Running head: TECHNICAL EVALUATION OF AN OPERATING SYSTEM TITLE PAGE 1

Technical Evaluation of an Operating System:

Karima Williams

December 23, 2019

Southern New Hampshire University

Karima Williams

**Description of Challenges Faced by TSI**

Top Secret Inc. creates embedded operating systems for secure terminals that control systems for Wall Street firms, drone aircraft camera systems for government contractors, and alarm systems for top-secret government installations. TSI is having trouble staying up to date with technology since the company uses its own in-house operating system. TSI is well-known for fast responses and reliable operation, TSI still needs modern technology to present to customers. For instance, TSI's operating system does not support more than one program running at a time. TSI also cannot provide CreateThread(), and pthread\_create() API calls API like Windows or Linux which can hinder the opportunity for new customers because the CreateThread function forms a new thread for a process; processes are needed to execute a program (Karl-Bridge-Microsoft, 2018). Programs cannot run if there are no processes. Lastly, TSI has hardly any security and the company’s software developers are not able to program new ideas for advancement.

**Computer Architecture**

Multiprogramming is a key feature for Fortune 500 companies that need more than one program running on one processor. The processor is running a piece of one program and then a piece of another so quickly that it seems to be in simultaneous operation (Karl-Bridge-Microsoft, 2018). Multiprogramming is the function of an operating system executing more than one program on a single processor machine (Tanenbaum & Bos, 2015). A computer running Photoshop and Chrome browser simultaneously is an example of multiprogramming. Because there is only one processor at TSI, the operating system freezes and old computer equipment is being used with single-core processors since TSI’s OS does not support multicore processors.

TSI cannot properly multithread because the company continues to buy old computer equipment to support single-core processors. TSI is using open-source software to support single-core processors and this limits the functionality of the programs the customers receive.

TSI’s virtual memory and system call interface cause OS crashes and compatibility issues. TSI must install a better security system because anyone that logs into TSI headquarters can also login to any TSI server, which is hazardous for a company that deals with Fortune 500 companies and government operations.

Device drivers are a huge concern for TSI because the company’s software developers write individualized device drivers for each customer instead of having a template for all customers. The issue with this is when the company’s software developers need to make code adjustments and repairs, each customer’s layout is different, and when those developers depart from the company it will be hard for freshly hired software developers to get familiar. Also, TSI’s system doesn't acknowledge and repair problems unless it is connected to an automated response.

**Process Management**

TSI currently uses a single processor system, which is not ideal for the type of hardware/software that TSI produces for government operations. Single processor systems perform one process at a time, and performs the next queued process after the current process is completed; this makes the OS operate at a slow speed. TSI should upgrade to a multiprocessor system because multiprocessing operating systems can control many programs simultaneously, most modern network operating systems (NOSs), such as Windows support multiprocessing (Marcraft International, 2005). A multiprocessing system applies multiple processors to any given workload, thus increasing the OS’s performance above that of a single processor. Multiple processors also have the scalability feature which tunes the server network for task performance functionality (Marcraft International, 2005). Scalability is imperative within multiprocessing system architectures. Scalability permits network administrators to tune a server network’s performance built upon the amount of needed processing nodes (Marcraft International, 2005). The NOS can operate multiprocessor systems by utilizing Symmetrical Multi-Processing (SMP). The high-traffic bandwidth for TSI’s networking servers have put outrageous stipulations on TSI’s current single processor systems, meaning these single processors cannot manage the heavy workload. The workload should be equally spread over multiple processors with SMP. The benefits of using SMP is high-speed performance and affordability.

**Analytical Organizational Profile**

|  |  |  |
| --- | --- | --- |
| **GPOS Feature** | **Profile Criteria** | **Student Analysis** |
| **Multiprogramming** | Tech Description | Multiprogramming is parallel processing on a single processor. Programs are not running simultaneously, but small increments of each program are scheduled until completed. |
| Business Requirement | Multiprogramming would benefit TSI because they utilize a single processor that processes one program one by one, which is a slow process. If a large-scale program has to run, it would stop the small-scale programs until it completes. |
| **Multiprocessing** | Tech Description | Multiprocessing deals with multiple processors that operate on two processes at once. Processes are larger than threads and need memory access finish the task. |
| Business Requirement | This would a huge improvement for TSI because the hardware would be high-speed compared the single processor system. |
| **Multithreading** | Tech Description | Multithreading is accessed within shared cache very quickly. Threads combine and form a process. Threads, processes and programs all function together; the smaller the increment is, the faster the operation and the larger an increment is, the slower the operation. Processes access core memory which takes a while, however threads are more expansive because of their RAM processing. |
| Business Requirement | TSI is taking away threading through the use of open source software since they do not have a system call interface that goes past the basic functionality. TSI cannot make multiple threads. |
| **Virtual Memory** | Tech Description | Data sheets are handed over from RAM to disk to circumvent system shutdowns. When RAM becomes surcharged, it becomes a fleeting transfer. |
| Business Requirement | TSI has poor memory and administration must frequently restart the system because of the shutdowns. TSI should have a virtual memory model to prevent system shutdowns. |
| **System Call Interface** | Tech Description | The system call interface is in the middle of the application and the kernel. Each click and command that’s initiated by the user, the system call interface interprets it for the kernel to understand and carry out the command. TSI should have individualized kernel speech for every customer while fixing issues. |
| Business Requirement | TSI can only make individualized system calls since they own a standard interface that isn't usable with other software versions. Developers have to do lots of hard work to fix and update current software, as well as bring in new customers. |
| **Security** | Tech Description | Security stops unauthorized persons from hacking into the software or having access to the software. Employees should have access to needed parts of the system or there is the possibility of a data breach made by an employee. |
| Business Requirement | TSI has barely any security with no network and file protection. Anyone can alter files which is contrasts for a company that creates software for government agencies and fortune 500 companies. |
| **Device Drivers** | Tech Description | Device drivers direct a device’s connectivity to the program. |
| Business Requirement | TSI writes its own device drivers since the company OS does not intermingle well with other device drivers. Software developers at TSI must make individualized device drivers for every customer since they have different devices and require individualized programming for normal operation. |
| **Fault Tolerance** | Tech Description | Fault tolerance deals with problems and stops the system from shutting down. If the problem is the cause of the system shutting down, a program will be created to fix the problem before the shutdown occurs. |
| Business Requirement | Poor programming skills are the cause of TSI not being able develop certain programs.TSI should hire a programmer with more expertise. |

TSI must have deadlock avoidance in order to setup their server for 24 hour operation, and so that customers can make daily purchases from the website. Multiple servers processing programs simultaneously and requesting the same threads is what causes deadlocks. For instance, two people do Occupation C and Occupation D which both request Software 3 and Software 4 (GeeksforGeeks, 2019). Occupation C uses Software 3 and requests Software 4, but Occupation D already has Software 4 and requests Software 3. Both Occupation C and Occupation D have to wait for another software program, which won’t ever be accessible. This example depicts the process in which threads are requested to complete the entire process. At times processes require identical threads that aren’t ever going to be accessible because no process has the necessary threads to execute the operation (GeeksforGeeks, 2019).

**Avoiding Deadlocks**

There are systems that stop deadlocks by starting the process only if all of the threads are accessible. For instance, Occupation C above wouldn't have taken Software 3 unless all of the needed software were accessible. Additionally, deadlock can be stopped by taking the software piece-by-piece, and place it back after a certain amount of time has passed. Such as Occupation C had 5 of 6 software applications specified for the task for 15 minutes, now the system can return them back to their origin prior to the occurrence of the deadlock (GeeksforGeeks, 2019).

Anticipating the accessibility of the threads prior to securing them within the mutex is chaotic because the precise wait time isn’t defined and is probably indefinite. For instance, two out of six threads being utilized by another process never obtain accessibility to all six simultaneously. Once threads come across a mutex they are locked. This is the function that depicts a locked mutex: (pthread\_mutex\_lock()) (Tanenbaum & Bos, 2015). When threads leave the mutex they are unlocked, this is the function that depicts an unlocked mutex: (pthread\_mutex\_unlock()) (Tanenbaum & Bos, 2015). Once a thread is locked into It is impossible for a thread that has been locked into a mutex to cause a deadlock.

Locking threads individually for a planned amount of time will prevent deadlocks because those unreleased threads will be released if the remaining aren't accessible and locked within that time frame. The thread locking process restarts. The deadlock avoidance in the aforementioned example explains, Occupation C holds Software 3 and waits for Software 4. Because Occupation D was in the process of holding Software 4 and asking for Software 3 as needed to finish their task. Then Occupation C would ask to use and hold all of their software again until all of it is accessible.

**Memory Management**

TSI has poor memory and administration must frequently restart the system because of the shutdowns. TSI should have a virtual memory model to prevent system shutdowns. Virtual and physical memory are split into pages, these pages are all equal in size (Sneath, 2010). If the pages were not all equal sizes, the system’s administration would be difficult, such as TSI’s. TSI should utilize the Windows OS for memory management. Windows’ OS memory pages are 4KB for both virtual and physical memory; for efficiency purposes, the memory can also be allocated in large pages (Sneath, 2010). Every program instruction loads a register with the contents of a location in memory, the CPU maps from a virtual address to a physical one (Rusling, n.d.). Memory management monitors all memory location on the system, controlling the transition of memory and processes between the RAM and physical memory during execution (Philips, 2019). Memory management chooses where to allocate memory. When a program is shut down, memory is allocated to other processes. When memory management is full of system errors, it causes system shutdowns. That is what has been happening with TSI’s OS. The following list is the cause of TSI’s OS memory management errors (Philips, 2019):

• Faulty RAM

• Issues with new hardware, such as a graphics card

• Faulty drivers

• Software issues, including corrupt system and operating system files

• Disk errors

To prevent these errors, TSI should make sure the system is updated to its entirety. Old system files will cause strange errors. Windows’ OS automatically displays pending updates, the system will reboot during the update. TSI should also run diagnostic system tests to check the RAM to see if the system RAM is operating properly (Philips, 2019). The Memory Diagnostic tool checks the system memory for errors and logs the scan to a text file for analysis. Another speedy fix is updating the system drivers. New software or hardware is often accompanied with new drivers, and the system isn’t familiar with these drivers (Philips, 2019). Driver faults are less common with the Windows 10 OS because it provides driver updates.

**I/O and Mass Storage**

TSI needs to constantly update OS hardware and software. There are numerous input and output (I/O) objects that come with a computer. The usual I/O objects are keyboards, mice, audio controllers, video controllers, disk drives, networking ports, etc. Device drivers connect software to the hardware devices and the operating system, I/O is imperative to the device driver writer. TSI writes its own device drivers since the company OS does not intermingle well with other device drivers. Software developers at TSI must make individualized device drivers for every customer since they have different devices and require individualized programming for normal operation. TSI needs an OS like Windows with its own device drivers. The Windows 10 kernel-mode I/O controls the communication between applications and the interfaces provided by device drivers, the messages transmitted amongst the operating system and device drivers is mainly done through I/O request packets (IRPs) (Tedhudek-Microsoft, 2018). IRPs are transmitted from the operating system to specific drivers (Tedhudek-Microsoft, 2018). FAT is the most basic file system supported by Windows NT (Microsoft, 2008). The FAT file system is recognized by the file allocation table (FAT), which is a table that lies at the "peak" of the volume. To protect the volume, there are copies made of the FAT in the event of damage. An entry is created once a file is produced in the directory and the first cluster number containing data is established (Microsoft, 2008). A major disadvantage to having a FAT table is updating the FAT table is very time consuming. If the FAT table is not frequently updated, it can cause data loss. The disk read heads must be repositioned each time the FAT table is updated, which causes the process to be time consuming (Microsoft, 2008). Another disadvantage to having a FAT table is that there is no organization to the FAT directory structure, and files are given the first open location on the drive (Microsoft, 2008). The purpose of I/O interrupt handling within the Windows OS is to increase system performance. Any thread can be intercepted by a thread with a higher priority, and any driver's interrupt service routine (ISR) can be intercepted by a routine that operates at an increased interrupt request level (IRQL) (Tedhudek-Microsoft, 2017). Kernel-mode code is an interrupt with a higher IRQL value which causes another part of kernel-mode code that has a higher IRQL to operate instantly on that processor (Tedhudek-Microsoft, 2017). The lowest IRQL level is called PASSIVE\_LEVEL, where no interrupt vectors are concealed. Threads usually run at IRQL=PASSIVE\_LEVEL, device interrupts have higher IRQL values. The kernel saves the highest IRQL values for system-critical interrupts, such as system clock or bus errors (Tedhudek-Microsoft, 2017). Within the Windows OS, all threads have a thread context, which contains information that recognizes the process that owns the thread.

**Security**

TSI has barely any security with no network and file protection. Anyone can alter files which is contrasts for a company that creates software for government agencies and fortune 500 companies. It would be best for TSI’s OS to have enterprise security features such as identity protection, threat protection, and information protection. Windows 10 OS has these security features. Microsoft Defender ATP is a security service that is available to enterprise customers that detects, investigates, and responds to advanced threats on the network (Greg-Lindsay-Microsoft, 2018). The proper Windows license is required to deploy Microsoft Defender ATP. Then, the geolocation must be configured for data storage. Afterwards, start using the service. Microsoft Defender ATP integrates other products and tools to enhance its security features such as the following (Greg-Lindsay-Microsoft, 2018):

* SIEM tools
* Create custom alerts
* Use APIs
* Build Power BI reports

Windows 10 also connects to hardware to identify palm vein prints, use iris recognition and 3D facial recognition (Branscombe, 2015). Including biometrics with the standard Windows user password provides extra protection from phishing attempts and data breaches from hacked cloud services where employees have reused work passwords. Enterprise Data Protection (EDP) is another great security option to Windows 10, which is a new feature (Branscombe, 2015). This feature protects enterprise files and uses policies that automatically store corporate content in encrypted locations without encryption needing to be turned on manually for each file (Branscombe, 2015). Every file has its own individualized container.

**Full Organizational Profile**

|  |  |  |
| --- | --- | --- |
| **GPOS Feature** | **Profile Criteria** | **Student Analysis** |
| **Multiprogramming** | Tech Description | Multiprogramming is parallel processing on a single processor. Programs are not running simultaneously, but small increments of each program are scheduled until completed. |
| Business Requirement | Multiprogramming would benefit TSI because they utilize a single processor that processes one program one by one, which is a slow process. If a large-scale program has to run, it would stop the small-scale programs until it completes. |
| Application Benefit | The benefit of multiprogramming is its performance speed is high, which allows multiple processes to be completed that same day. |
| Implementation Tasks | Implementation is making the program respond to the set sequence the programmer coded the tasks to finish. |
| **Multiprocessing** | Tech Description | Multiprocessing deals with multiple processors that operate on two processes at once. Processes are larger than threads and need memory access finish the task. |
| Business Requirement | This would a huge improvement for TSI because the hardware would be high-speed compared the single processor system. |
| Application Benefit | The benefit of multiprocessing is the performance speed is high and more efficient for a computer that has just one processor. |
| Implementation Tasks | Implementation includes an additional processor. The OS should be programmed to acclimate the additional processor. |
| **Multithreading** | Tech Description | Multithreading is accessed within shared cache very quickly. Threads combine and form a process. Threads, processes and programs all function together; the smaller the increment is, the faster the operation and the larger an increment is, the slower the operation. Processes access core memory which takes a while, however threads are more expansive because of their RAM processing. |
| Business Requirement | TSI is taking away threading through the use of open source software since they do not have a system call interface that goes past the basic functionality. TSI cannot make multiple threads. |
| Application Benefit | Multithreading is quicker than multiprocessing. Since processes are composed of threads, multithreading speeds up multiprocessing. |
| Implementation Tasks | TSI should have a system call interface that has more controls and multithreading. |
| **Virtual Memory** | Tech Description | Data sheets are handed over from RAM to disk to circumvent system shutdowns. When RAM becomes surcharged, it becomes a fleeting transfer. |
| Business Requirement | TSI has poor memory and administration must frequently restart the system because of the shutdowns. TSI should have a virtual memory model to prevent system shutdowns. |
| Application Benefit | The system won’t shutdown as much. The programs won’t crash into each other because the threads will have individualized address space. |
| Implementation Tasks | Implementation could be accomplished by providing each thread a separate address space to be recovered in cache; which allows the threads to be used simultaneously by programs. |
| **System Call Interface** | Tech Description | The system call interface is in the middle of the application and the kernel. Each click and command that’s initiated by the user, the system call interface interprets it for the kernel to understand and carry out the command. TSI should have individualized kernel speech for every customer while fixing issues. |
| Business Requirement | TSI can only make individualized system calls since they own a standard interface that isn't usable with other software versions. Developers have to do lots of hard work to fix and update current software, as well as bring in new customers. |
| Application Benefit | This would benefit TSI because they won’t have to create individualized plans for customers. TSI’s systems would easy to follow. |
| Implementation Tasks | There can be a template created for system call software operation. |
| **Security** | Tech Description | Security stops unauthorized persons from hacking into the software or having access to the software. Employees should have access to needed parts of the system or there is the possibility of a data breach made by an employee. |
| Business Requirement | TSI has barely any security with no network and file protection. Anyone can alter files which is contrasts for a company that creates software for government agencies and fortune 500 companies. |
| Application Benefit | TSI should provide better security, it will boost their customer intake. |
| Implementation Tasks | TSI could hire a security company. TSI could have certain people with certain positions to access the system. |
| **Device Drivers** | Tech Description | Device drivers direct a device’s connectivity to the program. |
| Business Requirement | TSI writes uses its own device drivers since the company OS does not intermingle well with other device drivers. Software developers at TSI must make individualized device drivers for every customer since they have different devices and require individualized programming for normal operation. |
| Application Benefit | Compatible device drivers would benefit TSI. |
| Implementation Tasks | Additional software would have to be designed to implement device driver operation on TSI’s operating system. That way they customers won’t have to have an individualized program made for them. |
| **Fault Tolerance** | Tech Description | Fault tolerance deals with problems and stops the system from shutting down. If the problem is the cause of the system shutting down, a program will be created to fix the problem before the shutdown occurs. |
| Business Requirement | Poor programming skills are the cause of TSI not being able develop certain programs.TSI should hire a programmer with more expertise. |
| Application Benefit | An expert software engineer could assist TSI in stopping the system from shutting down a lot. This would also give TSI the opportunity to market the OS to new clients. |
| Implementation Tasks | An expert software engineer would implement code that stops the system from shutting down a lot. |
| *TSI must have multiprocessing and multiprogramming on each processor to enhance their operating system. TSI will need to speak with customers about the strength of their security and virtual memory. TSI’s security system and virtual memory should be updated. TSI could hire a third-party source for security and hire expert software engineers to implement virtual memory and multiprogramming. Since TSI wants to conduct business internally, they should have more experienced staff.* | | |

The company I used work for, Bellview Electronics, only had 20 employees. Bellview Electronics also tried to save money while in competition with larger companies. For instance, Bellview Electronics could not deliver all of the packages like a large retail facility, neither did Bellview Electronics have software developers that could create individualized software, so Bellview Electronics used FedEx and an ERP system. A difficulty Bellview Electronics faced with the ERP system is that it could not process a materials order bill. When employees decided to produce an item that used several items in the inventory, each item was placed on an order and invoiced, but the items could not have been grouped together into a single bill of materials item. It was costly to upgrade to that type of system, it was not done. Bellview Electronics also had a lot of personal information from the customers such as names, addresses, credit card information and place of employment. Bellview Electronics had an IT consultant that dealt with security and backups to ensure the company information and customer information was protected and secure.

Once proper security measures are in place, TSI can fix the rest of the challenges with the company one-by-one.

References

Branscombe, M. (2015, July 30). How to get the most out of Windows 10 enterprise security fea tures. Retrieved from https://www.cio.com/article/2954912/how-to-get-the-most-out-of-windows-10-enterprise-security-features.html/.

GeeksforGeeks. (2019, July 22).Deadlock Prevention And Avoidance. Retrieved from https:// www.geeksforgeeks.org/deadlock-prevention/.

Greg-Lindsay-Microsoft. (2018, May 31). Step 5: Deploy Windows 10 Enterprise security fea tures. Retrieved from https://docs.microsoft.com/en-us/microsoft-365/enterprise/windows10-enable-security-features

Karl-Bridge-Microsoft. (2018, May 30). About processes and threads - Win32 apps. Retrieved from https://docs.microsoft.com/en-us/windows/win32/procthread/about-processes-and-threads

Marcraft International. (2005, December 22). Multiprocessing Fundamentals for the Server+ Exam. Retrieved from http://www.pearsonitcertification.com/articles/article.aspx?p=426771&seqNum=2

Microsoft. (2008, October 01). Overview of FAT, HPFS, and NTFS File Systems. Retrieved from https://support.microsoft.com/en-us/help/100108/overview-of-fat-hpfs-and-ntfs-file-systems

Phillips, G. (2019, December 03). How to Fix the Windows Stop Code Memory Management BSOD . Retrieved from https://www.makeuseof.com/tag/windows-stop-code-memory-management-bsod/.

Rusling, D. A. (n.d.). An Abstract Model of Virtual Memory. Retrieved from http://www.sci ence.unitn.it/~fiorella/guidelinux/tlk/node25.html.

Sneath, T. (2010, October 28). PDC10: Mysteries of Windows Memory Management Revealed: Part One. Retrieved from https://blogs.msdn.microsoft.com/tims/2010/10/28/pdc10-mysteries-of-windows-memory-management-revealed-part-one/.

Tanenbaum, A.S. & Bos, H. (2015). Modern Operating Systems (4th ed.). Upper Saddle River, New Jersey: Pearson Education, Inc.

Tedhudek-Microsoft. (2017, June 15). Always Preemptible and Always Interruptible. Retrieved from https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/always-preemptible-and-always-interruptible

Tedhudek-Microsoft. (2018, October 16). Windows Kernel-Mode I/O Manager. Retrieved from https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/windows-kernel-mode-i-o-manager