

A Study on One-way Communication using PF_RING ZC

Jin-Hong Kim*, Jung-Chan Na**

* UST(Korea University of Science and Technology), South Korea

** ETRI(Electronics and Telecommunications Research Institute), South Korea

lggreen53@gmail.com, njc@etri.re.kr

Abstract— Commercial Off The Shelf(COTS) based one-way communication is advantageous in that support a low cost communication and high speed one-way communication.

This paper provides a implementation method of one-way communication through modified device driver for COTS NIC. Then, to verify the advantage of the COTS based one-way communication method, We present a sample implementation using Intel 82580 NIC and PF_RING ZC(Zero Copy). Then, we present a possibility that can contribute to the realization of one-way communication through experiments on performance and reliability.

Keywords— One-way Communication, Unidirectional network , Data Diode, PF_RING ZC, ICS

I. INTRODUCTION

Unidirectional communication technology makes it impossible to transmit data in a specific direction, and physical unidirectional communication and a firewall exist as related technologies. However, in the case of a firewall, there is a vulnerability that bypasses the allow and block policy for a specific service, making it impossible to defend against external threats. For this reason, it is recommended to apply physical unidirectional communication technology recently[1].

In this paper, we describe the configuration for unidirectional communication through register modification to control the operation of COTS NIC(Commercial Off The Shelf Network Interface Card), and propose a unidirectional communication method using PF_RING ZC (Zero Copy) to improve the performance of unidirectional communication.

II. UNIDIRECTIONAL COMMUNICATION

Unidirectional Communication means that the transmission of data flows one direction. Unidirectional communication consists of a unidirectional transmitting computer and a unidirectional receiving computer. The most common design of unidirectional communication platform relies on ethernet optical fiber based standard connectivity. The send node only transfer the data through unidirectional transmitting computer and the receive node only receive the data through unidirectional receiving computer.

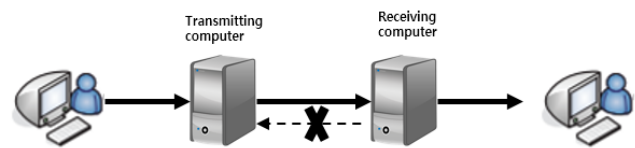


Figure 1. Unidirectional communication

And, the inter-computer communication uses a non-connection oriented protocol such as UDP. However, UDP communication doesn't guarantee that the data received safely, the reliability of the transmission can't be sure.

III. FAST PACKET PROCESSING USING PF_RING ZC

PF_RING ZC was introduced in 2014 as a high-speed network packet processing framework developed by ntop as an open source[2].

PF_RING ZC provides its own functions without using Linux system calls. PF_RING ZC's API emphasizes that multicore environments are easily available by allocating packet buffers to memory areas directly accessible by the CPU using NUMA[3].

PF_RING ZC is developed based on Linux driver and has the advantage that it can be used without running special application such as netmap[4]. However, there is the disadvantage that existing Linux networking applications cannot send or receive while applications using PF_RING ZC is running.

IV. DESIGN OF PF_RING ZC BASED UNIDIRECTIONAL COMMUNICATION PLATFORM

This chapter presents the design of a COTS NIC based unidirectional communication platform through register modification for unidirectional link setup and the PF_RING ZC library to improve performance [5]. COTS NIC based unidirectional communication design can be divided into two stages: register setting for unidirectional communication link and creation of transmission user application.

A. Register setting for unidirectional communication

1) Enable link state for physical one-way communication

A software approach to overcome link disruption in a unidirectional communication platform is to modify the registers of the network interface card at the completion of device initialization routines.

➤ Device Control Register

Device Control Register has a 32-bit length value that controls the main operating mode of the ethernet device. When reassigning values using the PF_RING API to change the device's speed, duplex, or flow control, certain bits that affect the functioning of the NIC can be changed.

TABLE 1. DEVICE CONTROL REGISTER SETTING LIST

Define	offset
E1000_CTRL_SPD_1000	0x0200
E1000_CTRL_FRCSPD	0x0800
E1000_CTRL_SLU	0x0040
E1000_CTRL_FD	0x0001
E1000_CTRL_FRCDPX	0x1000
E1000_CTRL_ILOS	0x0080
E1000_CTRL_RFCE	0x8000000
E1000_CTRL_TFCE	0x10000000

➤ PCS Link Control Register

PCS Link Control Register is used to control the link-related parts of the network interface of the physical layer such as SerDes (Serialization / Deserialization), Serial Gigabit Media-Independent Interface (SGMII) and 1000BASE-KX PCS. Since the Intel 82580 NIC used in this paper that uses the SerDes interface, we set the speed, duplex, and forced link of the SerDes interface statically.

TABLE 2. PCS LINK CONTROLS REGISTER SETTING LIST

Define	offset
E1000_PCS_LCTL_FLV_LINK_UP	0x0001
E1000_PCS_LCTL_FSV_1000	0x0004
E1000_PCS_LCTL_FDV_FULL	0x0008
E1000_PCS_LCTL_FSD	0x0010
E1000_PCS_LCTL_FORCE_LINK	0x0020
E1000_PCS_LCTL_FORCE_FCTRL	0x0080
E1000_PCS_LCTL_AN_RESTART	0x20000

2) Interface definition of the NIC

➤ Extended Control Register

In order to define the network interface card used in this paper, the Link.MODE bit of the Extended Control Register is defined as SerDes.

TABLE 3. EXTENDED CONTROL REGISTER SETTING LIST

Define	offset
E1000_CTRL_EXT_LINK_MODE_PCIE_SERDES	0xC00000

3) Disable the receive port of the device

In the unidirectional communication through physical isolation, considering the case where all the Tx / Rx ports are connected due to user's carelessness, the receiver of the transmitting computer should be stopped to prevent the packet from going up to the network upper layer.

To do this, modify the contents of the RX Control Register so that the received packet can be discarded immediately.

➤ RX Control Register

RX Control Register controls all receive functions of the Intel 82580 controller. In our unidirectional communication platform, the bit that activates the receiver is modified by AND NOT.

TABLE 4. RX CONTROL REGISTER SETTING LIST

Define	offset
E1000_RCTL_RXEN	0x02

B. PF_RING ZC application for unidirectional communication

Since the PF_RING ZC based unidirectional communication does not use the network stack of the kernel, the network technologies are necessary for data transmission such as buffer setting or packet generation. It should be further implemented using the PF_RING ZC API.

V. EXPERIMENT

To verify the improved performance of commercial NIC based unidirectional communication using PF_RING ZC, the throughput and reliability evaluation were performed.

A. Experimental environment

Unidirectional communication uses a platform designed as shown in Fig.2. One NIC is installed in each of the transmitting computer and the receiving computer. The LKM (Loadable Kernel Module) and the PF_RING ZC application

for unidirectional communication are loaded into platform and the experimental environment is configured.



Transmitting computer Receiving computer

Figure 2. Unidirectional communication platform used in experiments

The network interface card selected the Intel 82580 EB, which allows modification of communication link related registers and supports PF_RING ZC. Also, igb.ko driver for Native linux and igb.ko and pf_ring.ko which support PF_RING ZC were used. Other environments are as follows.

- CPU : Intel Core i5-4590 Processor(3.3Ghz)
- OS : Ubuntu 14.04 LTS
- RAM : 8GB (4GB * 2) Hynix Original
- NIC : Intel 82580 EB
- Driver : igb.ko, pf_ring.ko

Since the unidirectional communication environment is physically blocked, a trust protocol such as TCP cannot be used. Therefore, a unidirectional communication application is implemented using UDP, which is a non-connection oriented protocol. The packet structure and size used in the experiment are as follows.

- Ethernet Header : 14 bytes
- IP Header : 20 bytes
- UDP Header : 8 bytes
- Data : 16 byte ~ 1024 bytes

B. Experiment result and Discussion

First, we measured the throughput according to the data size of the frame in relation to the transmission performance.

TABLE 5. TRANSMISSION THROUGHPUT MEASUREMENTS

Frame Size (Bytes)	Throughput(Mbps)	
	LKM (standard Linux kernel)	PF_RING ZC
16	8.82	920
32	17.6	1,000
64	35.3	1,000

128	70.6	1,000
192	100.0	1,000
256	102.0	1,000
512	102.0	1,000
1024	102.0	1,000

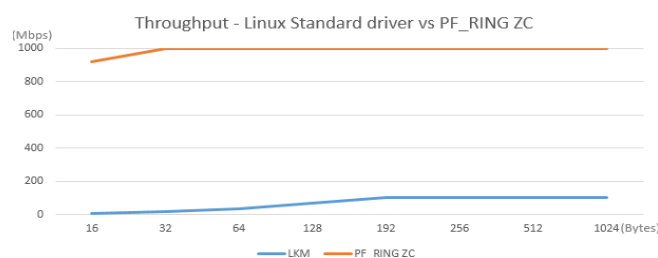


Figure 3. Transmission throughput measurements

In the second experiment, we measured the loss rate according to the frame size when transferring 1GB file. And, the transmission rate used in the experiment is the same as the throughput of the previous experiment.

TABLE 6. COMPARISON OF FRAME LOSS RATE

Frame Size (Bytes)	Frame loss rate(%)	
	LKM (standard Linux kernel)	PF_RING ZC
16	24.00	0
32	26.00	0
64	31.00	0
128	40.00	0
192	55.00	0
256	66.00	0
512	99.63	0
1024	99.58	0

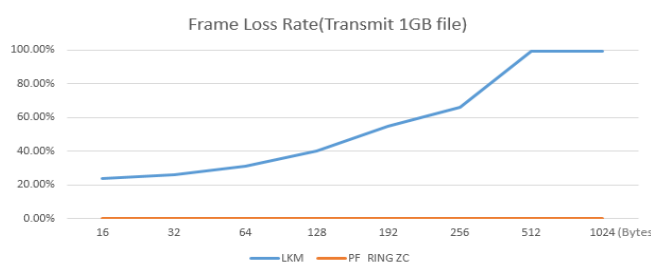


Figure 4. Frame loss rate

In transmitting step, PF_Ring ZC copies directly to the network interface card without copying from the user memory to the kernel memory. In addition, since copying from the user memory to the network interface card memory is performed through the DMA without CPU involvement, the utilization rate of the CPU used in the copying process approaches 0%. This process is referred to as Zero Copy(ZC). you can increase the da

ta throughput by using the cycle of the CPU obtained through Zero Copy for the packet transmission / reception process. This Zero Copy mechanism also applies to the processing of received packets. As the throughput of received packets increases, the rate at which received data accumulates in the buffer is reduced, and the frame loss rate in the buffer is also reduced.

In the second experiment, we can see that frame loss compared to LKM is greatly reduced through Zero Copy mechanism of PF_RING when receiving a packet. In Figure 5, you can see difference in data copy. PF_RING ZC based application send data to NIC directly without kernel copy. But Linux kernel stack-based applications use double-copying, which increases the overhead.

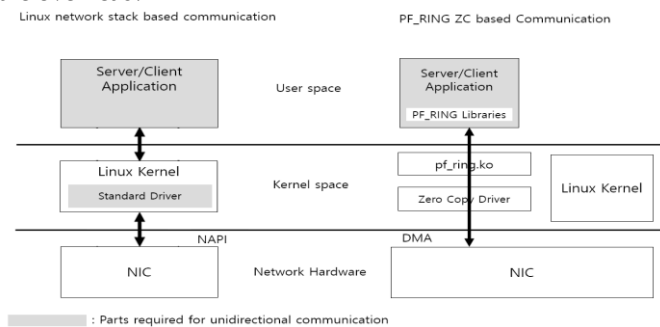


Figure 5. Comparison of standard networking and PF_RING ZC based networking

VI. CONCLUSION

In this paper, we discuss the necessity of physical unidirectional transmission for data linkage in a network separation environment. We also proposed a high performance unidirectional transmission method based on Intel 82580 by combining PF_RING ZC, a high performance packet processing framework, with COTS NIC based unidirectional communication platform.

Through experiments of LKM based unidirectional communication platform and PF_RING ZC based unidirectional communication platform, we confirmed that the performance of unidirectional communication platform using PF_RING ZC is improved overall.

Future research is needed to apply Forward Error Correction(FEC) in order to overcome packet loss and improve platform reliability[6].

ACKNOWLEDGMENT

This study was conducted as part of the R&D on the development of core technology for information protection of the future creation science department and information and communication technology promotion center [R0126-15-1095, development of physical unidirectional security gateway in cyber and physical systems].

REFERENCES

- [1] Sung-Hoon, Lee, "A Study on Separate Plan of Efficient Information System Network in Partitioned Network Environment", Soongsil University, Jun, 2011.
- [2] ntop website, Introducing PF_RING ZC (Zero Copy), [Online]. Available: http://www.ntop.org/pf_ring/introducing-pf_ring-zc-zero-copy/, Apr, 2014
- [3] ntop website, PF_RING ZC API, [Online] Available: http://www.ntop.org/pfring_api/pfring_zc_8h.html, Mar. 2015.
- [4] L.Rizzo, "Netmap: a novel framework for fast packet I/O", 21st USENIX Security symposium, 2012,
- [5] Intel Networking Division (2015), "Intel 82580EB/82580DB Gigabit Ethernet Controller DataSheet", pp 1-760.
- [6] Forward Error Correction from Wikipedia,[Online] Available: https://en.wikipedia.org/wiki/Forward_error_correction



JinHong Kim was born in South Korea in 1990. He received the B.S. degrees in Computer Engineering from Chonnam National University, Korea, in 2015. He is currently a M.S. student in information security engineering at the Korea University of Science and Technology, Korea. His current research interests include network security, system security and cryptanalysis.



JungChan Na was born in South Korea in 1962. He received B.S. degrees in Calculation of Statistics from Chungnam National University in 1986, and He received M.S. degrees in computer engineering from Soongsil University in 1989, respectively. He also received Ph.D degree in Computer Science from Changnam National University in 2004. He joined Electronics and Telecommunications Research Institute(ETRI), Daejeon, Korea, in 1989. He is the leader of the Industrial Control System(ICS) Security Research Section. Currently, his main areas of research interest are Network Security and ICS Security.