Operating Systems
Project Report

Introduction

The following report describes the objective of our project, the methods used, and the final results. The Task Manager Application for Android smartphones collects rooted information from the Android Operating System. It concentrates on the specifics, services, and CPU consumption. A device must be rooted to utilize the program. This application runs Linux commands to acquire information about the operating system.

Objective

The primary goal of this application is to collect information on a rooted device's details, services, and CPU utilization. It will take the information, disassemble it, and provide it to the user via a scrollable user interface. You may also use this application to examine the various permissions and memory, as well as forcibly stop or entirely uninstall it. The benefit of this application is that it allows you to collect device details, services, and CPU information.

Features

There are 3 different tabs of information in the Task Manager Application. The 3 different tabs show 3 different key pieces of information. And we implemented it as follows:

- **App Details Button**
  The App Details button shows you all the available information about the Task Manager Application. This is including if the user would like to know more about the application’s storage, permissions, notifications, and memory. This allows the user to also force stop or uninstall the application.

- **Search view**
  It is found at the top to filter the different processes if you are looking for a specific one without having to spend time finding it.

- **Details Tab**
  The Details Tab shows all processes from all available users. To retrieve the running processes from all the available users, the command “su -c ps” is being run. A SearchView is found at the top to filter the different processes if you are looking for a specific one without having to spend time finding it.

- **Services Tab**
  The Services Tab shows all available services, as well as the currently running ones. To retrieve all available services the command “su -c service list” is being run. To retrieve only the running services, the command “cmd -l” is being run.

- **CPU Tab**
  The CPU tab shows a lot of useful information about the CPU. This would include uptime, load average, CPU utilization, and the CPU % of each process.

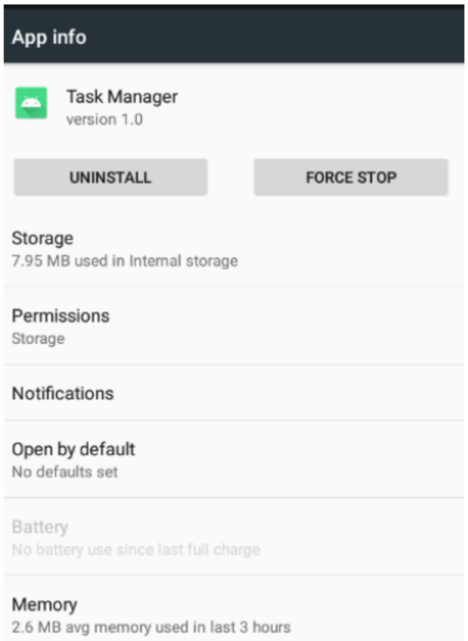
- **Stopping the Task Manager Application**
  There are several ways that you can stop the Task Manager Application.
  - Stop button
  - From toolbar
  - Keyboard shortcut
  - Force stop

Methodology

- The application requires root access to gather and execute certain Linux commands. This information pertains to details, services, and CPU utilization.
- To gather information for each tab in the application, we executed different Linux commands in the superuser. After getting the output from these commands as one large string, we broke it up line by line to display the information in a ListView format with an ArrayAdapter.
- On the details tab being that there were so many processes, we gave the user the ability to search the list and find the exact one they want.
- On the services tab we were able to use two separate commands to determine which ones are active, so made it possible to just look at active or all services.
- Our CPU tab gives us CPU utilization information as well as the CPU% of each process running. There are many different aspects that we would like to improve on with our application. Our team would like to eventually find a way to be able to kill processes and services. Overall, this application gathers accurate protected operating system information and displays the output in a readable format for the user.

Results

App Details Button



App info

Task Manager
version 1.0

Storage
7.95 MB used in Internal storage

Permissions
Storage

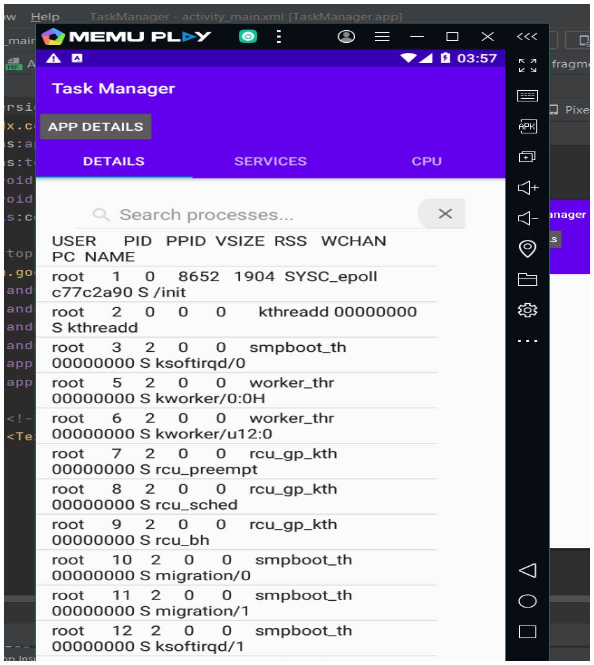
Notifications

Open by default
No defaults set

Battery
No battery use since last full charge

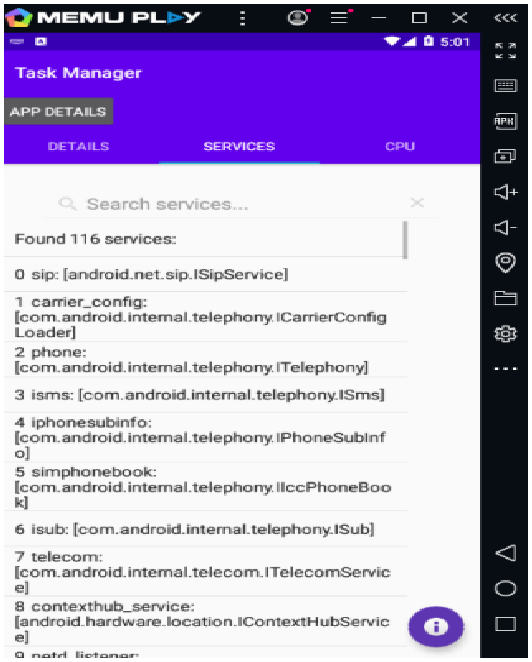
Memory
2.6 MB avg memory used in last 3 hours

Details Tab



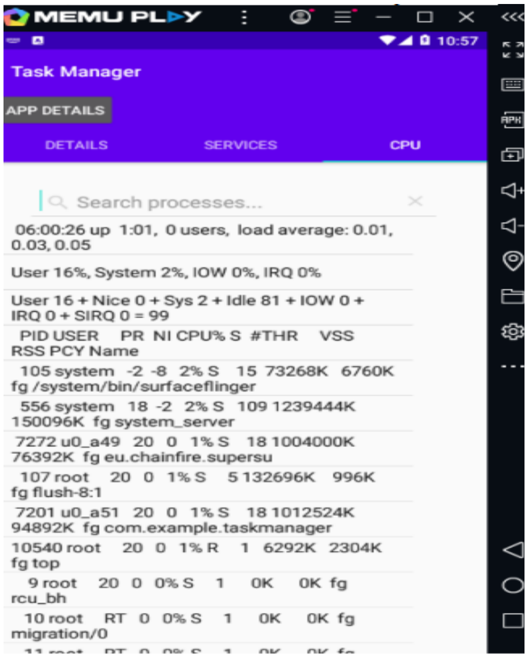
```
USER    PID  PPID  VSIZE  RSS   WCHAN
PC  NAME
root     1    0    8652   1904  SYSC_epoll
c77c2a90 S /init
root     2    0    0      0     kthreadd 00000000
S kthreadd
root     3    2    0      0     smpboot_th
00000000 S ksoftirqd/0
root     5    2    0      0     worker_thr
00000000 S kworker/0:0H
root     6    2    0      0     worker_thr
00000000 S kworker/u12:0
root     7    2    0      0     rcu_gp_kth
00000000 S rcu_preempt
root     8    2    0      0     rcu_gp_kth
00000000 S rcu_sched
root     9    2    0      0     rcu_gp_kth
00000000 S rcu_bh
root     10   2    0      0     smpboot_th
00000000 S migration/0
root     11   2    0      0     smpboot_th
00000000 S migration/1
root     12   2    0      0     smpboot_th
00000000 S ksoftirqd/1
```

Service Tab



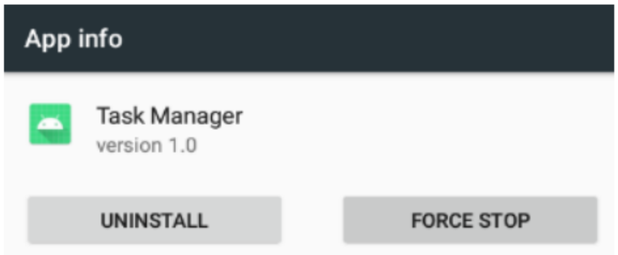
```
Found 116 services:
0 sip: [android.net.sip.ISipService]
1 carrier_config:
[com.android.internal.telephony.ICarrierConfig
Loader]
2 phone:
[com.android.internal.telephony.ITelephony]
3 isms: [com.android.internal.telephony.ISms]
4 iphonesubinfo:
[com.android.internal.telephony.IPhoneSubInf
o]
5 simphonebook:
[com.android.internal.telephony.IIccPhoneBoo
k]
6 isub: [com.android.internal.telephony.ISub]
7 telecom:
[com.android.internal.telecom.ITelecomServic
e]
8 contexthub_service:
[android.hardware.location.IContextHubServic
e]
9 netd_listener:
```

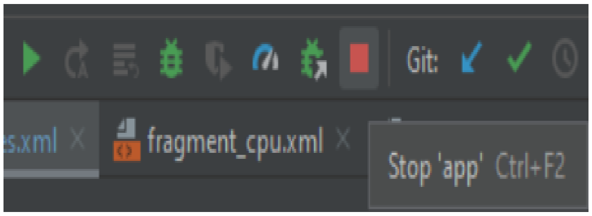
CPU Tab

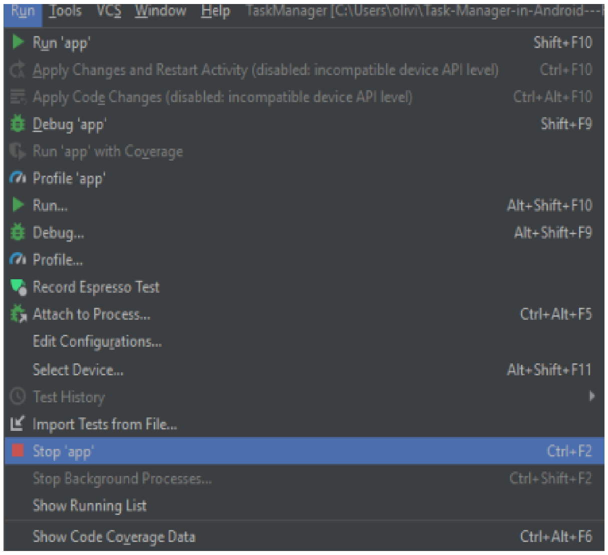


```
 06:00:26 up  1:01,  0 users,  load average: 0.01,
0.03, 0.05
User 16%, System 2%, IOW 0%, IRQ 0%
User 16 + Nice 0 + Sys 2 + Idle 81 + IOW 0 +
IRQ 0 + SIRQ 0 = 99
  PID USER     PR  NI CPU% S  #THR     VSS
RSS PCY Name
  105 system   -2  -8   2% S    15  73268K   6760K
fg /system/bin/surfaceflinger
  556 system   18  -2   2% S   109 1239444K
150096K  fg system_server
 7272 u0_a49   20   0   1% S    18 1004000K
76392K  fg eu.chainfire.supersu
  107 root     20   0   1% S     5 132696K    996K
fg flush-8:1
 7201 u0_a51   20   0   1% S    18 1012524K
94892K  fg com.example.taskmanager
10540 root     20   0   1% R     1   6292K   2304K
fg top
    9 root     20   0   0% S     1      0K      0K  fg
rcu_bh
   10 root     RT   0   0% S     1      0K      0K  fg
migration/0
```

Stopping the Task Manager Application






Run
Tools
VCS
Window
Help
TaskManager [C:\Users\olivi\Task-Manager-in-Android--
Run 'app' Shift+F10
Apply Changes and Restart Activity (disabled: incompatible device API level) Ctrl+F10
Apply Code Changes (disabled: incompatible device API level) Ctrl+Alt+F10
Debug 'app' Shift+F9
Run 'app' with Coverage
Profile 'app'
Run... Alt+Shift+F10
Debug... Alt+Shift+F9
Profile...
Record Espresso Test
Attach to Process... Ctrl+Alt+F5
Edit Configurations...
Select Device... Alt+Shift+F11
Test History
Import Tests from File...
Stop 'app' Ctrl+F2
Stop Background Processes... Ctrl+Shift+F2
Show Running List
Show Code Coverage Data Ctrl+Alt+F6

Challenges

- The biggest challenge for this project was related to the security of the privileged information we want to extract. The information we want is protected because the operating system relies on it. Any app installed will not have access to this operating system information.

- In our application, we wanted to enable the user to be able to kill the processes and services listed in each tab. Our team was unsuccessful in accomplishing this task. We were able to get the PID by using a substring() method inside the for loop to display the output, but when each item was clicked in the ListView, it was not retrieving the correct item in the list, but only the last one in the list.

Contributions

- Mruganshi Gohel:
  - Coded, implemented and debugged the service tab
  - Presentation
- Harshita Gupta:
  - Coded, implemented and debugged the CPU tab
  - Report
- Atul Khobragade:
  - Coded, implemented and debugged the details tab
  - Readme