JYOTI NIVAS COLLEGE

POST GRADUATE CENTRE

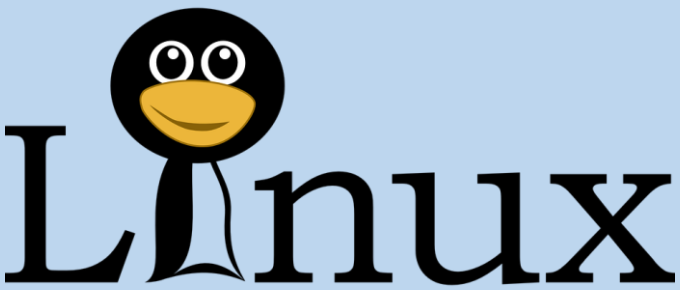
DEPARTMENT OF MCA

TECH-ON-TAP

E- JOURNAL ON

OPERATING SYSTEMS - LINUX

ISSUE:7 **MARCH 2022**

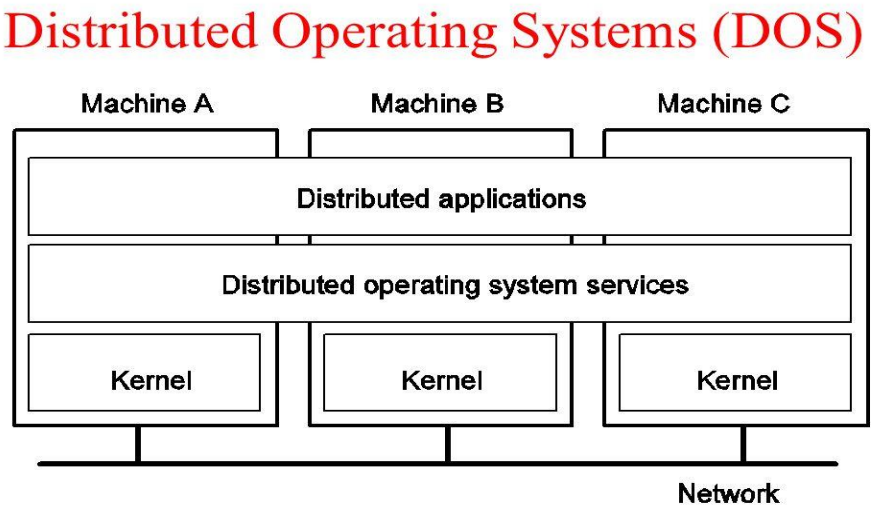


Distributed Operating System(DOS)

Vandana.R
21MCA44

Distributed operating system is based on autonomous but interconnected computers communicating with each other via communication lines or a shared network.

Each autonomous system has its own processor that may differ in size and function. A distributed operating system serves multiple applications and multiple users in real-time. The data processing function is then distributed across the processor. Distributed Operating systems are used for tasks such as telecommunication networks, airline reservation controls and peer-to-peer networks.



Advantages of distributed operating system include:

- They allow a faster exchange of data among users.
- They allow remote working.
- They reduce delays in data processing.
- Failure in one site may not cause much disruption to the system.
- They minimize the load on the host computer.
- They enhance scalability since more systems can be added to the network.

Disadvantages of distributed operating system include:

- If the primary network fails, the entire system shuts down.

- Distributed systems are very expensive to install.
- They require a high level of expertise to maintain.

References:

- https://en.m.wikipedia.org/wiki/Distributed_operating_system#:~:text=A%20distributed%20operating%20system%20is,the%20global%20aggregate%20operating%20system

Fuchsia OS

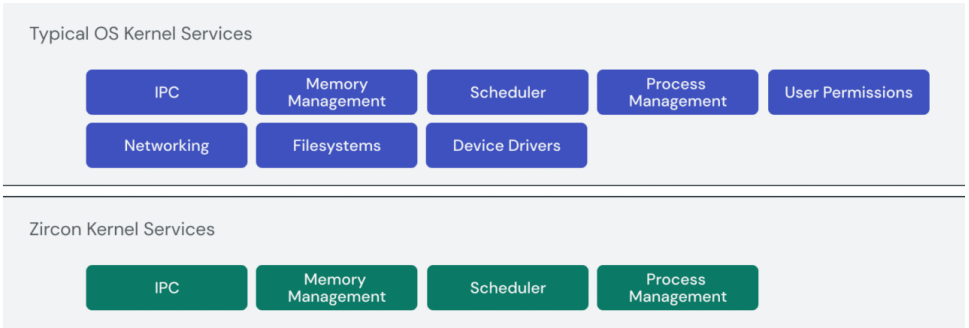
Dipika
21MCA14

Google's unique open-source operating system is known as Fuchsia. Google started developing fuchsia as an experiment back in 2016. Many aspects of fuchsia operating systems are inspired by the web and how a web works in general.

Fuchsia is not a Linux based OS.

This means fuchsia doesn't use Linux as its kernel. Instead, it uses a new kernel (specifically, a microkernel) called **Zircon.** Fuchsia's microkernel architecture helps to reduce the amount of trusted code running in the system.

Comparison between general OS kernel services and Zircon kernel services.



Fuchsia is based on four core principles:

1. **Secure:** All software that runs on Fuchsia receives the least privilege it needs to perform its job, and gain access only to information it needs to know.
2. **Updatable:** Much like the web, software on fuchsia is designed to come and go as needed, and security patches can he passed to all products on demand.
3. **Inclusive:** Fuchsia is an open-source object that currently supports a variety of languages and runtime, including C++, web, Go, Flutter and Dust.
4. **Pragmatic:** Fuchsia is not a science experiment, it's a production grade- operating system that must adhere to fundamentals like, performance.

References: https://blog.codemagic.io/fuchsia-os-preview/

https://en.wikipedia.org/wiki/Fuchsia_(operating_system)

KUBERNETES

Prerana D Saligram

(21MCA29)

Kubernetes (also known as k8s or "kube") is an open source container orchestration platform that automates many of the manual processes involved in deploying, managing, and scaling containerized applications.

You can cluster together groups of hosts running Linux containers, and Kubernetes helps you easily and efficiently manage those clusters.

Kubernetes clusters can span hosts across on-premise, public, private, or hybrid clouds. For this reason, Kubernetes is an ideal platform for hosting cloud-native applications that require rapid scaling, like real-time data streaming through Apache Kafka.

Kubernetes was originally developed and designed by engineers at Google. Google was one of the early contributors to Linux container technology and has talked publicly about how everything at Google runs in containers. (This is the technology behind Google's cloud services.)

Google generates more than 2 billion container deployments a week, all powered by its internal platform, Borg. Borg was the predecessor to Kubernetes, and the lessons learned from developing Borg over the years became the primary influence behind much of Kubernetes technology.

The primary advantage of using Kubernetes in your environment, especially if you are optimizing app dev for the cloud, is that it gives you the platform to schedule and run containers on clusters of physical or virtual machines (VMs).

More broadly, it helps you fully implement and rely on a container-based infrastructure in production environments. And because Kubernetes is all about

automation of operational tasks, you can do many of the same things other application platforms or management systems let you do—but for your containers.

Developers can also create cloud-native apps with Kubernetes as a runtime platform by using Kubernetes patterns. Patterns are the tools a Kubernetes developer needs to build container-based applications and services.

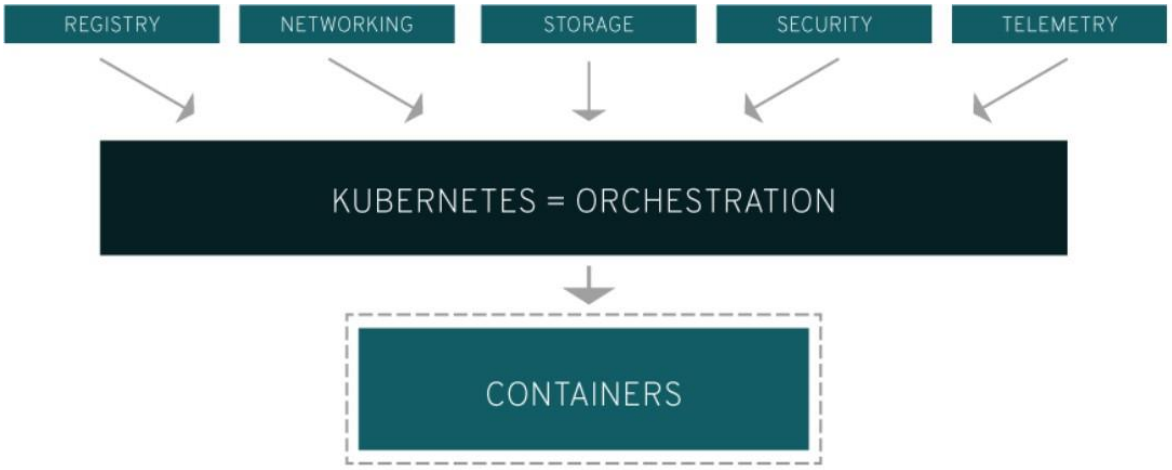
With Kubernetes you can:

- Orchestrate containers across multiple hosts.
- Make better use of hardware to maximize resources needed to run your enterprise apps.

- Control and automate application deployments and updates.
- Mount and add storage to run stateful apps.
- Scale containerized applications and their resources on the fly.
- Declaratively manage services, which guarantees the deployed applications are always running the way you intended them to run.
- Health-check and self-heal your apps with autoplacement, autorestart, autoreplication, and autoscaling.

However, Kubernetes relies on other projects to fully provide these orchestrated services. With the addition of other open source projects, you can fully realize the power of Kubernetes. These necessary pieces include (among others):

- Registry, through projects like Docker Registry.
- Networking, through projects like OpenvSwitch and intelligent edge routing.
- Telemetry, through projects such as Kibana, Hawkular, and Elastic.
- Security, through projects like LDAP, SELinux, RBAC, and OAUTH with multitenancy layers.
- Automation, with the addition of Ansible playbooks for installation and cluster life cycle management.
- Services, through a rich catalog of popular app patterns.



Learn to speak Kubernetes

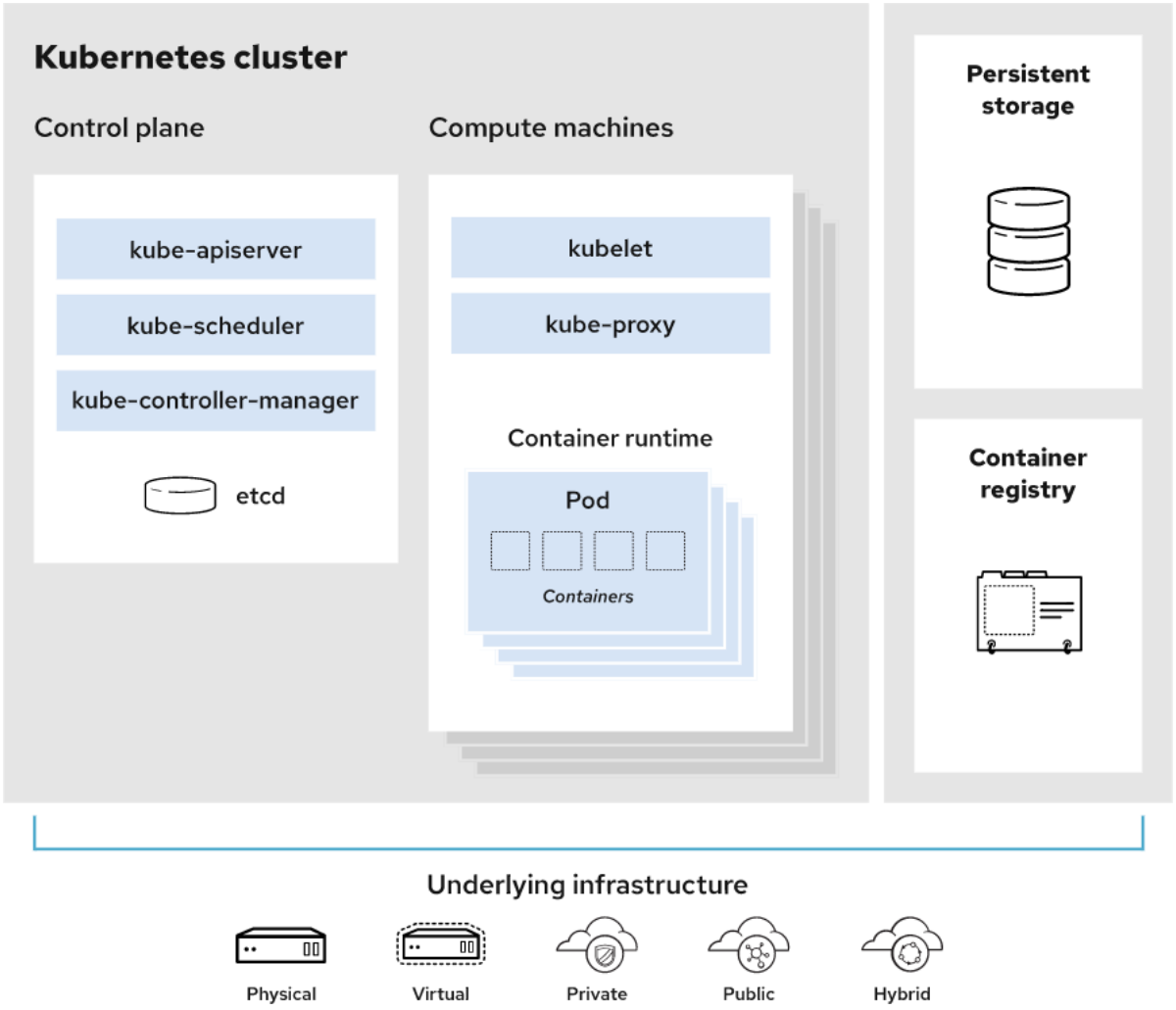
As is the case with most technologies, language specific to Kubernetes can act as a barrier to entry. Let's break down some of the more common terms to help you better understand Kubernetes.

- **Control plane:** The collection of processes that control Kubernetes nodes. This is where all task assignments originate.
- **Nodes:** These machines perform the requested tasks assigned by the control plane.
- **Pod:** A group of one or more containers deployed to a single node. All containers in a pod share an IP address, IPC, hostname, and other

resources. Pods abstract network and storage from the underlying container. This lets you move containers around the cluster more easily.

- **Replication controller:** This controls how many identical copies of a pod should be running somewhere on the cluster.
- **Service:** This decouples work definitions from the pods. Kubernetes service proxies automatically get service requests to the right pod—no matter where it moves in the cluster or even if it's been replaced.
- **Kubelet:** This service runs on nodes, reads the container manifests, and ensures the defined containers are started and running.
- **kubectl:** The command line configuration tool for Kubernetes.

How does Kubernetes work?



A working Kubernetes deployment is called a cluster. You can visualize a Kubernetes cluster as two parts: the control plane and the compute machines, or nodes.

Each node is its own Linux environment, and could be either a physical or virtual machine. Each node runs pods, which are made up of containers.

The control plane is responsible for maintaining the desired state of the cluster, such as which applications are running and which container images they use. Compute machines actually run the applications and workloads.

Kubernetes runs on top of an operating system and interacts with pods of containers running on the nodes.

The Kubernetes control plane takes the commands from an administrator (or DevOps team) and relays those instructions to the compute machines.

This handoff works with a multitude of services to automatically decide which node is best suited for the task. It then allocates resources and assigns the pods in that node to fulfill the requested work.

The desired state of a Kubernetes cluster defines which applications or other workloads should be running, along with which images they use, which resources should be made available to them, and other such configuration details.

From an infrastructure point of view, there is little change to how you manage containers. Your control over containers just happens at a higher level, giving you better control without the need to micromanage each separate container or node.

Your work involves configuring Kubernetes and defining nodes, pods, and the containers within them. Kubernetes handles orchestrating the containers.

Where you run Kubernetes is up to you. This can be on bare metal servers, virtual machines, public cloud providers, private clouds, and hybrid cloud environments. One of Kubernetes' key advantages is it works on many different kinds of infrastructure.

What about Docker?
Docker can be used as a container runtime that Kubernetes orchestrates. When Kubernetes schedules a pod to a node, the kubelet on that node will instruct Docker to launch the specified containers.

The kubelet then continuously collects the status of those containers from Docker and aggregates that information in the control plane. Docker pulls containers onto that node and starts and stops those containers.

The difference when using Kubernetes with Docker is that an automated system asks Docker to do those things instead of the admin doing so manually on all nodes for all containers.

References: www.redhat.com

Memory Resource Management in VMware ESX Server

Devika K

21MCA13

Introduction

VMware ESX Server is a thin software layer designed to multiplex hardware resources efficiently amongvirtual machines. The current system virtualizes the Intel IA-32 architecture. A ballooning technique reclaims the pages considered least valuable by the operating system running in a virtual machine.

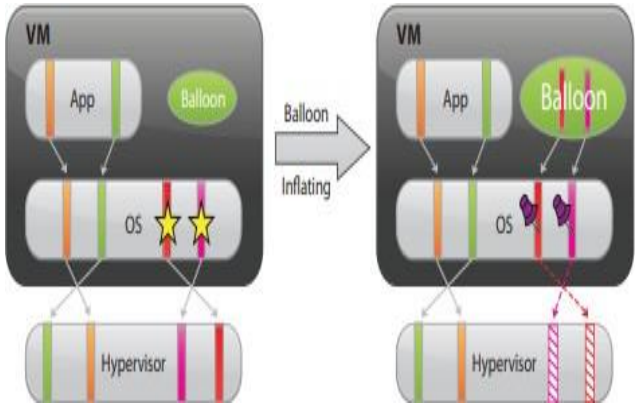
Terminology

- Host physical memory: refers to the memory that is visible to the hypervisor as available on the system.
- Guest physical memory: refers to the memory that is visible to the guest operating system running inthe virtual machine.
- Guest virtual memory: refers to a continuous virtual address space presented by the guest operating system to applications. It is the memory that is visible to the applications running inside the virtual machine.
- Guest Operating System: refers to a software installed on either a virtual machine (VM) or partitioned disk that describes an operating system that is different than the host operating system.

Memory Ballooning

In ESX, a balloon driver is loaded into the guest operating system as a pseudo-device driver. It has no external interfaces to the guest operating system and communicates with the hypervisor through a privatechannel. The balloon driver polls the hypervisor to obtain a target balloon size. If the hypervisor needs to reclaim virtual machine memory, it sets a proper target balloon size for the balloon driver, making it "inflate" by allocating guest physical pages within the virtual machine. Figure illustrates the process ofthe balloon inflating. Four guest physical pages are mapped in the host physical memory. Two of the pages are used by the guest application and the other two pages (marked by stars) are in the guest operating system free list. Note that since the hypervisor cannot identify the two pages in the guest freelist, it cannot reclaim the host physical pages that are backing them. Assuming the hypervisor needs to reclaim two pages from the virtual machine, it will set the target balloon size to two pages. After obtaining the target balloon size, the balloon driver allocates two guest physical pages inside the virtualmachine and pins them dashed arrows point at these pages.

The hypervisor can safely reclaim this host physical memory because neither the balloon driver nor the guest operating system relies on the contents of these pages. This means that no processes in the virtual machine will intentionally access those pages to read/write any values. Thus, the hypervisor does not need to allocate host physical memory to store the page contents. If anyof these pages are re-accessed by the virtual machine.



The hypervisor will treat it as normal virtual machine memory allocation and allocate a new host physical page for the virtual machine. When the hypervisor decides to deflate the balloon — by setting a smaller target balloon size — the balloon driver deallocates the pinned guest physical memory, whichreleases it for the guest's applications.

Advantages

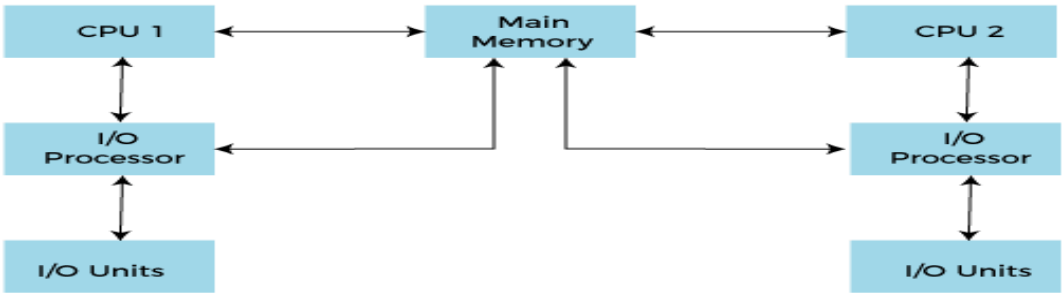
Resource optimization, Memory availability, Lower cost

References

www.vmware.com/content/dam/digitalmarketing/vmware/resource-management-white-paper.pdf

Multiprocessing Operating Systems

Lakshmi Shruthi J
21MCA23



Working of Multiprocessor System

Multiprocessing operating system is one in which two or more central processing units control the functions of the computer. Each CPU contains a copy of the OS, and these copies communicate with one another to coordinate operations. The use of multiple processors allows the computer to perform calculations faster, since tasks can be divided up between processors. Multiprocessing operating systems perform the same functions as single-processor operating systems. They schedule and monitor operations and calculations in order to complete user-initiated tasks. The difference is that multiprocessing operating systems divide the work up into various subtasks and then assign these subtasks to different central processing units. Multiprocessing uses a distinct communication architecture to accomplish this.

A multiprocessing OS needs a mechanism for the processors to interact with one another as they schedule tasks and coordinate their completion. Because multiprocessing operating systems rely on parallel processing, each processor involved in a task must be able to inform the others about how its task is progressing. This allows the work of the processors to be integrated when the calculations are done such that delays and other inefficiencies are minimized. Multiprocessing operating systems can handle tasks more quickly, as each CPU that becomes available can access the shared memory to complete the task at hand so all tasks can be completed the most efficiently.

There are two types of multiprocessing systems

1. Symmetrical multiprocessing operating system
2. Asymmetric multiprocessing operating system

. The symmetric multiprocessing operating system is also known as a shared every-thing system, because the processors share memory and the Input output bus or data path. In a Symmetrical multiprocessing system, each processor executes the same copy of the operating system, takes its own decisions, and cooperates with other processes to smooth the entire functioning of the system. The CPU scheduling policies are very simple.

In an Asymmetric Multiprocessing system, there is a master slave relationship between the processors. Asymmetric Multiprocessing system is a multiprocessor computer system where not all of the multiple interconnected central processing units (CPUs) are treated equally. In asymmetric multiprocessing, only a master processor runs the tasks of the operating system.

References:

http://The Art of Multiprocessor/ Second Edition,Maurice Herlihy.html

https://science.jrank.org/programming/Multiprocessing_Operating_Syst.html

FAULT TOLERANCE IN OPERATING SYSTEM

Shilpa V
21MCA39)

Fault tolerance is a process that enables an operating system to respond to a failure in hardware or software. This fault-tolerance definition refers to the system's ability to continue operating despite failures or malfunctions. An OS's ability to recover and tolerate faults without failing can be handled by hardware, software, or a combined solution leveraging load balancers. Some computer systems use multiple duplicate fault tolerant systems to handle faults gracefully. Even after performing the so many testing processes there is possibility of failure in system. Practically a system cannot be made entirely error free hence, systems are designed in such a way that in case of error availability and failure, system does the work properly and gives the correct result.

Depending on the fault tolerance issues that cope with, there may be different fault tolerance requirements for our system. That is because fault-tolerant software and fault-tolerant hardware solutions both offer very high levels of availability, but in different ways. Fault-tolerant servers use a minimal amount of system overhead to achieve high availability with an optimal level of performance. Fault-tolerant software may be able to run on servers we already have in place that meet system standards.

There is more than one way to create a fault-tolerant server platform and thus prevent data loss and eliminate unplanned downtime. Fault tolerance in computer architecture simply reflects the decisions administrators and engineers use to ensure a system persists even after a failure.

At the drive controller level, a redundant array of inexpensive disks (RAID) is a common fault tolerance strategy that can be implemented. Other facility level forms of fault tolerance exist, including cold, hot, warm, and mirror sites.

Fault tolerance computing also deals with outages and disasters. For this reason a fault tolerance strategy may include some uninterruptible power supply (UPS) such as a generator—some way to run independently from the grid should it fail.

REFERENCES

https://www.techopedia.com/definition/3362/fault-tolerance#:~:text=Fault%20tolerance%20is%20the%20way,or%20a%20combination%20of%20both.

https://avinetworks.com/glossary/fault-tolerance/

Operating system security

Rupa Kumari
21MCA36

Security refers to providing a protection system to computer system resources such as CPU, disk software programs and data or information stored in the computer system.

The goal of OS security is to protect the OS from various threats, including malicious software such as worms, trojans and other viruses, misconfigurations, and remote intrusions.

1.Authentication:-Ensures that user who is running the program is authentic.

OS identifies authenticate user using three ways

- .**Username / Password** − User need to enter a registered username and password with Operating system to login into the system.
- **User card/key** − User need to punch card in card slot, or enter key generated by key generator in option provided by operating system to login into the system.
- **User attribute - fingerprint/ eye retina pattern/ signature** − User need to pass his/her attribute via designated input device used by operating system to login into the system.

2.One time password:- one-time password offer an additional layer of security when combined with standard authentication measures. users must enter a unique password generated each time they log in to the system .A one-time password cannot be reused.

3.virtualization:-virtualization enables you to abstract software from hardware, effectively

. effectively separating the two. The main advantage of virtualization is that it introduces a high level of efficiency and flexibility, while providing greater security coverage. There are many types of virtualization, including desktop, application, network, server, network, storage, and OS virtualization.

There are several type of virtualization.

Fully locked- down VM:-should be used to provide access to sensitive data and corporate system, such as IT environments, payment systems, and sensitive customer.

Unlocked, open VM:-should be used to provide unrestricted access to corporate resources.

Semi- locked- down VM:-should be used to provide access to standard corporate applications and resources, such as office documents, company email, and internal services.

4.Testing and Validating Operating System Security:-Securing an operating system any is an ongoing process that requires constant testing. Depending on the risk and priority of a system, security posture tests may take place on a monthly, weekly or daily basis.

Improving Operating System Security with Hysolate :- Hysolate is a full OS isolation solution for Windows10 or Windows 11, splitting your endpoint into a more secure corporate zone and a less secure zone for daily tasks. This means that one OS can be reserved for corporate access, with strict networking and security policies, and the other can be a more open zone for accessing untrusted websites and applications.

References :- **https://www.tutorialspoint.com/**

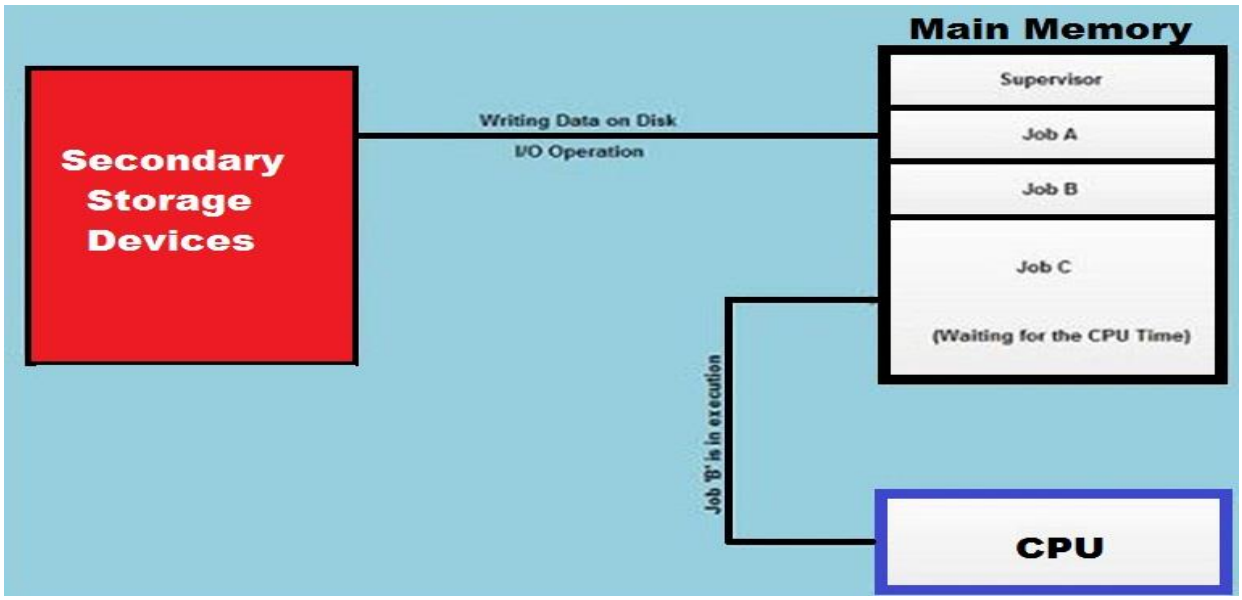
https://www.hysolate.com/

Multiprogramming Operating System

RakshithaJ
21MCA34

Multiprogramming operating system has ability to execute multiple programs with using only one processor machine. .In multiprogramming operating system, if single program gets to wait for I/O transfer, then other programs are always ready to CPU utilization. Due to this, multiple jobs can share time of its CPU. If program is in the execution process then it is known as "Process" or "Job" or "Task". The concurrent executions of programs help to improve performance of system resources utilization as well as improve system throughput than to serial and batch processing system.

One real life Example:-User can use MS-EXCEL, download apps, transfer data from one point to another point, Firefox or Google chrome browser, and more at a same time.



❖ ADVANTAGES

- Increase CPU utilization and it never gets idle.
- Resources are utilized smartly.
- Less response time.
- Short time jobs are done fastest compare to long time jobs.
- Multiple users can use multiprogramming system at once.
- It can help to execute multiple tasks in single application at same time duration.

❖ DISADVANTAGES

- CPU scheduling is needed.
- Memory management is required because all types of jobs are stored in the main memory.
- If, it contains massive load of jobs then its long time jobs have to need long waiting time.
- It is highly complex.

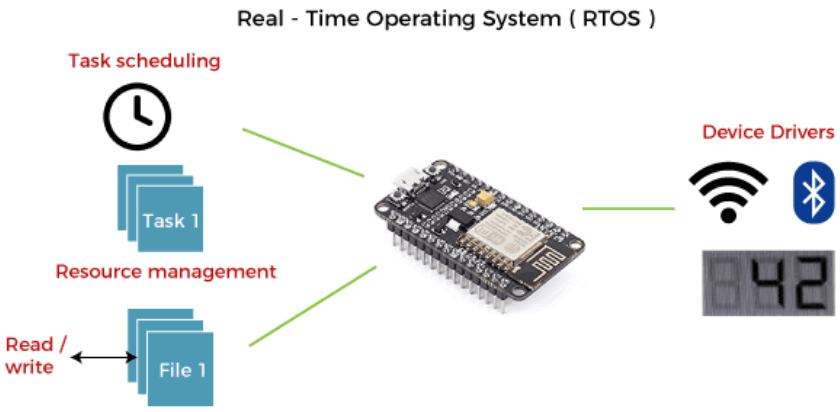
References: wikipedia

Multiprogramming Operating System (digitalthinkerhelp.com)

Real-Time operating system

Smruthi.R
21MCA40

A real-time operating system (RTOS) is a special-purpose operating system used in computers that has strict time constraints for any job to be performed. It is employed mostly in those systems in which the results of the computations are used to influence a process while it is executing. Whenever an event external to the computer occurs, it is communicated to the computer with the help of some sensor used to monitor the event. The sensor produces the signal that is interpreted by the operating system as an interrupt. On receiving an interrupt, the operating system invokes a specific process or a set of processes to serve the interrupt.



This process is completely uninterrupted unless a higher priority interrupt occurs during its execution. Therefore, there must be a strict hierarchy of priority among the interrupts. The interrupt with the highest priority must be allowed to initiate the process , while lower priority interrupts should be kept in a buffer that will be handled later. Interrupt management is important in such an operating system.

Real-time operating systems employ special-purpose operating systems because conventional operating systems do not provide such performance.

The various examples of Real-time operating systems are:

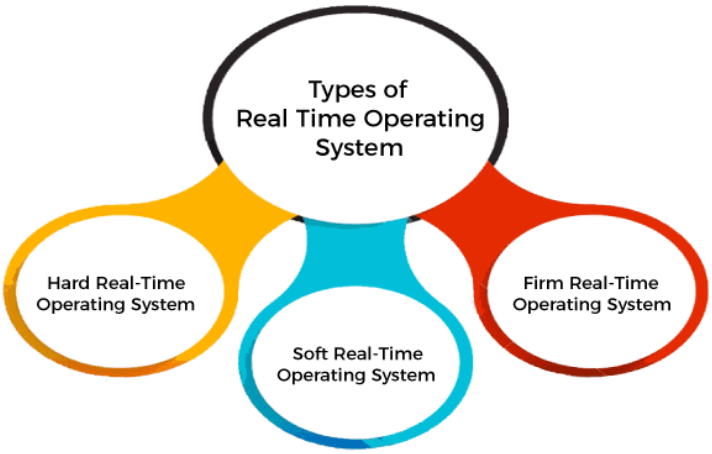
- MTS
- Lynx
- QNX
- VxWorks etc.

Applications of Real-time operating system (RTOS):

- Real-time running structures are used inside the Radar gadget.
- Real-time running structures are utilized in Missile guidance.
- Real-time running structures are utilized in on line inventory trading.
- Real-time running structures are used inside the cell phone switching gadget.
- Real-time running structures are utilized by Air site visitors to manipulate structures.
- Real-time running structures are used in Medical Imaging Systems.
- Real-time running structures are used inside the Fuel injection gadget.
- Real-time running structures are used inside the Traffic manipulate gadget.
- Real-time running structures are utilized in Autopilot travel simulators.

Types of Real-time operating system

Following are the three types of RTOS systems are:



Hard Real-Time operating system:

In Hard RTOS, all critical tasks must be completed within the specified time duration, i.e., within the given deadline. Not meeting the deadline would result in critical failures such as damage to equipment or even loss of human life.

Soft Real-Time operating system:

Soft RTOS accepts a few delays via the means of the Operating system. In this kind of RTOS, there may be a closing date assigned for a particular job, but a delay for a small amount of time is acceptable. So, cut off dates are treated softly via means of this kind of RTOS.

Firm Real-Time operating system:

In Firm RTOS additionally want to observe the deadlines. However, lacking a closing date might not have a massive effect, however may want to purposely undesired effects, like a massive discount within the fine of a product.

Advantages of Real-time operating system:

- Easy to layout, develop and execute real-time applications under the real-time operating system.
- The real-time working structures are extra compact, so those structures require much less memory space.
- In a Real-time operating system, the maximum utilization of devices and systems.
- These types of systems are error-free.
- Memory allocation is best managed in these types of systems.

Disadvantages of Real-time operating system:

- Real-time operating systems have complicated layout principles and are very costly to develop.
- Real-time operating systems are very complex and can consume critical CPU cycles.

References:

https://www.javatpoint.com/real-time-operating-system

https://www.geeksforgeeks.org/types-of-operating-systems/?ref=lbp

Multitasking/Time Sharing Operating System

Rabiya Basri
21MCA33

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

Time sharing Operating system:

Time-Sharing Operating Systems is one of the important type of operating system.

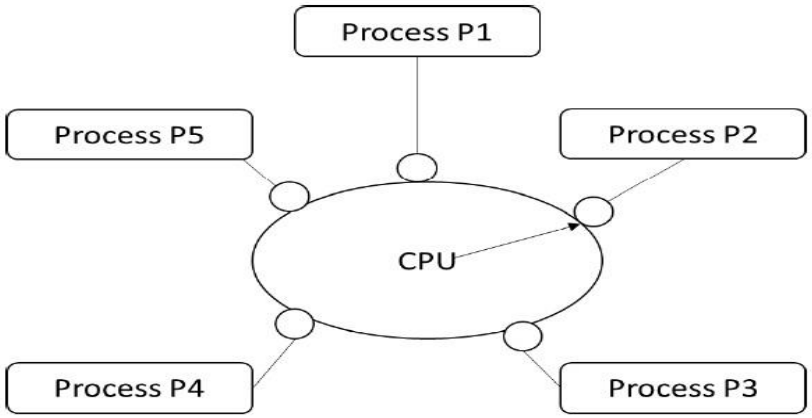
Logically, it is an extension of multiprogramming as in multi programming the user cannot interact properly with the system but in time-sharing it is possible because in this CPU executes multiple processes by switching among them and giving the CPU time to each process and in this time the user can also interact with the process.

Time-sharing OS provides short response time as the user can also directly give instruction to the operating system.

It uses or utilizes the complete CPU and not let the CPU sit idle as it uses various CPU scheduling to provide the multi programming features to the user by providing each process a short time to execute until it is not finished.

It also utilizes some more memory as each process requires swap-in and swap-out when it starts executing and stops executing.

The figure given below depicts the functioning of the time sharing operating system



Advantages:

The advantages of time sharing operating system are as follows −

- In time sharing each process gets equal opportunity to execute because equal time quantum is given to each process.
- The CPU is always busy because of maintaining time slots, there is no wastage of CPU time.
- This type of operating system avoids duplication of software.

Disadvantage:

The disadvantage of time sharing operating system is as follows –

- In the Time sharing operating system the process having higher priority does not get the chance to be executed first. This is because an equal priority is given to each process.
- Problem of data communication and security.
- It creates reliability issue.
-

Examples of Time Sharing Operating System:

For an example, In transaction processing system, all **types of processors** have ability to execute every user program in small burst or quantum of computation,, like as when n users are exist, then every user is capable to grab a time quantum.

Some Examples of Time Sharing Operating System:

- UNIX: UNIX is a general-purpose, interactive time-sharing operating system
- Linux: Linux provides time sharing environment to processes.

References:

https://www.tutorialspoint.com/time-sharing-operating-system

Android Operating System

HARI PRIYA S
21MCA18

Android is a mobile operating system based on a modified version of the Linux kernel and other open source software, designed primarily for touchscreen mobile devices such as smartphones and tablets. Android is developed by a consortium of developers known as the Open Handset Alliance and commercially sponsored by Google.

The source code has been used to develop variants of Android on a range of other electronics, such as game consoles, digital cameras, portable media players, PCs, each with a specialized user interface.

Features:

1. Interface: Android's default user interface is mainly based on direct manipulation, using touch inputs that loosely correspond to real-world actions, like swiping, tapping, pinching, and reverse pinching to manipulate on-screen objects, along with a virtual keyboard.Game controllers and full-size physical keyboards are supported via Bluetooth or USB. The response to user input is designed to be immediate and provides a fluid touch interface, often using the vibration capabilities of the device to provide haptic feedback to the user.

2. Home screen: A home screen may be made up of several pages, between which the user can swipe back and forth. Android home screens are typically made up of app icons and widgets; app icons launch the associated app, whereas widgets display live, auto-updating content, such as a weather forecast, the user's email inbox.

3. Status bar: Along the top of the screen is a status bar, showing information about the device and its connectivity.

4. Notifications: Notifications are "short, timely, and relevant information about your app when it's not in use", and when tapped, users are directed to a screen inside the app relating to the notification.

5. App lists: An "All Apps" screen lists all installed applications, with the ability for users to drag an app from the list onto the home screen.

6. Navigation buttons:
Many early Android OS smartphones were equipped with a dedicated search button for quick access to a web search engine and individual apps' internal search feature. More recent devices typically allow the former through a long press or swipe away from the home button.

7. Charging while powered off:
When connecting or disconnecting charging power and when shortly actuating the power button or home button, all while the device is powered off, a visual battery meter whose appearance varies among vendors appears on the screen, allowing the user to quickly assess the charge status of a powered-off without having to boot it up first. Some display the battery percentage.

Android has been the best-selling OS worldwide on smartphones since 2011 and on tablets since 2013. As of May 2021, it has over three billion monthly active users, the largest installed base of any operating system and as of January 2021, the Google Play Store features over 3 million apps. Android, released on October 4, 2021, is the latest version.

References:https://en.wikipedia.org/wiki/Android (operating system) #Features

ROBOT OPERATING SYSTEM

AMRITHA BALAKRISHNAN
21MCA02

Robot operating system, a framework for building robot applications, allows developers to assemble a complex system by connecting existing solutions for smaller problems.

The Robot Operating System (ROS) is not an actual operating system, but a framework and set of tools that provide functionality of an operating system on a heterogeneous computer cluster. Its usefulness is not limited to robots, but the majority of tools provided are focused on working with peripheral hardware. ROS is split up in more than 2000 packages, each package providing specialized functionality. The number of tools connected to the framework are probably its biggest power.

The key feature of ROS is the way the software is run and the way it communicates, allowing you to design complex software without knowing how certain hardware works. ROS provides a way to connect a network of processes (nodes) with a central hub. Nodes can be run on multiple devices, and they connect to that hub in various ways. The main ways of creating the network are providing requestable services, or defining publisher/subscriber connections with other nodes. Both methods communicate via specified message types. A Meta Operating system has a huge amount of functionality, so much that it cannot be classified as a framework or a cluster of libraries but not so much that it can be categorized as an operating system either.

An operating system uses software that provides an interface between the applications and the hardware. It deals with the allocation of resources such as memory, processor time etc. by using scheduling algorithms and keeps record of the authority of different users, thus providing a security layer. ROS depends on the underlying operating system. ROS demands a lot of functionality from the operating system. There is close proximity between ROS and OS, so much so that it becomes almost necessary to know more about the operating system in order to work with ROS. Using Linux as a newbie can be a challenge, one is bound to run into issues with Linux especially when working with ROS, and a good knowledge of Linux will be helpful to avert/fix these issues. Proprietary Operating Systems such as Windows 10 and Mac OS X may put certain limitations on how we can use them. This may lead to rigidity in the development process, which will not be ideal for an industry-standard like ROS. Hence, most people prefer to run ROS on Linux particularly Debian and Ubuntu since ROS has very good support with Debian based operating systems especially Ubuntu. But the support is limited and people may find themselves in a tough situation with little help from the

community. RViz is a 3D Visualization tool for ROS. It is one of the most popular tools for visualization.

ROS provides functionalities of both Operating Systems as well as frameworks but not fully hence, it cannot be classified as either e.g., it does not provide the core functionalities that an operating system is supposed to provide but provides APIs.The design allows support for any language by wrapping the C++ communication classes, or manually developing classes for the language interface. ROS is made by its own community, meant for its community. After several years, that resulted in a great number of reusable packages that are simple to integrate. ROS increases the speed of software development and helps to redistribute as it included integrated framework and toolsets for robotics development. These factors help researchers and developers to adopt robot operating system (ROS) into their robotics research and innovation.

REFERENCES:

Robot Operating System - Wikipedia

Introduction to ROS (Robot Operating System) - GeeksforGeeks

Solaris Operating System

NIVEDITHA
21MCA26

Solaris or SunOS is the name of the Sun company's Unix variant operating system that was originally developed for its family of Scalable Processor Architecture-based processors (SPARC) as well as for Intel-based processors. The UNIX workstation market had been largely dominated by this operating system during its time. As the Internet grew Sun's Solaris systems became the most widely installed servers for Web sites. Oracle purchased Sun and later renamed to Oracle Solaris

Features:

- Solaris is known for its scalability. It can handle a large workload and still delivers indisputable performance advantages for database, Web, and Java technology-based services.
- Solaris systems were known to their availability meaning that these operating systems hardly crashes at any time and because of its internet networking oriented design and broad scope of features it makes the job of adding new features or fixing any problems easy.
- It is built for network computing as it provides optimized network stack and support for advanced network computing protocols that delivers high-performance networking to most applications.

Drawbacks:

- Solaris is quite expensive since it's an enterprise operating system. Also, Solaris doesn't provide updates for free.
- Solaris lacks a good graphical user interface support and is not user friendly.
- Hardware support is not nearly as good as many other operating systems.
- Solaris sometimes becomes unstable and crashes due to total consumption of CPU and memory.

References:

https://www.geeksforgeeks.org/different-operating-systems

THE AURORA OPERATING SYSTEM

Feba Biju
21MCA16

Aurora is open-source operating system. It is aimed at large business and government organizations with a large staff of field employees. Developed on the basis of the kernel. It is the first domestic operating system for mobile devices, but someof its components are closed. This is necessary to ensure protection, since it was planned to be used for Russian civil servants from the very beginning. In 2021, Auroraeven received a positive FSB opinion on compliance with all security requirements applicable to software used in government agencies.

With recent storage hardware like NVMe SSDs and NVDIMMs, Aurora is able to continuously checkpoint entire applications with millisecond granularity. Aurora is a novel single level store that enables the persistence and manipulation of execution state.Aurora is based on the FreeBSD kernel and is the first SLS to run POSIX applications.Aurora supports all POSIX abstractions and complex multi-process applications like the Firefox web browser. Moreover, by providing new ways to interact with and manipulate application state, it enables applications to provide features that would otherwise be prohibitively difficult to implement.

Aurora is called Russian OS, but in reality, this is not entirely true. It is a fork of anotheroperating system called sailfish. Sailfish has a pretty nice interface that is visually lighter than Android.

Some of the applications for Aurora OS are Trusted boot and integrity control of the bootloader and file system, Built-in verification of software installation and launch, Built-in security policies, Full remote-control over-all smartphone functions (needed incase external control is needed), Own mobile device management (needed to protect and encrypt data), Protection of communication channels (in fact, a built-in state VPN),Multi-factor authentication (with token support), Data encryption, Working with an electronic signature.

Can Aurora replace Android? Well, objectively speaking, no. There are several reasonsfor this: Firstly, Aurora is primarily aimed at government employees, and lackssoftware; Secondly, Aurora is installed only on selected devices, and it is impossible toinstall it instead of Android; Thirdly, Aurora, whatever one may say, is still a foreign platform, albeit licensed, which means that Jolla can simply revoke the license at any time.

Aurora OS does not support Android apps. This is a feature exclusively of the originalSailfish, but its forks do not have such a privilege. Thus, Aurora users will be able to use only the software that was developed specifically for it. There are not many devicesbased on Aurora OS on the market. Some of the smartphones on Aurora OS are MIG C55, Qtech QMP-M1-N, Qtech QMP-M1-N IP, INOI R7. Tablets on Aurora OS are Aquaris Cmp NS 220R, Aquaris Cmp NS 208R, F+ Life Tab Plus, F+ R570, MIG-T8.

REFERENCES:

https://en.m.wikipedia.org/wiki/Aurora_OS#:~:text=Aurora%20(formerly%20named%20Eeebuntu)%20

https://tadviser.com/index.php/Product:Aurora_OS_(formerly_SailfishOS)

https://gamingsym.in/aurora-os-what-kind-of-operating-system-is-it-and-can-it-replace-android-in-russia/

OPERATING SYSTEM DESIGN & IMPLEMENTATION

KAVYA SHREE B
21MCA38

Design Goals:

The first problem that we face in the design goal is:

1st problem:- Defining Goals and specification

- This is a problem because; it is not very easy to specify all the goals and specifications that we need for an operating system. That is because depending on the user and system requirements that we have, there may be different sets of requirements and it is not possible to fulfill all those goals and requirements. So, that is why we call it a problem.

There are some of the requirements which we can specify.

- **Choice of Hardware**
- **Type of System**

1. Choice of Hardware:- It is the kind of hardware that we are going to choose, on which we are going to build our operating system.
2. Type of system:- It can specify the choice of hardware and the types of system that we need. Beyond this highest design level, the requirements may be much harder to specify.

Beyond this, the kind of requirements that they have, be difficult to specify or difficult to achieve all of those requirements.

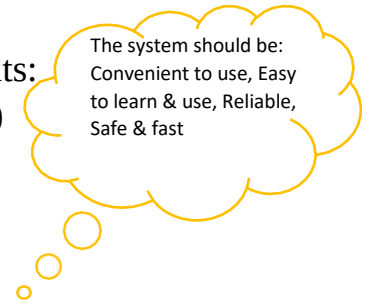
Requirements:

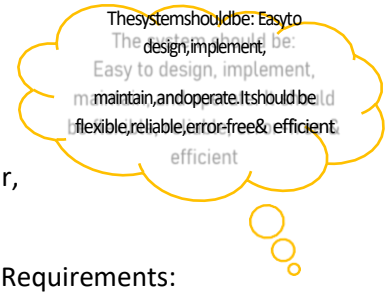
- **User Goals**
- **System Goals**

User Goals:- user goals are the requirements that are there from the user perspective or from the user side.

System Goals:- The system goals are the requirements that are there from the system or from the developer, that are developing or designing the system.

User Requirements: (User goals)





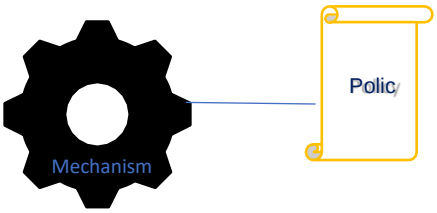
Designer,

Engineer Requirements: (System goal)

Mechanisms and Policies:

Mechanisms determine how to do something. Policies determine what will be done.

One important principle is the separation of policy from mechanism.



Good and flexible

Not good

Implementation:

- Once an operating system is designed, it must be implemented.
- Traditionally, operating systems have been written in assembly language.
- Now, however, they are most commonly written in higher-level languages, such as C or C++

Advantages of writing in high-level languages:

- The code can be written faster
- It is more compact
- It is easier to understand and debug
- It is easier to port

Conclusion:

MS-DOS was written in Intel 8088 assembly language. Consequently, it is available on only the intel family of CPUs.

The Linux operating system, in contrast, is written mostly in C and is available on a number of different CPUs, including Intel 80X86, Motorola 680X0, SPARC, and MIPS RX000.

References:

https://nesoacademy.org/cs/03-operating-system/ppts/02-os-structures

Fedora Operating System

Priyadarshini N

21MCA31

Fedora operating system is an open-source operating system that is based on the Linux OS kernel architecture. A group of developers developed the Fedora operating system under the Fedora Project. It is sponsored by Red Hat. It is designed as a secure operating system for the general-purpose. Fedora operating system offers a suite of virus protection, system tools, office productivity services, media playback, and other desktop application.

According to the Fedora Project, it is always free to use, modify, and distribute. Fedora OS is integrated with applications and packaged software. This operating system enhances the abilities of the software. It offers the same consistency, procedures, and functionality as a traditional OS. Fedora operating system is the second most commonly used distribution of Linux after Ubuntu. There are over 100 distributions based on the Fedora operating system, including the XO operating system of Red Hat Enterprise Linux.

Features of Fedora Operating System

List of the Fedora OS features:

- Fedora OS offers many architectures.
- Fedora OS is a very reliable and stable operating system.
- It provides unique security features.
- Fedora OS provides a very powerful firewall.
- Fedora OS is very easy to use.
- It supports a large community.
- Fedora OS is actively developed.
- Fedora OS is an open-source OS.
- The interface of Fedora OS is very attractive.
- This operating system offers live mode tools.
- This operating system enhances internet speed.

Fedora OS comes with many pre-installed applications and tools, such as Internet Browser, PDF and Word files Viewer, Pre-installed Games, Libre Office Suite, Programming language Support, etc.

Fedora is a very stable, secure, and light-weight operating system. It supports

different types of architectures, such as IBM Z, AMD x86-x64, Intel i686, IBM Power64le, ARM-hfp, MIPS-64el, ARM AArch64, IBM Power64, etc. Usually, it also works on the latest Linux kernel.

Fedora Server

Fedora Server is a very flexible and powerful OS. It keeps all your infrastructure and services under your control. Fedora operating system offers the latest data center technologies.

Advantages of Fedora Operating System

1. Fedora OS is a very reliable and stable operating system.
2. It enhances the security in this operating system.
3. It offers many graphical tools.
4. This operating system updates automatically.
5. This OS supports many file formats.
6. It also offers many education software.
7. It supports a large community.
8. It provides unique security features.

Disadvantages of Fedora Operating System

1. It requires a long time to set up.
2. It requires additional software tools for the server.
3. It does not provide any standard model for multi-file objects.
4. Fedora has its own server, so we can't work on another server in real-time

Reference

https://www.javatpoint.com/fedora-operating-system