IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 10, Issue 3, March 2022

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.165

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165 |

|| Volume 10, Issue 3, March 2022 ||

| DOI: 10.15680/IJIRCCE.2022.1003097 |

A Linux Based Lightweight Command Line Operating System

Aditya Uday Apte, Praharsh Ravindra Shinde, Preyas Ravindra Shinde, Soheb Samad Mujawar

Final Year Student, Dept. of I.T., Thakur Polytechnic, Kandivali East, Mumbai, Maharashtra, India

Final Year Student, Dept. of I.T., Thakur Polytechnic, Kandivali East, Mumbai, Maharashtra, India

Final Year Student, Dept. of I.T., Thakur Polytechnic, Kandivali East, Mumbai, Maharashtra, India

Final Year Student, Dept. of I.T., Thakur Polytechnic, Kandivali East, Mumbai, Maharashtra, India

ABSTRACT: The Linux based operating system named as "Aspire Linux", is a basic Command Line operating based on the latest available Linux Kernel. The main motive of this project to create an operating system which can run on any kind of Desktop/Laptop, be it a slow or a real fast system, the main focus is to target on old systems who has less available Random Access Memory (RAM) and storage available, this will give a new life to an old system, the operating system which consumes less RAM and is easy on your CPU. This project will help people understand how a Linux Operating System works internally.

KEYWORDS: Linux, Operating System, Lightweight, Kernel, Source Code, Bootloader.

I. INTRODUCTION

The first version of the Linux Kernel was written on 17 September 1991 by a Finnish Software Engineer named Linus Torvalds, he offered the people to use, modify and distribute his kernel for entirely free since the Linux is an Open Source project, which means any one can modify, distribute their own copy of the Linux Kernel.

The Aspire Linux Operating System is based on the latest Linux Kernel which is the Linux-5.17.5 Kernel at this current date. It is a command line operating system, meaning that this Operating System (OS) does not have a Graphical user Interface (GUI), all the interaction is done via a Command Line Interface (CLI). This poses a major problem of novice users who are not familiar or just are not comfortable using an operating system which just a command line interface. However, there are some major advantages over the most popular Linux Distros or Distributions which is less RAM usage, low in size or uses less disk space and speedy. This operating uses a systemd daemon since it is a core part of a Linux system. Systemd is a new modern replacement for sysvinit systems, since sysvinit all the way originates to UNIX systems. Many modern Linux Distributions like Fedora Linux, Arch Linux, Ubuntu Linux ship with systemd daemon as it is stable and updated frequently.

The Aspire Linux project is a Linux Distribution which is compiled entirely from source which means it gives full control over what gets installed which significantly reduces the risk factor of the system getting hacked after applying security patches.

An extremely compact Linux Distributions which can only use about 1 GB of space if there are no other packages installed except the base packages which are needed in order for the operating system to function correctly.

RAM usages can be as low 100 MB since there are no preloaded services and software included.

II. RELATED WORK

Aspire Linux is built entirely from source by gathering some base packages required to get the operating system up and running. The major advantage of building the operating system from source is that we can control on what is getting installed by applying all the necessary patches to the respective packages.

Aspire Linux is compiled on top of an already well established Linux Distribution, in this case we used Manjaro Linux, which is based on Arch Linux, as it is a rolling release, meaning that the Distribution gets constant updates which include Security Updates, package updates and other miscellaneous updates.

The Aspire Linux distribution is a totally independent operating system which is not based on any other existing operating system like Ubuntu Linux, which is based on Debian Linux. This project aims to grow continuously and implement new features to become a better Linux Distribution.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165 |

|| Volume 10, Issue 3, March 2022 ||

| DOI: 10.15680/IJIRCCE.2022.1003097 |

III. PROPOSED METHODOLOGY

The Aspire Linux project is built with a set steps starting from downloading the necessary source files and all the way to booting it.

The steps were carried out as follows:

A. The appropriate source files were downloaded from their respective websites by following the LinuxFromScratch (LFS) Book.

B. After downloading the base system files, it was time to check the authenticity of those files to verify their integrity, their MD5 Sums were checked, after all the packages passed the MD5 Sums check, it was time to build a Cross Compiler.

C. Now we have to format the target disk drive in a specific layout, means creating GRUB BIOS Partition of size 1 MB, creating a FAT32 Partition of 512 MB, an EXT4 Partition of size respective to the remaining disk space before creating a fixed SWAP partition of 8 GB.

D. Now that the partitions are created and formatted, we need to mount the root volume of the target disk, in this case is the EXT4 partition to the /mnt/linux directory of the host system.

Important Note: The 1MB GRUB BIOS Partition need not be formatted as that 1MB is required by GRUB to boot the Linux System which uses Legacy BIOS.

Important Note: /mnt/linux needs to be created by running /mnt/linux (or any other name of choice) and then setting the environment variable as $LINUX=/mnt/linux

Important Note: The purpose of building a Cross Compiler is to create an isolation between the host system and the target system, means building basic cross compiler which will build a new compiler which is used to build the new operating system in which all the tools and sources are isolated from the host system on which the operating system is made on.

E. Switch to the root user of your host system to avoid any permission error while compiling packages

F. Now we need to cd into the target disk directory, like cd $LINUX

G. The Cross Compiler is now being built in this stage. Some base packages like Binutils, GCC (GNU Compiler Collection), Linux API Headers, Glibc, libstdc++ from the GCC package.

H. Now that we have our temporary tools cross compiler, the full-fledged compiler will be built for the new operating system eventually fully isolating from the host environment.

I. Some packages are built here, like M4, Make, Patch, Ncurses, Bash, Coreutils, Grep, Tar, XZ, Gzip, etc.

J. In this same section Binutils and GCC are getting recompiled to create a fully new GCC compiler for building packages.

K. After a new Compiler is built, it was finally time to go into the chroot environment.

Important Note: The chroot environment is a root environment in which we go into the target system while still running our host system for kernel purposes. In this stage we are basically in the target system which is just newly built or for recovery purposes.

L. When in the chroot environment, we have to build and finalize the target system by installing all the basic system software.

M. To start off, we have to first mount some Virtual Kernel File Systems which are /dev, /sys, /run, /dev/pts.

N. After mounting those respective Virtual Kernel File Systems, enter into the chroot environment by typing chroot "$LINUX" /usr/bin/exec and then setting some environment variables like the HOME directory for the chroot environment for example /root and then append /bin/bash –login at the end of the command to change the root to the target disk and login as a root user in that target disk.

O. Now, in chroot we need to create some essential directories /boot, /home, /mnt, /opt, /srv.

P. Then creating some essential Symlinks (Symbolic links) for proper functioning of the system as mentioned in the Linux from Scratch(LFS) Book.

Q. Now it is time to install some basic system software like, Binutils, Coreutils, Diffutils, Findutils, Grep, Gzip, Make, Patch, etc.

R. In this stage of compilation of these packages the password for the root user is set.

S. The installation of these packages are placed in their final location in the Linux chroot system.

T. After the installation of these packages is over, it was time install the Bootloader. We used GRUB (Grand Unified Bootloader) for this purpose.

U. We need to mount the boot partition, in this case is the FAT32 partition, it should be mounted to /boot directory for the bootloader installation. Since target device is a UEFI Compliant Machine, we will be installing GRUB as a UEFI Bootloader and keeping that 1 MB GRUB BIOS Partition for Legacy BIOS Booting, install grub by executing

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165 |

|| Volume 10, Issue 3, March 2022 ||

| DOI: 10.15680/IJIRCCE.2022.1003097 |

grub-install --target=x86_64-efi --efi-directory=/boot --bootloader-id=Aspire. This will install GRUB for an x86_64-efi capable machine, if the machine is using a 32-bit processor then GRUB should be installed for the i386 platform.

V. Now a new kernel needs to be built to boot into the newly built Linux system, some settings need to be modified to boot a UEFI Compliant System. After that is done, adjust some more settings to suit your personal needs, means adding appropriate drivers for your system in the kernel configuration, we configured the kernel by untarring the tarball of the downloaded kernel source and cd into it. We executed make mrproper to clean the kernel source tree and make a fresh kernel by executing make menuconfig, this will launch an Ncurses based menu driven interface which makes it easy to select the features needed to include or exclude or modularize the kernel options. After configuring the kernel, save the configuration file and run make to compile the kernel with the selected features, this will take some time depending on your system speed. After the Kernel is compiled install the Kernel Modules by running make modules_install. Now copy the bzImage file which is in the arch/x86/boot/bzImage to /boot/vmlinuz-linux. Do the same for the .config file and the System.map file.

W. Once it is done, generate the GRUB configuration file by running grub-mkconfig –o /boot/grub/grub.cfg
This will generate a new GRUB configuration file which includes our newly made kernel.

X. The operating system is now complete and it time to boot from it, before that we need to create an /etc/os-release file to identify our OS name, Version, etc.

Y. Now unmount all the Virtual Kernel File Systems and also unmount the target disk from the $LINUX variable we set earlier.

Z. Finally, shutdown your host system and boot from the target disk and there is newly built Aspire Linux operating system.

Now that the Operating System is booted it is time to hint out some features:

a. Less RAM Usage.
b. Using less Disk Space.
c. Less CPU Usage.
d. Command Line Wi-Fi Connection via the iwctl command which is installed by the iwd package.
e. Fast system boot time.

IV. Discussion & Results

To start off, we gathered all the information we needed to start the project, then the host system was decided. Everybody in the group had some clever ways of building the Linux system to which we ended up referring LinuxFromScratch (LFS) book. Each member was assigned to a certain task. One was appointed for gathering the information and creating a roadmap on building this system. Other was appointed for searching and organizing the packages. Then, one was appointed for information on the packages which were downloaded and creating the install scripts for speeding up the installation process.

The final results after booting the Linux system was mind blowing. The RAM usage was as low as 68 MB and used disk space was about 3 GB.

Then we analyzed and improved our Linux System to be more efficient and organized. Hence, we learned that how a Linux System works internally.

V. Conclusion

Finally, after many trials and errors the Linux system is up and running, the takeaway from this project is that we get to know how the kernel works, how the bootloader works and how does it boot the kernel, etc. This poses a major foundation for the development of the Aspire Linux project. This project can be improved to include a Desktop Environments like GNOME, KDE Plasma, XFCE, etc. Also possibly implementing an all new package manager to efficiently install, modify, delete packages, without the need of building everything from source code.

References

1. Linux from Scratch(LFS) Book r11.1-92-systemd by Gerard Beekmans

INNO SPACE
SJIF Scientific Journal Impact Factor

Impact Factor: 8.165

doi cross ref

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

9940 572 462 6381 907 438 ijircce@gmail.com

www.ijircce.com

Scan to save the contact details