Module Code & Module Title

CT5052NP Network Operating System

Assessment Type

Logbook 1

Semester

2023/24 Spring/Autumn

Student Name: Pujan Jung Thapa

London Met ID: 23057035

Assignment Due Date: 7 December 2024

Assignment Submission Date: 7 December 2024

Submitted To: Mr. Prasant Adhikari

Word Count (Where Required):

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

Contents

1. Introduction

The kernel is the central part of any operating system, its algorithms have gained prominence in machine learning and signal processing also it acts as a bridge between hardware and software. These methods utilize kernels to map data into high-dimensional feature spaces, enabling complex nonlinear analyses. Support Vector Machines (SVMs) are a key example, offering theoretical elegance and practical effectiveness in various domains. The kernel distance, interpreted as an L2 distance between probability measures in a Reproducing Kernel Hilbert Space (RKHS), provides efficient solutions to data analysis problems. Its cardinal task is to ensure that system resources-maybe memory, CPU, input/output devices, file systems, to name a few-are managed and presented for maintaining a stable and consistent environment for application execution. Kernel adaptive filtering extends these concepts to online learning scenarios, addressing nonlinear adaptive algorithms in signal processing and communications. This approach offers a new design methodology for nonlinear adaptive filters, incorporating algorithms such as kernel least mean squares and kernel recursive least squares. Overall, kernel methods provide a powerful and flexible framework for tackling complex learning tasks across diverse fields, from bioinformatics to engineering. (Phillips, 2011)

An operating system kernel is made up of several blocks that should interact to achieve set goals of:

1. Process Management: It is responsible for process scheduling, process creation, process termination, and synchronization of processes. This can be used to equitably distribute CPU resources among several processes.

2. Memory Management: The operating system manages main memory and virtual memory, if the former is available. The latter provides the facility of running big applications, even when the size of the physical memory is smaller than required.

3. Device Management: Standardizes the interface through which hardware components-keyboard, disk drives, network interfaces-communicate with the operating system; at the same time, it abstracts the hardware details from the user applications.

These components provide the facilities for the kernel to manage the entire system, sitting between the hardware-software interaction and ensuring a neat, secure, and efficient computing environment.

2. Objective

The main objective of this log is to examine the various kernel types, based on two different classifications. The evolution of kernels applied in popular operation system, describe the process thought which a system boots and summarize the modern computing role that a kernels plays.

3. Types of Kernels

The five main types of kernels are:

- Monolithic Kernel
- Microkernel
- Hybrid Kernel
- Nano Kernel
- Exo Kernel

a. Monolithic Kernel

In a monolithic kernel, services such as process management, memory management, file systems, drivers, and so forth are implemented in the kernel core itself and executed in the same address space. Efficiency is very high since there is no context switching between different components. It adds hardware drivers, VFS, and protocol stacks, so everything that a microkernel does is performed there. It will reduce the inter-process communications requirements while improving the system security, but this would need careful system design and should reduce the flexibility available to the designer. Examples of monolithic kernels include UNIX and Linux. (Malallah, 2021)

b. Microkernel

Microkernel mainly consists only three things, that is built-in function, scheduling and memory management and ipc. User-space Hardware Entities, through which all functions of the operating system for any other than RTEMS RMA are assigned: via user-space hardware drivers, though at control-level kernel-controlled. Application drivers and window managers are in user space, and therefore can be coded up very quickly and swapped out for programs compiled into languages not used by the kernel-without changing the kernel. Certainly such a high IPC overhead has its negatives too. Examples: Mach and MINIX. (Malallah, 2021)

4. Popular Kernels and Their History

1. iOS (XNU Kernel)

Apple Inc. develops and operates iOS, which is a mobile operating system. Initially designed and built for the iPhone, it has expanded to provide support for Apple TV and iPads. Like several other operating systems, iOS is regularly updated with new versions, with 4.0 as the prior version and 5.1 as the most recent one. The bottom of the iPhone operating system architecture is the Main OS layer. At this same location in the architecture are included: an extra activity layer, a media layer, a cocoa-touch layer, and the core services layer specific to iOS. For all components including the scheduler, file system, and Mach kernel, memory system management and hardware drivers, the network and security frameworks, and inter-process communication, the OS core layer contains the liner for protecting the data of both the system and programs.

It has developed and maintains iOS, which is an operating system for smartphones. Although developed for the iPhone, it has been extended to support the Apple TV and the iPad. Like many other OS, iOS regularly gets its updates, the latest being iOS version 5.1 and the older 4.0 version. The main operating systems layer lies at the bottom of the iPhone operating system architecture. An additional Preoccupation layer, the media, cocoa-touch layer, and the core services layer of the iOS architecture are included. Besides all the components like the scheduler, the file system, the Mach kernel, the management system of memory, and hardware drivers, the network and protection framework, and inter-process communication, the OS core layer also includes a planner for protection of both the data of the systems and programs.(Malallah, 2021)

2. Windows (NT Kernel)

Windows NT is a 32-bit, preemptive multitasking operating system with comprehensive networking capabilities and multiple security levels. Its architecture consists of a native API, which provides system services to both user and kernel mode programs, similar to Unix system calls. The Windows NT kernel architecture remains fundamentally unchanged in version 5.0, with extensions to support new features like plug and play, job objects, and 64-bit large memory support. The system is designed with key components that interact within specific contexts, adhering to original design goals and specifications . While the Win32 API offers limited access to the operating system's full functionality, the native API allows deeper interaction with Windows NT internals, enabling developers to create advanced tools, debug user mode applications, and develop kernel mode code. .(Malallah, 2021)

3. Ubuntu (Linux Kernel)

Based on the Linux kernel, Ubuntu is one of the most widely used Linux distributions. In 1991, Linus Torvalds developed the Linux kernel. As an operating system, Ubuntu is a free, user-friendly Linux distribution designed for novices and experts alike, offering regular updates and comprehensive support. Its monolithic kernel enables it to use a single, sizable piece of code for memory, devices, file systems, and process management. The main reasons this kernel has gained popularity are that it is open source, adaptable, and supported by many developers worldwide. In addition to Ubuntu, it runs a variety of other operating systems, such as Android and powerful PCs. (Newman, 2011)

5. Boot Process

The boot process is a critical initial step in operating system functionality, enabling subsequent processes to execute . It varies across different operating systems, with Windows, Linux, and Mac exhibiting both similarities and differences in their booting paradigms. The process typically involves a boot loader that prepares the system to load the kernel, often from a remote server into memory. After the kernel is loaded, it undergoes its own bootstrapping process, initializing hardware and loading drivers . Different boot configurations can determine specific actions for initialization after device reset, as seen in Texas Instruments' digital signal processors. Recent advancements in non-volatile memory technology have led to proposals for faster booting techniques, such as using a single kernel image for every boot, potentially reducing user wait times. These developments aim to streamline the boot process and enhance overall system efficiency. (Lee, 2012)

6. Conclusion

In conclusion, this current log stressed the crucial function of the kernel as far as being an OS is concerned; that is, it is the so-called backbone of modern computing. Besides, kernels have been distinguished into types; based on the architecture, they are monolithic and microkernel and then the types based on execution mode, patching out preemptive and non-preemptive kernel. In such a case, discussions were carried on how the two kernels evolved from iOS-, Windows-based operating systems, and afterwards Ubuntu because of brief descriptions of what the word booting entails in computing. In short, a kernel happens to be among the most critical components for any OS: giving one perspective on resource management, safety, and then stability over the latter itself.

7. References

Lee, D. a. Y. W., 2012. *"Booting Linux faster.".* s.l.:IEEE International Conference on Network Infrastructure and Digital Content (2012).

Newman, B., 2011. Ubuntu: I in You and You in Me (review). In: s.l.:A Journal of Christian Spirituality, pp. 136-138.

Phillips, J. &. V., 2011. A Gentle Introduction to the Kernel Distance. In: s.l.:arXiv.org .

Malallah, H. e. a., 2021. A Comprehensive Study of Kernel. *Asian Journal of Research in Computer Science,* pp. 11-16.