreate vg\_data /dev/sda1

**Explanation:**

* + No additional flags; this command creates a volume group named vg\_data using /dev/sda1.
* **lvcreate**  
  **Example:**
* sudo lvcreate -n lv\_home -L 20G vg\_data

**Options Explained:**

* + **-n lv\_home**: Sets the name of the new logical volume to “lv\_home.”
  + **-L 20G**: Specifies that the size of the logical volume should be 20 gigabytes.
* **mkfs.ext4**  
  **Example:**
* sudo mkfs.ext4 /dev/vg\_data/lv\_home

**Explanation:**

* + No flags are used in this example; it simply formats the volume with the ext4 file system.
* **mount**  
  **Example:**
* sudo mount /dev/vg\_data/lv\_home /mnt/home

**Explanation:**

* + No additional options; the command attaches the formatted volume to the specified directory.
* **mdadm**  
  **Example:**
* sudo mdadm --create /dev/md0 --level=1 --raid-devices=2 /dev/sda /dev/sdb

**Options Explained:**

* + **--create**: Instructs mdadm to create a new RAID array.
  + **--level=1**: Sets the RAID level to 1 (mirroring).
  + **--raid-devices=2**: Indicates that two devices will be used in the RAID array.
* **fsck**  
  **Example:**
* sudo fsck /dev/sda1

**Explanation:**

* + No options are used; fsck checks and repairs file system inconsistencies on the specified device.
* **smartctl**  
  **Example:**
* sudo smartctl -a /dev/sda

**Option Explained:**

* + **-a**: Displays all available S.M.A.R.T. information about the disk, including health status and error logs.

**2. User and Group Management**

* **useradd**  
  **Example:**
* sudo useradd -m johndoe

**Option Explained:**

* + **-m**: Creates a home directory for the new user if it does not already exist.
* **passwd**  
  **Example:**
* sudo passwd johndoe

**Explanation:**

* + No flags are used; this command is used to set or change the password for the specified user.
* **groupadd**  
  **Example:**
* sudo groupadd developers

**Explanation:**

* + No options are used; it creates a new group called “developers.”
* **usermod**  
  **Example:**
* sudo usermod -aG developers johndoe

**Option Explained:**

* + **-aG developers**: The -G option specifies a list of supplementary groups, and the -a flag ensures that the user is appended to the group list (rather than replacing existing groups). Here, it adds “johndoe” to the “developers” group.

**3. Process Management and Scheduling**

* **ps**  
  **Example:**
* ps aux | grep apache

**Options Explained:**

* + **a**: Lists processes for all users.
  + **u**: Displays process details in a user-oriented format.
  + **x**: Includes processes without a controlling terminal.
* **top**  
  **Example:**
* top

**Explanation:**

* + No flags are used; it provides a real-time view of system processes.
* **htop**  
  **Example:**
* htop

**Explanation:**

* + No flags are used; it’s an enhanced interactive process viewer.
* **nice**  
  **Example:**
* nice -n 10 tar -czf backup.tar.gz /var/log

**Option Explained:**

* + **-n 10**: Sets the niceness value to 10, lowering the priority of the command.
* **tar**  
  **Example (as used with nice):**
* tar -czf backup.tar.gz /var/log

**Options Explained:**

* + **-c**: Creates a new archive.
  + **-z**: Compresses the archive using gzip.
  + **-f backup.tar.gz**: Specifies the filename for the archive.
* **renice**  
  **Example:**
* sudo renice -n -5 -p 1234

**Options Explained:**

* + **-n -5**: Sets the new niceness value to -5, which increases the priority of the process.
  + **-p 1234**: Specifies the process ID (PID) of the process to be reniced.
* **cgcreate**  
  **Example:**
* sudo cgcreate -g cpu:/mygroup

**Option Explained:**

* + **-g cpu:/mygroup**: Creates a new control group called “mygroup” for the CPU subsystem.
* **cgset**  
  **Example:**
* sudo cgset -r cpu.shares=512 mygroup

**Option Explained:**

* + **-r cpu.shares=512**: Sets the CPU shares (a relative weight for CPU time allocation) to 512 for “mygroup.”
* **cgexec**  
  **Example:**
* sudo cgexec -g cpu:mygroup your\_command\_here

**Option Explained:**

* + **-g cpu:mygroup**: Executes the given command within the control group “mygroup” for the CPU subsystem.
* **chrt**  
  **Example:**
* sudo chrt -f 99 your\_command\_here

**Options Explained:**

* + **-f**: Specifies that the FIFO (first-in, first-out) scheduling policy is to be used.
  + **99**: Sets the real-time priority to 99.
* **sar**  
  **Example:**
* sar -u 1 5

**Option Explained:**

* + **-u**: Instructs sar to display CPU utilization statistics.
  + The numbers 1 and 5 indicate that samples should be taken every 1 second, for a total of 5 iterations.

**4. Containerization and Cloud-Based Application Management**

* **docker run**  
  **Example:**
* docker run -d -p 80:80 nginx

**Options Explained:**

* + **-d**: Runs the container in detached mode (in the background).
  + **-p 80:80**: Maps port 80 of the host to port 80 of the container, enabling access to the container’s services.
* **kubectl apply**  
  **Example:**
* kubectl apply -f nginx-deployment.yaml

**Option Explained:**

* + **-f nginx-deployment.yaml**: Specifies the YAML configuration file to be applied, which defines the deployment settings.

**5. Networking and Security**

* **ip addr show**  
  **Example:**
* sudo ip addr show

**Explanation:**

* + This command uses the subcommand “addr show” to display IP addresses for all network interfaces. No additional flags are used.
* **ip route show**  
  **Example:**
* ip route show

**Explanation:**

* + Similarly, this command uses “route show” to display the routing table without extra flags.
* **tcpdump**  
  **Example:**
* sudo tcpdump -i eth0

**Option Explained:**

* + **-i eth0**: Tells tcpdump to capture packets on the network interface eth0.
* **ping**  
  **Example:**
* ping google.com

**Explanation:**

* + No flags are used in this basic connectivity test.
* **traceroute**  
  **Example:**
* traceroute google.com

**Explanation:**

* + No additional options are used in this example.
* **ss**  
  **Example:**
* ss -tuln

**Options Explained:**

* + **-t**: Displays TCP sockets.
  + **-u**: Displays UDP sockets.
  + **-l**: Shows only listening sockets.
  + **-n**: Displays numerical addresses (skipping DNS resolution).
* **iptables**  
  **Example:**
* sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT

**Options Explained:**

* + **-A INPUT**: Appends this rule to the INPUT chain.
  + **-p tcp**: Specifies that the rule applies to TCP packets.
  + **--dport 22**: Matches packets destined for port 22 (typically used for SSH).
  + **-j ACCEPT**: Specifies the target action—here, to ACCEPT (allow) the matching packets.
* **firewall-cmd**  
  **Example:**
* sudo firewall-cmd --zone=public --add-port=80/tcp --permanent
* sudo firewall-cmd --reload

**Options Explained:**

* + **--zone=public**: Indicates that the rule applies to the “public” firewall zone.
  + **--add-port=80/tcp**: Opens port 80 for TCP traffic.
  + **--permanent**: Makes the rule persistent across reboots.
  + **--reload**: Reloads the firewall configuration to apply the changes.
* **aws ec2 authorize-security-group-ingress**  
  **Example:**
* aws ec2 authorize-security-group-ingress --group-id sg-xxxxxxxx --protocol tcp --port 80 --cidr 0.0.0.0/0

**Options Explained:**

* + **--group-id sg-xxxxxxxx**: Specifies the target security group by its ID.
  + **--protocol tcp**: Sets the protocol to TCP.
  + **--port 80**: Opens port 80.
  + **--cidr 0.0.0.0/0**: Allows traffic from any IP address.
* **openvpn**  
  **Example:**
* sudo openvpn --config client.ovpn

**Option Explained:**

* + **--config client.ovpn**: Specifies the configuration file (client.ovpn) to use when starting OpenVPN.