Technical Evaluation of an Operating System

Dicksania Rivas

Operating Systems

IT-600

Southern New Hampshire University

August 20, 2020

Dr. Robert Flowers

Table of Contents

**Introduction……………………………………………………………………………….4**

**Organizational Needs and Requirements………………………………………**

**Computer Architecture………………………………………………………………..**

**Process Management…………………………………………………………………….**

**Memory Management…………………………………………………………………….**

**I/O and Mass Storage……………………………………………………………………...**

**security………………………………………………………………………………………...**

**Evaluation……………………………………………………………………………………**

**References…………………………………………………………………………………….**

Abstract

This report will evaluate Top Secret, Inc. a successful renowned operation system company. In this evaluation we will analyze TSI’s operating system, asses the hardware-software interface, the procedures used at implementation, the systems used to support the operating system as well as the security features.

**Technical Evaluation of an Operating System**

TSI-Top Secret Incorporated is a prosperous operating system organization which clients include Fortune 500 companies, International government, and leading contractors in the United States. TSI makes embedded operating systems for secure terminals that control incoming/outgoing control systems for Wall Street firms, camera systems for drone aircraft for government contractors, and alarm systems for top-secret government installment. TSI operating systems are world-renowned for their swift feedback to sensor input, exceptional quality of operations, contained memory utilization, the minor size on disk, and low power usage.

**Organizational Needs and Requirements**

**Organizational Profile**

TSI needs a system that provides services of the operating system to its user programs through an application programming interface. Kernels use the system calls as entry points to perform essential tasks. TSI needs to complete the responsibility that goes further than opening and closing files and reading and writing. A thread is the virtual interpretation of a central processing unit. It allows a single task to have numerous code segments. Multithreading is the capacity for a process to conduct multiple users at a time and manage requests from the same user without the need to open multiple instances.

Another issue TSI is experiencing is that it drops background operations. TSI needs to make multithreading changes to prevent further hang-ups. A thread that shares resources of the parent process to create new threads and swap between them is an option. The TSI operating systems uses a flat memory model without paging. The operating system needs to swap, which is writing the entire process, not just some of it onto a disk. Without paging file the moment that the memory runs out, there is no paging to fall over to causing a fight of resources and cause a crash or not sufficient memory to complete the background process.

**Security Requirements**

TSI operating system uses sensor input operations which are not secured. Tasks such as login, modifications to the file system, or structure security are not protected. Because of this, anyone can make changes to the application. There are many reasons why security is essential in business, to provide safe service, protect personal data from employees and customers, protect passwords, protect the database and service crash. By adding security to any application, it provides the benefit of data integrity, by providing or restricting access to various parts of the application based on permissions. An additional benefit of security is that with the data integrity and any added authentication needs, it allows the application data to remain confidential. (Abousen, 2015)

**Performance Requirements**

Discuss how performance is important to the organization and whether it is a large or small need. For example, do they need SMP, RAID 0, or 10Gbps networking drivers? (1-2 Paragraphs)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | RAID 0 | RAID 1 | RAID 5 | RAID 10 |
| Benefits | Provides very  fast access by  spreading data  over across all  member disks | An exact replica of disks, which theoretically improves performance by 2. Simplest to recover data from in the event of a controller failure, as it just needs to be connected to a functioning computer. | Spreads data across disks similar to RAID0; however, it adds a parity block to each row. The parity block is used in the event of a disk failure to reconstruct the contents so that it can be accessed. | Secure, fast, reliable, and ample storage. Works best in environments that call for high security and performance. Data can be written together on many drives. |
| Drawbacks | If one disk fails,  all data is lost  because each  block  represents the  ONLY copy of  the data | If a drive were to fail, there is limited ability to support hot-swapping of the hard drive. This results in downtime, where the system is not accessible to the end-users. 50% capacity is available due to data being duplicated. | Read performance impacted if one drive fails. No significant write speed improvements. Requires a specific type of software to recover data should the controller fail | Read speed is twice as slow. Overhead is higher due to the number of disks, making it the most expensive in large capacity. Data works on half the storage due to mirroring, and in the case, more than one drive in the same mirror fails, then data is lost. |
| Business Impact | Businesses that  need the fastest  access with no  redundancy will  benefit.  however,  businesses  sensitive to  data loss should  avoid RAID 0 | While from a financial aspect, this is the lowest cost for redundancy with the added benefit of improved Read speeds, the potential for downtime when a drive does fail needs to be included in any discussions. | Best compromise between RAID0 and RAID1 by having a minimum of 3 drives, which increases the cost. There is still a decrease in overall storage. However, data is more protected in the event a single drive fails. | Helps businesses in performance and data redundancy. Four storage drives would be needed to extend capacity. While RAID 10 has the most significant benefits, it is also at the highest cost. |

**Reliability Requirements**

Fault tolerance is the ability of the computer system to continue working while manipulating the failure of hardware or software. Fault tolerance uses backup elements that systematically replace the failed components making sure the service is not affected. Without fault tolerance being enabled, it puts the company at risk of data loss in the event of a hardware failure, such as a hard drive failing.

By TSI implementing a fault tolerance solution, this will minimize the impact and time to recover if and when a drive were to fail. More importantly, the data that the application is accessing would remain redundant, limiting the impact to user performance in the event of a failure (SafeBytes, 2016) It would be recommended for TSI to create a RAID10 storage array to store the application data, along with a shared copy, so there were two locations for the data. For fault tolerance of the operating system, it would be recommended to implement a minimum RAID1 configuration to avoid as much user impact as possible in a drive failure (Datapacket, 2020)

**Management and Maintenance**

Discuss any specific needs the organization has with regard to management and maintenance (1-2 Paragraphs). Here are a few examples of what should appear here, but you should add your own:

1. Do they need patch management in order to meet the Security Requirements above?
2. Do they require remote operating system management software to meet the Reliability Requirements above?
3. What challenges have they had with operating system maintenance?

As previously discussed, TSI operating systems use a primary system call interface. The interface is essential because it is a way for systems to communicate. In this case, the software compatibility issues resolved by adding an arduous unnecessary process to address the problem by going around and modifying traps to kernel mode and making call responses.

TSI coders create their own customized drivers. TSI is currently experiencing functionality problems with their drivers having various limitations due to their size. Part of the issues within the background operations is a result of creating customized device drivers for each client. This practice is not scalable because they would have to maintain multiple drivers as opposed to using a single, standardized driver.

RAID cards are storage configuration. They require more than two drives to complete. RAID can stand hard drive failure. TSI is having problems because all their data gets stored in a single hard drive, creating a single point of failure with no redundancy.

**Computer Architecture**

**High Level Hardware**

Describe the hardware needed to implement your proposed solution. Be sure to discuss the type of microprocessors as well as the memory and disk (e.g., RAID 1, RAID 5, etc.) details. (2-3 Paragraphs)

**Specific Hardware Architecture**

Describe the specific architecture of microprocessors (e.g., multi-core, hyperthreading, SMP, CISC/RISC, ARM, x86, etc.) and any exotic or specialized hardware you are recommending to support your new operating system. (2-3 Paragraphs)

**Hardware Limitations**

Talk about any constraints your proposed solution may present. For example, is your solution better for certain applications and not others or does your solution require real-time processing? Is it poorly suited for mobile computing because your SMP architecture draws lots of power or does it require lots of cooling in a data center for that same reason? Identifying your own counterpoints is a key component of graduate writing. Predicting the side effects of your own technical proposals is also a valuable engineering skill. Use this section to demonstrate you truly understand your operating system recommendation from all perspectives. (3-5 Paragraphs)

**Process Management**

**Processes & Requirements**

Describe how the typical user can utilize process management tools to evaluate the features of your proposed operating system during its useful life. For example, much like you ran [tasklist /v] earlier in the course, how will TSI users determine the scheduler type and process states when they are using your proposed solution? (2-3 Paragraphs)

**Operational Processes Tools**

Describe the software tools a TSI engineer can use to determine if there is a thread or process deadlock while running your proposed operating system. Think about what the Linux command [ps -ef] or Microsoft’s Task Manager do and write about how similar functionality would operate in your proposed solution. (2-3 Paragraphs)

**Multi-Processing Support**

Discuss how your proposed solution will support concurrently executing processes and threads. For example, are the applications in your implementation multithreaded and if so, what method of synchronization (e.g., mutexes, semaphores, critical sections, etc.) will they use? (2-3 Paragraphs)

**Memory Management**

**Memory Abstraction**

See Page 2, Bullet IV, Sub-Bullet **A** for details. Remember, discuss how YOUR proposed solution will address the items listed in bold. (2-3 Paragraphs)

**Virtual Memory, Paging, and Segmentation Support**

See Page 2, Bullet IV, Sub-Bullet **B** for details. Remember, discuss how YOUR proposed solution will address the items listed in bold. (2-3 Paragraphs)

**Memory Management Techniques**

See Page 2, Bullet IV, Sub-Bullet **C** for details. Remember, discuss how YOUR proposed solution will address the items listed in bold. (2-3 Paragraphs)

**I/O and Mass Storage**

**Hardware-Software Interface**

See Page 3, Bullet V, Sub-Bullet **A** for details. Remember, discuss how YOUR proposed solution will address the items listed in bold. (2-3 Paragraphs)

**File Systems**

See Page 3, Bullet V, Sub-Bullet **B** for details. Remember, discuss how YOUR proposed solution will address the items listed in bold. (2-3 Paragraphs)

**Interrupts**

See Page 3, Bullet V, Sub-Bullet **C** for details. Remember, discuss how YOUR proposed solution will address the items listed in bold. (2-3 Paragraphs)

**Security**

**Security Model**

See Page 3, Bullet VI, Sub-Bullet **A** for details. Remember, discuss how YOUR proposed solution will address the items listed in bold. (2-3 Paragraphs)

**Security Techniques**

See Page 3, Bullet VI, Sub-Bullet **B** for details. Remember, discuss how YOUR proposed solution will address the items listed in bold. (2-3 Paragraphs)

**Conclusion**

With the considerable limitations within the TSI system, the following are my recommendations for improvements. Security and Fault Tolerance are two areas that require immediate attention, or TSI will risk the loss of data via data theft or a single point of failure from lack of fault tolerance. The next areas of improvement I would focus on are adding support for multithreading and improving virtual memory support with the operating system currently only supporting single processors, attempting to proceed with that model with becoming cost-prohibitive as the single-core processors become challenging to purchase. My final recommendation would be to implement multiprocessing and multiprogramming support, as it will cover all processor deployments (single processor, dual processor, multi-core) and expand deployment potential.

If, at this point, you have more than 20 pages (not including title, abstract, and reference pages) you may want to trim some content. If you do not have 15 pages (not including title, abstract, and reference pages), you may want to beef up your content above. Find additional sources and expand your base of technical knowledge. You will have lots more to write about.

Multiprogramming, by definition, is when a computer runs more than one process at a time; for example, simultaneously running Chrome and Word. Regularly, computers run multiple operations at any given time. Concerning TSI, their operating system does not support various processes. TSI clients need the ability and access to numerous services. They need one program that would help their needs for input and output.

In a multiprogramming system, when one task finishes, another goes through the in and out process. Meanwhile, the operating system pauses a job and puts it in a waiting pool that gives CPU to the new tasks and starts its completion process. A background operations system must perform multitasking duties and have complex organization components. The operating system is responsible for managing all of the procedures effectively and efficiently. (Singh & Darshan, 2019)

TSI needs a system that provides services of the operating system to its user programs through an application programming interface. Kernels use the system calls as entry points to perform essential tasks. TSI needs to complete the responsibility that goes further than opening and closing files and reading and writing. A thread is the virtual interpretation of a central processing unit. It allows a single task to have numerous code segments. Multithreading is the capacity for a process to conduct multiple users at a time and manage requests from the same user without the need to open multiple instances.

Another issue TSI is experiencing is that it drops background operations. TSI needs to make multithreading changes to prevent further hang-ups. A thread that shares resources of the parent process to create new threads and swap between them is an option.

The TSI operating systems uses a flat memory model without paging. The operating system needs to swap, which is writing the entire process, not just some of it onto a disk. Without paging file the moment that the memory runs out, there is no paging to fall over to causing a fight of resources and cause a crash or not sufficient memory to complete the background process.

As previously discussed, TSI operating systems use a primary system call interface. The interface is essential because it is a way for systems to communicate. In this case, the software compatibility issues resolved by adding an arduous unnecessary process to address the problem by going around and modifying traps to kernel mode and making call responses.

TSI operating system uses sensor input operations which are not secured. Tasks such as login, modifications to the file system, or structure security are not protected. Because of this, anyone can make changes to the application.

TSI coders create their own customized drivers. TSI is currently experiencing functionality problems with their drivers having various limitations due to their size. Part of the issues within the background operations is a result of creating customized device drivers for each client. This practice is not scalable because they would have to maintain multiple drivers as opposed to using a single, standardized driver.

RAID cards are storage configuration. They require more than two drives to complete. RAID can stand hard drive failure. TSI is having problems because all their data gets stored in a single hard drive, creating a single point of failure with no redundancy.

**Case study**

I believe that the main problem TSI is experiencing is that they did not update or keep up with technological advances. They have the right product for their clients, but it lacks more elements than average startup companies. Thus they have outstanding contracts and a reputation they have extensive work before it gets to the point those legacy processors are no longer available.

In the case study referenced below called “Performance Issues for Multiprocessor Operating Systems” discusses ways shared-memory can create system software issues, and it’s importance. The case study also explains the difference between using two basic classes of multiprocessors and a larger-scale system.

**Workplace Experience**:

The most critical feature that would impact my productivity for school would be multiprogramming. The inability to have both the SNHU web-based library and the program I am using to develop my assignment (i.e., MS Word, Powerpoint, or Visio) would cause me to print out all of the references and need to retype any quotations cited. This process would both be time-consuming as well as not very eco-friendly.

\

References

Rouse, M. (2015, September 28). What is multiprogramming? - Definition from WhatIs.com. Retrieved July 11, 2020, from <https://whatis.techtarget.com/definition/multiprogramming>

Mishra, N. (2016, March 05). Difference Between Multiprogramming, Multitasking, Multiprocessing and Multithreading. Retrieved July 11, 2020, from <https://www.thecrazyprogrammer.com/2014/12/difference-between-multiprogramming-multita>

Goodacre, J. (n.d.). The Design Dilemma: Multiprocessing Using Multiprocessors and Multithreading. Retrieved July 12, 2020, from <https://www.design-reuse.com/articles/13914/the-design-dilemma-multiprocessing-using-multiprocessors-and-multithreading.html>

A. (2019, August 16). Virtual Memory in Operating System. Retrieved July 12, 2020, from https://www.geeksforgeeks.org/virtual-memory-in-operating-system/

IBM. (n.d.). Multiprogramming and multiprocessing. Retrieved July 12, 2020, from https://www.ibm.com/support/knowledgecenter/zosbasics/com.ibm.zos.zconcepts/zconcepts\_75.htm

Castro, K. (2018, September 10). What are system calls in the Operating System? Retrieved July 12, 2020, from https://www.tutorialspoint.com/what-are-system-calls-in-operating-system

Abousen, D. (2015). Computer Security. Retrieved July 12, 2020, from http://www.contrib.andrew.cmu.edu/~dabousen/Default - Copy (4).html

What is a Device Driver? (2020, April 30). Retrieved July 12, 2020, from <https://www.computerhope.com/jargon/d/driver.htm>

Rehman, J. (2017, November 22). Advantages and disadvantages of multiprogramming systems. Retrieved August 09, 2020, from https://www.itrelease.com/2017/09/advantages-disadvantages-multiprogramming-systems/

E. (2008). The Advantages of Multithreaded Applications. Retrieved August 09, 2020, from <https://www.evaluationengineering.com/home/article/13000965/the-advantages-of-multithreaded-applications>

Python - Multithreaded Programming. (2009). Retrieved August 09, 2020, from <https://www.tutorialspoint.com/python/python_multithreading.htm>

Point, T. (2020). Operating System - Virtual Memory. Retrieved August 09, 2020, from https://www.tutorialspoint.com/operating\_system/os\_virtual\_memory.htm

Datapacket, .. (2020, January 28). Advantages and Disadvantages of Various RAID Levels. Retrieved August 02, 2020, from <https://blog.datapacket.com/advantages-disadvantages-various-raid-levels/>

Mandal, S. (2019, August 16). Introduction of System Call. Retrieved August 10, 2020, from <https://www.geeksforgeeks.org/introduction-of-system-call/>

*IT 600 Final Project Milestone One Guidelines and Rubric* [PDF]. (n.d.). Manchester: Southern New Hampshire University.

Singh, H., & Darshan, L. (2019, September 16). Difference between Multiprogramming, multitasking, multithreading and multiprocessing. Retrieved July 05, 2020, from <https://www.geeksforgeeks.org/difference-between-multitasking-multithreading-and-multiprocessing/>

Gamsa, B., Krieger, O., Parsons, E., & Stumm, M. (1995). *Performance Issues for Multiprocessor Operating System* (pp. 1-16, Tech. No. CSRI-339). Toronto: Computer Systems Research Institute. doi:https://www.researchgate.net/profile/Michael\_Stumm/publication/2247884\_Performance\_Issues\_for\_Multiprocessor\_Operating\_Systems/links/02e7e53219eff59705000000/Performance-Issues-for-Multiprocessor-Operating-Systems.pdf?origin=publication\_detail

Final Checklist

a) Have you incorporated all of the feedback and suggestions you received from Dr. Flowers?

b) Does your submission address all of the critical elements in the grading rubric?

c) Does the operating system you selected match the specific requirements your organization identified?

d) Do you have title and references pages?

e) Is your submission at least 15 pages of content not including title, abstract, and reference pages?

f) Have you supported each of your positions with a credible source (in-text citations used)?

g) Have you properly cited your sources and made proper attribution for any copyrighted graphics or photos?

h) Does your submission make a clearly stated recommendation for an operating system?

i) Is your submission written in the third person?

j) Did you delete all of the blue instructional text before finalizing your submission?

k) Did you use in-text citations so that each source on your reference page has a matching instance within your text? (<https://www.youtube.com/watch?v=qzKlb7E7ERc&t=3s>)