Green University of Bangladesh

Department of Computer Science and Engineering (CSE)
Semester: (Spring, Year: 2024), B.Sc. in CSE (Day)

Device Health Checker System

Course Title: Operating System Lab

Course Code: CSE-310

Section: 221_D21

Students Details

Name	ID
Md. Harun-Ur-Roshid	221002138
Md. Mostakim Rahman Shipon	221002098

Submission Date: 10/06/2024
Course Teacher's Name: Sudip Chandra Ghoshal

[For teachers use only: Don't write anything inside this box]

Lab Project Status	
Marks:	**Signature:**
Comments:	**Date:**

Contents

Chapter 1

Introduction

1.1 Overview

The Device Health Checker System is a shell scripting-based project aimed at providing users with insights into the health condition of their Linux system. The system will monitor key metrics such as CPU usage, RAM usage, disk space utilization, and other relevant parameters to generate a comprehensive health report. The report will be presented in HTML format for easy visualization and analysis.

1.2 Motivation

With the increasing complexity of modern computing systems, monitoring the health and performance of devices has become crucial for ensuring optimal functionality and reliability. This project seeks to address the need for a simple yet effective solution to monitor and analyze the health condition of Linux-based systems, empowering users to take proactive measures to maintain system stability and performance.

1.3 Problem Definition

1.3.1 Problem Statement

The project aims to address the following challenges:

- Designing shell scripts to collect and analyze system health metrics.
- Generating HTML reports dynamically based on the collected data.
- Ensuring accuracy and reliability of health status information.
- Providing an intuitive interface for users to view and interpret the health report.

1.3.2 Complex Engineering Problem

One of the complex engineering problems involves efficiently collecting and processing system metrics such as CPU usage, RAM usage, and disk space utilization in real-time. Additionally, designing shell scripts to generate HTML reports dynamically while ensuring compatibility across different Linux distributions presents a challenge.

Name of the P Attributess	Explain how to address
P1: Depth of knowledge required	Conduct thorough research on Linux system metrics and shell scripting.
P2: Range of conflicting requirements	Prioritize system metrics based on their importance and impact on system health.
P3: Depth of analysis required	Develop algorithms to analyze system metrics efficiently.
P4: Familiarity of issues	Provide documentation and resources to familiarize stakeholders with system health concepts.
P5: Extent of applicable codes	Ensure compatibility across different Linux distributions.
P6: Extent of stakeholder involvement and conflicting requirements	Communicate with stakeholders to understand their needs and expectations
P7: Interdependence	Implement modular design to manage interdependencies effectively.

1.4 Design Goals/Objectives

- To develop shell scripts to monitor key system metrics including CPU usage, RAM usage, disk space, and other relevant parameters.
- To implement algorithms to analyze the collected data and generate a comprehensive health report.
- To design the HTML report format for easy visualization and interpretation.
- To ensure the scalability and portability of the system across different Linux distributions.
- To provide a user-friendly interface and options for customization.

1.5 Application

The Device Health Checker System has several potential applications:

- **System administrators:** Can use the system to monitor the health and performance of servers and other Linux-based devices.
- **Individual users:** Can utilize the system to optimize the performance of their personal computers and laptops.
- **DevOps teams:** Can integrate the system into their infrastructure monitoring tools for proactive system health management.

Chapter 2

Design/Development/Implementation of the Project

- ## Introduction

The Device Health Checker System is a shell scripting-based project aimed at providing users with insights into the health condition of their Linux system. The system will monitor key metrics such as CPU usage, RAM usage, disk space utilization, and other relevant parameters to generate a comprehensive health report. The report will be presented in HTML format for easy visualization and analysis and also the report will sent to your mail instantly.

- ## Project Details

In this Device Health Checker System projects we use shell script language and also we use html, css and javascript to generate a html file for the report. After generating a report the full report file gonna send to your email.

- ## Projects Image

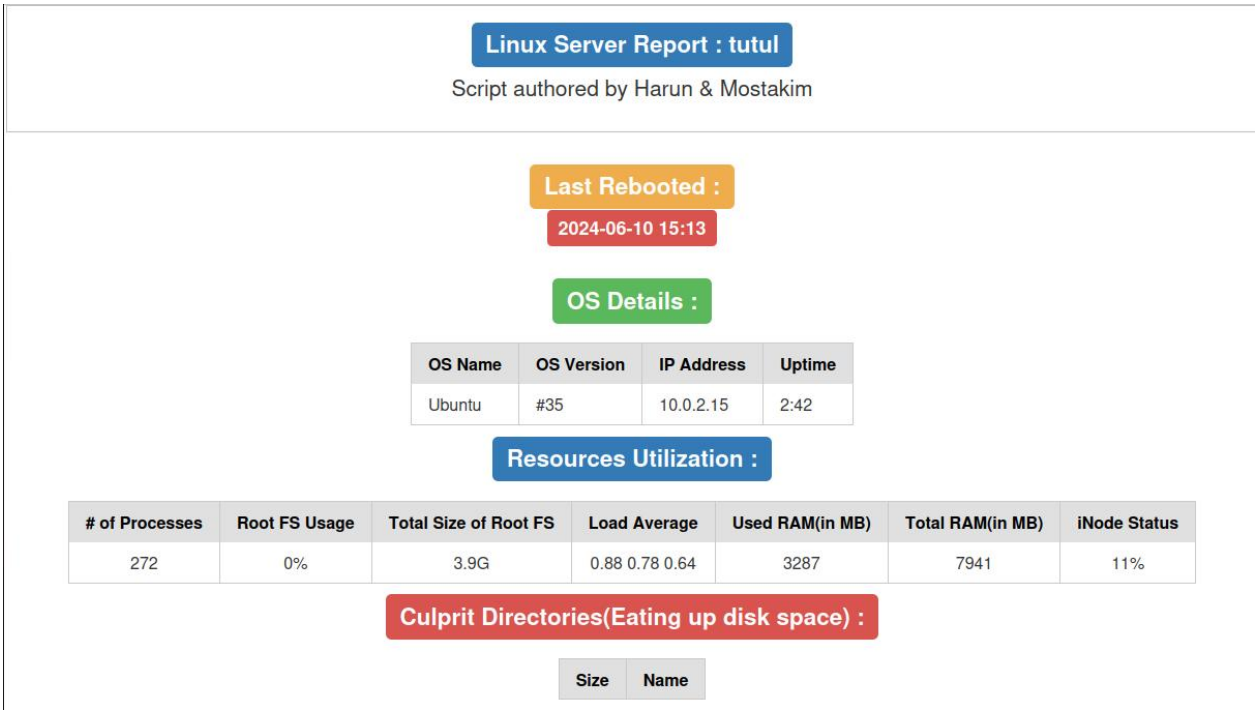


Figure 2.1: Projects Image

- **Implementation**

Here is the full code of our projects:

```
#! /bin/bash
#Author : - Harun & Mostakim
#Date : - 12 June, 2024

#Checking if this script is being executed as ROOT.
if [ "$(id -u)" -ne 0 ]
then
  echo "Please run this script as root so as to see all
details! Better run with sudo."
  exit 1
fi

#Declaring variables
#set -x
os_name=`uname -v | awk {'print$1'} | cut -f2 -d'-'`
upt=`uptime | awk {'print$3'} | cut -f1 -d','`
ip_add=`ip addr | grep "inet " | grep -v "127.0.0.1" |
awk '{print $2}' | cut -f1 -d'/'`
num_proc=`ps -ef | wc -l`
root_fs_pc=`df -h /dev/sda1 | tail -1 | awk
'{print$5}'`
root_fs_pc_numeric=`df -h /dev/sda1 | tail -1 | awk
'{print$5}' | cut -f1 -d'%'`
total_root_size=`df -h /dev/sda1 | tail -1 | awk
'{print$2}'`
#load_avg=`uptime | cut -f5 -d':'`
load_avg=`cat /proc/loadavg  | awk {'print$1,$2,$3'}`
ram_usage=`free -m | head -2 | tail -1 | awk
{'print$3'}`
ram_total=`free -m | head -2 | tail -1 | awk
{'print$2'}`
ram_pc=`echo "scale=2; $ram_usage / $ram_total * 100" |
bc | cut -f1 -d '.'`
inode=`df -i / | head -2 | tail -1 | awk {'print$5'}`
inode_numeric=`df -i / | head -2 | tail -1 | awk
{'print$5'} | cut -f1 -d '%'`
os_version=`uname -v | cut -f2 -d'~' | awk {'print$1'}
| cut -f1 -d'-' | cut -c 1-5`
num_users=`w | head -1 | awk '{print$4}'`
cpu_free=`top -n 1 | awk 'NR>7 {print$9}' | sed -n
'1p'`
last_reboot=`who -b | awk '{print$3, $4}'`
df -h | awk '{print$6, $5}'| grep -v Mounted | cut -f1
-d '%' > /tmp/fsstat.txt

#Creating a directory if it doesn't exist to store
reports first, for easy maintenance.
```

```
if [ ! -d ${HOME}/health_reports ]
then
  mkdir ${HOME}/health_reports
fi
html="${HOME}/health_reports/Server-Health-Report-
`hostname`-`date +%y%m%d`-`date +%H%M`.html"
email_add="221002138@student.green.edu.bd"
for i in `ls /home`; do sudo du -sh /home/$i/* | sort -
nr | grep G; done > /tmp/dir.txt
ps aux | awk '{print$2, $4, $6, $11}' | sort -k3rn |
head -n 10 > /tmp/memstat.txt
top -n 1 | head -17 | tail -11 | awk '{print $1, $2, $9,
$12}' | grep -v PID > /tmp/cpustat.txt

#Generating HTML file
echo "<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">" >> $html
echo "<html>" >> $html
echo "<head>" >> $html
echo "<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/c
ss/bootstrap.min.css" integrity="sha384-
BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K6
8vbdEjh4u" crossorigin="anonymous">" >> $html
echo "<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/c
ss/bootstrap-theme.min.css" integrity="sha384-
rHyoN1iRsVXV4nD0JutlnGaslCJuC7uwjduW9SVrLvRYooPp2bWYgmg
JQIXwl/Sp" crossorigin="anonymous">" >> $html
echo "<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js
/bootstrap.min.js" integrity="sha384-
Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wG
NIcPD7Txa" crossorigin="anonymous"></script>" >> $html
echo "<script type="text/javascript"
src=\"https://www.gstatic.com/charts/loader.js\"></scri
pt>" >> $html
echo -e "<script type="text/javascript">
google.charts.load('current', {'packages':['gauge',
'corechart']});
google.charts.setOnLoadCallback(drawChart);
google.charts.setOnLoadCallback(drawPieChart);
function drawChart() {

  var data = google.visualization.arrayToDataTable([
  ['Label', 'Value'],
  ['# of Processes', $num_proc],
  ['# of Users', $num_users]
  ]);
```

```
  var options = {
    width: 600, height: 175,
    redFrom: 90, redTo: 100,
    yellowFrom:75, yellowTo: 90,
    minorTicks: 5
  };

  var chart = new
google.visualization.Gauge(document.getElementById('cha
rt_div'));

  chart.draw(data, options);
}
function drawPieChart() {

  var data = new google.visualization.DataTable();
  data.addColumn('string', 'Filesystem');
  data.addColumn('number', 'Usage');
  data.addRows([ " >> $html
  while read fs_name fs_usage;
  do
    echo "['$fs_name',$fs_usage]," >> $html
  done </tmp/fsstat.txt
  echo "]);" >> $html
  echo -e "var options = {title:'Overall Filesystem
Usage',
  width:600,
  height:400,
  is3D: true
};
var chart = new
google.visualization.PieChart(document.getElementById('
pie_chart_div'));
chart.draw(data, options);
}
</script>" >> $html
echo "<link rel="stylesheet"
href="https://unpkg.com/purecss@0.6.2/build/pure-
min.css">" >> $html
echo "<body>" >> $html
echo "<fieldset>" >> $html
echo "<center>" >> $html
echo "<h2><span class=\"label label-primary\">Linux
Server Report : `hostname`</span></h2>" >> $html
echo "<h3><legend>Script authored by Harun &
Mostakim</legend></h3>" >> $html
echo "</center>" >> $html
echo "</fieldset>" >> $html
echo "<br>" >> $html
echo "<center>" >> $html
echo "<h2><span class=\"label label-warning\">Last
```

```
Rebooted :</span></h2>" >> $html
echo "<h3><span class=\"label label-
danger\">$last_reboot</span></h3>" >> $html
echo "</center>" >> $html
echo "<br>" >> $html
echo "<center>" >> $html
echo "<h2><span class=\"label label-success\">OS
Details : </span></h2>" >> $html
echo "<br>" >> $html
echo "<table class="pure-table">" >> $html
echo "<thead>" >> $html
echo "<tr>" >> $html
echo "<th>OS Name</th>" >> $html
echo "<th>OS Version</th>" >> $html
echo "<th>IP Address</th>" >> $html
echo "<th>Uptime</th>" >> $html
echo "</tr>" >> $html
echo "</thead>" >> $html
echo "<tbody>" >> $html
echo "<tr>" >> $html
echo "<td>$os_name</td>" >> $html
echo "<td>$os_version</td>" >> $html
echo "<td>$ip_add</td>" >> $html
echo "<td>$upt</td>" >> $html
echo "</tr>" >> $html
echo "</tbody>" >> $html
echo "</table>" >> $html
echo "<h2><span class=\"label label-primary\">Resources
Utilization : </span></h2>" >> $html
echo "<br>" >> $html
echo "<table class="pure-table">" >> $html
echo "<thead>" >> $html
echo "<tr>" >> $html
echo "<th># of Processes</th>" >> $html
echo "<th>Root FS Usage</th>" >> $html
echo "<th>Total Size of Root FS</th>" >> $html
echo "<th>Load Average</th>" >> $html
echo "<th>Used RAM(in MB)</th>" >> $html
echo "<th>Total RAM(in MB)</th>" >> $html
echo "<th>iNode Status</th>" >> $html
echo "</tr>" >> $html
echo "</thead>" >> $html
echo "<tbody>" >> $html
echo "<tr>" >> $html
echo "<td><center>$num_proc</center></td>" >> $html
echo "<td><center>$root_fs_pc</center></td>" >> $html
echo "<td><center>$total_root_size</center></td>" >>
$html
echo "<td><center>$load_avg</center></td>" >> $html
echo "<td><center>$ram_usage</center></td>" >> $html
echo "<td><center>$ram_total</center></td>" >> $html
```

```
echo "<td><center>$inode</center></td>" >> $html
echo "</tr>" >> $html
echo "</tbody>" >> $html
echo "</table>" >> $html
echo "<h2><span class=\"label label-danger\">Culprit
Directories(Eating up disk space) : </span></h2>" >>
$html
echo "<br>" >> $html
echo "<table class=\"pure-table pure-table-
bordered\">" >> $html
echo "<thead>" >> $html
echo "<tr>" >> $html
echo "<th>Size</th>" >> $html
echo "<th>Name</th>" >> $html
echo "</tr>" >> $html
echo "</thead>" >> $html
echo "<tbody>" >> $html
echo "<tr>" >> $html
while read size name;
do
  echo "<td>$size</td>" >> $html
  echo "<td>$name</td>" >> $html
  echo "</tr>" >> $html
done < /tmp/dir.txt
echo "</tbody>" >> $html
echo "</table>" >> $html
echo "<br>" >> $html
echo "<h2><span class=\"label label-info\">Top Memory
Consuming Processes : </span></h2>" >> $html
echo "<br>" >> $html
echo "<table class=\"pure-table pure-table-
bordered\">" >> $html
echo "<thead>" >> $html
echo "<tr>" >> $html
echo "<th>PID</th>" >> $html
echo "<th>%RAM</th>" >> $html
echo "<th>Resident Memory (KB)</th>" >> $html
echo "<th>Command Name</th>" >> $html
echo "</tr>" >> $html
echo "</thead>" >> $html
echo "<tbody>" >> $html
echo "<tr>" >> $html
while read pid ram resident pname;
do
  echo "<td>$pid</td>" >> $html
  echo "<td>$ram</td>" >> $html
  echo "<td>$resident</td>" >> $html
  echo "<td>$pname</td>" >> $html
  echo "</tr>" >> $html
done < /tmp/memstat.txt
echo "</tbody>" >> $html
```

```
echo "</table>" >> $html
echo "<br>" >> $html
echo "<h2><span class=\"label label-primary\">Top CPU
Consuming Processes : </span></h2>" >> $html
echo "<br>" >> $html
echo "<table class=\"pure-table pure-table-
bordered\">" >> $html
echo "<thead>" >> $html
echo "<tr>" >> $html
echo "<th>PID</th>" >> $html
echo "<th>User</th>" >> $html
echo "<th>%CPU</th>" >> $html
echo "<th>Command Name</th>" >> $html
echo "</tr>" >> $html
echo "</thead>" >> $html
echo "<tbody>" >> $html
echo "<tr>" >> $html
while read pid_cpu user cpu_prc cpu_cmd_name;
do
  echo "<td>$pid_cpu</td>" >> $html
  echo "<td>$user</td>" >> $html
  echo "<td>$cpu_prc</td>" >> $html
  echo "<td>$cpu_cmd_name</td>" >> $html
  echo "</tr>" >> $html
done < /tmp/cpustat.txt
echo "</tbody>" >> $html
echo "</table>" >> $html
echo "<br>" >> $html
echo "<h2><span class=\"label label-primary\">Pictorial
Data : </span></h2>" >> $html
echo "<br>" >> $html
echo "<div class=\"panel panel-info\" style=\"width:
40%;\"> " >> $html
echo "<div class="panel-heading"> " >> $html
echo "<h3 class="panel-title">General Stats</h3> " >>
$html
echo "</div> " >> $html
echo "<div class="panel-body"> " >> $html
echo "<center><div id=\"chart_div\"></div></center>" >>
$html
echo "</div> " >> $html
echo "</div> " >> $html
echo -e "<div class=\"panel panel-info\" style=\"width:
50%;\">
  <div class="panel-heading">
    <h3 class="panel-title">Filesystem Usage Stats</h3>
  </div>
  <div class="panel-body">
<div id=\"pie_chart_div\"></div></div>
</div>
</div>" >> $html
```

```
echo -e "<br>
<div class=\"panel panel-primary\" style=\"width:
40%;\">
<div class=\"panel-heading\">
<h3 class=\"panel-title\">Resources Overview</h3>
</div>
<div class=\"panel-body\">
<b>Root Filesystem</b>
<div class=\"progress\" style=\"width: 55%;\">" >>
$html
#set -x
if [ $root_fs_pc_numeric -ge 85 ]
then
  echo "<div class=\"progress-bar progress-bar-danger
progress-bar-striped\" role=\"progressbar\" aria-
valuenow="60" aria-valuemin=\"0\" aria-valuemax="100"
style=\"width: $root_fs_pc;\">" >> $html
else
  echo "<div class=\"progress-bar progress-bar-info
progress-bar-striped\" role=\"progressbar\" aria-
valuenow="60" aria-valuemin=\"0\" aria-valuemax="100"
style=\"width: $root_fs_pc;\">" >> $html
fi
echo "$root_fs_pc" >> $html
echo -e "</div>
</div>
<b>Memory</b>
<div class="progress" style=\"width: 55%;\"> " >> $html
if [ $ram_pc -ge 90 ]
then
  echo "<div class=\"progress-bar progress-bar-danger
progress-bar-striped\" role="progressbar" aria-
valuenow="60" aria-valuemin="0" aria-valuemax="100"
style=\"width: $ram_pc%;\">" >> $html
else
  echo "<div class=\"progress-bar progress-bar-info
progress-bar-striped\" role="progressbar" aria-
valuenow="60" aria-valuemin="0" aria-valuemax="100"
style=\"width: $ram_pc%;\">" >> $html
fi
echo "$ram_pc%" >> $html
echo -e "</div>
</div>
<b>CPU Free</b>
<div class="progress" style=\"width: 55%;\"> " >> $html

if [ $cpu_free -le 50 ]
then
  echo "<div class=\"progress-bar progress-bar-danger
progress-bar-striped\" role="progressbar" aria-
valuenow="60" aria-valuemin="0" aria-valuemax="100"
```

```
style=\"width: $cpu_free%;\"> " >> $html
else
  echo "<div class=\"progress-bar progress-bar-info
progress-bar-striped\" role="progressbar" aria-
valuenow="60" aria-valuemin="0" aria-valuemax="100"
style=\"width: $cpu_free%;\"> " >> $html
fi

echo "$cpu_free%" >> $html
echo -e "</div>
</div>
<b>iNodes</b>
<div class="progress" style=\"width: 55%;\"> " >> $html
if [ $inode_numeric -gt 90 ]
then
  echo "<div class=\"progress-bar progress-bar-danger
progress-bar-striped\" role="progressbar" aria-
valuenow="60" aria-valuemin="0" aria-valuemax="100"
style=\"width: $inode;\"> " >> $html
else
  echo "<div class=\"progress-bar progress-bar-info
progress-bar-striped\" role="progressbar" aria-
valuenow="60" aria-valuemin="0" aria-valuemax="100"
style=\"width: $inode;\"> " >> $html
fi
echo "$inode" >> $html
echo -e "</div>
</div>
</div>
</div>" >> $html
echo "</body>" >> $html
echo "</html>" >> $html
echo "Report has been generated in
${HOME}/health_reports with file-name = $html. Report
has also been sent to $email_add."

#Sending Email to the user
cat $html | mail -s "`hostname` - Daily System Health
Report" -a "MIME-Version: 1.0" -a "Content-Type:
text/html" -a "From: Tutul <root@tutul.com>" $email_add
```

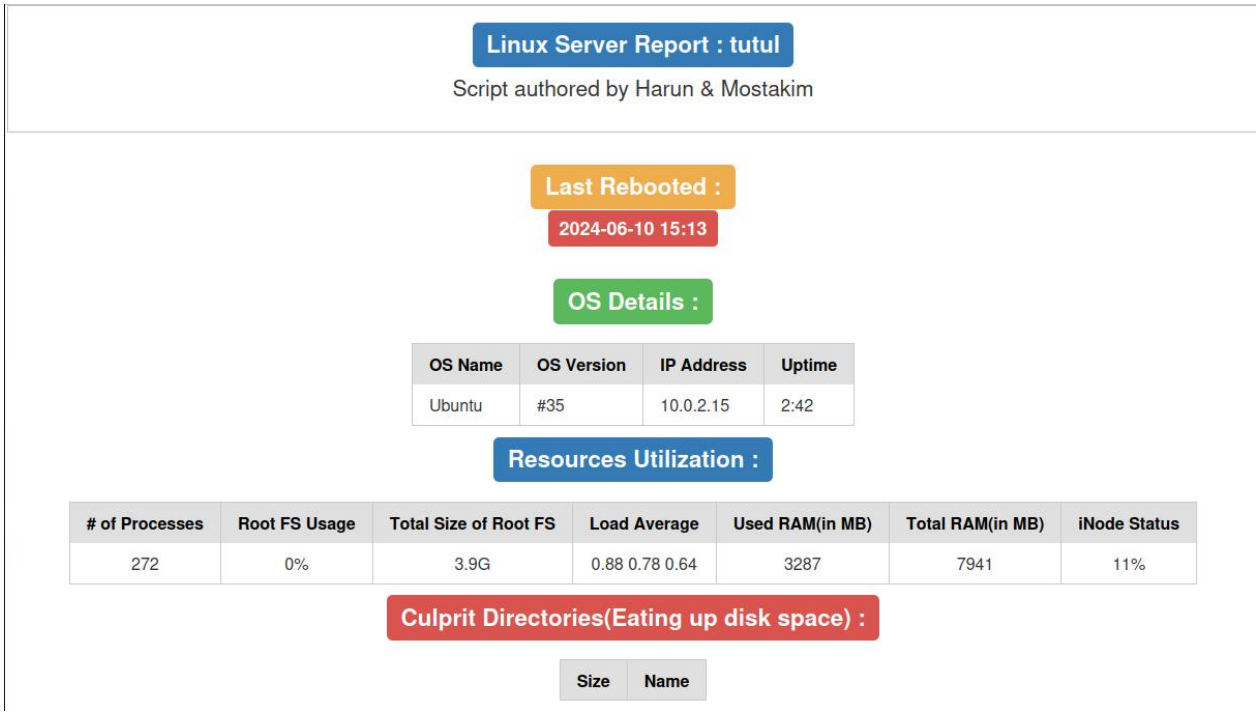
Algorithms

1. START
2. CHECK if the script is run as ROOT
    a. IF not, PRINT "Please run this script as root so as to see all details! Better run with sudo."
    b. EXIT
3. DECLARE variables
    a. Retrieve and assign system metrics to variables
4. CREATE directory ${HOME}/health_reports if it doesn't exist
5. INITIALIZE HTML report file
    a. ADD HTML structure and styles
6. INSERT system metrics into HTML
    a. Last reboot time
    b. OS details
    c. Resource utilization
7. ADD tables for
    a. Directories consuming the most disk space
    b. Top memory consuming processes
    c. Top CPU consuming processes
8. ADD charts and graphical representations
9. ADD resource overview with progress bars
    a. Conditional formatting based on thresholds
10. FINALIZE and close HTML structure
11. PRINT message about report location and email
12. SEND email with the HTML report
13. END

Chapter 3

Performance Evaluation

3.1 Simulation Environment/ Simulation Procedure



3.2 Results Overall Discussion

From this projects we can see our device name, os name, version, ip address, up time, last rebooted time, no of running process, used ram and etc. Most of the feature was working well.

3.2.1 Complex Engineering Problem Discussion

Creating a Device Health Checker involves making a system that can monitor and report the health status of various devices, especially servers. This project is complex because of the variety of devices, environments, and performance measures involved. The aim is to provide real-time monitoring, collect accurate data, create clear visual reports, and send timely updates, all while keeping the system secure and not slowing down the devices.

Chapter 4

Conclusion

4.1 Discussion

Our projects was working well but few feature was not running well. We tried our best to run every feature but cpu proccess feature was not running well. Also file system feature was not working. We tried to send our report to our mail but that feautre is also not working properly.

4.2 Limitations

- **Static Data:**
  - This projects only gives us static data not real time dynamic data.
- **Root file not access:**
  - Though we have to see the report of system sometimes root file was not access properly.
- **Send Data to Email:**
  - Sending report of system data is not working.

4.3 Scope of Future Work

- **Computer Application:**
  - We will make a computer application of our projects in future.
- **Real time Monitor:**
  - Though our projects gives us static value we will upgrade realtime dynamic feature of our projects in next update.
- **Send Data to Email:**
  - In next update we can add send report to your email feature.