BAHIRDAR UNIVERSITY

INSTITUTE OF TECHNOLOGY

FUCULTY OF COMPUTING DEPARTMENT OF

SOFTWARE ENGINEERING

COURSE: OPERATING SYSTEM INDIVIDUAL

ASSIGNMENT

Name:Tinsae Birhanu

ID:1602608

Submitted to:Mr.Wendimu Baye

Submittion date:16/08/17 E.C

Table of Contents page no.

1. Introduction to Linux operating system

Linux is an open source, community-developed operating system (OS) for PCs, servers mainframes, and mobile devices that resembles Unix. It is among the most extensively used operating systems since it runs on practically all of the major computer platforms.

In 1991, Linus Torvalds launched Linux, an operating system kernel, as a side project. He began the project because he wanted to run an operating system based on Unix without investing a lot of money. He also wanted to become familiar with the 386 processor's features. Under the terms of the General Public License, Linux was made freely available to the public so that anybody could research and enhance it.

1.1 Key features

1. Free and Open Source

This Linux code is freely available to all, and it's a community-based development project which helps the user to modify the code or analyze the codes and redistribution of codes.

2. Multiuser Capacity

In Multiuser Capacity, Linux allows its users to share the system resources such as RAM, hard drive, and application program at same time. But they have to use it in different terminals.

3. Multitasking

It allows multiple applications to run simultaneously by effectively dividing the CPU time. Examples of multitasking are UNIX, IBM's OS/390, Windows Vista, Windows XP, and Windows 7, 8, 9, and 10.

4. Security

It secures data through Authentication, By using a password and login ID; Authorization, By using read, write and execute permissions; Encryption, It converts the files into an unreadable format.

5. **Graphical User Interface (GUI)**

It is a command line that can be converted into a Graphical User Interface by installing some packages. The most popular method for getting a GUI in a Linux system is logging into a Ubuntu server and installing its desktop environment.

6. File System

It provides a hierarchical system to arrange files and directories. The folders are classified into binary directories, configuration directories, data directories, and memory directories based on the type of files they contain.

7**. Application Support**

It consists of a software repository for users to download and install several applications, which gives the user a vast field of applications to download the application of their choice.

8. Frequent New Updates

Linux operating systems receive frequent updates, including security patches, bug fixes, and new features, ensuring ongoing improvements and staying up-to-date with evolving technology.

9. Portability

Here the term portability doesn't mean the smaller size. Here, it means that Linux supports different kinds of hardware, which means that the software can perform similarly on different types of hardware.

10. Performance

Linux operating systems are known for their performance, offering high stability, efficiency, and scalability, making them widely used in various applications.

11. Live CD/USB

Live CD/USB of a Linux operating system allows booting and running the OS from external media without installation, useful for testing, recovery, or portable computing.

12. Support's customized keyboard

Linux offers support for customized keyboard layouts, allowing users to define and use specific key mappings for various languages and preferences.

13. Compatible with cloud computing

Linux is highly compatible with cloud computing platforms, offering a wide range of distributions and tools for cloud deployment, scaling, and management. It's a popular choice for cloud servers and containers.

14. Interoperability

Linux demonstrates strong interoperability, allowing it to work seamlessly with various software, file formats, and systems, promoting open standards and collaboration across diverse environments.

15. Shell

The Shell command-line interpreter on Linux acts as a line between the user and the kernel, which runs the program known as commands. It is efficient and takes less time and space while executing the tasks.

1.2 Linux distributions

Linux distributions, often referred to as Linux distros, are operating systems made from a software collection that is based upon the Linux kernel and, often, a package management system. They include the Linux kernel, supporting utilities and libraries, and usually a large amount of application software to fulfill the distribution's intended use. It's a complete, ready-to-use OS built upon the Linux kernel, often with added features and customization

Today, there are hundreds of different Linux distributions, each aimed at a particular user or system, such as PCs, servers, mobile devices, or embedded devices. Certain distributions are packed as source code that must be compiled by the user during installation, although the majority are ready to use. Some distributions are community-developed, like Debian and Slackware, while others are commercial, including Fedora and Red Hat Enterprise Linux from Red Hat, openSUSE from SUSE, Ubuntu from Canonical, and Oracle Linux from Oracle. Red Hat and Oracle are two examples of commercial distributions that charge users for services like maintenance and custom development, even though open source license forbids charging for the open source software itself.

- From those distribution this documentation is about **Slackware Linux**

2. Introduction of Slackware Linux

Slackware Linux is one of the oldest Linux distributions still maintained today. It is known for its simplicity, minimalism, and a strong focus on UNIX-like behavior. Unlike modern distributions that automate most system processes, Slackware gives users full control of the installation and configuration process. The motivation for using Slackware is to develop a deeper understanding of the Linux operating system, practice manual system setup, and experience working with a text-based installer. distribution that he and his friends could use. This private distribution quickly gained popularity, so Volkerding decided to name it Slackware and make it publicly available. Along the way, Patrick added new things to Slackware; a user friendly installation program based on a menuing system, as well as the concept of package management, which allows users to easily add, remove, or upgrade software packages on their system and It's known for its simplicity, stability, and Unix-like philosophy, making it a popular choice for advanced users and those who want a deeper understanding of how Linux works.

2.1 Key Features Slackware Linux

- Control and Simplicity: Slackware's design emphasizes "do it yourself" and simplicity. It appeals to people who want more control over their operating system because it closely follows Unix ideals. Its simple package management system, which installs applications using the conventional "tarball" format, reflects this idea.
- Flexibility and Power: Slackware provides a balance of simplicity and power, allowing users to customize their environment extensively. It includes a modern software package set while retaining a classic Unix-style operating system environment, which can be appealing to both new and experienced users.
- Stability & Tradition: Known as one of the "Grandfathers" of the Linux ecosystem, Slackware is among the oldest actively developed Linux distributions. It has resisted the fads and trends that

frequently affect other distributions by keeping a constant structure and design over time. It is a dependable option for consumers that value a stable operating system because of its dedication to tradition and reliability.

- Minimalism: The distribution follows the "Keep It Simple, Stupid" This means that Slackware Linux does not have complex graphical tools to configure the system. As a result the learning curve of Slackware Linux can be high for inexperienced GNU/Linux users, but it provides more transparency and flexibility. Besides that you get a deeper understanding of GNU/Linux with no-frills distributions like Slackware Linux.

2.2 Historical background of slackware linix

Slackware Linux was created by Patrick Volkerding in 1993, making it the oldest Linux distribution still actively maintained. It originated as a personal project to improve upon the Softlanding Linux System (SLS), which was the most popular Linux distribution at the time. Volkerding aimed to fix bugs and streamline the installation process, leading to the release of Slackware 1.0 on July 17, 1993. Over the years, Slackware has influenced many other distributions, including early versions of SUSE Linux

Version 1.0 of Slackware was announced on the July 16, 1993, and project lead Patrick Volkerding, who still maintains the distribution today, celebrated with a modest announcement:
The size of Slackware quickly increased with the addition of included software, and by version 2.1, released October 1994, it had more than tripled to comprise seventy-three 1.44M floppy disk images.

In 1999, Slackware saw its version jump from 4 to 7. Slackware version numbers were lagging behind other distributions, and this led many users to believe it was out of date even though the bundled software versions were similar. Volkerding made the decision to bump the version as a marketing effort to show that Slackware was as up-to-date as other Linux distributions, many of which had release numbers of 6 at the time. He chose 7, estimating that most other distributions would soon be at this release number.

In April 2004, Patrick Volkerding added X.Org Server packages into the testing/ directory of -current as a replacement for the Xfree86 packages currently being used, with a request for comments on what the future of the X Window System in Slackware should be. A month later, he switched from XFree86 to X.Org Server after stating that the opinions were more than 4 to 1 in favor of using the X.org release as the default version of X. He stated the decision was primarily a technical one, as XFree86 was proving to cause compatibility problems. Slackware 10.0 was the first release with Server.

In March 2005, Patrick Volkerding announced the removal of the GNOME in the development Change Log. He stated this had been under consideration for more than four years and that there were already projects that provided a more complete version of GNOME for Slackware than what Slackware itself provided. Volkerding stated future GNOME support would rely on the community. The community responded and as of October 2016, there are several active GNOME projects for Slackware. These include Dlackware, MATE, and SlackMATE. The removal was deemed significant by some in the Linux community due to the prevalence of GNOME in many distributions.

In May 2009, Patrick Volkerding announced the public (development) release of an official x86_64 variant, called Slackware64, maintained in parallel with the distribution. Slackware64 is a pure 64-bit distribution in that it does not support running or compiling 32-bit programs, however, it was designed as "multilib-ready". Eric Hameleers, one of the core Slackware team members, maintains a multilib repository that contains the necessary packages to convert Slackware64 to multilib to enable running of 32-bit software. Hameleers started the 64-bit port as a diversion from the pain of recovering from surgery in September 2008. Volkerding tested the port in December 2008, and was impressed when he saw speed increases between 20 and 40 percent for some benchmarks compared to the 32-bit version. To minimize the extra effort of maintaining both versions in parallel, Slackware's build scripts, called SlackBuilds, were slowly transitioned to supporting either architecture, allowing for one set of sources for both versions. Slackware64 saw its first stable release with version 13.0.

Between the November 2013 release of 14.1 and June 2016, Slackware saw a 31-month gap between releases, marking the longest span in release history. During this time the development branch went without updates for 47 days. However, on April 21, 2015, Patrick Volkerding apologized on the ChangeLog for the absence of updates and stated that the development team used the time to get "some good work done." There were over 700 program changes listed on that ChangeLog entry, including many major library upgrades. In January 2016, Volkerding announced the reluctant addition of PulseAudio, primarily due to bluez dropping direct ALSA support in v5.x. while various other projects were in turn dropping support for BlueZ v4.x. Knowing some users would not be happy with the change, he stated that "Bug reports, complaints, and threats can go to me." These changes culminated in the release of Slackware 14.2 in June 2016.

The current version, Slackware 15, went into beta in 2021 and was released early last year.

Slackware 15 is very much *not* a lightweight distribution. Running a full update brought us perilously close to filling up our 16GB root partition, and it wasn't particularly snappy on the elderly Thinkpad W500 that we chose to test it on – although it was a bit quicker than the copy of Linux Mint 21.2 it was dual-booting alongside. However, all the controls and config is right there, laid out for you, and if you manually prune some things, you could trim it to size quite easily.

3. Objectives

The primary objective of this project is to gain hands-on experience with the complete lifecycle of a traditional UNIX-like installation by working directly with Slackware Linux in a virtual environment. Through manual disk partitioning, filesystem selection, and package configuration, you will develop a deep understanding of how Linux distributions are structured under the hood. A secondary goal is to strengthen system administration skills: configuring network interfaces, setting up a bootloader, and managing user accounts. By writing and executing shell scripts—such as implementing system calls in C.

4 . Requirements

Hardware Requirements:

- 486 processor.
- 64MB RAM (1GB+ suggested)
- About 5GB+ of hard disk space for a full install.
- CD or DVD drive (if not bootable, then a bootable USB flash stick or PXE server/network card)

Software Requirements:

- Slackware Linux ISO image: Download the appropriate ISO image from the Slackware website.
- Virtualization software like VMware workstation, virtual box
- Bootable medium(Optional): Create a bootable CD/DVD or USB flash drive using the ISO image.

5. Installation steps

Step 1 preparing software and hardware requirements

- Download Slackware Linux from the official website
- install vm ware workstataion
  - ✓ Create New Virtual Machine
    - Open VMware Workstation.
    - Click "Create a New Virtual Machine".
    - Choose Typical (recommended) → Next.
    - Select Installer disc image file (iso) and load your Slackware ISO.
    - OS type:

      Linux

      Version: Other Linux 5.x and later kernel (64-bit) → Next
      - Name the VM and choose storage location.

✓ Set VM Hardware

- Assign RAM (recommend 1 GB or more).
- Create a virtual hard disk (recommend 20 GB or more).
- Finish setup.

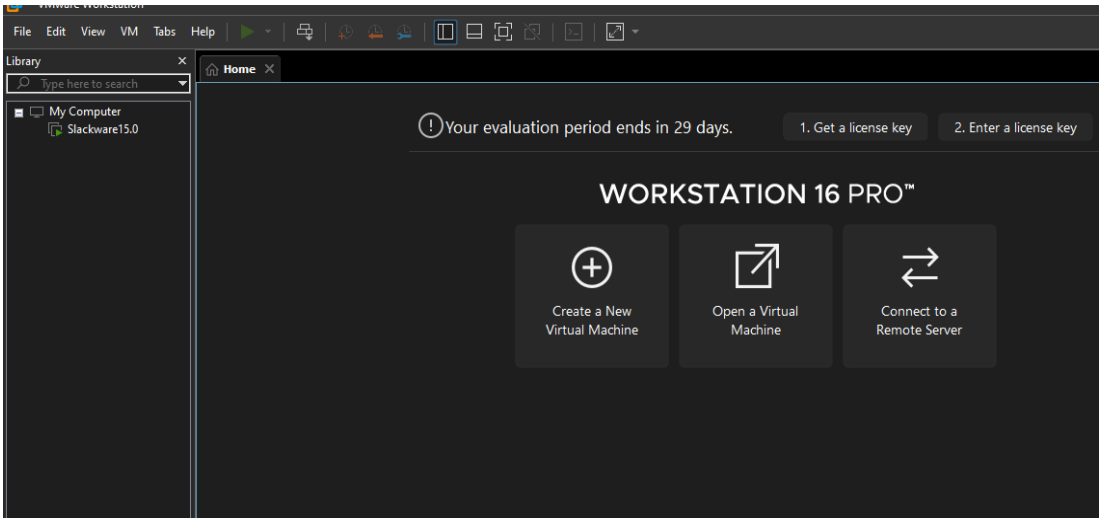


Fig1 VMware workstation interface

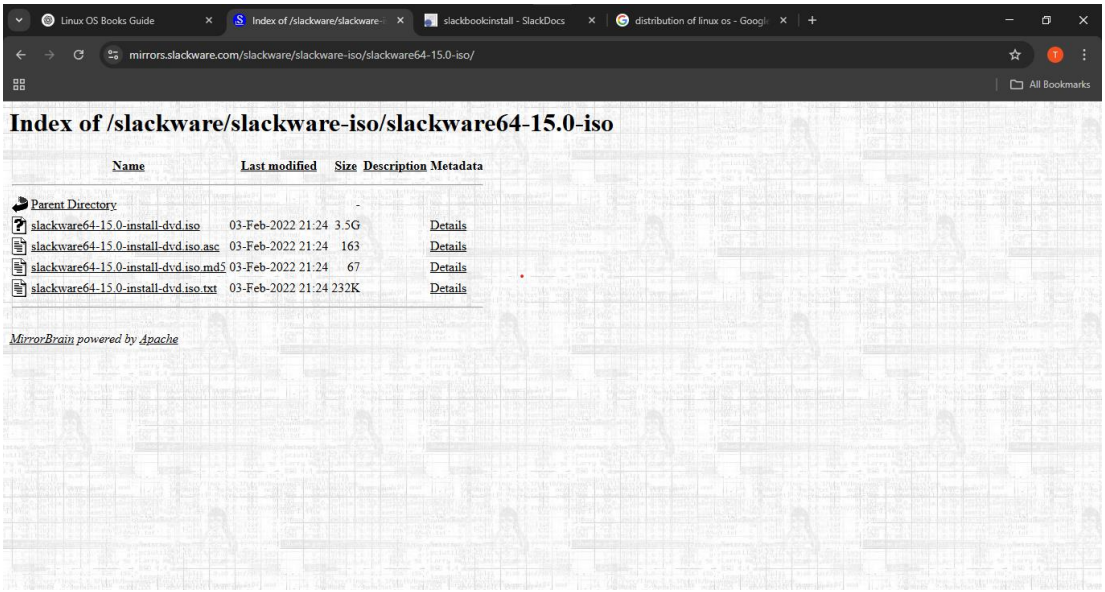


Fig2 Slackware Linux download interface

Step2: boot in to Slackware installer and login as root

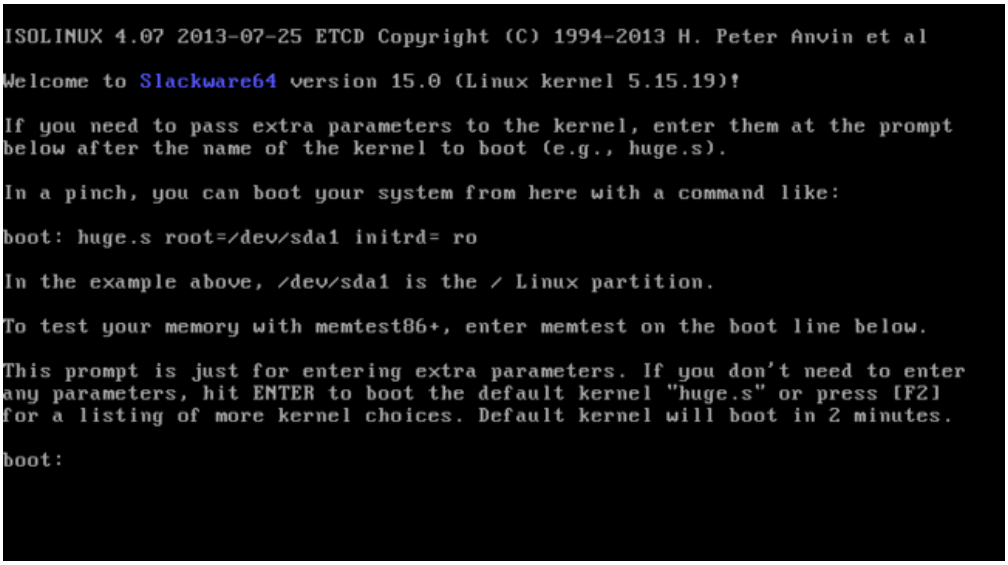


Fig 3 boot interface

Step 3 Use cfdisk to create partitions:

- Swap (e.g., 2 GB)
- Linux root (/) with ext4

Fig 4 partitioning

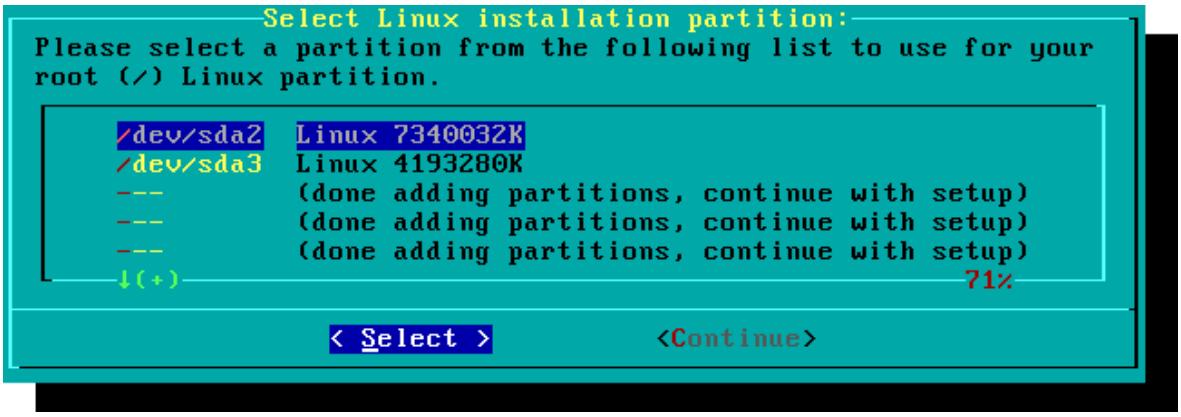
Step 4: Set up the program

Addswap

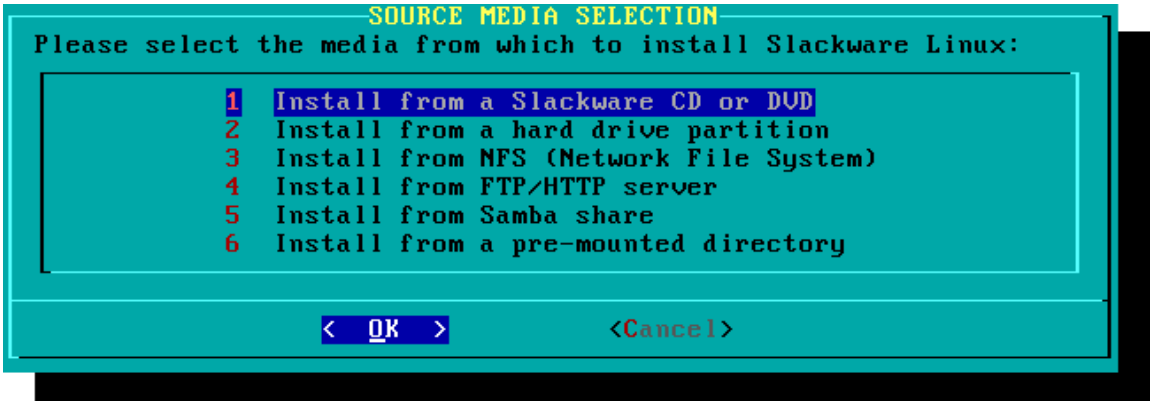
If you created a swap partition, this step will allow you to enable it before running any memory-intensive activities like installing packages. swap space is essentially virtual memory. It's a hard drive partition where regions of active system memory get copied when your computer is out of useable RAM. This lets the computer “swap” programs in and out of active RAM, allowing you to use more memory than your computer actually has.

select

Our next step is selecting our root partition and any other partitions we'd like Slackware to utilize..

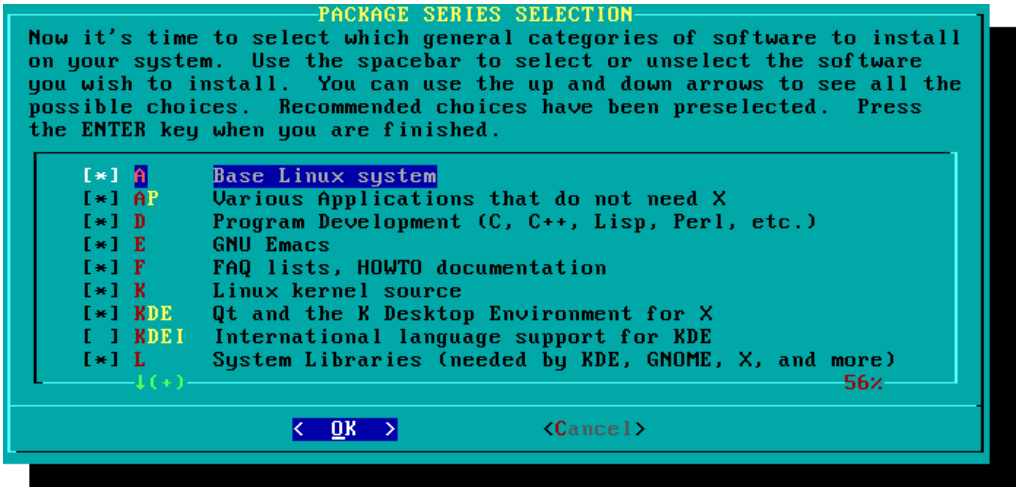


telling the installer where to find the Slackware packages. The most common method is to use the Slackware install DVD or CDs, but there are various other options are available.



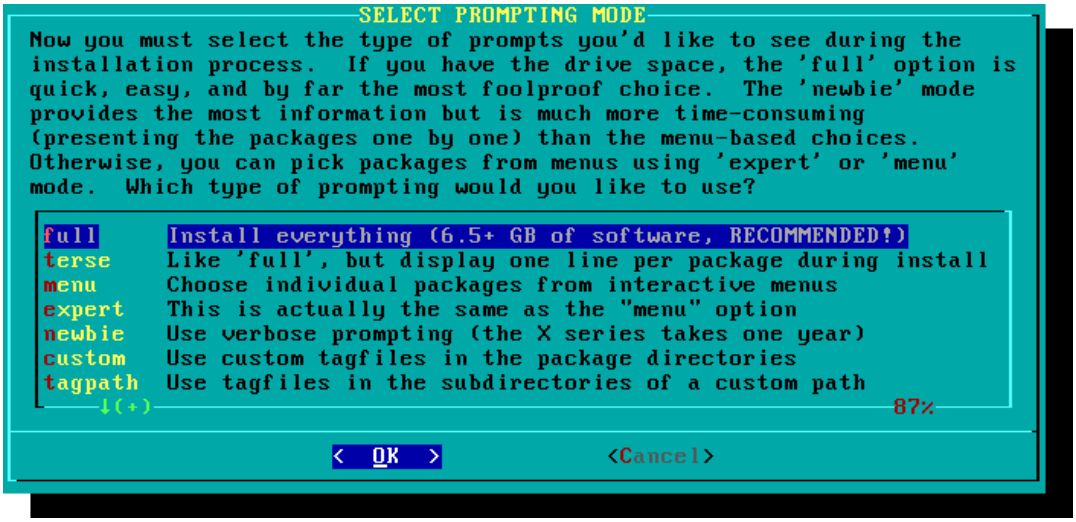
Series Selection

One unique feature of Slackware is its manner of dividing packages into disksets. At the beginning of time, network access to FTP servers was available only through incredibly slow 300 baud modems, so Slackware was split into disk sets that would fit onto floppy disks so users could download and install only those packages they were interested in.

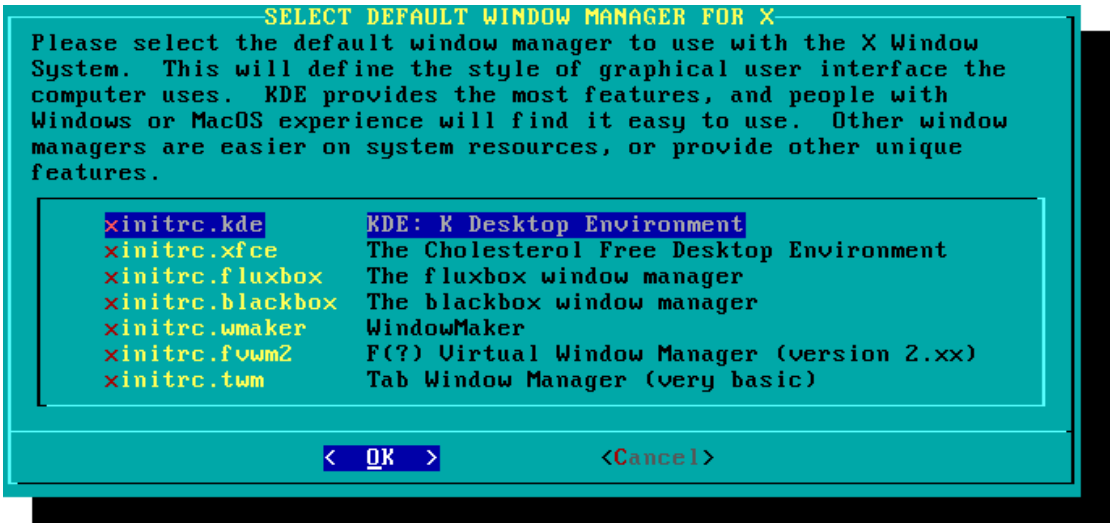


Install

Finally we get to the meat of the installer. At this stage, Slackware will ask what method to use to choose packages.



Select default desktop environment

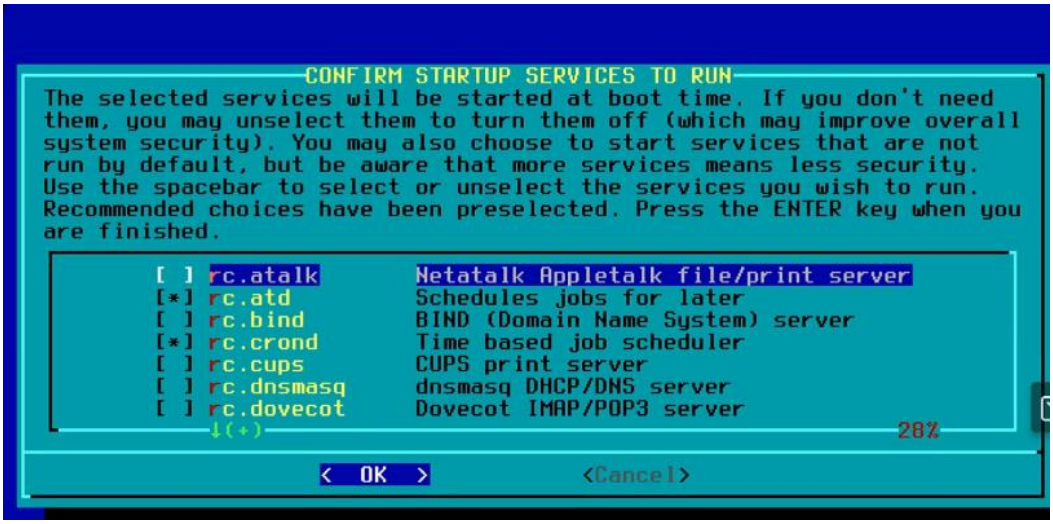


Configure

- Network using netconfig

- Timezone and hardware clock

Bootloader (LILO or GRUB)

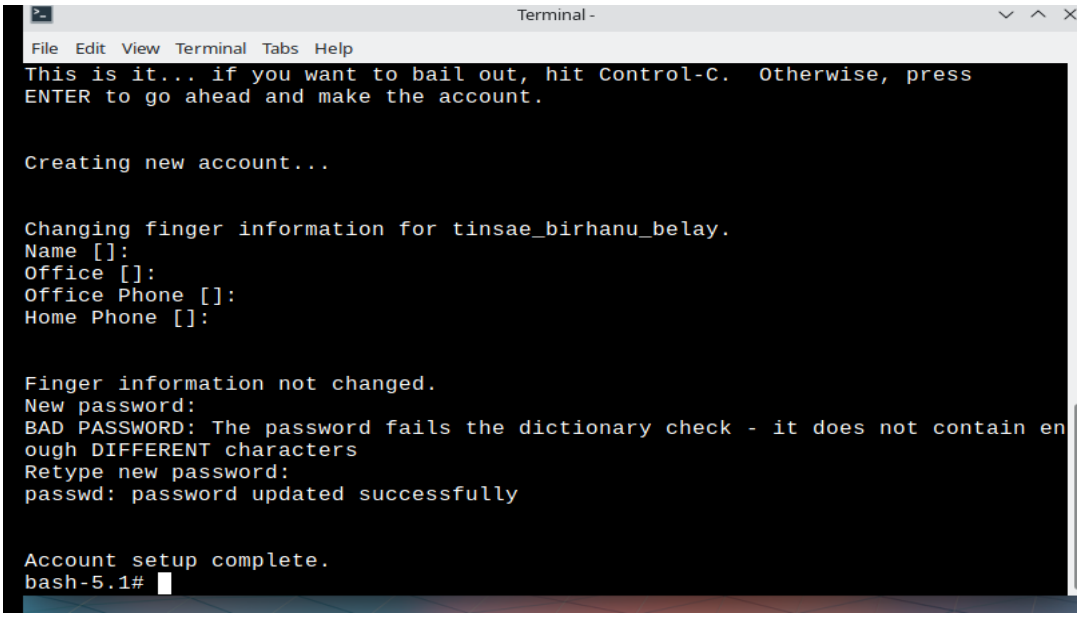


Step5: After configuration exit from the set up and Reboot the operating system

login to your system as root and then startx or install a display manager.

Step 6: Create a new user with My full name

After rebooting the system, I logged in as root and created a new user with my full name. Since Slackware Linux does not allow spaces in usernames, I used tinsae_birhanu_belay as the username.



6. Issues Faced and solution undertaken

1. the first issue I faced was to get direct link to download one of the virtual machine, VMware workstation on chrome

   ⇒ Solution: changing the browser

2. the second was VT-x Was Disabled in BIOS

   During the first boot of the Slackware virtual machine in VMware Workstation, I received an error related to virtualization support (Intel VT-x or AMD-V). The VM could not launch the 64-bit Slackware installer.

   Solution: first I asked what what VT-x and how can fix it then I follow the following steps and fix it

   Step 1: Reboot and Enter BIOS/UEFI

   Restart my PC and during the first few seconds of booting, press esc key

   Step 2: Find the Virtualization Setting

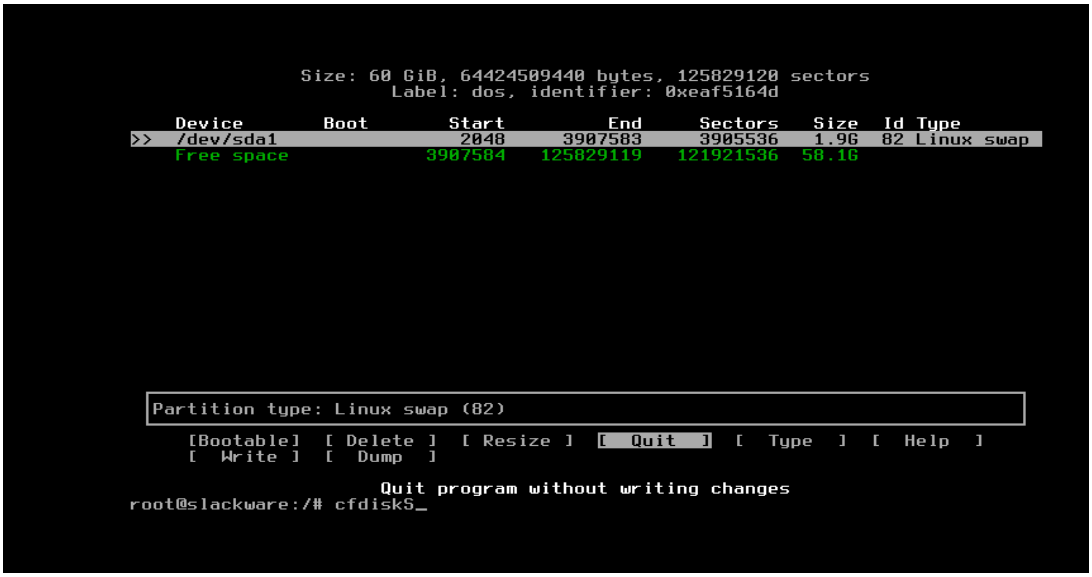
   - Advanced, Advanced CPU Configuration, Processor, or Security then I Look for Intel Virtualization Technology (VT-x) then I set it enabled

   Step 3: finally I Press F10 to Save and Exit By doing so I fixed it.

3. Navigation Difficulty During Partitioning Without Mouse Support

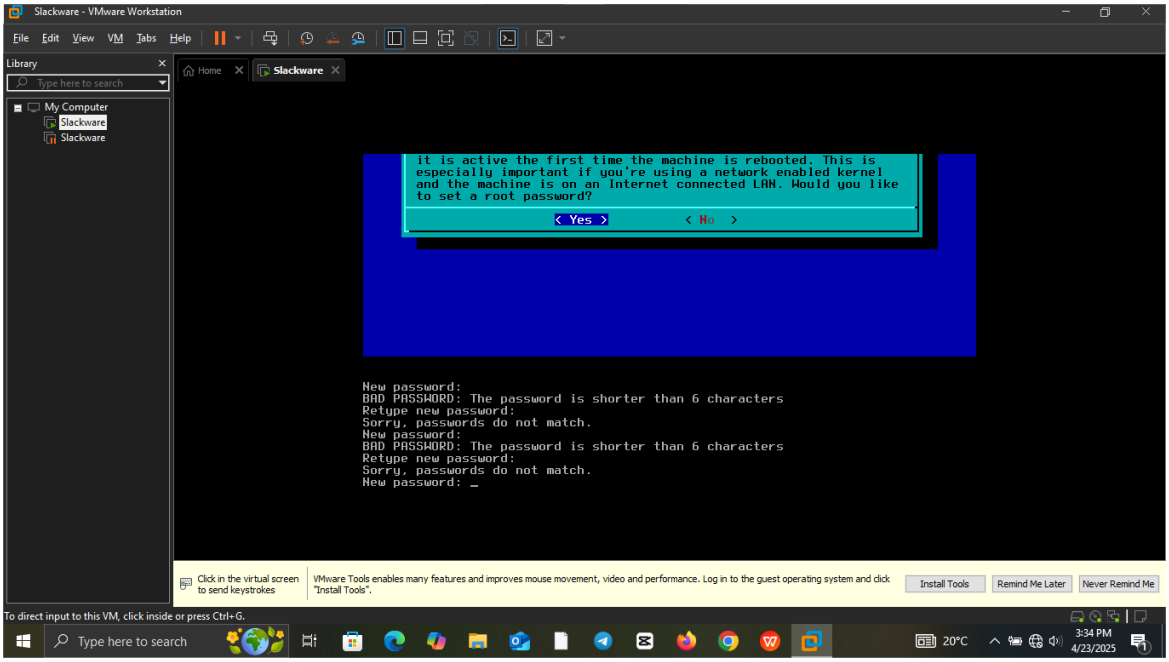
   In the Slackware text-mode installer, I was limited to using only the keyboard (arrow keys and Enter). This made it difficult to navigate the partitioning tool (cfdisk) efficiently. At one point, I made a mistake in partitioning and was unsure how to go back, so I had to restart the process to get back to the previous step.

⇒ Solutions: Since mouse support was not available during the text-based setup, I relied entirely on keyboard shortcuts. When I made an error during partitioning with cfdisk, I exited the tool, restarted it using the terminal by typing cfdisk, and carefully used arrow keys and Enter to navigate and recreate the correct partitions.



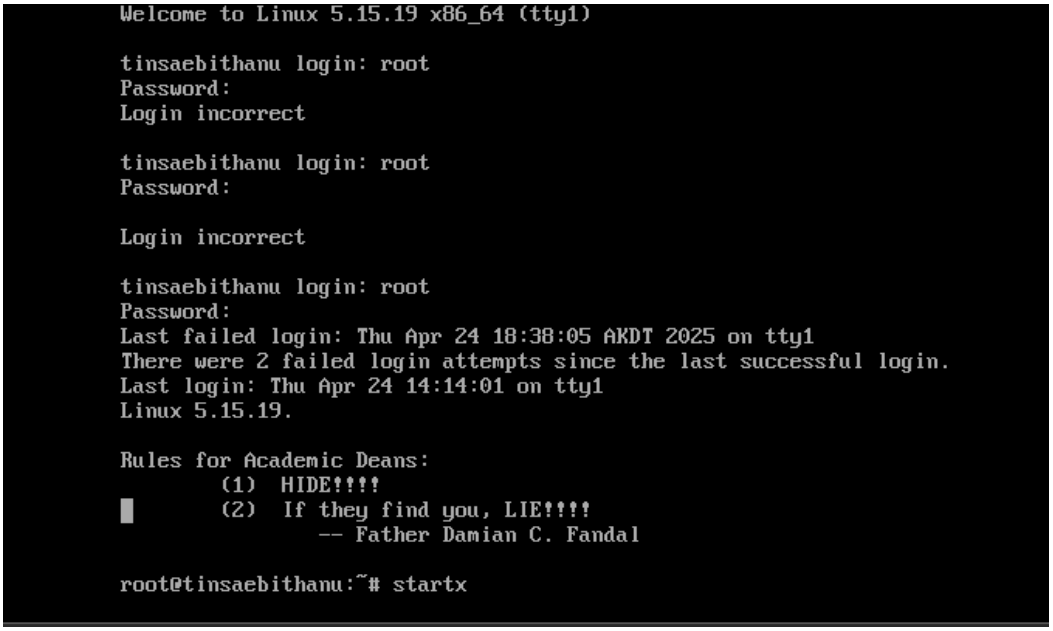
4. During changing password and power on the system, no characters appeared while typing. This is a default Linux security behavior to hide password length and prevent screen observation. I made mistake couple of times

⇒ Solution: because of many attempt it insist to change the password so I let it be as root and after robot I changed



Case 2

⇒ Solution: I confirmed the password was accepted when the system passed to the next step.



5. I also experienced problem dealing on terminals , tracking down missing editors (nano), to implement system call.

⇒ Solution: I solved it by watching videos related to system call in Slackware Linux and practice until it works.

Overall, in this project I struggle to familiarized with Slackware Linux's interface, minimalist philosophy and manual workflow, practical skills in virtualization, system administration, and low-level programming.

7. File support system

Slackware Linux supports a variety of Linux filesystems during installation and use. Each has unique features and trade-offs. Below is a detailed explanation of the most commonly supported filesystems:

- **ext4**

ext4 (Fourth Extended Filesystem) is the default and most widely used filesystem in Linux distributions. It supports journaling, large file sizes, extents, and backward compatibility with ext3 and ext2. Slackware recommends ext4 due to its stability, speed, and reliability. It is the best choice for the root (/) partition.

- **ext3**

(Third Extended Filesystem) -ext3 adds journaling to ext2, which helps prevent you from data disruption in case of unexpected system crashes. Although stable and dependable, it just lacks ext4's optimizations and scalability. It is considered obsolete by now.

- **ext2**

(Second Extended Filesystem) is a non-journaling file system type, and is the default for Linux before the 2.4 series. It's lightweight and still in use on small partitions such as /boot but it is not recommended for general usage because it lacks of crash recovery features.

- **XFS**

XFS is a high-performance journaling filesystem developed by SGI. It is optimized for parallel I/O operations and performs well with large files and filesystems. While robust, it is more complex to manage and not ideal for small-scale desktop use.

- **Btrfs**

Btrfs (B-tree Filesystem) is a modern filesystem designed for advanced features such as snapshots, compression, and self-healing. While promising, it is still considered experimental by some distributions, including Slackware, especially for use as a root filesystem.

- **ReiserFS**

ReiserFS is a journaling filesystem known for excellent performance with small files. However, it is now deprecated and rarely used due to lack of active maintenance and support.

- **FAT32**

FAT32 is a legacy Windows-compatible filesystem used primarily for USB drives and shared partitions. Slackware supports reading/writing to FAT32 via `dosfstools`, but it lacks journaling and Linux file permissions.

- **NTFS**

NTFS is a Windows-native filesystem. Slackware supports NTFS via the `ntfs-3g` driver. While suitable for external storage or dual-boot scenarios, it is not used for Linux root partitions due to permission handling limitations.

8. Advantage and Disadvantage of Slackware linux

Slackware Linux, known for its simplicity and stability, offers a unique experience for technically inclined users. Its advantages include a solid, predictable, and resource-light operating system that aligns with the core Unix philosophy. However, it also presents challenges like a lack of automatic dependency resolution and a steeper learning curve for beginners.

8.1 Advantages

- **Simplicity, Reliability and Stability**

Slackware prioritizes simplicity and stability, making it a good choice for those who value predictability and a well-defined system.

- **Resource-Light**

It tends to be lighter on system resources compared to some modern Linux distributions. Slackware sticks closely to upstream packages with very little “frankenstein” customization. What you install is what the original developers intended.

All system settings live in flat text files (e.g. /etc/rc.d/*, /etc/fstab), making it easy to read, version-control, and troubleshoot without hidden GUIs or database back-ends.

Slackware only ships well-tested, mature software. Major version upgrades happen infrequently, minimizing the risk of breaking your system. By avoiding forced dependency resolution, Slackware gives you control—there’s no sudden cascade of tens of packages being removed or downgraded by the package manager.

- **Unix Philosophy**

Slackware's design aligns with the core Unix philosophy of making programs do one thing well, promoting a modular and efficient approach.

- **Minimal Modifications**

Slackware aims to make few modifications to software packages from upstream, ensuring a more "Unix-like" experience.

- **No Systemd**

It doesn't use the systemd init system, which may appeal to users who prefer a more traditional approach.

Updates only when needed:

Slackware updates only what's necessary, avoiding "bleeding edge" updates; Not a "fix" for what isn't broken; It doesn't try to fix things that aren't broken, keeping the system simple and reliable; Focus on Command-Line:

- Slackware's emphasis on command-line interface (CLI) can be beneficial for users who prefer a text-based environment.

- Control and Flexibility: Users have complete control over their system, allowing for extensive customization and optimization.
- Learning and Education: Slackware provides a valuable learning experience for those who want to gain a deeper understanding of Linux.
- Server and Workstation: Slackware is a solid choice for both servers and workstations.
- Stability and Reliability: Slackware is known for its stability and reliability, making it a good choice for mission-critical systems.

8.2 Disadvantages

No Automatic Dependency Resolution:

Unlike many modern distros, Slackware requires manual dependency resolution for software packages.

- **Steeper Learning Curve**:

The lack of graphical installation and automated processes can make it challenging for beginners.

- **Lack of Formal Support:**

Slackware doesn't have a formal bug tracking facility or public code repository, and the community is smaller than some other distros.

- **No Automatic Package Removal:**

There's no automatic tool to remove unneeded dependencies when deleting a package, which can be frustrating, though sbopkg provides some assistance.

- **Default Package Selection:**

Slackware may recommend installing many packages by default, potentially leading to a bloated system if not carefully managed Steeper Learning Curve: Slackware's command-line focus and lack of graphical tools can be challenging for new Linux users.

- **Manual Configuration:** Many tasks, such as software installation and configuration, require manual intervention.

- **Potential for Compatibility Issues:** While Slackware's package system is generally reliable, there can be compatibility issues with some software.
- **Not User-Friendly:** The installer and overall system are not user-friendly, making it less appealing to casual users.

9. Recommendations

In the future It remains a strong choice for those who value stability, simplicity, and a deep understanding of their operating system. While it may not be the most mainstream choice due to its conservative approach and learning curve, Slackware's dedicated community and unique features ensure its place in the Linux ecosystem.

Reasons to consider Slackware in the future:

Stability and reliability: it is known for its robust stability and reliability, making it a good choice for long-term use and servers.

Traditional UNIX principles: it adheres to traditional UNIX conventions, which can be valuable for those who want to learn how their operating system works in a more fundamental way.

Customization and control: it offers a high degree of customization and control, allowing users to tailor their system to their specific needs.

Staying the course: By not adopting Systemd, it caters to users who prefer a more traditional approach to system management.

Reasons to consider other distributions:

- User-friendliness:

Debian-based distributions like Ubuntu and Mint are generally considered more user-friendly and offer more extensive support, which may be better suited for newcomers to Linux.

- Ease of use:

Other distributions often have more streamlined package management systems and graphical interfaces, making them easier to use and administer.

- Community support:

Larger and more established distributions typically have larger and more active communities, which can be helpful for troubleshooting and finding information.

In conclusion: Slackware is not a "one-size-fits-all" distribution, and its future recommendation depends on the user's priorities and technical skills. It's a great choice for those who value stability, customization, and a hands-on approach to Linux, while other distributions may be more suitable for those who prioritize ease of use and mainstream support.

- Use Slackware as a foundation for advanced system programming
- Build automated scripts for repetitive configuration
- Create custom Slackware ISOs with preinstalled software
- Recommend GUI setup for daily usage

10 virtualization: What,Why and How

10.1 What is virtualization

Virtualization is technology that you can use to create virtual representations of servers, storage, networks, and other physical machines. Virtual software mimics the functions of physical hardware to run multiple virtual machines simultaneously on a single physical machine. Businesses use virtualization to use their hardware resources efficiently and get greater returns from their investment. It also powers cloud computing services that help organizations manage infrastructure more efficiently.

It is also a process that allows a computer to share its hardware resources with multiple digitally separated environments. Each virtualized environment runs within its allocated resources, such as memory, processing power, and storage. With virtualization, organizations can switch between different operating systems on the same server without rebooting.

Virtual machines and hypervisors are two important concepts in virtualization.

Virtual machine

- A *virtual machine* is a software-defined computer that runs on a physical computer with a separate operating system and computing resources. The physical computer is called the *host machine* and virtual machines are *guest machines*. Multiple virtual machines can run on a single physical machine. Virtual machines are abstracted from the computer hardware by a hypervisor.

Hypervisor

- The *hypervisor* is a software component that manages multiple virtual machines in a computer. It ensures that each virtual machine gets the allocated resources and does not interfere with the operation of other virtual machines. There are two types of hypervisors.

Type 1 hypervisor

- A type 1 hypervisor, or bare-metal hypervisor, is a hypervisor program installed directly on the computer's hardware instead of the operating system. Therefore, type 1 hypervisors have better performance and are commonly used by enterprise applications. KVM uses the type 1 hypervisor to host multiple virtual machines on the Linux operating system.

Type 2 hypervisor

- Also known as a hosted hypervisor, the type 2 hypervisor is installed on an operating system. Type 2 hypervisors are suitable for end-user computing.

10.2 Why virtualization

- **Efficient resource use**

Virtualization improves hardware resources used in your data center. For example, instead of running one server on one computer system, you can create a virtual server pool on the same computer system by using and returning servers to the pool as required. Having fewer underlying physical servers frees up space in your data center and saves money on electricity, generators, and cooling appliances.

- **Automated IT management**

Now that physical computers are virtual, you can manage them by using software tools. Administrators create deployment and configuration programs to define virtual machine templates. You can duplicate your infrastructure repeatedly and consistently and avoid error-prone manual configurations.

- **Faster disaster recovery**

When events such as natural disasters or cyberattacks negatively affect business operations, regaining access to IT infrastructure and replacing or fixing a physical server can take hours or even days. By contrast, the process takes minutes with virtualized environments. This prompt response significantly improves resiliency and facilitates business continuity so that operations can continue as scheduled.

10.3 How does virtualization

Virtualization uses specialized software, called a hypervisor, to create several cloud instances or virtual machines on one physical computer.

Cloud instances or virtual machines

After install virtualization software on the computer, we can create one or more virtual machines. You can access the virtual machines in the same way that you access other applications on your computer. Your computer is called the host, and the virtual machine is called the guest. Several guests can run on the host. Each guest has its own operating system, which can be the same or different from the host operating system.

From the user's perspective, the virtual machine operates like a typical server. It has settings, configurations, and installed applications. Computing resources, such as central processing units (CPUs), Random Access Memory (RAM), and storage appear the same as on a physical server. You can also configure and update the guest operating systems and their applications as necessary without affecting the host operating system.

Hypervisors

The hypervisor is the virtualization software that you install on your physical machine. It is a software layer that acts as an intermediary between the virtual machines and the underlying hardware or host operating system. The hypervisor coordinates access to the physical environment so that several virtual machines have access to their own share of physical resources.

For example, if the virtual machine requires computing resources, such as computer processing power, the request first goes to the hypervisor. The hypervisor then passes the request to the underlying hardware, which performs the task.

The following are the two main types of hypervisors.

Type 1 hypervisors

A type 1 hypervisor—also called a bare-metal hypervisor—runs directly on the computer hardware. It has some operating system capabilities and is highly efficient because it interacts directly with the physical resources.

Type 2 hypervisors

A type 2 hypervisor runs as an application on computer hardware with an existing operating system. Use this type of hypervisor when running multiple operating systems on a single machine.

11 System call implementation

In Slackware Linux, system calls are the interface through which user-level programs request services from the kernel. They allow programs to interact with the operating system, enabling tasks like accessing hardware, managing memory, and performing file operations. Slackware, being a Linux distribution, inherits the same system call mechanisms as other Linux distributions.

There are vast number of system calls for various purposes.

Examples of common system calls:

- open(): Opens a file for reading or writing.
- read(): Reads data from a file or socket.
- write(): Writes data to a file or socket.
- fork(): Creates a new process.
- exec(): Replaces the current process with a new one.
- exit(): Terminate a process

➢ **from those I will emphasize on clock_gettime() system call and its implementation**

The clock_gettime system call is a successor to the gettimeofday system call with a few key changes: higher precision and the ability to request specific clocks. It fills in a structure containing two fields: a seconds and a nanosecond count of the time since the Epoch (00:00 1 January, 1970 UTC).

Steps for the implementation of clock_gettime()

1. Open the Terminal

In Slackware virtual machine:

- If you're in the desktop (GUI), look for an app called “Terminal”, “Xterm”, or “Konsole”.

- Or press Alt + F2, type xterm, and press Enter.

2. Open a Text Editor (I Use vi)

vi gettime.c

3. Type This Code:

Once inside vi, do the following:

1. Press i (this puts you in Insert mode).
2. type this code:

```
#include <stdio.h>
#include <time.h>
int main() {
   struct timespec ts;
   if (clock_gettime(CLOCK_REALTIME, &ts) == -1) {
      perror("clock_gettime");
      return 1;
   }
   printf("Current time: %ld seconds and %ld nanoseconds\n", ts.tv_sec,
ts.tv_nsec);
   return 0;
}
```

3. When it is done, press the Esc key.
4. Then type :wq (to save and exit):

4. Compile the Program

Type gcc gettime.c -o gettime –lrt and then press enter

5. Run the Program

Finally run the program ./gettime

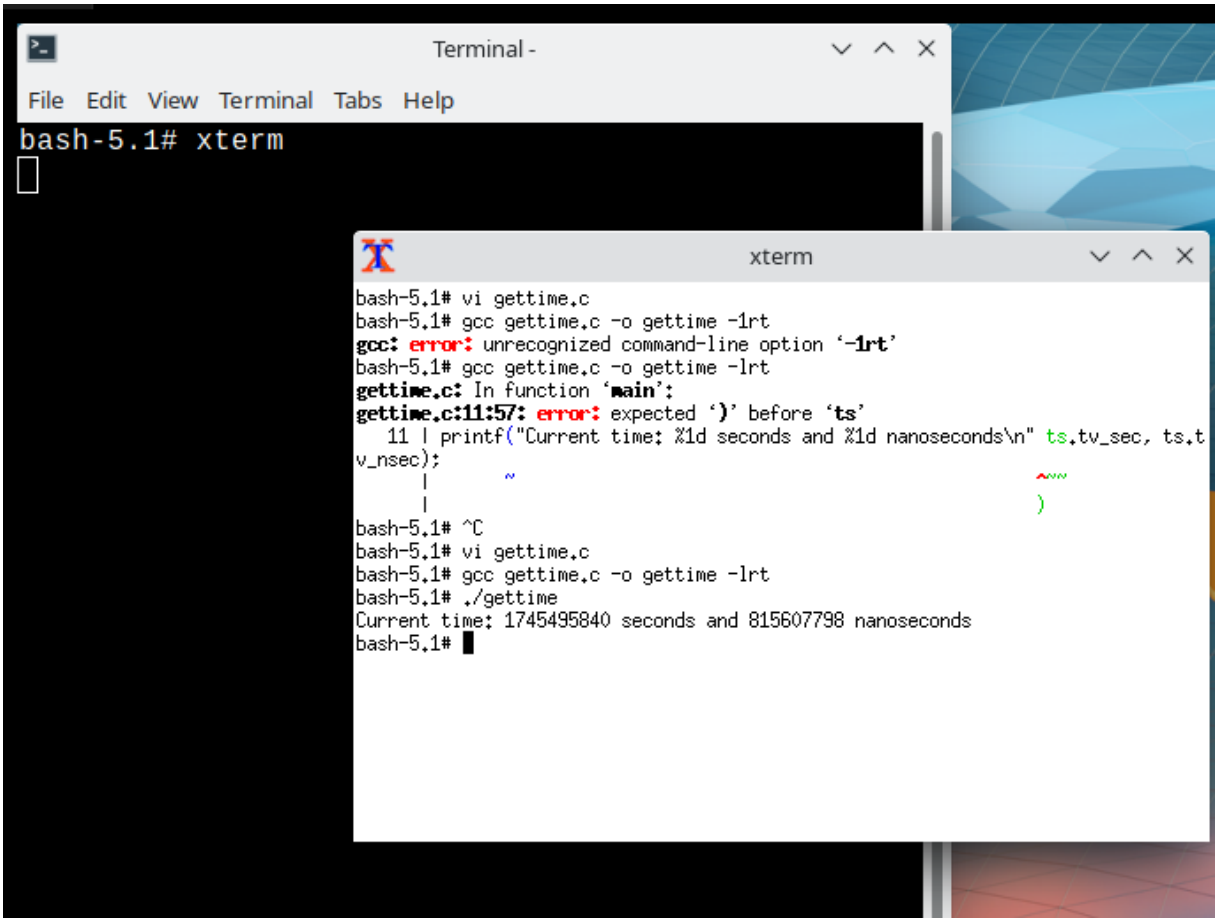


Fig Clock_gettime() system call implementation

12. Conclusion

Installing and configuring Slackware Linux has provided a distinctly hands-on experience with a classic UNIX-like operating system. In contrast to more automated distributions, Slackware's text-mode installer and plain-text configuration files require direct engagement with each step, from bootloader configuration with LILO to network interface setup with netconfig and manual disk partitioning with cfdisk and formatting with mkfs.ext4. This process deepened understanding of Linux internals and the fundamental role that each configuration file and system script plays in making the OS work.

Running Slackware in a virtual environment (VirtualBox or VMware Workstation) illustrated the power and flexibility of virtualization technology. Installing VMware Tools or the VirtualBox Guest Additions further demonstrated how guest-host integration enhances usability—enabling proper mouse support, shared clipboards, and seamless window resizing.

Beyond installation, We can extended Slackware’s capabilities through system programming and system calls: writing a shell script for automated file backups and a C program leveraging the clock_gettime() system call. Compiling and running these utilities under Slackware’s minimalist environment reinforced how user-level applications interact directly with kernel services.

13. references

Slackware Linux Project. (n.d.). How to configure networking in Slackware. Slackware Documentation Project. https://docs.slackware.com/howtos:network_services:network_configuration

Slackware Linux Project. (n.d.). Slackpkg tool documentation. Slackware Documentation Project. https://docs.slackware.com/slackware:slackpkg

Slackware Linux Project. (n.d.). Slackware Linux essentials. Slackware Documentation Project. https://docs.slackware.com

Slackware Linux Project. (n.d.). System startup and boot process. Slackware Documentation Project. https://docs.slackware.com/howtos:slackware_admin:boot_process

Volkerding, P. (2022). Slackware ChangeLog.txt – Slackware 15.0 release notes. Slackware Linux. https://www.slackware.com/changelog/current.php?cpu=x86_64