STM JOURNALS

Journal of
Operating Systems
Development & Trends

ISSN: 2454-9355
Volume 10, Issue 2, 2023
DOI (Journal): 10.37591/JoOSDT

http://computers.stmjournals.com/index.php?journal=JoOSDT&page=index

Review JoOSDT

Quality of Services (QoS) in Linux Operating System

Mohammad Salem Hamidi[1,*], Fahim Nasiri[2], Wisal Ahmad Zai[3], Himal Rahman Zai[3], Aminullah[3], Mohammad[3]

Abstract

Quality of Service (QoS) is a fundamental characteristic of contemporary computer networks, enabling the prioritization of network traffic and the allocation of network resources according to specific criteria. The Linux operating system provides a powerful set of QoS tools, including Traffic Control, Differentiated Services, and Network Emulator. These utilities enable you to establish traffic categories, impose bandwidth constraints, and give precedence to traffic according to diverse criteria such as IP address, port number, protocol, and packet size. In addition, the Netem module allows you to emulate various network conditions for testing and debugging network applications and services. In summary, Quality of Service (QoS) functionalities within Linux offer a versatile and effective system for overseeing network traffic, guaranteeing that essential applications and services obtain the required bandwidth and resources for optimal operation.

Keywords: QoS, Linux, traffic control, Traffic Control (TC), differentiated services, Diffserv, network emulator, Netem module

INTRODUCTION

Quality of Service (QoS) is a critical feature in modern computer networks that allows network traffic to be prioritized and network resources allocated based on specific criteria. On a network, some applications and services require a higher priority than others to function properly. As an illustration, real-time applications like video conferencing and Voice over IP (VoIP) demand a dependable and steady network connection characterized by minimal latency and negligible packet loss. On the other hand, less critical applications such as file transfer or web browsing can tolerate higher latency and packet loss [1].

*Author for Correspondence
Mohammad Salem Hamidi
E-mail: sshamidi13@gmail.com

[1]Ph.D. Scholar, Vivekananda Global University, Jaipur, Rajasthan, India, and Vice Chancellor, Jahan University, Kabul, Afghanistan
[2]Lecturer, Department of Computer Science, Jahan University, Kabul, Afghanistan
[3]Student, Department of Computer Science, Jahan University, Kabul, Afghanistan

Received Date: July 28, 2023
Accepted Date: October 03, 2023
Published Date: October 21, 2023

Citation: Mohammad Salem Hamidi, Fahim Nasiri, Wisal Ahmad Zai, Himal Rahman Zai, Aminullah, Mohammad. Quality of Services (QoS) in Linux Operating System. Journal of Operating Systems Development & Trends. 2023; 10(2): 1–5p.

THE GOAL

QoS in Linux can be achieved using various techniques such as traffic shaping, packet scheduling, and congestion control. These mechanisms allow network administrators to allocate bandwidth and prioritize traffic based on criteria such as source and destination addresses, application type, and user identity.

LITERATURE REVIEW

Quality of Service (QoS) is an important aspect of any operating system, including Linux. QoS

refers to the system's ability to prioritize certain types of traffic or applications over others based on their importance or criticality. In this literature review, we will explore some of the research that has been done on QoS in Linux operating systems [2].

RESEARCH METHODOLOGY

QoS (Quality of Service) in the Linux operating system is a set of features that allow you to prioritize network traffic and allocate network resources based on specific criteria. These features can be used to improve the performance of critical applications and services, such as VoIP, video conferencing, and online gaming, while ensuring that less critical traffic does not consume all available bandwidth [3].

QoS in Linux Operating System

In Linux, QoS (Quality of Service) refers to the ability to prioritize and control network traffic to ensure that certain types of traffic receive more bandwidth and are served with higher priority than others. This holds significance in networks where various types of traffic contend for restricted network resources. Using QoS in Linux can help ensure that critical network traffic, such as VoIP or video conferencing, receives sufficient bandwidth and is delivered with low latency, while less critical traffic, such as file transfers or email, is given a lower priority. In Linux, Quality of Service (QoS) is a mechanism for managing and prioritizing network traffic. It allows the allocation of network resources based on the importance of traffic and ensures that critical traffic is given higher priority than less important traffic. QoS in Linux is typically implemented using various queuing disciplines that define how packets are queued and prioritized in the network stack. These queuing disciplines can be configured to prioritize traffic based on various criteria.

Examples of these criteria include the protocol used, the source or destination IP address, or the port number. Quality of Service (QoS) is crucial in networks where diverse forms of traffic are present. vying for constrained network assets like bandwidth and computational capacity. By prioritizing traffic based on its importance, QoS can help ensure that critical network applications, such as VoIP or video conferencing, receive sufficient bandwidth and are delivered with low latency, while less important traffic, such as file transfers or e-mails, will receive less preference. In Linux, QoS can be implemented using a variety of tools and techniques, including the Traffic Control (TC) command-line utility, network schedulers, and various queuing disciplines such as class-based queuing (CBQ) and differentiated service (DiffServ) [4].

Types of QoS in Linux Operating System

Different types of QoS in Linux operating system are as follows.

1. Traffic shaping;
2. Traffic police;
3. Queuing disciplines;
4. Differentiated services;
5. Queuing by class; and
6. Valued fair queuing.

Forming Traffic

This involves regulating the volume of traffic transmitted or received via a network interface. Traffic shaping can be used to limit the amount of bandwidth used by specific types of traffic, such as file transfers, to ensure that more critical traffic such as VoIP or video conferencing receives sufficient band [5].

Traffic Police

This includes monitoring network traffic and enforcing predetermined limits on the amount of traffic that can be sent or received. Utilizing traffic policing can help avert network congestion and safeguard critical traffic from being impacted by less essential traffic.

Sorting Disciplines

This includes controlling how packets are queued and prioritized in the network stack. Various queuing methods can be employed to give precedence to traffic based on distinct criteria, including protocol, source or destination IP address, or port number [6].

Differentiated Services

This is a standard for QoS that uses a code point in the IP header to classify and prioritize packets. DiffServ allows the definition of multiple classes of service, each with its own QoS requirements.

Queuing by Class

This is a queuing discipline that allows hierarchical classification and prioritization of packets based on various criteria such as source or destination IP address, protocol or port number.

Dear Fair Queuing

This is a queuing discipline that fairly allocates bandwidth between different traffic flows. WFQ can be used to ensure that each traffic flow receives a fair share of the available bandwidth [7].

HOW DOES QOS WORK ON LINUX?

In Linux, Quality of Service (QoS) is a mechanism for managing network traffic by setting priorities for different types of traffic. The objective of Quality of Service (QoS) is to guarantee that vital traffic, like real-time applications such as voice and video, is given higher priority than less important traffic such as file downloads. QoS in Linux is implemented using the Traffic Control (TC) subsystem, which is part of the Linux kernel. The TC command is used to configure QoS settings for network devices and interfaces. The TC subsystem uses various techniques to prioritize network traffic, such as traffic shaping, traffic control [8].

QOS IN THE LINUX OPERATING SYSTEM TECHNIQUES

There are several QoS techniques used in the Linux operating system to manage network traffic. Some of the most common techniques include:

- *Traffic shaping:* Traffic shaping is a technique used to control the rate of outgoing traffic from a network interface. The mechanism functions by either postponing or storing packets to restrict the volume of data transmitted across the network. This approach is beneficial for guaranteeing that essential traffic, like real-time applications, obtains the necessary bandwidth to operate effectively.
- *Traffic police:* Traffic policing is a technique used to control the rate of incoming traffic on a network interface. It works by dropping or marking packets that exceed a predetermined rate or size limit. This technique is useful for preventing network congestion and ensuring that important traffic is prioritized over less important traffic [9, 10].
- *Labeling of packets:* Packet marking is a technique used to identify packets with different priority levels. Achieving this involves affixing a distinct tag to the packet header, designating its priority level. Packet marking proves valuable in ensuring that critical traffic receives preference over less significant traffic.
- *Queue management:* Queuing is a method employed to control the sequence in which packets are dispatched across a network. It works by prioritizing packets based on their priority level and queuing accordingly. This technique is useful for ensuring that critical traffic is transferred before less important traffic.
- *QoS by class:* Class-based QoS is a technique used to classify traffic into different classes based on its characteristics. Each class is then assigned a priority level, which is used to determine how traffic is managed. This technique is useful for ensuring that traffic is managed in a granular and flexible manner.

QoS (QUALITY OF SERVICE) FRAMEWORKS

Quality of Service (QoS) frameworks offer a comprehensive strategy for the administration of network traffic, prioritizing critical traffic over less significant traffic. Some of the most common QoS frameworks include:

- *Differentiated services:* It is a Quality of Service (QoS) framework that employs packet marking to categorize traffic into distinct classes according to its significance. Each class is assigned a DSCP (Differentiated Services Code Point) value that is used to prioritize network traffic.
- *Integrated services:* Integrated services is a QoS framework that uses the Resource Reservation Protocol (RSVP) to reserve resources such as bandwidth or buffer memory for specific applications or traffic types. This method proves beneficial in guaranteeing that essential applications have the necessary resources to operate effectively. COPS is a QoS framework that provides a policy-based approach to network traffic management. It works by allowing network administrators to define policies that determine how network traffic should be managed.
- *Multiprotocol label switching:* MPLS is a QoS framework that uses labels to identify and prioritize network traffic. This technique is useful for ensuring that critical traffic is prioritized over less important traffic.
- *Application layer topic optimization:* ALTO is a QoS framework that uses network topology information and traffic statistics to optimize traffic flows between applications.

CONCLUSION

The quality of service in the Linux operating system is generally considered to be very high. Linux is known for its stability, reliability and security, making it a popular choice for servers and other mission-critical applications. One of the key factors contributing to the quality of service in Linux is the open-source nature of the operating system. This allows developers and users to access the source code and make modifications or improvements to the system as needed. A community-driven development model also encourages collaboration and knowledge sharing, which can lead to better software quality. Another factor is the wide variety of tools and utilities available in Linux for monitoring and managing system performance. These tools can help administrators quickly identify and fix problems and ensure that the system remains stable and reliable.

REFERENCES

1. Hertzog R, Mas R. The Debian Administrator's Handbook. Sorbiers: Freexian SARL. 2013.
2. Elvis Pfützenreuter PU5EPX. (2016). Internet Quality of service (QoS). [Online]. EPx. Available from: https://epxx.co/artigos/qos_en.html
3. E-tutorials. (2023). Policy Routing: Chapter 13. Policy Routing, Bandwidth Management, and QoS: Integrated cisco and Unix network architectures: Networking: eTutorials.org. [Online]. Available from: https://etutorials.org/Networking/Integrated+cisco+and+unix+network+architectures/Chapter+13.+Policy+Routing+Bandwidth+Management+and+QoS/Policy+Routing/
4. e-Zest Solutions. (2023). Linux Services and Linux Consulting. [Online]. E-zest.com. Available from: https://www.e-zest.com/linux-services
5. Loscocco P, Smalley S. Integrating flexible support for security policies into the Linux operating system. In 2001 USENIX Annual Technical Conference (USENIX ATC 01). 2001.
6. Cotroneo D, Natella R, Pietrantuono R, Russo S. Software aging analysis of the Linux operating system. In 2010 IEEE 21st International Symposium on Software Reliability Engineering. 2010 Nov 1; 71–80.
7. Xiao G, Zheng Z, Yin B, Trivedi KS, Du X, Cai KY. An empirical study of fault triggers in the Linux operating system: An evolutionary perspective. IEEE Trans Reliab. 2019 Jun 5; 68(4): 1356–83.
8. Wang YJ. A summary of research on the Linux operating system. Adv Mater Res. 2013 Jul 23; 705: 565–8.

9. Al-Rayes HT. Studying main differences between android & Linux operating systems. Int J Electr Comput Sci (IJECS-IJENS). 2012 Oct; 12(05): 46–9.
10. Boras M, Balen J, Vdovjak K. Performance evaluation of Linux operating systems. In 2020 IEEE International Conference on Smart Systems and Technologies (SST). 2020 Oct 14; 115–120.

