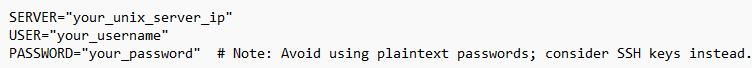
**Problem Description:** NAS utilization and instances running on Unix servers.

1. **Shebang**

****

**Explanation**: This line tells the system that the script should be run using the Bash shell.

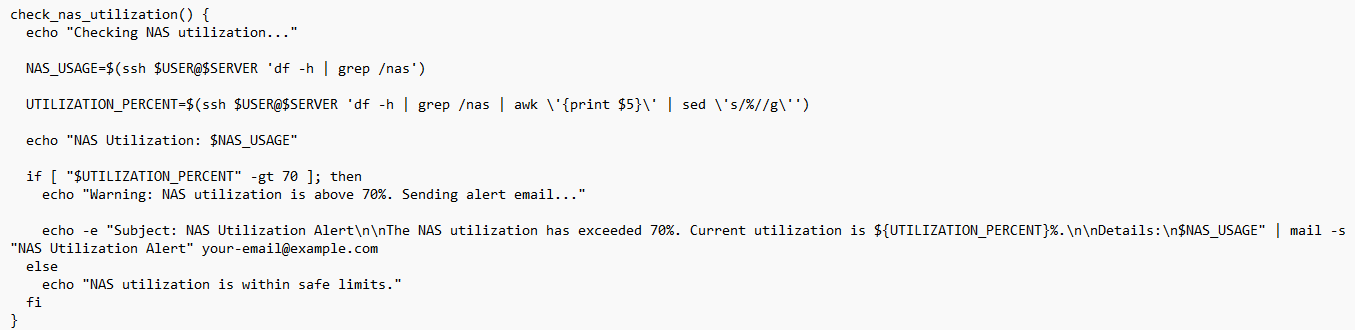
1. **Variables**

****

**Explanation**: These lines define three variables:

* SERVER: The IP address of the Unix server you want to check.
* USER: Your username for logging into the server.
* PASSWORD: Your password for logging into the server. Note: It's not safe to store passwords in plain text. It's better to use SSH keys for secure access.

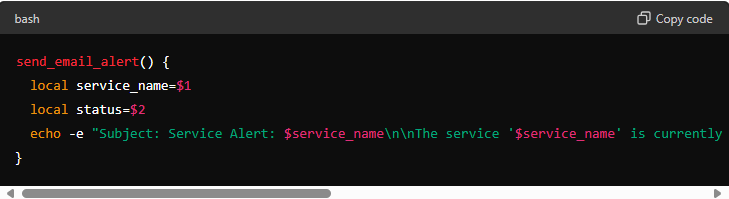
1. **Function to Check NAS Utilization**



**Explanation**:

* **Function Name**: check\_nas\_utilization
* **Purpose**: This function checks how much space is being used on a Network Attached Storage (NAS) system.
* **Steps**:
  1. echo "Checking NAS utilization.": Prints a message indicating that NAS utilization is being checked.
  2. NAS\_USAGE=$(ssh $USER@$SERVER df -h | grep /nas): Connects to the server using SSH and runs df -h | grep /nas to find storage details for /nas. It captures this information in NAS\_USAGE.
  3. UTILIZATION\_PERCENT=$(ssh $USER@$SERVER 'df -h | grep /nas | awk \'{print $5} \' | sed \'s/%//g\''): Runs a command to extract the usage percentage of /nas and removes the '%' sign.
  4. echo "NAS Utilization: $NAS\_USAGE": Prints the NAS utilization details.
  5. The if statement checks if the usage is more than 70%. If so, it sends an alert email. Otherwise, it prints a message saying the utilization is within safe limits.

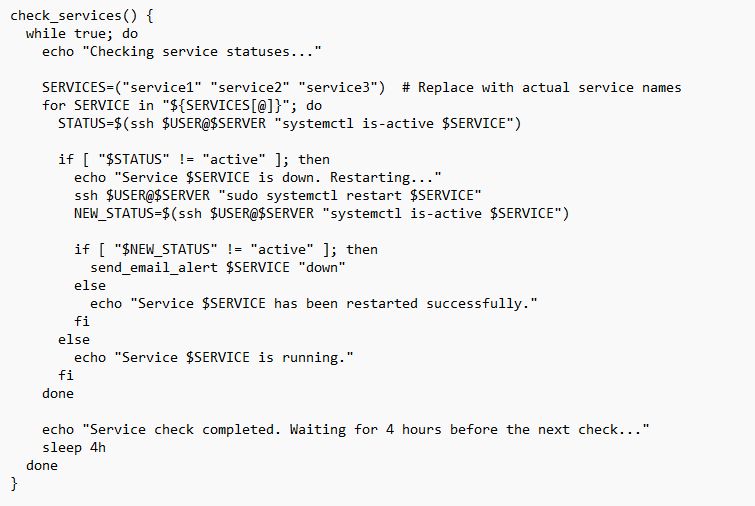
1. **Function to Send Email Alerts**



**Explanation**:

* **Function Name**: send\_email\_alert.
* **Purpose**: Sends an email alert about a specific service status.
* **Steps**:
  1. Accepts two parameters: service\_name and status.
  2. Constructs an email with the subject "Service Alert" and the body containing the service name and its current status.
  3. Sends the email to your-email@example.com.

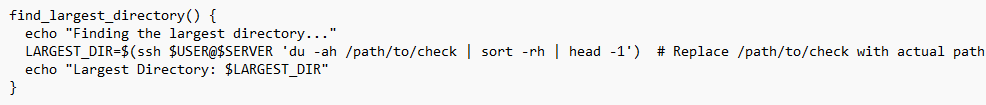
**5. Function to Check Services Every 4 Hours**

****

**Explanation**:

* **Function Name**: check\_services
* **Purpose**: Checks the status of specified services on the server and restarts them if they are down.
* **Steps**:
  1. Runs in an infinite loop (while true), checking services every 4 hours.
  2. SERVICES= ("service1" "service2" "service3"): Defines a list of services to check.
  3. For each service:
     + Uses SSH to check if the service is running (systemctl is-active $SERVICE).
     + If the service is not active, it tries to restart the service.
     + Checks the status again after attempting a restart.
     + Sends an email alert if the service is still down.
  4. Sleeps for 4 hours before the next check.

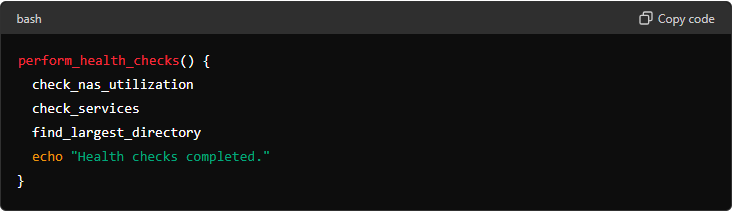
**6. Function to Find the Largest Directory**

****

**Explanation**:

* **Function Name**: find\_largest\_directory.
* **Purpose**: Identifies the largest directory in a specified path.
* **Steps**:
  1. Connects to the server and runs a command to find the largest directory in /path/to/check.
  2. Prints the largest directory found.

### **7. Main Function to Perform Health Checks**

****

**Explanation**:

* **Function Name**: perform\_health\_checks.
* **Purpose**: Calls the three main health check functions: check\_nas\_utilization, check\_services, and find\_largest\_directory.
* **Steps**:
  1. Runs check\_nas\_utilization to check NAS space usage.
  2. Runs check\_services to monitor and restart services if necessary.
  3. Runs find\_largest\_directory to find the largest directory.
  4. Prints a message indicating that the health checks are completed.

### **Summary**

* The script is designed to be run on a regular basis to monitor the health of a server.
* It checks the utilization of NAS storage, the status of specified services, and identifies the largest directory.
* If issues are found, it sends email alerts to notify you.

This script helps automate server maintenance tasks, ensuring that key resources and services are monitored, and issues are addressed promptly.