CMSC 125 Final Project

Tumulak, Patricia Lexa U.
Valles, Oscar Vian L.

June 2, 2020

Contents

1 Fedora

Fedora is a Linux distribution that is built by the Fedora Project and is sponsored by Red Hat. It advertises itself as an "innovative, free, and open source platform for hardware, clouds, and containers". The Fedora Project offers multiple editions of their operating system based on different use cases. Fedora Workstation is one of these editions. It is used as operating systems for laptops and desktop computers. Fedora Server, as its name suggests, is a carefully crafted version that is tailored for server use. Lastly, Fedora IoT is an Internet of Things (IoT) focused operating system that serves as a backbone for IoT ecosystems ("Get Fedora," n.d.).

1.1 Author(s)

Fedora is built by the Fedora Project and is sponsored by Red Hat. The Fedora Project describes themselves as "a community of people working together to build a free and open source software platform and to collaborate on and share user-focused solutions built on that platform".

The project is made up of different leadership groups based on different functional areas that may change over time. Examples of these groups include the Fedora Council, which is the topmost governing body of the project. It is made up of members from the entire community that are either appointed or elected. Another group is the Fedora Engineering Steering Committee which manages and directs the technical features of the distribution. Various other smaller groups exist, like the Fedora Mindshare Committee which manages community growth and support. These smaller groups are categorized under Working Groups, Subprojects, and Special Interest Groups. ("Fedora Leadership," n.d.)

1.2 History

Fedora Linux started out as a Computer Science academic project by Warren Togami in 2002 at the University of Hawaii. It got its name Fedora, from the hat used in the logo of Red Hat at the time. It was created to distribute additional

software for RHL from a central repository that was managed by community volunteers ("User:Wtogami," 2018).

Fedora Linux was then merged with RHL when Red Hat announced the open development process on July 21, 2003. By then, Red Hat had been developing RHL for nearly 9 years. On November 5, 2003, Red Hat released Fedora Core 1 to the world. Since then, Red Hat bases their enterprise Linux solution, Red Hat Enterprise Linux (RHEL) on release from Fedora.

Since the start of Fedora, two major versions of the operating system have been released every year. To date, there have been 34 versions released, with the 35th version under development. Some notable versions of Fedora include the following: Fedora 7 and Fedora 21 ("History of Red Hat Linux," 2016).

Fedora 7 was released on May 31 2007. This version moved from calling releases of Fedora as Fedora Core. It was shortened to Fedora. This version also marked the first time all of its development was 100% by the community. This was made possible by merging all of the code into a single external repository. This was also the first Fedora release that allowed distribution with Live CD/DVD, making it easier to share and install the operating system ("Announcing Fedora 7 (Moonshine)," 2007)

Fedora 21 was released on December 9, 2014. This was the version where Fedora started creating new flavors of Fedora based on different use cases. These flavors were the following: Cloud, Server, and Workstation. Cloud is the flavor to use in private cloud environments. Server is a flavor used for a wide array of application stacks. It can be used as a web server, file server, and database server, among others. Workstation is optimized for use on desktops and laptops. It includes a built desktop environment, KDE, for ease of use ("Announcing Fedora 21!" 2014). Other releases include general improvements which includes bumping version numbers from upstream packages, updating the Linux kernel to the latest stable version, and adding support for additional architectures.

1.3 Influence

Fedora has greatly influenced other distributions. The following active distributions were released as remixes or modifications of Fedora: Berry Linux, FX64 Linux, MontanaLinux, and Network Security Toolkit. Other remixes have also been created but they have not been updated for a long time. These outdated remixes include: Hanthana, FedBerry, Amahi, and Korora. ("List of Fedora remixes," 2021)

In addition, Fedora has a great influence on the Linux community as a whole. Fedora has used a lot of software first which was then parroted by the Linux community. These include PulseAudio and SELinux. Fedora has also created NetworkManager and D-bus. Now, NetworkManager is the default network configuration tool for various Linux distributions. D-bus is used by GNOME and KDE, the two most commonly used desktop environments today.

1.4 Motivation

The initial release of Fedora Linux was motivated to fix the limited number of packages available in RHL. Since the merge, Fedora has a set of values that guide them throughout the development process. They have named these values the "Four Foundations": Freedom, Friends, Features, First.

Freedom means that the Fedora Project is dedicated to free and open source software and content. This means choosing free alternatives to closed source software and to reduce the number of proprietary code on the project.

Friends means that the project cultivates a strong and caring community that allows anyone to help, as long as these people believe in the same core values that the project believes in. Technical skill is not a prerequisite.

Features means that the project cares about creating and developing excellent software. Fedora aims to create flexible and powerful software that empowers users.

First means that the project is committed to innovation. The project be-

lieves in creating better solutions to the problem first, even if that means long-term stability is compromised. ("Fedora's Mission and Foundations," n.d.)

1.5 Objective and Purpose

The Fedora Council creates short, medium, and long term objectives of the organization. The long term objective of Fedora, according to their mission, is "to create an innovative platform for hardware, clouds, and containers that enables software developers and community members to build tailored solutions for their users." ("Fedora's Mission and Foundations," n.d.)

The current medium and short term objective of the project is to revamp the community outreach teams within Fedora. This goal involves identifying struggling teams and bringing them together under CommOps in a clear structure to better serve the community.

Past medium and short term objectives that the project were able to accomplish include increasing involvement in the educational sector, modularization of the operating system, a flavor specifically for Internet of Things, and minimization. ("Current 12-18 Month Community Objectives," n.d.)

2 openSUSE

openSUSE is a Linux-based operating system and distribution sponsored by SUSE. It is developed by the openSUSE project which is a global community that collaborates on the development of openSUSE (formerly called SUSE Linux), aims to "promote Linux everywhere", and provide and promote free and open source software. The project itself is controlled by the community — almost nothing happens within the operating system without the express permission of the end user. openSUSE also allows users to select packages to include in their flavor of openSUSE which is unique among major Linux distributions. It is entirely free and open source with users able to choose between two major offerings — openSUSE Leap and openSUSE Tumbleweed. Leap is the stable version with releases almost every year and is intended for users who

are relatively new to Linux or those who want a usable and stable OS with regular updates. Tumbleweed is the rolling release that provides the latest stable updates of the software regularly whenever they are available. It is intended for the more advanced Linux users.

2.1 Author(s)

openSUSE is a community project — described as a "do-ocracy" wherein those who do the work are also the ones who decide what happens. Contributions on the OS come from the community. SUSE, the project's primary sponsor, contributes to the project just like any other contributor.

It is self-organized and has three organizational units: the openSUSE Board (which consists of 5 members elected every 2 years and the chairman who is elected by SUSE), Election Officials, and Membership-Officials.

According to Brown, former openSUSE chairman, contributors to the project are expected to collaborate. If there are differing ideas between contributors, they are expected to reach a compromise. Otherwise, if a conflict arises, the board steps in to mediate.

2.2 History

The company SUSE was founded on September 2, 1992 in Germany under the name Gesellschaft für Software- und Systementwicklung mbH (S.u.S.E. GmbH), which means "Software and System Development, Inc.". With their release of S.u.S.E. Linux 1.0 in 1994, SUSE is one of the oldest existing GNU/Linux distributions and is also one of the leading distributions today. SUSE was then acquired by Novell in 2003.

In 2005, Novell launched openSUSE — a community-based Linux distribution following in the footsteps of Red Hat Inc. and its Fedora Project Linux distribution. The openSUSE project was launched with the goal of involving the community more and opening up development.

The first stable release from the openSUSE project was SUSE Linux 10.0 on October 6, 2005 which was released as a freely downloadable ISO image. SUSE Linux 10.1 was subsequently released on May 11, 2006. On their third release, the distribution was renamed from SUSE Linux to openSUSE with openSUSE 10.2. For 10.2, the developers focused on reworking menus that launched KDE and GNOME, making ext3 the default file system, improving power management framework and the package management system.

Until version 13.2 released on November 4, 2014, the project released stable versions with separate maintenance streams from SUSE Linux Enterprise. These versions included but were not limited to updates in GNOME, KDE, the file system from ext3 to Ext4, and Linux kernel version as well as improvements to YaST and other new features.

In 2014, openSUSE Factory, which is the project's development branch, was stabilized enough to become a stable and tested rolling release distribution, called openSUSE Tumbleweed. Tumbleweed is the flagship of openSUSE with new software each day. It forms the base for SUSE Linux Enterprise Server and Desktop (SLES and SLED). Since late 2015, openSUSE split into two main offerings - Tumbleweed and Leap. Leap is the classic stable distribution which uses SUSE Linux Enterprise for the core system. It also has a form of long term support with major service packs released every year and major releases only happening every 3–4 years.

The next latest release is openSUSE Leap 15.3 wherein the repository for Leap and SLE will be merged, sharing the same source code and the exact same binary packages.

2.3 Influence

openSUSE is considered one of the most complete Linux distributions that is available today. It consistently is among the top five downloads in distrowatch.org. openSUSE Leap is particularly popular with Linux developers, system administrators, and software vendors due to its stability and reliability as well as regular stable updates. The branches of openSUSE are also the basis for SUSE Linux Enterprise releases. In 2012, 80% of Linux mainframe

systems ran SUSE Linux and half of the world's largest supercomputer clusters ran SUSE Linux.

2.4 Motivation

openSUSE was first started with the purpose of making a community-based Linux distribution that is both free and open source. This allows individuals to contribute to the project as testers, writers, translators, developers, usability experts, among others.

2.5 Objective and Purpose

The openSUSE project aims to provide easy access to software that is free and open source. It also aims to propagate the use of Linux. Free in "Free Software" refers to freedom and not price. The license on the software created and promoted permits users to study and modify the software however they want. Community members are free to pursue new initiatives and ideas - allowing for innovation.

According to their guiding principles, they aim to "create the best Linux distribution in the world" and to "foster the success of Linux everywhere" with the largest community which also provides the primary source for free software. Another aim is to create a distribution that is stable and easy to use and that is completely multi-purpose for everybody - for both users and developers, for desktop and server use, and for beginners and experienced users.

Aside from developing the Linux based distribution, the openSUSE project also develops tools such as Open Build Service (an open and complete distribution development platform), openQA (an automated testing tool for operating systems), Kiwi (an OS image builder for Linux supported hardware platforms), YaST (a Linux OS setup and configuration tool), and OSEM (an event management app). The project guiding principles also affirms that while doing all this work, it's motto is to "Have a lot of fun".

3 Feature Discussion

3.1 Kernel

Fedora and openSUSE both use the Linux Kernel. Fedora does not deviate from the upstream kernel source. The same is also true for openSUSE.@Therefore, there is no difference between the kernel of both operating systems, other than version differences.

The Linux kernel is a monolithic kernel — it is a large program composed of several components. It can also dynamically load and unload portions of itself on demand — these portions are called modules and are usually device drivers. It is also a preemptive kernel and it supports multithreaded applications. In addition, multiprocessor systems are also supported by the linux kernel. (Bovet & Cesati, 2006, p. 3)

The following sections discuss how the kernel manages the processor, memory, files, resources, and concurrency.

3.2 Processor Management

Both operating systems do not modify how Linux handles processor management, however, the supported processors differ between the two. Fedora primarily supports 64-bit processors, with support for other processors handled by the community ("Architectures," 2019). openSUSE supports both 32-bit and 64-bit processors. ("openSUSE Tumbleweed," n.d.).

In Linux, processes act as entities where resources such as CPU time and memory are allocated to. Linux uses lightweight processes as a method to support multithreading. Lightweight processes share resources and whenever one of these resources modifies a shared resource, the other process is notified of the changes. These processes are bundled together in a thread group. This thread group implements a multithreaded application and acts as a single process when interacting with certain system calls. The state of each process is stored in a process descriptor. (Bovet & Cesati, 2006, pp. 79–81)

Linux has three different mechanisms to create different types of processes with different ways parent and child processes can access resources. (1) Copy On Write enables the parent and child to read the same physical pages. Whenever one process writes to a page, the kernel copies the content into a new page that is assigned to a writing process. (2) Lightweight processes created through `clone()` allows parent and child to share many per-process data. (3) `vfork()` creates a process that shares memory address space of the parent process. The parent process is blocked from execution until the child exits or executes a new program (Bovet & Cesati, 2006, pp. 114–115)

Processes can be destroyed through two ways: `exit_group()` and `_exit()`. `exit_group()` terminates a full thread group while `_exit()` terminates a single process. The kernel is not allowed to discard data until the process terminates. In a case where the parent process terminates before their children, the children become orphans, which are then adopted by the init process. Whenever the init process terminates one of its children, the orphans will also be terminated at this point. (Bovet & Cesati, 2006, pp. 126–127)

Scheduling on Linux is based on exactly what kind of process is being scheduled. There are two types of processes, Real-time and conventional processes (Bovet & Cesati, 2006, pp. 262–263).

Real-time processes are scheduled using two algorithms, First-In-First-Out and Round Robin. The scheduler also takes into account the priority of the process. The process is replaced only if: a higher priority is in the queue, the process performs a blocking operation, the process is stopped or killed, the process yields, or it has exhausted its time quantum (Bovet & Cesati, 2006, pp. 265–266).

Conventional processes are scheduled using Completely fair scheduling (CFS). CFS is different from conventional preemptive scheduling where time slices are taken and priorities are known at the start. In this type of scheduling, the time given to a process is computed dynamically. In this scheduler, a target latency is set, usually 20ms. This target latency is then divided by the entire queue of runnable processes. In essence, each task is given a 1/N ms slice of the processor, where N is the number of runnable processes in the queue. However, this is different from conventional time sharing algorithms because the slice depends on

the number of processes in the queue, making it dynamic. As tasks are removed from the queue, the remaining processes get a greater slice of time.

In addition a nice value is used to weigh the slice. If a process has low-priority nice, it is given a fraction of the 1/N slice, and if it has a greater nice value, it is given a greater fraction of the slice. Do note that the nice value does not determine the 1/N slice, but it only modifies it per process.

Preemption in CFS occurs whenever the weighted slice is finished. However there is a minimum time spent for each process. This minimum time spent is known as the minimum granularity. This limits the amount of context switching that occurs as too much context switching incurs a lot of overhead.

This type of scheduling schedules by thread rather than by process. If a multithreaded application has N threads, then N scheduling actions are taken to schedule each thread. (Kalin, 2019)

3.3 Memory Management

Both operating systems do not modify how Linux handles processor management, however, the supported processors differ between the two. Fedora primarily supports 64-bit processors, with support for other processors handled by the community ("Architectures," 2019). openSUSE supports both 32-bit and 64-bit processors. ("openSUSE Tumbleweed," n.d.).

This means that openSUSE, on a 32-bit processor, has more limitations in accessing and allocating memory. A 32-bit openSUSE install is limited to 4 GB of addressable space and 3.2 GB of RAM. These limitations do not affect 64-bit openSUSE installs and all Fedora installs. ("32 bit vs 64 bit: Key Differences," n.d.)

Linux uses paging with a 4 KB page frame size as the standard memory allocation unit, rather than a 4 MB page size that may be used in some processors. This is because of two reasons: Page Fault exceptions are easily interpreted, and data transfer between main memory and disk are more efficient when page size is small. (Bovet & Cesati, 2006, pp. 294–295)

In cases where memory access time is not uniform, Linux supports the Non-Uniform Memory Access (NUMA) model. This type of model is used in some architectures, like MIPS, which both Fedora and openSUSE supports. (Bovet & Cesati, 2006, pp. 297–298)

Linux also manages memory in zones. There are three zones, `ZONE_DMA`, `ZONE_NORMAL`, and `ZONE_HIGHMEM`. These three zones were created since some architectures limit the way page frames can be utilized. Old Direct Memory Access (DMA) processors for ISA buses are only able to address the first 16MB of RAM and 32-bit computers cannot access all physical memory locations due a limitation in the size of the linear address space (Bovet & Cesati, 2006, p. 299). These zones are used by the Zoned Page Frame Allocator. This component handles requests for allocation and deallocation of dynamic memory. (Bovet & Cesati, 2006, p. 302)

To reduce external fragmentation, Linux uses the Buddy System Algorithm where free page frames are grouped into 11 lists of blocks which contain groups of 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, and 1024 contiguous page frames. These blocks are used to find the best possible fit for the requested memory. For example, 16 continuous page frames are requested. The algorithm goes to the 16-page-frame list and searches for free blocks in the list. If none are found, the algorithm goes to the next list, the 32-page-frame list. This continues on until a suitable block is found. From this, it can be determined that this algorithm is an extension of the Best-Fit algorithm. Whenever a block is allocated, the remaining free page frames are not left as internal fragments. Rather, these contiguous sets of blocks are then placed back into the lists, depending on the size left. If 32 page frames were requested and the only contiguous set of blocks are found in the 128 list, the remaining 96 frames are broken down into 64 continuous page frames and 32 continuous page frames. These are then inserted into the corresponding list.

Once frames are released, the kernel tries to merge pairs of contiguous free blocks of size b to form a single block, 2b. Once a pair, or a buddy, has been found, the pair finds another pair that is of the same size as them and merges together. The process is repeated until no pairs have been found and the largest contiguous page frames are formed. (Bovet & Cesati, 2006, pp. 311–312)

Swapping is also a feature built into the Linux kernel. The swapping subsystem does the following functions: Setup swap areas, manage the space on swap areas, manage swapping in and out of RAM, and track swapped pages. (Bovet & Cesati, 2006, p. 713)

3.4 File Management

File management systems in both Fedora and openSUSE are, in essence, the same since they both are built on the Linux kernel. Linux file management is built in a single hierarchical directory structure. The structure begins with a root directory, denoted by the forward slash /, which then expands into subdirectories. All these other directories are descendants of root. ("An intro to Linux file system management," n.d.)

File protection and directory protection in Linux systems are handled similarly as well. Three fields — owner, group, and universe — are associated with each file and directory. These fields consist of three bits "rwx" — r for read access, w for write access, and x for execution access. Each field separately indicates the permissions of the file owner, the file's group, and all the other users using these three bits. Groups can only be created by the manager of the facility or by any superuser. (Silberschatz et al., 2018, pp. 553–554)

Though both are similar, Fedora's default and recommended system is Ext4 while openSUSE Leap's default file system is Btrfs. There are pros and cons to each file system.

Most Linux distros use Ext4 as the default file system. It is a very robust file system with impressive limits — the largest volume/partition the Ext4 can make is 1 exbibyte and the largest file size is 16 tebibytes — but it is, however, built on an aging code base. Ext4, like most file systems, also uses journaling wherein changes to the file system are written sequentially to a journal. Despite these features, it does not support transparent compression, transparent encryption, data deduplication, or filesystem snapshots. (King, 2020)

On the other hand, Btrfs — which can be pronounced as "Butter FS" — is a newer system described as a Copy-on-Write (CoW) file system.This is a

resource management technique wherein when a process creates a child, it can exist as a reference to the original data. Only once the child is modified is an actual copy created and these modifications are only written on the copy ("What is Copy-on-write?" 2019). The main difference between Btrfs and Ext4 is that Btrfs is capable of addressing and managing more files, larger files, and larger volumes than the Ext2, Ext3, and Ext4 file systems. It was made because developers wanted to expand the functionalities of a modern file system to include snapshots and checksums, among others. (King, 2020)

Snapshots are an important feature for a filesystem to have. Before doing any risky modification, this feature allows you to take a snapshot of your filesystem so that if an error occurs down the line, a snapshot of this early state of your filesystem is available to restore. (Shovon, 1969)

Checksum is a block of data that is derived from another block of data to enable error detection for a file or during a data transfer. The data is produced by implementing a checksum algorithm to a block of data or a data packet. The algorithm returns a value which is appended to the data packet (usually at the end) and the receiver recalculates the checksums using the incoming data and compares the two values. If there is a discrepancy between the values, this means that an error occurred during data transmission. ("Isaac Computer Science," n.d.)

There is a clear advantage between Btrfs and Ext4. It was designed to have more useful functionalities for a modern filesystem and it was built for high-capacity and high-performance storage. However, there are still some pros of Ext4 over the other. Despite being built on an aging code base, Ext4 is still the most commonly used filesystem among Linux distros. This is because it has proven to be stable, resilient, and reliable despite its simplicity. Due to journaling support, data is kept safe even when there is an unexpected power failure.

Though these are the defaults, both operating systems still support other file systems such as Fedora still being capable of supporting Ext3, Ext2, and Btrfs. This is also the same case for openSUSE with the addition of FAT, XFS, Swap, and UDF. ("5.4.10.5. Device, File System and RAID Types," n.d.; "Expert Partitioner: Reference: openSUSE Leap 15.2," n.d.; "SUSE Linux En-

terprise Server 12 SP4: Chapter 1. Overview of File Systems in Linux (Storage Administration Guide)," n.d.)

3.5 Resource Management

Both Fedora and openSUSE share a way to manage resources called control groups or cgroups for short which is a feature of the Linux kernel. Cgroups are a collection of processes which can be bound to a set of parameters via the cgroup system. It allows for allocating resources such as CPU time, network bandwidth, and system memory among processes that are running on a system. Cgroups give system administrators fine-tuned control over allocation, prioritization, managing, and monitoring of system resources. This includes monitoring usage of various types of resources, not allowing cgroups access to some resources, and reconfiguring cgroups dynamically running on a system. ("cgroups(7) - Linux manual page," n.d.; "Chapter 1. Introduction to Control Groups (Cgroups)," n.d.)

Cgroups are organized hierarchically and child cgroups inherit some of the attributes of their parent cgroup. The cgroup model is "one or more separate, unconnected trees of tasks or processes". It is different to a single tree of processes such that there can be many different cgroups hierarchies simultaneously on a given system.("Chapter 1. Introduction to Control Groups (Cgroups)," n.d.)

3.6 Concurrency Management

Concurrency management for Fedora and openSUSE is also essentially the same since they share the same kernel. Multitasking in operating systems lets multiple threads execute in parallel to optimize system performance. However, if this is not implemented cautiously, it could lead to issues such as processes sharing the same resources at the same time. This is called the critical section of the process. Processes should be able to use the same resource but only one process at a time. In order to address these concurrency issues that could arise from multitasking, Linux kernel uses two synchronization mechanisms called Mutex (Mutual Exclusion Object) and Semaphore. ("Difference between Mutex and

Semaphore in Operating System," n.d.; Tewani, 2015)

Mutex is used to allow only one process at a time to have access to a specific resource. This allows all processes to use this resource but only one at a time. In order to handle the critical section issue, Mutex uses the lock-based technique. When a process requests to use a specific resource, a mutex object will then be generated by the system with a unique ID. This then allows the process to occupy a lock on the object whenever it wishes to use the resource. Once the process is finished it releases the mutex object and other processes can then create the mutex object and use it in the same manner. Locking the object allocates that resource to the specific process and other processes are essentially "locked out" and cannot use that resource.

Semaphore is used for process synchronization wherein an integer variable S is initialized with the number of resources present in the system. Two functions, such as `wait()` and `signal()`, are then used to modify the value of this variable but only one process at a time is allowed to change this value. The two categories of semaphores are counting semaphores and binary semaphores.

Counting semaphores have the semaphore variable initialized with the number of resources available. This is so that whenever a process takes a resource to use it, the value of the semaphore variable is decreased by one by the `wait()` function. After the resource is released, the `signal()` function then increases the value of the semaphore by 1 again. When the semaphore variable value reaches 0 this means that all resources are used and there is none left to be used. A process that wishes to use a resource has to wait for its turn.

Binary semaphores are implemented through setting the semaphore value to 0 or 1. It is first initialized to 1 and if a process requests some resource, the variable value is then changed to 0. When the resource is released, the value is then increased to 1. This is similar to Mutex wherein a process cannot use a certain resource at a particular instant of time when the value of the semaphore is 0 and has to wait for the process using the resource to release it. No locking is performed however. ("Difference between Mutex and Semaphore in Operating System," n.d.)

4 Conclusion

The main difference between the two operating systems lies in userland applications, and supported architecture. The underlying kernel that runs the major components of the operating system of the two systems is the same, the Linux kernel. However, there are minor differences between the two in the way they configure the kernel because Fedora does not support 32-bit processors while openSUSE does. This results in less addressable space and supported RAM capacity for 32-bit openSUSE. In addition, the default file systems for both operating systems are different. Fedora recommends ext4, while openSUSE defaults to btrfs. Other than those differences, the rest of the kernel remains the same for the two.

Because of these, choosing one operating system over the other depends on the following considerations: hardware, use case, preference of userland applications, and release styles. If the user's hardware is 32-bit, then they have no choice but to use openSUSE, otherwise if they have 64-bit processors, both operating systems are available. If the user's main use case is for cloud, or IoT, Fedora has an edge with their specific editions. For general server and desktop use, both operating systems would be good choices. Both operating systems have more or less the same applications, however the package manager between the two differs. Fedora uses dnf, while openSUSE uses YaST and ZYpper. Finally, the stability of the operating system differs between the two. openSUSE uses a rolling release model, while Fedora releases biannually. If stability is more important, Fedora takes the lead. If getting the latest version of software is more important, openSUSE takes the edge.

All in all, there really is not a lot of difference between Fedora and openSUSE when the way they manage the different resources of a computer is compared. Choosing one operating system over the other mostly comes down to user preference and available hardware.

References

32 bit vs 64 bit: Key differences. (n.d.). https://www.guru99.com/32-bit-vs-64-bit-operating-systems.html

5.4.10.5. device, file system and raid types. (n.d.). https://docs.fedoraproject.org/en-US/Fedora/25/html/Installation_Guide/sect-installation-gui-manual-partitioning-filesystems.html

Announcing fedora 21! (2014). https://lists.fedoraproject.org/pipermail/announce/2014-December/003242.html

Announcing fedora 7 (moonshine). (2007). https://listman.redhat.com/archives/fedora-announce-list/2007-May/msg00009.html

Appendix d: History and background - what is free software and open source and what is the history of opensuse. (n.d.). http://opensuse-guide.org/history.php

Architectures. (2019). https://fedoraproject.org/wiki/Architectures

Both, D. (2020). An introduction to swap space on linux systems. https://opensource.com/article/18/9/swap-space-linux-systems

Bovet, D. P., & Cesati, M. (2006). *Understanding the linux kernel*. O'Reilly.

Cgroups(7) - linux manual page. (n.d.). https://man7.org/linux/man-pages/man7/cgroups.7.html

Chapter 1. introduction to control groups (cgroups). (n.d.). https://docs.fedoraproject.org/en-US/Fedora/16/html/Resource_Management_Guide/ch01.html

Chauhan, M., & Chauhan, M. (2012). Half of the world's largest supercomputer clusters run suse linux: Tfir: Interviews, news analysis by swapnil bhartiya. https://www.tfir.io/half-of-the-world-s-largest-supercomputer-clusters-run-suse-linux/

Current 12-18 month community objectives. (n.d.). https://docs.fedoraproject.org/en-US/project/objectives/

Difference between mutex and semaphore in operating system. (n.d.). https://afteracademy.com/blog/difference-between-mutex-and-semaphore-in-operating-system

Expert partitioner: Reference: Opensuse leap 15.2. (n.d.). https://doc.opensuse.org/documentation/leap/reference/html/book-opensuse-reference/cha-expert-partitioner.html

Fedora leadership. (n.d.). https://docs.fedoraproject.org/en-US/project/leadership/

Fedora's mission and foundations. (n.d.). https://docs.fedoraproject.org/en-US/project/

Get fedora. (n.d.). https://getfedora.org/

History of red hat linux. (2016). https://fedoraproject.org/wiki/History_of_Red_Hat_Linux

An intro to linux file system management. (n.d.). https://www.rs-online.com/designspark/an-intro-to-linux-file-system-management

Isaac computer science. (n.d.). https://isaaccomputerscience.org/concepts/net_comm_error_checking

Kalin, M. (2019). Cfs: Completely fair process scheduling in linux. https://opensource.com/article/19/2/fair-scheduling-linux

King, B. (2020). Ext4 vs. btrfs: Which linux file system should you use? https://www.makeuseof.com/tag/ext4-btrfs-making-switch-linux/

List of fedora remixes. (2021). https://fedoraproject.org/wiki/List_of_Fedora_remixes

Opensuse tumbleweed. (n.d.). https://get.opensuse.org/tumbleweed/

Opensuse:guiding principles. (n.d.). https://en.opensuse.org/openSUSE:Guiding_principles

Shovon, S. (1969). The comparison of btrfs vs ext4 filesystems. https://linuxhint.com/btrfs-vs-ext4-filesystems-comparison/

Silberschatz, A., Galvin, P. B., & Gagne, G. (2018). *Operating system concepts*. Wiley.

Slashdot. (n.d.). https://linux.slashdot.org/story/05/10/06/2157223/suse-100-oss-released

Suse. (2018). Opensuse. https://susedefines.suse.com/definition/opensuse/

Suse linux enterprise server 12 sp4: Chapter 1. overview of file systems in linux (storage administration guide). (n.d.). https://documentation.suse.com/sles/12-SP4/html/SLES-all/cha-filesystems.html#sec-filesystems-major-btrfs

Tewani, P. D. (2015). https://sysplay.in/blog/linux-kernel-internals/2015/06/concurrency-management-in-linux-kernel/

User:wtogami. (2018). https://fedoraproject.org/wiki/User:Wtogami?rd=WarrenTogami

What is copy-on-write? (2019). https://www.computerhope.com/jargon/c/copy-on-write.htm

5 Appendix

5.1 Source Code

This project was written in LaTeX. The source code of this document can be found at GitHub: https://github.com/ohhskar-school/cmsc-125-final-project. The link for Overleaf can be found here: https://www.overleaf.com/project/60b74ef310d0ee358da40652

5.2 Division of Task

5.2.1 Tumulak, Patricia Lexa U.

- openSUSE
- File Management
- Resource Management
- Concurrency Management

5.2.2 Valles, Oscar Vian L.

- Fedora
- Kernel
- Processor Management
- Memory Management
- Conclusion