Cyber Security Fundamentals

Hyperiondev

An overview of the UNIX Architecture

Welcome

Your Lecturer for this session

Liano Naidoo

Lecture – Housekeeping

❑ The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment - please engage accordingly.

❑ No question is daft or silly - **ask them!**

❑ There are Q/A sessions midway and at the end of the session, should you wish to ask any follow-up questions.

❑ You can also submit questions here: hyperiondev.com/sbc4-cs-questions

❑ For all non-academic questions, please submit a query: www.hyperiondev.com/support

❑ Report a safeguarding incident: hyperiondev.com/safeguardreporting

❑ We would love your feedback on lectures: https://hyperionde.wufoo.com/forms/zsgv4m40ui4i0g/

Previously:

- Introduced the Client-Server architecture(model).

- Defined what a protocol is in Networking

- Explored HTTP and SSH protocols and some of their characteristics.

Objectives

- Become acquainted with what the UNIX operating system is.
- Fundamentals of UNIX architecture (monolithic and microkernel) and the function of the shell.
- Describe the directory structure, inodes, and file permissions in the UNIX file system.
- UNIX processes, including process states and fundamental memory management.
- To give a general overview of UNIX input and output, including basic terminal handling, standard input, output, and error.
- NFS, DNS Networking tools.

UNIX

- Written in C, UNIX is multi user, multitasking operating system (OS) designed for flexibility and adaptability.

- Created by Bell Labs in the 1960s. It has changed over time to take on different forms like BSD and macOS.

Don’t you mean “Linux”?

- A free and open-source operating system = Linux.

- Unix is a for-profit product that is sold by numerous companies, each with a unique version typically tailored to a specific piece of hardware.

Compatibility & Security

- A UNIX has been around for < 50 years and is widely used in a variety of sectors, ranging from banking/finance, research, and education.

- Many contemporary operating systems, including macOS and Linux, are built on or have incorporated UNIX concepts.

- Understanding UNIX architecture is therefore not only necessary,but critical for those who deal with these platforms.

- UNIX is also renowned for its strong security features, which could mainly be attributed to it’s design.

- Understanding UNIX architecture will help you better grasp how to protect your networks and avoid unauthorized entry.

- Understanding file permissions and how they are applied in UNIX, for example, is critical for protecting confidential data.

Customization and Utility

- Another benefit of UNIX design is its ability to be highly customized.
- UNIX gives users a plethora of tools and utilities to help them tailor the operating system to their particular requirements.
- Understanding UNIX architecture allows one to change and improve the system's efficiency.
- Modifying kernel parameters, for example, can greatly enhance system performance.
- UNIX offers powerful computing tools and utilities that are extensively used in the creation of software(Shell, Compilers,).
- It is critical to understand the architecture when creating and maintaining software on UNIX-based platforms(macOS, Solaris, FreeBSD).
- Understanding the system call interface and how it is used in UNIX, for example, is required for creating system-level applications.

Efficiency & Performance

- Because of its speed and effectiveness, UNIX is extensively used in high-performance processing settings.

- Understanding architecture allows users to better manage their systems for optimal effectiveness.

- Understanding the memory management system and how it is handled in UNIX, for example, is required for optimizing the performance of memory-intensive apps.

Questions and Answers

Questions around Client-Server Architecture and Protocols

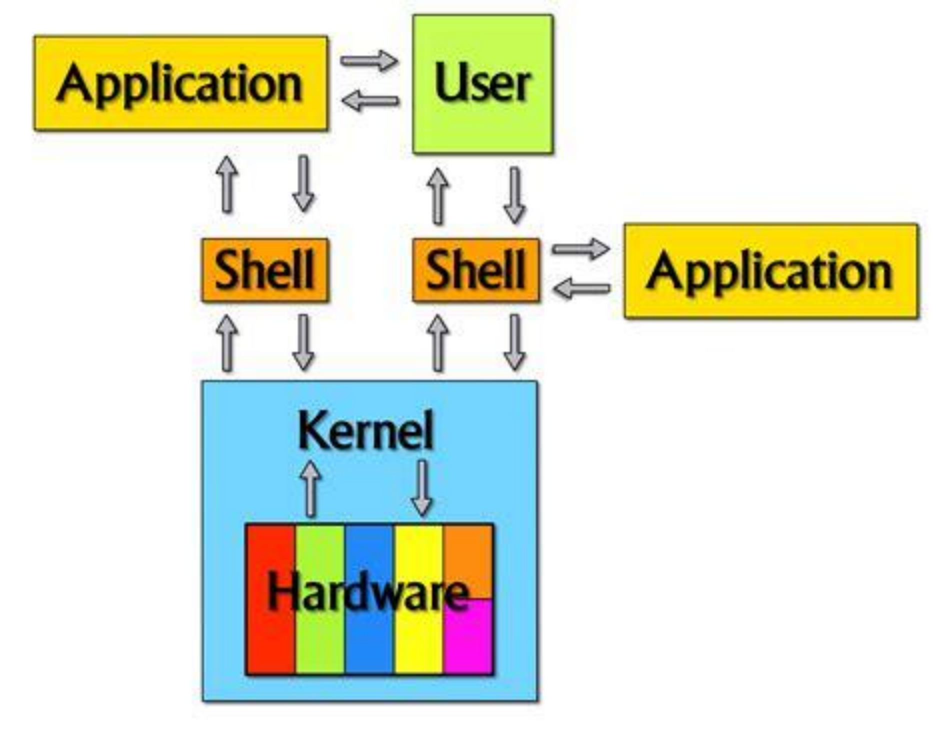
Architecture

- The general layout and design of a computer or software system is referred to as system architecture. It contains the system's different components, their connections, and how they work together to achieve the system's aims and objectives.

Kernel

- A kernel in computing is a key element of an operating system (OS) that serves as an interface between the hardware of a computer and the software programs running on it.
- The kernel is in charge of controlling system resources like the CPU, memory, and input/output (I/O) devices as well as offering services to the operating system's applications.
- When a computer starts up, the kernel is loaded into memory, and it stays there throughout the entire time the computer is in use.
- It gives operating systems and applications low-level access to the hardware, enabling them to carry out operations like reading and writing data to disks, interacting with other computers over a network, and displaying output on a screen.

Kernel


Application
User
Shell
Shell
Application
Kernel
Hardware

Monolithic Architecture

- The Monolithic Kernel Design is a standard UNIX architecture that has been used since the UNIX operating system's inception.

- The operating system kernel in this architecture is a singular, large program that offers all required operating system functions such as memory management, process management, file system management, and device drivers.

- The monolithic kernel design is a highly integrated system with great speed and economy.

- However, it has a few disadvantages, such as a absence of modularity and the possibility of crashes due to the bulk and intricacy of the kernel.

- Despite these disadvantages, the monolithic kernel design is still extensively used in UNIX-based operating systems like Linux and macOS.

MicroKernel Architecture

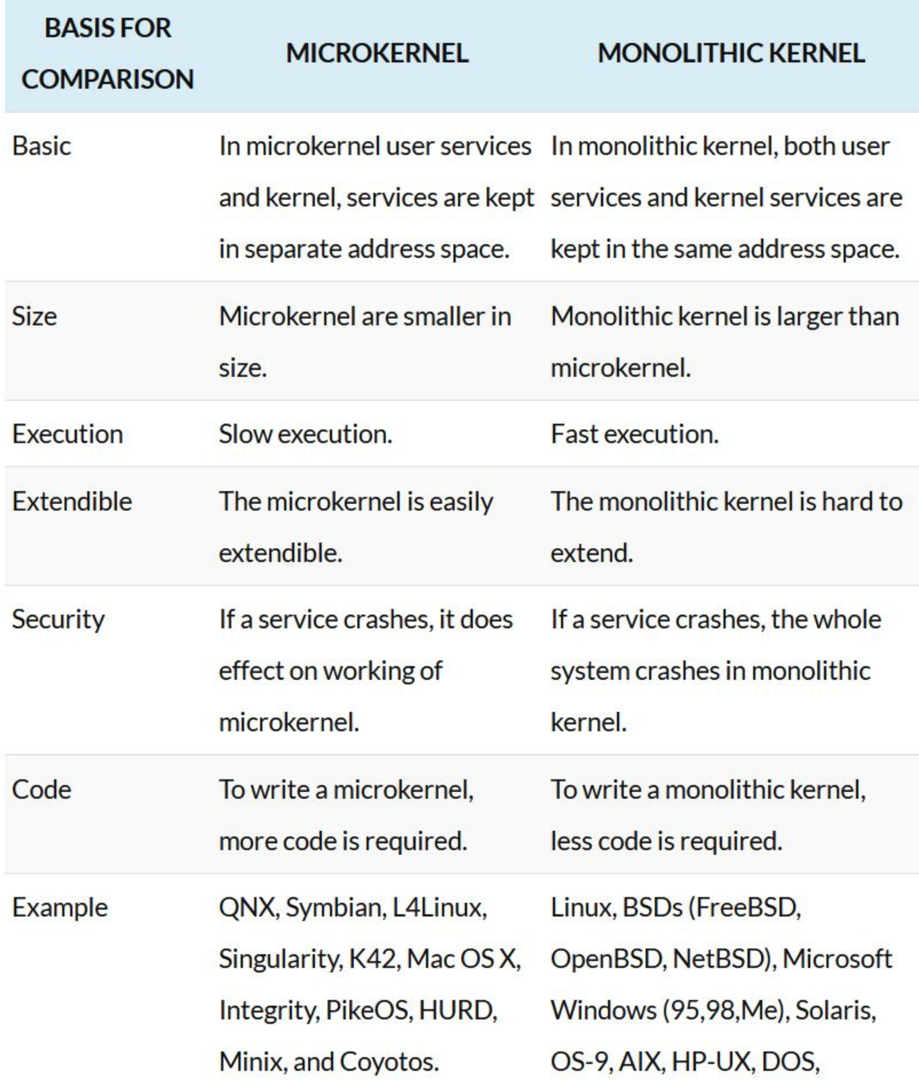
- The Microkernel Design is a newer UNIX architecture that tackles some of the shortcomings of the monolithic kernel architecture.

- The operating system kernel is split into tiny, modular components that provide only the necessary operating system functions, such as process management, memory management, and interprocess communication, in this design.

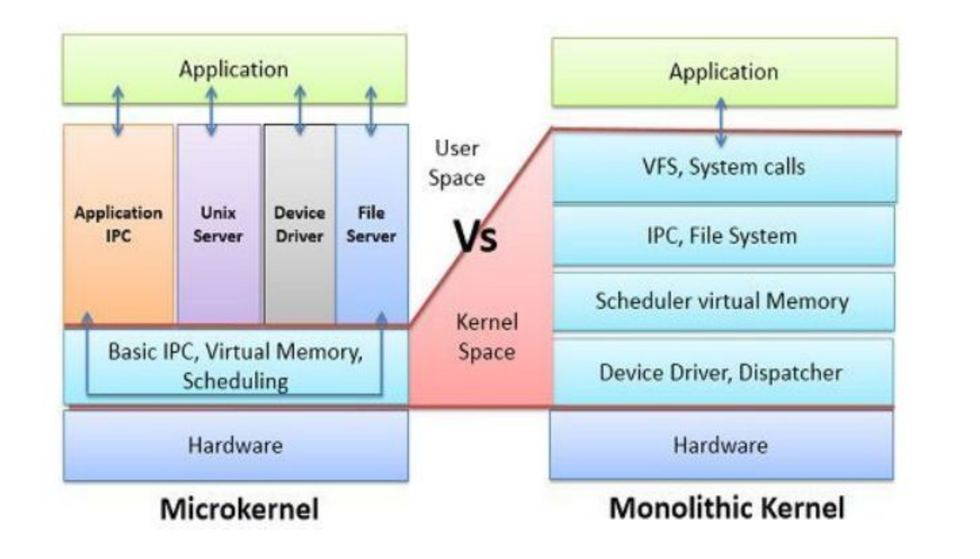
- Additional services, such as device drivers and file system administration, are implemented as distinct user-level programs that interact with the kernel via well-defined APIs.

- This division of kernel and user-level services increases modularity and freedom while decreasing the possibility of failures due to the kernel's size and complexity.

- The microkernel architecture is not commonly used in UNIX-based operating systems, but it has been implemented in some specialized systems, such as real-time and embedded systems.



BASIS FOR COMPARISON	MICROKERNEL	MONOLITHIC KERNEL
Basic	In microkernel user services and kernel, services are kept in separate address space.	In monolithic kernel, both user services and kernel services are kept in the same address space.
Size	Microkernel are smaller in size.	Monolithic kernel is larger than microkernel.
Execution	Slow execution.	Fast execution.
Extendible	The microkernel is easily extendible.	The monolithic kernel is hard to extend.
Security	If a service crashes, it does effect on working of microkernel.	If a service crashes, the whole system crashes in monolithic kernel.
Code	To write a microkernel, more code is required.	To write a monolithic kernel, less code is required.
Example	QNX, Symbian, L4Linux, Singularity, K42, Mac OS X, Integrity, PikeOS, HURD, Minix, and Coyotos.	Linux, BSDs (FreeBSD, OpenBSD, NetBSD), Microsoft Windows (95,98,Me), Solaris, OS-9, AIX, HP-UX, DOS,



https://techdifferences.com/difference-between-micro
kernel-and-monolithic-kernel.html

Question:

What does IP stand for?

NFS and DNS

- A user on a client computer can access files over a network as if they were on their local machine thanks to the Network File System (NFS) distributed file system protocol.
- In UNIX-based systems, NFS is frequently used to share files among computers connected to a network. B. DNS (Domain Name System) Domain names (like google.com) are translated into IP addresses using the Domain Name System (DNS), a hierarchical naming structure.
- The internet and many private networks use DNS to offer a simple way for users to access resources over a network. The DNS resolver is used in UNIX-based systems to look up domain names and convert them to IP addresses.

Putting it all together

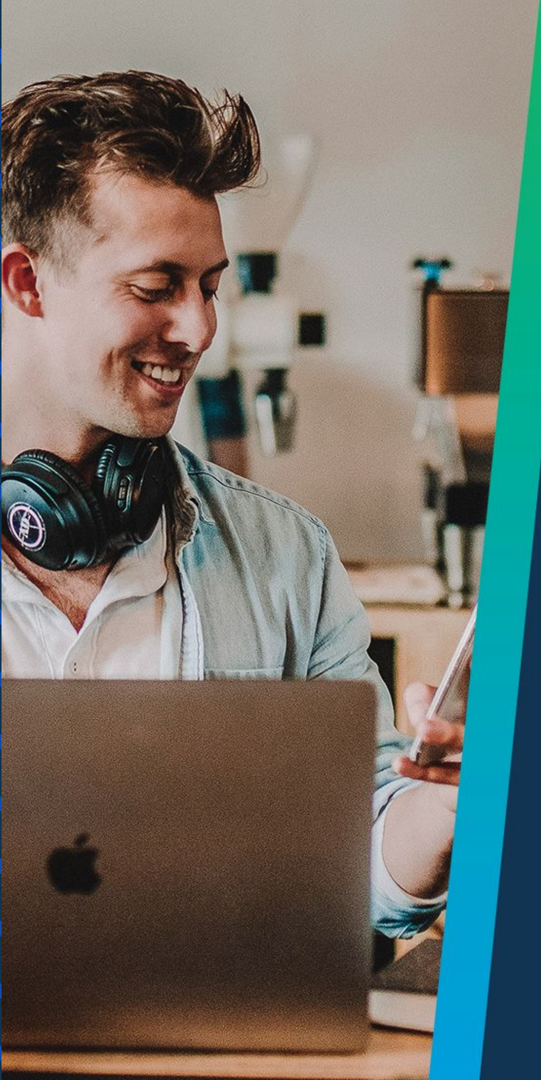
- Anyone working with UNIX-based systems or interested in learning about operating system design and implementation must be familiar with UNIX architecture.

- Understanding the fundamental concepts and workings of contemporary computing systems is made possible by the UNIX architecture.

- Understanding UNIX architecture allows one to more effectively secure their systems, modify the operating system to suit their unique requirements, and maximize the efficiency of their systems.

Questions and Answers

Questions around NFS, DNS



Hyperiondev

Thank you for joining us