**General Linux**

**User Management**

Configure the server to meet the following requirements:

1. Create a new user named cbjalice with the following parameters:
   * Home Directory: /data/home/cbj/alice
   * Password: DCEKgabE
   * Shell: Any **Non-Interactive** Shell
2. The existing user cbjbrian is currently unable to login via SSH, Terminal, or even via the su command. Resolve whatever issue is preventing this user from obtaining a shell. Note that the user's password is TCmZyY5Y.
3. The existing user cbjchuck is unable to create any new files in their $HOME directory. Resolve whatever issue is preventing the user from creating files. Note that the user's password is nuCAJHTN.
4. The existing user cbjdonald has been disabled and is unable to login. Please re-enable the account **without** deleting it or resetting the password.

**Cron Jobs**

Configure Cron to meet the following requirements, note that all times referenced are **server time**:

1. Create a job that runs **daily at 6:00 PM** and executes the command /usr/local/bin/rti-users.
2. Create a job that runs every **Monday at 5:00 PM** and executes the script /usr/local/bin/rti-openports. Redirect STDOUT and STDERR to /var/log/rti-ports.log.
3. Create a job that runs **hourly** and executes the script /usr/local/bin/rti-diskusage. Redirect STDOUT and STDERR to the file /var/log/rti-diskusage.log.
4. Create a job that runs on the **1st and the 15th of each month** that executes the script /usr/local/bin/rti-accounting -e payroll --report as the user rti4392.

**System Performance and Tuning**

Configure the server to meet the following requirements:

1. Limit the user tusercpb so that they are able to run no more than **25** processes at a time.
2. Tune the server to use swap space 50% of the time rather than the current value of 85%.
3. Disable the **SYN Cookies** feature, and increase the maximum **SYN Backlog** from 1024 to 4096 connections.
4. The maximum Core dump size is currently set to 0 for all users. While keeping Core Dumps disabled globally, configure the server to allow a maximum Core Dump size of 8192KB for the tdevcfb user **only**.

**Networking**

**Configure an Interface Alias (Additional IP)**

Add the IP Address **192.168.2.50/24** on the server as alias interface **eth1:0**.  Ensure that the interface alias is up, and will remain up after a reboot.

**Static Route Not Working**

This server has been configured with a TUN device '*tun-aib0*' that acts as an endpoint for a VPN tunnel.  The IP address of the device is 192.168.17.14 and the network it is on is using a 22 bit netmask (255.255.252.0).  The problem is that the server needs to be able to communicate with the 10.42.182.0 network with a 24 bit netmask (255.255.255.0) but none of the traffic appears to be reaching the other VPN endpoint and instead is trying to be sent out to the public internet over eth0.  Please repair the static route for 10.42.182.0 so that traffic goes to the right place.

**Configure IPTABLES to Block Traffic**

Configure IPTABLES to block traffic from the IP address **192.168.1.29** from being able to connect to any port on the server.  The rule should persist across reboots.

**General Scripting (Any Language)**

**Periodic Logging of Running Processes**

Create a script that meets the following criteria:

1. Script Name: /usr/local/bin/periodic\_log
2. The script must log the full path for the executable of all processes currently running on the server, in 10 second intervals.
3. Each iteration should log to a separate log file in the directory /proc-logs/ as log1, log2, etc.
4. The script should end after 6 iterations

**Organize Incoming Data Files by Type/Extension**

Create a script that meets the following requirements:

* Script Name: /usr/local/bin/move-by-ext
* The script must move files from the /data/downloads/incoming/ directory into the appropriate sub-directory based on the file extension.
  1. Files with a .mov extension should be moved to /data/downloads/movies
  2. Files with a .jpg extension should be moved to /data/downloads/images
  3. Files with a .doc extension should be moved to /data/downloads/docs
  4. Files with any other extension should be deleted.

The /data/downloads/incoming/ directory should be completely empty after running the script.

**Warning**: Prior to grading, the directory structure for /data/downloads/ will be removed and rebuilt so that your script is running in a pristine environment. You should assume that your script will need to create the movies/, images/, and docs/ sub-directories if they do not already exist.

**Create a Script to Filter Out Duplicate Data**

Create a script using any language you choose that meets the following requirements:

* Script Name: /usr/local/bin/dedupe-bafiles
* The script must parse all files in the directory /data/ba/infiles/, and remove all duplicate lines from the content read from each file.
* After processing a file's content, it must write the resulting data to a file in the directory /data/ba/outfiles/ with the same filename
* The script must also print a count of how many lines are in each resulting file to STDOUT in the format: FILENAME: LINE\_COUNT

The order of the resulting data of each file in /data/ba/outfiles/ does not need to be the same as from the original files, but they absolutely must be unique to each file.

**WARNING**: The files in /data/ba/infiles/ will be replaced with similar (but different) data before the script is executed during grading.