BAHIRDAR UNIVERSITY

INSTITUTE OF TECHNOLOGY

FACULTY OF COMPUTING

Department of software engineering

Operating System and System Programming individual assignment

Name: Hiyab Bayeh

Id no: BDU1601787

Section-B

Submitted to: Lec Wendmu Baye

Submission date: May 12,2017 E.C

TABLE OF CONTENT

INTRODUCTION

What is an operating system?

An operating system (OS) is system software that manages computer hardware and software resources, and provides common services for computer programs.

Operating systems provide an interface that abstracts the details of accessing hardware details (such as physical memory) to make things easier for programmers.

Operating systems provide common services, such as an interface for accessing network and disk devices. This enables an application to be run on different hardware without needing to be rewritten.

INFERNO OPERATING SYSTEM

Inferno is a distributed operating system started at Bell Labs and now developed and maintained by Vita Nuova Holdings as free software under the MIT License .Inferno was based on the experience gained with Plan 9 from Bell Labs, and the further research of Bell Labs into operating systems, languages, on-the-fly compilers, graphics, security, networking and portability.

Bell Labs, the same company that contributed to the creation of UNIX, developed the intriguing but outdated operating system Inferno. It was created with the intention of bringing disparate devices together via a network.

History of development

Bell Labs, the same company that contributed to the creation of UNIX, developed the intriguing but outdated operating system Inferno in the 1990s. It was created with the intention of bringing disparate devices together via a network.

In the mid-1990s, Plan 9 development was set aside in favor of Inferno. The new system's existence was leaked by Dennis Ritchie in early 1996, after less than a year of development on the system, and publicly presented later that year as a competitor to Java. At the same time, Bell Labs' parent company AT&T licensed Java technology from Sun Microsystems.

The Inferno Business Unit closed after three years, and was sold to Vita Nuova Holdings. Vita Nuova continued development and offered commercial licenses to the complete system, and free downloads and licenses (not GPL compatible) for all of the system except the kernel and VM. They ported the software to new hardware and focused on distributed applications. Eventually, Vita Nuova released the 4th edition under more common free software licenses, and in 2021 they relicensed all editions under mainly the MIT License.

Release timeline

Date	release	comment
1996	Infero Beta	Released by Bell Labs
May 1997	Inferno Release 1.0	Winter 1997 Bell Labs Technical Journal Article
May 1997	Inferno 2nd Edition	Winter 1997 Bell Labs Technical Journal Article
June 2001	Inferno 3rd Edition	Released by Vita Nuova
2004	Inferno 4th Edition	Open Source release; changes to many interfaces (incompatible with earlier editions); includes support for 9P2000.

Understanding the fundamentals of virtual machines, system programming, and network-transparent communication is what drives the exploration of Inferno OS. This project offers a hands-on opportunity to install, test, and document an operating system that was innovative in supporting distributed computing environments as part of the system programming course.

OBJECTIVE

This project's main goal is to give students hands-on experience installing and using an operating system in a virtualized setting. Students can learn more about the basic workings of operating systems by interacting with Inferno OS, a distinctive, lightweight, and distributed operating system. This project walks through the entire process of setting up a virtual machine, creating installation files, and resolving any problems that may come up during the setup in an effort to close the gap between theoretical knowledge and practical experience.

We will investigate how different operating systems varies from contemporary mainstream systems through this practical method. We will learn about its architecture, which prioritizes communication via a single file-based system model, virtual machines, and minimalism. A useful skill in systems programming, embedded development, and legacy system maintenance, the project also promotes learning how to communicate with older or less-supported systems.

This project will also teach us how to recognize and fix issues with out-of-date installation formats, unsupported hardware, and inadequate documentation. The ultimate objective is to improve students' problem-solving skills, help them understand the variety of operating system design, and get them ready for increasingly challenging system-level programming and research assignments.

In general

- To obtain practical experience configuring an operating system in a virtual setting.
- To learn about virtual machine configuration and ISO/image-based booting.
- To understand the role of system calls in a custom OS.
- To explore the unique architecture and design concepts of OS.

REQUIREMENT

To successfully complete the installation of Inferno OS in a virtualized environment, certain hardware and software components are required.

In terms of hardware, the user needs to have access to a modern computer with a processor that is at least basic and compatible with AMD64 or x86. Although Inferno OS can technically run on extremely low-end hardware, virtualization tools like Oracle VM Virtual Box require at least 512MB of RAM.

It is advised to set aside at least 1GB of RAM for smoother performance. Furthermore, it is recommended that you have at least 2GB of free storage space because this will hold the floppy images, downloaded ISO files, and virtual machine files.

On the software side, the host operating system (Windows 10 pro) must support and be compatible with the latest version of Oracle VM VirtualBox, which is used to create and manage the virtual machine environment.

Users must obtain a community-maintained ISO or floppy disk image, which is frequently accessible through reliable archives like ArchiveOS or repositories on GitHub, as Inferno OS lacks a widely distributed official ISO file. Usually, these community images are enough to start the computer and use its functions in the virtual machine.

ISSUES (PROBLEM FACED)

I ran into a number of problems when attempting to install Inferno OS in Oracle VM VirtualBox, which stopped the operating system from booting up properly:

Boot Failure: The error message "Could not read from the boot medium!" appeared on the screen after the virtual machine was configured and the floppy disk image was attached. Kindly reboot after inserting a bootable medium. This indicates that the floppy image was either corrupted or VirtualBox did not recognize it as bootable.

Lack of official ISO image: Unlike the majority of other operating systems, Inferno OS does not offer an official ISO image. The only alternative is to use community-maintained floppy images, which are more difficult to validate or debug.

Limited Support and Documentation: Because it is an experimental and legacy operating system, there isn't much official documentation or current community support, which made resolving the boot issue more difficult.

Missing Configuration File: /mkconfig: When you ran the build command (./makemk.sh or mk), it gave an error: mk: cannot load /mkconfig: No such file or directory

SOLUTION

I verified that the floppy image was properly attached by looking at the VirtualBox settings after the boot error appeared. I tried downloading the floppy image again just in case the file was broken, but everything appeared to be in order. It didn't work even after I changed the old one.

I also began by using the Ubuntu terminal to clone the Inferno OS source code from its GitHub repository. I then tried using the supplied scripts and commands to build and install it. However, out-of-date build tools and missing configuration files caused the procedure to fail. The installation failed after attempting a number of fixes, such as installing dependencies and investigating various build techniques.

I gained knowledge about how virtual machines operate and how challenging it can be to use older systems, despite the fact that it didn't work.

Filesystem Support

Unlike other contemporary filesystems, Inferno OS employs a proprietary filesystem known as Venti, or occasionally Plan 9-like file protocols. However, the base system (like Windows 10 Pro) typically uses standard filesystems like NTFS or FAT32 to manage disk storage when operating in a virtual environment like VirtualBox.

In this instance, I did not directly interact with a filesystem like ext4 or NTFS within Inferno itself because the operating system was operating from a floppy image rather than being installed on a virtual hard drive.

I was unable to examine the Inferno filesystem since the installation failed. It is known, nevertheless, that Inferno does not directly support popular formats like NTFS or ext4. Rather, it employs its own file management system, which is primarily intended for use in embedded or networked systems.

ADVANTAGES

Because of its small size and low system resource requirements, Inferno OS is ideal for learning, experimentation, and running on outdated hardware.

It requires little space or power to run on a variety of platforms, including Windows, Linux, macOS, and virtual machines.

Inferno is appropriate for distributed computing environments because it was created with networking in mind.

Understanding modern cloud and distributed computing concepts is made easier by Inferno's ability to run programs across networks as if they were local.

System-level and application code are written in Inferno using the Limbo programming language. Because you only need to learn one language to accomplish everything, learning becomes easier.

It is more secure against crashes and malicious code because it runs code in a virtual machine (Dis VM).

Inferno has a neat and consistent design that can make it easier for students to comprehend operating system structures because everything is handled as a file (devices, processes, and resources).

Inferno is perfect for embedded systems or devices with limited resources because of its simplicity and minimalism, which result in very low memory and CPU usage.

Compared to large and complicated modern operating systems like Windows or Linux, it is easier to understand due to its simplicity and small size.

DISADVANTAGES

It has been years since Inferno OS received significant updates. There is no official team that fixes bugs or adds features on a regular basis because it lacks long-term support (LTS).

Installing Inferno OS on modern hardware or virtual environments (like VirtualBox) is extremely difficult because there are no official ISO files or updated versions.

Inferno has a much smaller user base than Linux or Windows. When you run into issues, it can be challenging to locate tutorials, documentation, or assistance.

Most users are not familiar with the commands and interface. Beginners may find it confusing, especially if they've never used a Unix-like system before.

Inferno is nearly useless for routine tasks or software testing because it doesn't support well-known applications or contemporary drivers.

Filesystems such as NTFS, FAT32, and ext4 are not supported. Rather, it makes use of its own virtual filesystem, which new users may find challenging to comprehend and utilize.

Modern hardware was not considered when designing Inferno. Current CPUs, GPUs, and peripherals like USB drives and printers might not be compatible with it.

Despite being an excellent learning tool, Inferno OS is no longer utilized in production or industry settings. Thus, mastering it won't directly address the demands of contemporary employment.

CONCLUSION

Despite the installation's failure, working on the Inferno OS installation in Oracle VM VirtualBox was a worthwhile educational experience. I gained knowledge about configuring system settings, setting up a virtual machine, and trying to boot a floppy image through this project. In the real world, I also had to deal with issues like out-of-date tools, incompatible file formats, and the lack of official ISO files.

I obtained practical experience in handling dependencies, working in the Linux terminal, utilising Git, and resolving build issues. In addition to honing my problem-solving skills, this challenge taught me how to adjust to technical challenges, which will help me in future endeavours.

I was able to learn about the fundamentals of operating systems, how system calls might be implemented, and the challenges of working with older software technologies even though I was unable to fully install and run the Inferno operating system. My ability to use virtualization software, troubleshoot, and research technical issues has improved as a result of this experience.

In conclusion, even though Inferno OS is no longer appropriate for contemporary systems, it is still a fantastic educational resource for students wishing to investigate the ideas of virtual machines, distributed systems, and low-resource operating systems.

FURTHER RECOMMENDATION

Provide Working Resources:

My attempt to install Inferno OS in a virtual environment led me to conclude that, although the operating system has intriguing features for academic learning, particularly when it comes to comprehending distributed systems, virtual machines, and lightweight system design, it is not appropriate for the majority of contemporary learning environments without substantial support.

Offer Pre-configured Virtual Machines:

Sharing pre-configured virtual machine files (such as .ova or. vdi) that already have Inferno installed will help to streamline the process. Instead of wasting too much time on installation, this would allow students to concentrate more on learning system calls and file system structure.

More Detailed Documentation:

There are currently very few resources available for Inferno. If the community or organizations kept up-to-date FAQs, tutorials, and guides to assist with installation, basic usage, and system programming in Inferno, future students would gain a great deal.

Consider Modern Alternatives:

Working with modern operating systems like Ubuntu, Arch Linux, or even lightweight Linux distributions like Tiny Core Linux may be more efficient for real-world application development and practical system programming. In addition to being actively maintained, these systems are simpler to set up and operate for practical experiments.

VIRTUALAIZATION IN MODERN WORLD

Virtualization is a technology that allows a single physical computer (or server) to run multiple virtual machines (VMs)

Despite sharing the same physical hardware, each virtual machine (VM) functions as a separate, real computer with its own files, apps, and operating system (OS).

In computing, virtualization is a series of technologies that allows dividing of physical computing resources into a series of virtual machines, operating systems, processes or containers. Virtualization began in the 1960s with IBM CP/CMS. The control program CP provided each user with a simulated stand-alone System/360 computer.

Why is Virtualization Used?

Virtualization has become a crucial component of computer use and management in today's digital world. Its ability to optimize hardware resources is one of the primary factors contributing to its widespread popularity. A physical computer typically doesn't use all of its memory or processing power at once. With virtualization, we can divide that one machine into several smaller, virtual computers—each capable of running its own operating system and applications. Because they no longer need to purchase numerous physical machines to perform various tasks, businesses and even individual users can save money.

Flexibility is also another significant benefit. For instance, developers and students can quickly establish various virtual environments to test software or experiment with new operating systems without compromising their primary machine. This implies that the other virtual machines are unaffected if something goes wrong in one. Because each virtual machine operates independently, the system as a whole is more stable and secure.

How Does Virtualization Work?

The software layer that stands between the hardware and the virtual machines, known as a hypervisor (such as VMware or Oracle VirtualBox), is how virtualization operates. Each virtual machine (VM) is assigned hardware resources (CPU, memory, and storage) by the hypervisor, which also maintains their isolation from one another.