**“Київський фаховий коледж зв’язку”**

**Циклова комісія Комп’ютерної та програмної інженерії**

**ЗВІТ ПО ВИКОНАННЮ**

**ЛАБОРАТОРНОЇ РОБОТИ №2**

**з дисципліни: «Операційні системи»**

**Тема: “Знайомство з інтерфейсом та можливостями ОС Linux”**

**Виконали**

**студенти**

**групи КСМ-13Б**

1. **Міньков Ілля**

**2. Колотуша Микола**

**Робота студентів групи КСМ-13б**

**Команда Defense of the Ancients: Міньков І. ,Колотуша М.**

**Київ 2023**

Мета роботи:

1. Знайомство з інтерфейсами ОС Linux.

2. Отримання практичних навиків роботи в середовищах ОС Linux та мобільної ОС – їх графічною

оболонкою, входом і виходом з системи, ознайомлення зі структурою робочого столу, вивчення

основних дій та налаштувань при роботі в системі

Матеріальне забезпечення занять

1. ЕОМ типу IBM PC.

2. ОС сімейства Windows (Windows 7).

3. Віртуальна машина – Virtual Box (Oracle).

4. Операційна система GNU/Linux – CentOS.

5. Сайт мережевої академії Cisco netacad.com та його онлайн курси по Linux

**Хід роботи.**

**Готував матеріал студент: Міньков І.**

**1. Наведіть приклади серверних додатків Linux для сервера баз даних, серверів розсилки повідомлень та**

**файлообмінників.**

Database Servers:

MySQL: An open-source relational database management system (RDBMS) that supports SQL.

PostgreSQL: Another open-source RDBMS with advanced features and geospatial support.

MongoDB: A document-oriented NoSQL database that stores data in JSON-like documents.

Message Servers:

Postfix: A popular open-source mail transfer agent (MTA) used for email server setups.

Sendmail: Another MTA that has been widely used for email delivery.

Exim: A mail server software often used on Unix-like operating systems.

File Sharing Servers:

Samba: An open-source software suite that provides seamless file and print services to SMB/CIFS clients (Windows file sharing).

NFS (Network File System): A distributed file system protocol that allows you to share files and directories among Linux/Unix systems.

OwnCloud: A self-hosted file sharing and synchronization server that allows you to create your own cloud storage solution.

These applications serve various purposes in managing data, handling message communication, and managing files on Linux servers.

**Готував матеріал студент: Міньков І.**

**2. Порівняйте оболонки Bourne, C, Bourne Again (Bash), the tcsh, Korn shell (Ksh) та zsh.**

Bourne shell (sh):

The standard shell in most Unix-like systems.

Has a limited set of features but operates quite fast.

Lacks many of the features found in more modern shells like command history and auto-completion.

C shell (csh):

Has a syntax similar to C-like programming languages.

Supports command history, auto-completion, and other user conveniences.

Typically used for interactive work but not as popular for scripting.

Bourne Again shell (Bash):

The default shell for most modern Linux systems.

Features a rich set of functions, including command history, auto-completion, job control, etc.

A powerful tool for scripting and task automation.

tcsh:

An extended version of the C shell (csh) with additional features such as an enhanced command history and command line editing.

Convenient for users accustomed to the C shell but less widely used compared to Bash.

Korn shell (Ksh):

Offers a powerful feature set, including command history, job control, advanced arithmetic, and more.

Well-suited for writing complex scripts and automating tasks.

zsh:

An advanced shell with numerous modern features like powerful auto-completion, themes, and plugins.

Convenient for interactive use and scripting.

Considered one of the most powerful shells available.

Each of these shells has its own advantages and drawbacks, and the choice of which one to use depends on the specific needs of the user. Bash and zsh are the primary shells for most Linux users due to their power and ease of use.

**Готував матеріал студент: Міньков І.**

**3. Для чого потрібен менеджер пакетів. Які менеджери пакетів ви знаєте у Linux?**

A package manager is a software tool used to install, update, remove, and manage software packages and libraries on an operating system. The primary purpose of a package manager is to streamline the management of software, ensure system stability, and provide an easy way to install and update software.

APT (Advanced Package Tool): Used in Debian, Ubuntu, and their derivatives. One of the most popular package managers.

YUM (Yellowdog Updater, Modified): Used in distributions based on Red Hat, such as CentOS and Fedora.

DNF (Dandified YUM): A newer replacement for YUM, also used in CentOS and Fedora.

Pacman: Used in Arch Linux and other Arch-based distributions.

Portage: Used in Gentoo Linux. Known for its source-based package compilation system.

ZYpp (ZENworks Package Management): Used in openSUSE and SUSE Linux Enterprise.

Snap: A package management system that provides additional isolation and portability. Used in various distributions.

Flatpak: Another package management system that offers isolation and portability. Used in various distributions.

These package managers help users efficiently manage software on their Linux systems, providing convenience and security when working with packages and updates.

**Готував матеріал студент: Міньков І.**

**4. Які засоби безпеки використовуються в Linux?**

DAC (Discretionary Access Control): Linux uses discretionary access control, allowing owners to set permissions on files and resources based on user and group ownership.

MAC (Mandatory Access Control): Some Linux distributions, such as SELinux (Security-Enhanced Linux) and AppArmor, implement mandatory access control, which enforces stricter and more flexible access controls over system resources and processes.

Firewalls: Linux offers various tools for configuring firewalls, including iptables and nftables, which allow you to control network traffic and filter packets.

SELinux (Security-Enhanced Linux): SELinux introduces an additional layer of mandatory access control, providing enhanced security by restricting program and user access to resources.

AppArmor: Another mandatory access control system that allows defining rules for applications and controlling their access to resources.

Security Auditing: Linux supports security auditing, which logs security-related events to help detect unusual or suspicious activities.

Password and Password Policy: Linux allows for strong password configuration and setting password policies to enhance user authentication security.

Digital Signatures and Encryption: Linux supports digital signatures for verifying package integrity and encryption for securing data in transit and at rest.

Updates and Patches: Regular updates and security patches are crucial for closing system vulnerabilities and protecting against attacks.

Authentication and Authorization: Linux supports various authentication methods, including passwords, SSH keys, biometric data, and more, to enhance user access security.

Intrusion Detection Systems (IDS): IDS tools like Snort and Suricata are used to detect unusual or aggressive network or server activity.

Network Security: Linux offers various network security protocols and tools, including VPNs, packet filtering, and more.

**Готував матеріал студент: Міньков І.**

**5. Чому використання віртуалізації зараз стало таким актуальним?**

The use of virtualization has become highly relevant for several reasons:

Resource Efficiency: Virtualization allows for more efficient utilization of physical server resources. A single physical server can be divided into multiple virtual machines (VMs), each with its own operating system and software stack. This optimizes resource utilization, reduces hardware costs, and saves power.

Cost Reduction: Virtualization reduces hardware costs and simplifies management. Organizations can use fewer physical servers and allocate resources as needed, leading to significant cost savings in terms of hardware procurement, maintenance, and power consumption.

Management Ease: Virtualization facilitates easy creation, movement, and duplication of VMs. This simplifies administration and backup tasks, making it easier to manage and maintain the infrastructure.

Isolation and Security: VMs can be isolated from each other, enhancing security. If one VM is compromised, others remain unaffected, reducing the risk of a security breach affecting the entire environment.

Rapid Deployment: Virtualization enables quick deployment of new VMs or infrastructure. This agility is valuable for development, testing, and scaling up services as needed.

Disaster Recovery: Virtualization makes it easier to create backups and snapshots of VMs, streamlining disaster recovery procedures. VMs can be restored quickly, minimizing downtime in case of failures.

Cloud Computing: Virtualization is a fundamental component of cloud computing platforms. Cloud providers leverage virtualization to offer scalable, on-demand resources to users.

Scalability: Virtualized environments are easily scalable. Organizations can add or remove VMs to adapt to changing workloads or demand, ensuring flexibility and performance.

**Готував матеріал студент: Міньков І.**

**6. Як ви розумієте поняття контейнеризації?**

Containerization is a methodology of virtualization that allows applications and their dependencies to be packaged, executed, and deployed in isolated containers. Containers create self-contained environments where applications can run independently of each other and the host system. Key concepts of containerization include:

Containers: A container is a standardized package that includes all necessary components to run an application, including code, runtime environment, libraries, and configuration. They are isolated from each other and the host system.

Resource Isolation: Containers are isolated from each other and have limited access to resources such as CPU, memory, and network. This ensures that programs in containers do not interfere with one another.

Portability: Containers are highly portable and can be run on any host server that supports containerization, without modifying the application code.

Resource Efficiency: Containerization allows efficient use of resources, as multiple containers can share the same host OS kernel while maintaining separate user spaces.

Ease of Deployment: Containers make it easy to create, deploy, and scale applications. They simplify administration and enable rapid deployment of new instances.

Security Isolation: Containers provide a level of security isolation, where if one container is compromised, it does not affect others, enhancing security.

Disaster Recovery: Containers can be easily backed up and restored, facilitating disaster recovery procedures and minimizing downtime in case of failures.

Orchestration: Orchestration tools like Kubernetes and Docker Swarm automate the management of multiple containers across multiple hosts.

Containerization is a valuable technology for developers and system administrators as it simplifies application deployment and management, ensures consistency, and enhances efficiency and security in both development and production environments.

**Готував матеріал студент: Міньков І.**

**7. Які переваги/недоліки використання програмного забезпечення з відкритим кодом?**

Advantages of Using Open-Source Software:

Open Access to Source Code: Open-source software provides access to the source code, allowing users to view, analyze, modify, and enhance it. This promotes transparency and security.

Community Support: Many open-source projects have active developer and user communities that offer free support, bug fixes, and improvements.

Reliability and Stability: With a large user base and developer community, open-source software tends to be reliable and stable, with quick bug identification and resolution.

Adaptability and Customization: Open-source software can be adapted and customized to meet specific organizational needs, providing flexibility.

Cost: Open-source software is often free to use, reducing software licensing costs.

Disadvantages of Using Open-Source Software:

Lack of Official Support: Open-source projects may not have guaranteed support or accountability, and support may rely on the community or third-party vendors.

Limited Features: Some open-source projects may lack certain features or functionalities compared to commercial alternatives.

Compatibility and Integration: Integration with existing systems and software can be challenging due to compatibility issues.

Maintenance Responsibility: Organizations using open-source software may be responsible for ongoing maintenance, updates, and security patches.

Legal and Licensing Issues: Understanding and complying with open-source licenses can be complex, and there may be legal risks if not managed properly.

Documentation and Training: Open-source software may lack comprehensive documentation and training materials compared to commercial alternatives.

Patent Concerns: Some open-source projects may include patented technologies, potentially leading to legal issues.