International Journal of Advanced Computer Technology (IJACT)

ISSN:2319-7900

# LINUX NETWORK ACCELERATOR FOR MULTICAST PROTOCOL

Mrs. Mayura Kinikar, Assistant Professor ,Department of computer Engineering, MIT AOE Alandi(D),Pune; Mrs. Minakshi N.Vharkate , Assistant Professor ,Department of computer Engineering, MIT AOE, Alandi(D),Pune.

## Abstract

This paper is aimed to measure the performance of Linux Network stack for Multicast protocol. Aim of this paper is to find out whether we can accelerate the rate of packet Transmission. The Multicast protocol can be used well for skipping some of the Linux Network Stack levels so as to reduce the possible CPU overhead and to speed up the transmission.

By using designed architecture of Multicast Protocol and possible changes within it, Linux Network Stack can be improved for its efficiency and possible alternatives for fast developing technology can be analyzed. Number of possible approaches to attain maximum possible speed with optimal use of CPU can be identified and the one which can be completed by Multicast architecture will be implemented.

Keywords: Infiband (IB), Multicast, Stack Bypass, Linux Kernel

## Introduction

OSI and TCP/IP are the two standard architectures used for network programming which makes use of different communication protocols which provides them a modularized approach for programming. Each of these protocols or layers of reference models provide specific functionalities which altogether contributes to the final flow of data.

In most of the cases, communication doesn't require all the functionalities of any layer and need only few of them, but as protocols are restricted to follow set of rules defined by model, it's hard to make changes in existing model. And hence there is need to design a new sub-model based on these standard model which will provide a specific functionality by removing unnecessary steps required in previous model.

Thus project idea revolved around this thing and we decided to work for the speed of communication over single LAN which may be expanded later for subnets. The main motivation for the project is from Infiniband Protocol (IB) which also aims at the same objective i.e. to accelerate the data over a network[3]. Infiniband trade association was formed with seven Industry leaders namely Compaq, Dell, HP, IBM, Intel, Microsoft and Sun.

The Architecture of IB is combined effect of Software and Hardware advancements. The primary structure of IB includes changes in wires used for networking of Computers and also with the connectors. Special kind of hardware is required by this technology known as Host Channel Adapter (HCA) and/or Target Channel Adapter (TCA). This hardware provides interaction of machine with IB Fabric which consists of switches, routers, links, etc. If the software changes, this architecture skips some of the network stack layers and passes the enclosed data packets to lower layers which are explicitly handled by IB hardware.

Being apart from hardware changes, we are a bit impressed by the concept of '**Skipping Network Stack Layers**'. This means that, we can design new ways for how packet must go through the network stack. We can avoid few steps from the standard architecture so that at the cost of skipping these steps we can speed up the network. So the general idea of project comes from the software structure of IB.

## Stack Bypass and Multitasking

### A. Network stack Bypass

Implementation of network stack bypass concept by skipping some of the network stack levels while transmission of packets through multicast protocol, we reduce many system calls, so system will accelerate the packet transmission rate.

The concept of Stack Bypass comes into picture as we will be providing different implementation for Standard stack as shown by arrow. In this technique, we will be passing Network packets from Application Layer (or User space) to direct Network Access Layer (in our case is Ethernet). During this step, we will be providing an equivalent implementation as that of UDP-IP layers in one pass only so that the two pass, each by UDP and IP can be reduced to one pass and thus time required can be reduced.

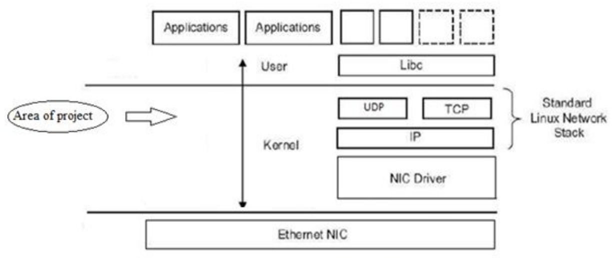

**Figure 1: Area of Project**

Network stack bypass is achieved by taking certain packets from application layer and directly transmitted over the Ethernet LAN which will transmit these packets to other machine.

## B. Multicasting

By using designed architecture of Multicast Protocol and possible changes within it, Linux Network Stack can be improved for its efficiency and possible alternatives for fast developing technology can be analyzed. Consider only multicast protocol we are transmitting packets to group using multicasting mechanism.

In multicasting there is at least one sender and several receivers (group of receivers called multicast group).In multicast routing, the router may forward the received packet through several of its interfaces.

Why Multicast Protocol?
Multicasting is a process when there is a single sender and multiple receivers of a message. This primarily involves the use of duplication of packets at routers which is capable of multicasting. TCP involves one-to-one dedicated connection establishment rules for communication and hence it is not considered to be a multicast protocol[5]. So in this project we worked on UDP/IP stack so that it will not require doing much of work for maintaining reliability of message.

## Implementation Approach

To achieve these goals we made some changes in the Linux kernel. By designing new data structure for the building and transmission of packets from application layer to directly on hardware through our modified module we achieved these goals of the project. For this we used different machines with same Ethernet hardware.

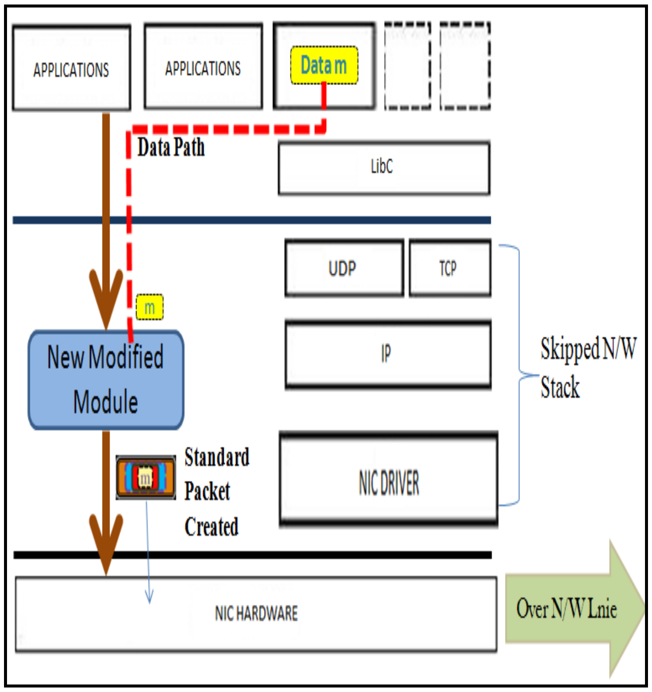

**Figure 2: Implementation Approach**

## A. Algorithmic Strategy

1. Receive file containing data to be transmitted.
2. Open file and read file format header.
3. Divide data into standard size blocks.
4. Select one packet serially.
5. Read Packet Stream
6. Build UDP IP equivalent header.
7. Add packet data after header
8. Compute checksum of composed packet
9. Build the frame header value which contains UDP IP headers of packet.
10. Pass Packet over NIC for transmission.
11. Repeat step 4-10 until all data being transmitted.

## B. Performance Improvement of Network Stack

The project is made to measure the performance of the Linux network stack when certain packets are transmitted to other machine through alternate mechanism, by designing new data structure instead of using available structures of UDP, IP, and Ethernet. New structure is designed in which we filled all information in single structure. So by making single structure and used only once by CPU we reduced CPU overhead and decreased the network stack load so there is great improvement in the network stack.

## C. Reducing CPU Overhead

Accelerating the packets through network stack we reduced CPU overhead and speed up the transmission. In this we skipped some network stack level so as to speed of network increased and by skipping the stack level and transmitting packet with single pass we reduces CPU overhead for multicast protocol. While sending packets from the application layer through network stack over physical media there will be much CPU overhead. Hence more time can be taken by the CPU to process this task. Aim of this project is to reduce the CPU overhead by skipping some of the network stacks which gives the speed for transmission of the packet.

To achieve these goals some changes are made in the Linux kernel. By designing new data structure for the transmission of packets from application layer to directly on hardware though modified module, we can achieve these goals of the project. For this we used different PCs with same Ethernet hardware.

## Result Analysis

Figure 3 shows result analysis of project. In this, first standard stack is used to send text file to multicast group with standard module through standard network stack levels. The readings of which are shown in Blue line and user program used for its execution is "regular.c".

User program "noqueue.c" uses project module to skip standard network stack and pass packet directly on driver. Orange Line with "noqueue.c" shows less time than standard module without use of any internal buffer for storing overflowing packets. Hence this shows acceleration of packets.

Third i.e. Grey line shows readings for "queue.c" uses project module with buffer maintained inside, which prevents packet drops. Hence this result shows how packets are accelerated.

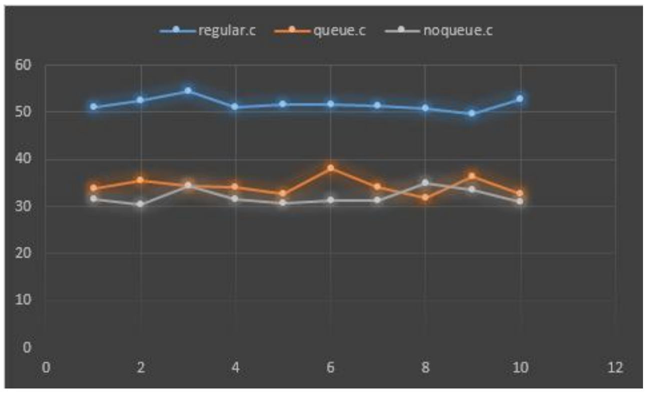

**Figure 3: Time Chart**

Figure 4 shows average time taken by standard module and project module. First bar with time 51.78 µs is average time for standard module with standard stack.

Second bar with time 34.38 µs is project module and third bar with time 32.11 µs is project module with use of internal buffer. Less time is required for project module hence there is acceleration of packets.

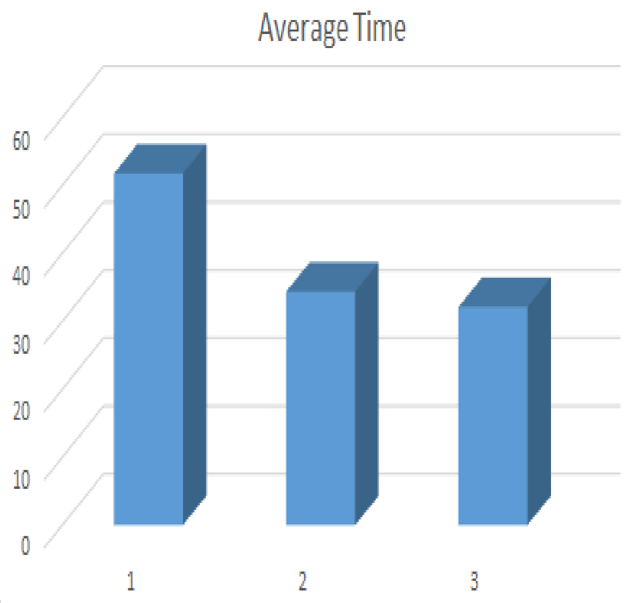

**Figure 4: Average time**

## Conclusion

Network acceleration is very important while communicating with each other through the network. This project shows the acceleration of certain Ethernet packet for multicast protocol through the Linux network stack over the network. The concept of bypassing the network stack by skipping some of the network stack levels shows that this project is best option for multicast protocols like UDP. So this optimizes the UDP multicasting mechanism.

While sending the packets through the network stack there is more CPU use so there will be more utilization of CPU for packet transmission. So accelerating the network stack, CPU overhead will be reduced so this leads to increase in the transmission speed. So less CPU utilization is achieved for transmission of the packets which is more important task while communicating over the network. So CPU can be utilized for other processes in the system.

## References

[1] TCP-IP Network Stack Performance in Linux Kernel, OLS 2002, V. Anand & B. Hartner

[2] Path of the Packet in Linux Kernel Stack, University of Kansas 2005, Ashwin Chimata

[3] Introduction to Infiniband, Infiniband Trade Association, Paul Grun

[4] TCP Bypass, Informatix Solutions 2010.

[5] Multicasting & Multicasting Routing Protocols, Tata McGraw-Hill 2000.

[6] Network Handbook Protocol, Dr. Ke Yan, Chief Architect of Juniper Networks.

[7]OReilly - Ethernet Definitive Guide, by Charles E. Spurgeon 2000.

[8] The Linux TCP/IP Stack: Networking for Embedded Systems byTHOMAS F. HERBERT

## Biographies

**MRS. MAYURA KINIKAR** received the BE(Computer Science & engineering) degree from the University of Dr. B.A.M.U. Aurangabad , Maharashtra, in 2000, the M.E(Computer Science & Engineering)degree from the University of Dr. B.A.M.U. Aurangabad, Maharashtra, in 2006. Currently, she is an Assistant Professor of Computer Engineering at MIT AOE Alandi (D) Pune in University of Pune. Her teaching and research areas include Database management, data mining, and operating systems.

**MRS. MINAKSHI N. VHARKATE** received the BE (Computer Science & engineering) degree from the University of Dr. B.A.M.U. Aurangabad , Maharashtra, in 1999 the M.E. degree in Computer science & Engineering from the University of Shivaji University, Kolhapur ,Maharashtra in 2007. Currently, she is an Assistant Professor of Computer Engineering at MIT AOE Alandi (D) Pune in University of Pune. Her teaching and research areas include Database management, data mining, Machine learning, and operating system.