CPSC 4240: Machine Performance Tool

Spring 2023

Michael Harris
Student
Clemson University
Clemson SC USA
mah6@g.clemson.edu

John Mathews
Student
Clemson University
Clemson SC USA
mathew7@g.clemson.edu

ABSTRACT

This paper summarizes a programming project called Machine Performance. Machine Performance is a command-line tool built for Linux (VM - Ubuntu 22.04.2) operating systems that monitors system and network performance. The tool was developed using Visual Studio Code with Python 3 in combination with the psutil (5.9.5) library. Machine Performance is open source and found at https://github.com/thejohnmathews/Machine-Performance.

1. INTRODUCTION

Computers are extremely powerful and useful machines. By using hundreds of millions of transistors within loads of other hardware components, these technologies create better opportunities for humans to learn and grow as a whole. The mind-boggling amount of power and intelligence that these machines contain is incredible and impressive, therefore it is extremely important to ensure these machines are running at peak efficiency while retaining their longevity. Many components are very important when it comes to overall machine performance and efficiency, but some of the most important sectors are hardware performance, system performance, machine processes, and the machine's network. The combination of these factors are imperative when it comes to maintaining a fully functional machine that can be used for such a wide range of beneficial and important purposes.

Hardware performance is the foundation of having a properly functioning machine. The environment in which hardware is assembled in the machine is such a fragile yet incredibly important factor. The amount of microscopic moving parts and electronically powered systems allow for a multitude of problems and mistakes to arise throughout the machine's lifespan. This makes hardware monitoring and maintenance an extremely important aspect in machine function.

Overall system performance is another huge factor when it comes to the efficiency of a machine. The system and firmware that runs on the built-in hardware is what makes the machine actually useful to us. These systems are also very fragile due to the varying factors that can impact them. The system should be monitored regularly to ensure that there are no bugs, glitches, or problems that have occurred in the machine. This monitoring will also allow for fixes to be made quicker and more effectively.

A process in a machine is an instance of a computer program that is being executed by one or many threads. These processes are essential to having a machine that can do a wide variety of tasks and applications. At any time, there can be hundreds or thousands of processes running on a machine, and all of them need system and hardware resources to complete their tasks. The idea of monitoring and controlling these processes allows for fine-tuning the system and understanding what is actually happening in the machine in terms of real-time programs.

In a world that is so interconnected through our technology, the network communication and especially the machine's network is a pivotal part into the uses and purpose of today's machines. Since the development of the Internet and other network-dependent tools, monitoring and evaluating a machine's network performance is extremely common. In order for these machines to be useful, the network must be working at maximum efficiency. Like all other computer systems, the network can be very fragile and sensitive, so it is imperative for network-specific monitoring of efficiency and effectiveness.

These four factors are massive pillars in the world of computers. They all play a huge role in the usefulness of these machines, and the combination of them must be perfect. Because of this fact, ensuring that all these components are at maximum efficiency and health makes computers the most useful and innovative tool in the history of the world.

2. BACKGROUND

Performance analysis is a critical component of system administration and software development. From designing large scale programming systems to defending against bad actors, knowing the limits and specifications of your machine can make a large difference.

2.1 Psutil

The backbone of Machine Performance is the psutil library. Psutil is the most useful library because it includes major

documentation useful for UNIX systems by combining lots of common command-line programs into one place. According to the documentation for psutil it is useful for: "system monitoring, profiling and limiting process resources and management of running processes … [and] it implements many functionalities offered by classic UNIX commands such as ps, top, iotop, lsof, netstat, ifconfig, free, etc."[1].

2.2 System Security

Working with computers can pose security threats unknown to the user. Typically security flaws in a machine is the operating system/kernel; however, there can be threats that can be exploited in the hardware of a machine. For experts who need to defend hardware attacks, they must know all the specifications of the hardware to ensure they are thorough in fixing vulnerabilities. A common practice invented in the 1970s, information flow tracking (IFT), can help make a difference in hardware security. According to an article published in ACM, "Hardware IFT techniques specifically target security vulnerabilities related to the design, verification, testing, manufacturing, and deployment of hardware circuits … [it] provides a powerful technique for identifying hardware security vulnerabilities as well as verifying and enforcing hardware security properties" [2]. As you can see Machine Performance can have utility in helping the user get to know the hardware better, in turn helping prevent hardware vulnerabilities.

3. MOTIVATIONS AND OBJECTIVES

3.1 Motivations

Simply put, Machine Performance reduces the amount of system monitoring applications that a user may need to run. Combining programs such as top and ps save the user the time and effort needed to navigate through these applications. This allows the user to focus on the objectives of Machine Performance: monitor, control, and inform about the system's performance and networking.

3.2 Objectives

The brainstorming stage had four key objectives that needed to be drawn out and implemented into the code.

3.2.1 Objective One: Identify Components

The first objective for Machine Performance is to: identify all hardware and input/output (IO) devices that are currently using machine resources and prompt the user to make decisions based on these findings.

3.2.2 Objective Two: Processes

The second objective for Machine Performance is to: list processes running on the CPU and show how this can affect system performance. This objective can help the user find out why their CPU may be slow and help identify the main processes affecting CPU utilization. Additionally, this allows the user to fully control the statuses of each process.

3.2.3 Objective Three: Memory

The third objective for Machine Performance is to: display the machine's current memory statistics and show how this can affect system performance. This can help users free up memory from programs or find resource limited processes on the machine.

3.2.4 Objective Four: Networking

The final objective for Machine Performance is to: list how processes are using network resources on the machine. Networking information allows the user to see what protocols and resources each process uses.

4. METHODOLOGY AND DESIGN

4.1 Core Design

The core of Machine Performance is that it is command-line based so that it can function virtually the same across different distributions of Linux operating systems. To make navigation easy for the user, Machine Performance uses a main menu to access the menu options. These options then display a submenu that allows the user to perform the needed actions. When the user selects an action for Machine Performance to perform, it outputs the text to the terminal in an easy to read format. After displaying the text, the program returns automatically to the main/sub menu.

4.2 Methodology

The backend code for Machine Performance mainly uses library function calls to psutil. All of our objectives have a focus on displaying its specific information to the user. Implementing these objectives with something such as: "Basic Hardware Information" is just a print() statement with the psutil function calls concatenated to the output string. Objective two and three share a goal of letting the user see CPU/RAM utilization for current processes. In order for the user to fully grasp and understand the live utilization, the program allows the option to display a live graph (similar to Windows Task Manager's Performance Tab). When the user selects "Show CPU Performance" or "Show Memory Performance" in the "System Performance" submenu, it prints the typical information followed by a prompt to see the graph by entering "g". When the user enters "g", a live ASCII graph will display the utilization over fifteen seconds. The graph is made of asterisks(*) and higher utilization is represented by a tall bar of asterisks. Since the graphs record over fifteen seconds, the user can open/close other processes to see how it affects the CPU/RAM utilization. This effect is shown below in Figure 1 (each asterisk is 5% utilization).

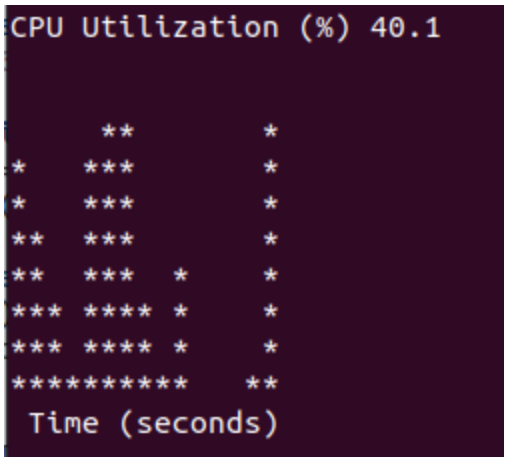

Figure 1. CPU utilization graph with varying loads

Lastly, objective two states that it allows the user to fully control the statuses of every process. The "Process Information" submenu has another submenu called "Process Control Menu" that lets the user change the statuses of a process using its PID. This submenu was built with Task Manager in mind and can: resume, suspend, terminate, kill, and force wait processes. This allows the user to force kill any process including Machine Performance as seen in Figure 2.

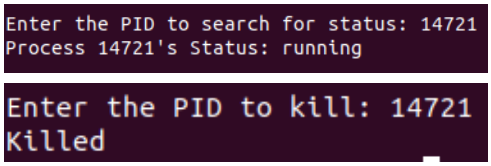

Figure 2. Killing Machine Performance's PID

5. ANALYSIS AND RESULTS

In terms of the four major components when it comes to machine performance and efficiency, our Machine Performance tool covers all the bases. The hardware information section of the program covers very critical and important hardware information, with a multitude of different information points and statistics. This information includes CPU type and cores, RAM and Swap memory, Disk Usage, Boot Time, and some sensor statistics. This large array of information can be used to make determinations about performance and to ensure that the hardware efficiency is still high.

The system performance section of our program also plays a big role in the user's goal to monitor and evaluate their machine. Our program uses the system performance section to present information by showing CPU, Memory, and Disk information. The CPU section shows CPU utilization and speed, number of processes and threads, and the system cache. The Memory section current memory usage, total memory space, and committed and cache memory usage. The Disk section shows total capacity, used memory, read and write counts, read and write speed, and the machine's number of partitions. These three subgroups in the overall system performance section allow for a wide range of information and statistics to be shown about the entire system. These subgroups can be used for many different tasks, like troubleshooting and evaluation of the system.

The Current Processes section of our program is a very versatile and effective part of our application. There is a large amount of functionality and choices that can be made by the user. The sections in the Processes section include showing the current processes, general information, child and parent processes, status, CPU time, memory information, environment variables, and the process control menu. This menu can be used to resume, suspend, terminate, kill, and force wait any process that is inputted. This section has a huge amount of information about all of the processes on the machine. Since these processes make up the entire workflow of the machine, this section is very useful to fine-tune the machine and monitor the efficiency of the computer.

Finally, we have the network information section of our program which is an integral part of machine performance. The options in this section include input/output counters, network addresses, network statistics, and showing network connections of a specified type. This part allows for user input in order to specify which type of connection should be shown. The network portion of our program shows information about the machine's connections outside of the hardware of the computer. This can be used to troubleshoot and monitor the entire system so it can continue to be used in an online environment.

The four major parts of our program assist the user of the machine in a multitude of ways. The amount of information it returns can be used to complete many tasks and to guarantee the health and efficiency of the machine stays at a high mark. Our program is effective in its functionality to control and monitor the entire system to uphold a valuable and effective tool.

6. CONCLUSIONS & FUTURE WORK

6.1 Conclusions

6.1.1 Summary and Conclusions

We have created a programming project called Machine Performance. Machine Performance is a command-line tool built for Linux (VM - Ubuntu 22.04.2) operating systems that monitors system and network performance. The tool was developed using Visual Studio Code with Python 3 in combination with the psutil (5.9.5) library. Machine Performance is open source and freely available.

Throughout the design, implementation, and output of our program, we learned the complex and massive amount of information that is contained in the machine. We had to research the most important components of a computer and

find out ways to efficiently monitor and evaluate the machine in a user-friendly manner. We also had to learn how a majority of the psutil library worked, and figured out a way to use this library in the most effective way. This project helped us understand the importance of monitoring and controlling the system in order to keep it at maximum efficiency and to uphold the longevity of the system.

6.1.2 Weaknesses of Machine Performance

Weaknesses of Machine Performance are seen from the front end to the back end of the program. An obvious front end weakness is that Machine Performance does not have a simple GUI to make the program look and feel easy to navigate. This roughness can also be seen in the utilization graphs, as they do not have visible axes and can result in confusion. On the back end, Machine Performance is a long program that could take more time to run than other UNIX based programs. Also, some calls to psutil functions could be printed to standard output to reduce user confusion.

6.2 Future Work

Even though Machine Performance achieved all of its initial objectives, there are plenty of ways to improve the program and add to its core functionality, readability, and versatility . A core option that could be beneficial to the main menu would be “System Energy Usage”. This option would allow the user to see active energy consumption by hardware component as well as an overall utilization graph. Another key to the program’s functionality would be to provide hardware and utilization information for the system’s GPU. To improve Machine Performance’s readability, colored text and a proper Python 3 graph library can be added to the program. Colored text could help separate the differences between the main menu and submenus as well as separate psutil’s output text. A Python 3 graphing library such as Plotly would help make the graphs look more professional and readable to the user. Finally to make the program smoother, multi-threading could be implemented to allow the utilization graphs to exist and record utilization concurrently with the rest of the program (similar to Task Manager).

7. REFERENCES

[1] Anon. PSUTIL documentation. Retrieved May 2, 2023 from https://psutil.readthedocs.io/en/latest/

[2] wei Hu, Armaiti Ardeshiricham, and Ryan Kastner. 2021. Hardware information flow tracking. *ACM Computing Surveys* 54, 4 (2021), 1–39. DOI:http://dx.doi.org/10.1145/3447867