Exploring Linux: Installation, Usage, and Maintenance

Guru Swarupa

May 30, 2025

Abstract

This document outlines my personal experiences and general insights using Linux-based operating systems. It covers choosing a distribution, preparing for installation, setting up the system, common usage practices, and essential maintenance habits. Whether you're a new user or exploring advanced distributions like Arch or Debian, this guide provides a practical foundation.

1 Introduction

Linux is a family of open-source Unix-like operating systems that has become a favorite among developers, sysadmins, and privacy-conscious users. It offers greater control, flexibility, and efficiency than many proprietary alternatives. Over the years, I've worked with several Linux distributions—primarily Arch Linux and Debian—and gained insights into installation practices, daily usage, and system upkeep.

It's been over six years since I first ventured into the world of Linux, and I'm still learning something new every day. The depth and versatility of the Linux ecosystem continuously encourage exploration and problem-solving. In this guide, I'll share the most important things I've learned—stuff that applies no matter which distro you use. It's a mix of my own experience and a lot of wisdom from the awesome Linux community.

2 What I Learned from Installing Linux

I'll never forget my first Linux install—it felt like diving under the hood of my computer for the first time. It wasn't just about putting a new OS on my laptop; it was about understanding how things actually worked behind the scenes. Here are the most important lessons and realizations I gained through the process:

1. Understanding the Boot Process and Installation Internals

Before Linux, the boot process was something I never really thought about. During installation, I learned about:

- BIOS vs UEFI firmware
- Partitioning (MBR vs GPT, root/home/swap setups)
- Bootloaders like GRUB and systemd-boot

This gave me insight into how an operating system boots and manages hardware at a low level.

2. Discovering the Power and Speed of Linux

Once installed, I was amazed at how fast Linux booted and ran—even on older hardware. The system felt snappy and responsive, with fewer background processes and no unnecessary bloat. It taught me that a well-configured system doesn't have to be resource-heavy to perform well.

3. Learning About Package Management

One of the standout features of Linux is its package management system. Installing software became incredibly easy:

- `apt` for Debian/Ubuntu
- `pacman` and AUR helpers for Arch
- `flatpak`, `snap`, and `AppImage` for universal packages

It was refreshing to manage software through a simple terminal command, without hunting down installers and manually updating apps.

4. Exploring Distributions Out of Curiosity

Curiosity led me to try multiple distros: Debian for its stability, Arch for its simplicity and control, and others like Linux Mint, Pop!_OS, Fedora, Manjaro, KDE Neon, Opensuse, Zorin, Elementary, EndeavourOS and you name it. Each had a unique flavor, package philosophy, and community. This exploration made me appreciate the diversity and freedom within the Linux ecosystem.

5. Embracing Tinkering and Ricing

Ricing—customizing the desktop for both aesthetics and function—became an enjoyable pastime. I learned to configure:

- Window managers like i3 and awesome
- Status bars like Polybar
- Tools like `neofetch`, `rofi`, and `picom`

This hands-on tweaking deepened my understanding of how the Linux desktop environment is structured.

6. Mastering the Command Line and Scripting

Spending time in the terminal felt intimidating at first, but over time, I became comfortable using the shell for daily tasks. Learning commands, flags, pipes, and scripting gave me confidence in navigating Linux more efficiently.

Eventually, I even contributed by using community scripts that help newcomers. One such tool is Chris Titus Tech's `Linux Utility`, a robust distro-agnostic script that helps optimize and streamline system setup.

7. Using `linutil`: A Community-Driven Utility

Chris Titus Tech's Linux Utility (known as `linutil`) was a great example of how the open-source community builds helpful tools. Written in Rust, `linutil` simplifies tasks like installing apps, tweaking system settings, and improving performance:

- GitHub: `https://github.com/ChrisTitusTech/linutil`
- AUR: `linutil` or `linutil-bin`

To run it, simply execute:

```
curl -fsSL https://christitus.com/linux | sh
```

Tools like this not only make Linux more accessible but also demonstrate the strength of the open-source community in creating user-friendly solutions.

3 Choosing a Linux Distribution

The Linux ecosystem is vast. Each distribution (distro) serves a different user base. Here's a brief overview:

- **Linux Mint**: User-friendly, great for beginners, and based on Debian.
- **Debian**: Stable and well-supported; ideal for servers or users who prefer reliability.
- **Arch Linux**: Rolling release model, minimal by default, and highly customizable—best for advanced users.

When selecting a distro, consider hardware compatibility, community support, and whether you prefer stability (Debian) or bleeding-edge features (Arch).

4 Preparing for Installation

Before installing Linux, preparation is crucial:

Checklist

- **Backup important data** from your current operating system.
- **Check hardware compatibility**, especially Wi-Fi adapters and GPUs.
- **Choose a desktop environment (DE)** or window manager (WM) depending on your needs and preferences:
  - **DEs:** GNOME, KDE Plasma, Xfce, Cinnamon – provide a complete graphical interface.
  - **WMs:** i3, dwm, awesome, hyprland – lightweight and minimal, for more control and speed.
- **Create a bootable USB drive** using tools like:
  - `Rufus` (Windows)
  - `Balena Etcher`, `Ventoy`, or `dd` (Linux/macOS)

5 Installing Linux

Graphical Installers (Beginner-Friendly)

Most beginner-oriented distros (e.g., Ubuntu, Mint) offer GUI-based installers:

1. Boot from the USB stick.
2. Choose your language and keyboard layout.
3. Configure disk partitions (use entire disk or manual setup).
4. Create a user and set a strong password.
5. Install and reboot.

Manual Installation (Advanced)

For advanced users or those who want full control, manual installation (such as with Arch Linux) involves several key steps. Here's a clear overview:

- **Boot into the live environment:** Start the system using the bootable USB containing the Arch Linux ISO.
- **Connect to the internet:** Ensure you have a working internet connection (wired or wireless).
- **Set system clock:** Synchronize the system clock with network time.

- **Partition the disk:** Create required partitions like EFI (for UEFI systems), root (`/`), optional home, and optionally swap.
- **Format the partitions:** Choose and apply file systems (e.g., ext4, btrfs) to each partition.
- **Mount the partitions:** Mount the root partition and any others (e.g., boot, home) to appropriate directories.
- **Install the base system:** Install the core operating system and kernel onto the mounted root.
- **Generate the filesystem table (fstab):** Automatically generate the fstab file to define mount points.
- **Chroot into the new system:** Switch into the newly installed system environment to continue setup.
- **Set the timezone:** Define your geographical timezone.
- **Set locale and language settings:** Enable your preferred language and character encoding.
- **Set the hostname and configure network:** Assign a name to your system and set up basic networking files.
- **Set the root password:** Define a secure password for the root user.
- **Install a bootloader:** Set up GRUB, systemd-boot, or another boot manager to load the OS.
- **Create a regular user (optional but recommended):** Add a new user account and configure sudo permissions.
- **Finish and reboot:** Exit the setup environment, unmount partitions, and reboot into your new system.

Tip: Arch's official wiki is the best place for detailed guidance and troubleshooting.

6 Using Linux: Daily Practices

Basic Terminal Commands

The terminal is essential in Linux:

- `ls`, `cd`, `pwd` – navigation
- `cp`, `mv`, `rm` – file operations
- `sudo apt update && sudo apt upgrade` (System Update for Debian)
- `sudo pacman -Syu` (System Update for Arch)
- `htop`, `journalctl`, `df -h` – system monitoring

Package Management

Each distro has its own package manager:

- `apt` – Debian-based distros
- `pacman` – Arch Linux
- `dnf` – Fedora
- `yay`, `paru` – AUR helpers (for Arch users)

GUI software managers like GNOME Software or KDE Discover make software discovery easier for beginners.

7 Maintaining the System

System Updates

- Update regularly to receive security patches and new features.
- Use `apt`, `pacman`, or other tools depending on your distro.

System Cleanup

- Remove unused packages:
  - `sudo apt autoremove`
  - `sudo pacman -Rns` *`package-name`*
- Clear cache:
  - `sudo apt clean`
  - `sudo pacman -Sc`

Backup Solutions

Always maintain backups:

- `rsync` – command-line tool for syncing directories
- `Timeshift` – system restore points (ideal for Debian-based systems)
- Cloud backups via `rclone` or dedicated services

8 Troubleshooting Common Issues

- **Boot issues**: Use live USB to chroot into system and reinstall bootloader.
- **Broken updates**: Downgrade packages or rollback using Timeshift (if available).
- **Hardware drivers**: For Wi-Fi or GPU, consult Arch/Debian wiki or use proprietary drivers where necessary.
- **Logs and debugging**: Use `dmesg`, `journalctl -xe`, or logs in `/var/log`.

Communities such as AskUbuntu, Arch Forums, and Unix Stack Exchange are excellent resources for help.

9 Why Linux is Better Than Windows (In Many Cases)

After years of using both Linux and Windows, I've come to appreciate the many reasons why Linux often outshines Windows—especially for developers, power users, and privacy-conscious individuals. Here are some of the key advantages:

1. Open Source and Transparent

Linux is fully open-source, meaning its source code is publicly available. This fosters trust, community contributions, and faster bug fixes. In contrast, Windows is proprietary and closed, limiting user insight into what the OS is doing in the background.

2. Lightweight and Efficient

Many Linux distributions are far more lightweight than Windows. They consume fewer system resources, boot faster, and are highly customizable. This makes Linux ideal for both modern and older hardware.

3. No Bloatware or Forced Updates

Linux gives you full control over what's installed and when updates happen. Windows, on the other hand, often comes preloaded with unnecessary software and enforces automatic updates that can interrupt your work.

4. Stronger Privacy and Security

Linux does not track user activity the way Windows does. Its permission model, open-source nature, and lack of widespread malware make it inherently more secure out of the box.

5. Power of the Terminal and Scripting

The Linux terminal is far more powerful and integrated than Windows Command Prompt or even PowerShell. It allows for efficient system management, automation, and scripting without third-party tools.

6. Vast Customization and Control

From choosing your desktop environment to configuring system daemons, Linux gives you control over almost every aspect of your system. Windows offers limited customization by comparison.

7. Package Management and Software Freedom

Linux's package managers (like `apt` or `pacman`) simplify software installation, updates, and removal. You can also choose from a wide variety of open-source alternatives to commercial software.

8. Ideal for Development and Learning

Linux is the preferred OS for most programming and DevOps tasks. Tools like Docker, Git, SSH, and build systems are natively supported and often more stable on Linux.

9. Community Support and Documentation

Linux has a vibrant community and extensive documentation. The Arch Wiki, Stack Exchange, GitHub, and various forums provide help and tutorials for nearly every issue you might face.

10. It Respects You

Perhaps most importantly, Linux respects your choices. You are not treated as a product. You are in control of your system, your updates, your data, and your experience.

Note: While Linux isn't perfect for everyone (e.g., gamers reliant on Windows-only titles or users with specific software needs), it's increasingly viable for general computing, development, media, and even gaming thanks to advancements like Proton and Flatpak.

10 Linux Alternatives to Popular Windows Software

One of the most common concerns for new users switching to Linux is software availability. The good news? Almost everything you can do on Windows has an excellent counterpart on Linux—many of which are open-source, lightweight, and privacy-respecting. Here's a list of powerful alternatives:

Windows Software	**Linux Alternative**
Microsoft Office	**OnlyOffice**, LibreOffice, WPS Office
Microsoft Edge	**Brave**, Firefox, Vivaldi, Chromium
uTorrent / BitTorrent	**qBittorrent**, Transmission
Windows Media Player	**VLC Media Player**, MPV
Adobe Photoshop	**GIMP** (GNU Image Manipulation Program)
Adobe Premiere Pro	**Kdenlive**, Shotcut
Audacity	**Audacity** (also available on Linux)
Notepad++	**Geany**, Gedit, Kate, Sublime Text
FileZilla	**FileZilla** (cross-platform)
Visual Studio	**Visual Studio Code**, VSCodium
Steam (Gaming)	**Steam** for Linux (via Proton)
Paint.NET / MS Paint	**Pinta**, Krita
Windows Explorer	**Nautilus**, Dolphin, Thunar
Task Manager	**htop**, System Monitor
WinRAR / 7-Zip	**PeaZip**, Ark, Engrampa, unzip, tar
Windows Terminal / PowerShell	**bash**, zsh, fish

Bonus: Universal Package Formats

To simplify app installation across all Linux distributions, many of the above apps are available in formats like:

- `Flatpak` – sandboxed apps from Flathub
- `AppImage` – portable apps with no installation required

You can explore thousands of Linux-compatible apps on `https://flathub.org`

11 Conclusion

Linux is more than just an operating system—it's a mindset of openness, customization, and control. Whether you prefer the stability of Debian or the rolling edge of Arch, Linux provides an environment tailored to both newcomers and experts. With the right tools, practices, and curiosity, it becomes a rewarding journey into the world of free software and computing empowerment.

References

- `https://www.archlinux.org`
- `https://www.debian.org`
- `https://wiki.archlinux.org`
- `https://linuxcommand.org`
- `https://ubuntu.com/tutorials`