**IMPLEMENTING AN AUTOMATED SYSTEM UPDATE IN UBUNTU LINUX OPERATING SYSTEMS**

**Charity Spoko**

***Abstract:*** This project is aimed at automating system updates in Linux to streamline system maintenance and optimize performance. The system is designed to automatically install available updates, remove unused packages, and reboot the machine to ensure that updates take effect. Additionally, an email notification feature was intended to inform the user once the update process is completed, though this functionality has not yet been successfully implemented. This documentation outlines the project's purpose, workflow, challenges, and significance in system administration. Automated updates are critical to maintaining the security and efficiency of systems in a world where software vulnerabilities are continually evolving (Kizza, 2020; Stallings, 2019).

**1.0 Introduction**

Maintaining a secure and up-to-date system is a core aspect of system administration, particularly in enterprise and critical infrastructure environments. System maintenance is crucial for ensuring security and optimal performance in any environment, especially in Linux-based systems used for personal, enterprise, or critical infrastructure. A significant aspect of system maintenance is keeping software up to date with the latest patches and updates, which ensures that vulnerabilities are fixed and new features are integrated. Regular updates not only ensure the system runs smoothly but also patch vulnerabilities and address security concerns (Kizza, 2020). However, manual updates can be time-consuming, error-prone, and inconsistent while automating tasks saves time, improves system security and helps prevent human error, which is often a major contributor to system misconfigurations (Stallings, 2019).

This project automates the process of updating the Ubuntu Linux operating system. It performs essential tasks like checking the health of the system so that there is enough memory for the new updates, checking for software updates, installing them, removing unused packages to free up system resources, and rebooting after the update process completes. One key aspect of the system, which is still under development, is an email notification system that alerts users when updates are successfully applied.

* 1. **Project Overview**

The automated system update script performs five primary tasks:

1. **System Health Checks**: Before applying updates, the system verifies whether there is enough free disk space and memory available to avoid failures during the update process.
2. **Automated Updates**: Using a Bash script and **cron** jobs, the system automatically checks for available updates at scheduled intervals (For this project it has been set for 10 minutes every day) and installs them without user intervention.
3. **Package Cleanup**: After each update, the system removes unused or obsolete packages(orphans) that accumulates over time, ensuring that the system runs efficiently and freeing up storage space.
4. **Email Notifications**: Although not fully functional yet, the system aims to send a notification email to the user when updates are successfully installed.
5. **Reboot Process**: The system automatically reboots when the process of updating has finished to ensure changes take effect immediately.

**1.2 Objectives**

1. Automate the process of updating packages on a Linux system.

2. Remove unused packages after updates to free up disk space and maintain system cleanliness.

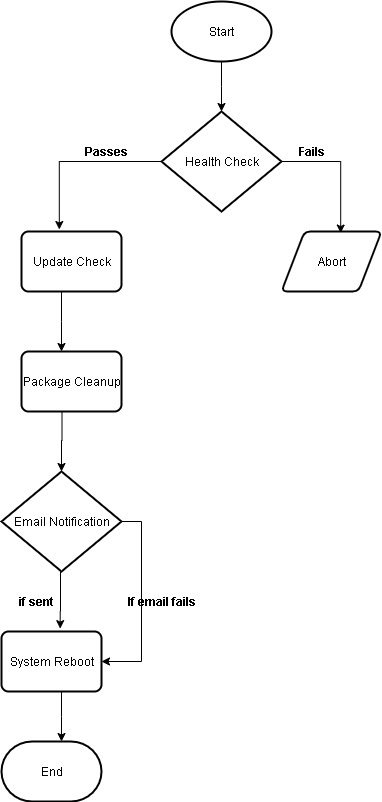
3. Send email notifications upon the successful completion of system updates

4. Reboot the system after every update made

**2.0 Process Workflow**

The process follows a sequential flow to ensure the system is always in a healthy state before proceeding with any changes:

1. **Health Checks**: Check disk space and memory.to ensure that the system is in a suitable state for updates
2. **Update and Upgrade**: Fetch and install updates for all installed software packages.
3. **Package Cleanup**: Remove unnecessary packages to optimize performance.
4. **Email Notification**: (Pending completion) Send an email to notify the user about the completion of updates.
5. **System Reboot**: The system automatically reboots to apply the updates.
6. **3.6 Diagram Representation of the Process Flow**



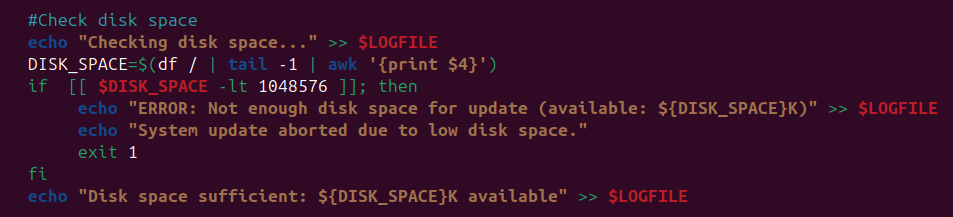
**2.1 Implementation**

The project consists of a Bash script that integrates with Ubuntu’s package management system (**apt**). The script is run by a cron job, which schedules the update process at user-defined intervals.

* **Health Checks**: The system first checks if there’s enough disk space and memory to perform the update, aborting if conditions aren’t met.

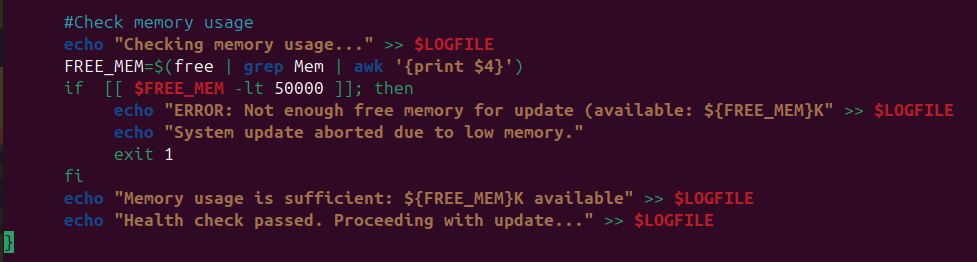
**Disk Space Check:** The script verifies that there is sufficient free disk space available. This prevents update failures due to inadequate storage. The minimum required disk space is set to 1.05 GB (1,04,8576 KB).

**Implementation:** The script checks disk space using the **df** command and compares it against the minimum required value. If there is insufficient space, the script logs an error and aborts the update process.

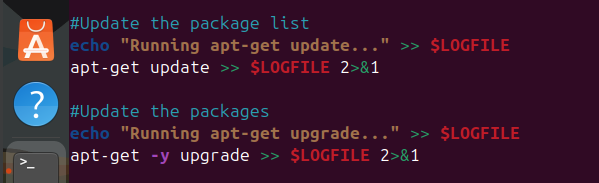


**Memory Check:** The script ensures that there is enough free memory available before proceeding with updates. This prevents issues related to insufficient memory during the update process. The minimum required free memory is set to 0.05 GB (50000 KB).

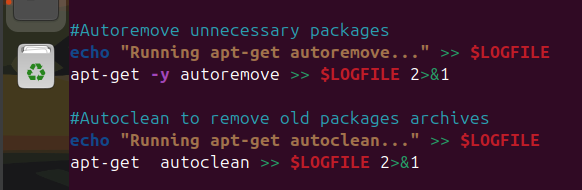
**Implementation:** The script checks free memory using the **grep** command on **/proc/meminfo** and compares it against the minimum required value. If there is insufficient memory, the script logs an error and aborts the update process.



* **Update Process**: The script runs **apt-get update** and **apt-get upgrade** to fetch and install the latest updates.

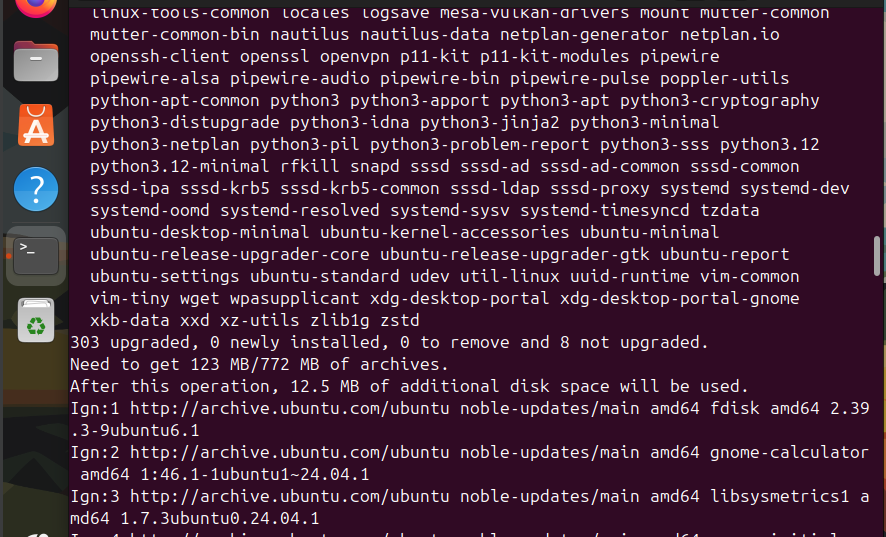


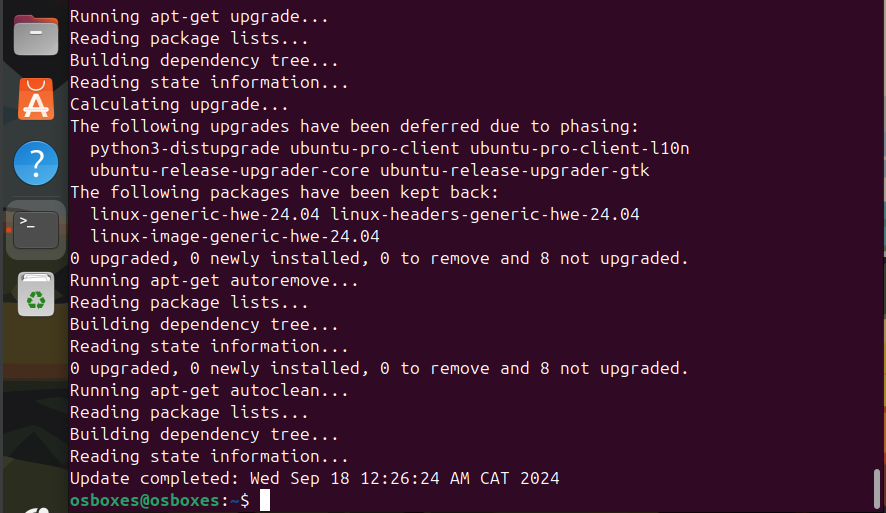
* **Package Cleanup**: After updating, it runs **apt-get autoremove** and **apt-get autoclean** to remove orphaned packages which cleans up obsolete dependencies and frees disk space.



* **Email Notifications:** The script was intended to notify the user via email when the update process is complete. The notification would include information about the success or failure of the update process. This feature requires further troubleshooting, particularly with the configuration of **mailutils** and **postfix**, which has been challenging.

In the absence of email notifications, the system logs the update process, including errors and successes. Below is a log entry sample to verify that updates were applied correctly.

****

****

* **System Reboot**: After completing updates and cleaning up unnecessary packages, the system reboots to apply the changes. Even if the email notification fails, the reboot still occurs, ensuring that the system runs the latest updates without manual intervention. The automated reboot minimizes downtime and ensures that updates take effect immediately.

**4.0 Benefits of the Project**

This project brings numerous advantages to Linux system administrators and users:

**4.1 Security and Stability**: By automating updates, the system ensures that security patches and bug fixes are applied promptly, reducing the risk of vulnerabilities being exploited. This is especially critical in environments with sensitive data or high security requirements.

**4.2 Time-Saving**: Automation frees up administrators from manually checking and applying updates, allowing them to focus on other important tasks. With the system scheduled to run daily, it keeps systems up to date without the need for constant supervision.

**4.3 Consistency**: The process of applying updates is standardized, reducing the risk of human error. This is particularly beneficial when managing multiple systems, as it ensures that all machines are uniformly updated.

**4.4 Performance Optimization**: Removing unused packages after updates helps free up disk space and prevent system bloat, maintaining system performance over time.

**4.5 Reduced Downtime**: With the implementation of health checks, the system ensures that updates are applied only when sufficient resources are available. Moreover, rebooting only when necessary minimizes unnecessary downtime (only when there has been an update), especially for critical servers that need to maintain high availability.

**5.6 Notifications**: (When fully implemented) The email notification feature will allow system administrators to stay informed about the status of system updates, making it easier to monitor multiple systems remotely and respond to potential issues in real-time.

**5.0 Challenges and Issues**

The most notable challenge is the inability of the system to send email notifications after completing the update process. Although **mailutils** and **postfix** were installed, the system is unable to send emails through the local mail server. A possible solution involves configuring an external SMTP server, such as Gmail's SMTP, to handle email notifications (Kizza, 2020). This issue remains unresolved and requires further troubleshooting of mail server configurations.

**7.0 Conclusion**

By automating the update process, this system ensures that the Linux environment remains secure and free from vulnerabilities that could arise from outdated software (Snyder, 2021). The automation process reduces the risk of human error and guarantees that updates are applied without the need for constant monitoring, making it a highly efficient solution for system administrators (Hatch, Lee, & Kurtz, 2019).And this project has successfully automated the update and maintenance process for Ubuntu Linux systems, ensuring timely updates and efficient performance with minimal manual intervention which streamlines system administration tasks, improves security, and keeps systems running smoothly. Once the email notification feature is completed, the system will be fully capable of managing updates in remote or headless environments, offering peace of mind to administrators managing large fleets of systems.

Automating these processes ensures that the Linux environm

References

Bash Scripting Guide: <https://tldp.org/LDP/abs/html/>

Hatch, B., Lee, J., & Kurtz, G. (2019). Hacking Linux Exposed, 3rd Edition. McGraw Hill.

Kizza, J. M. (2020). Guide to Computer Network Security (4th ed.). Springer.

Postfix Mail Server Documentation: <https://www.postfix.org/documentation.html>

Snyder, A. (2021). Linux System Administration: A Beginner's Guide (7th ed.). McGraw Hill.

Stallings, W. (2019). Operating Systems: Internals and Design Principles (9th ed.). Pearson.

Ubuntu Linux Official Documentation: <https://help.ubuntu.com/>