

Performance of Packet Analysis between Observer and WireShark

HyunHo Kim*, HoonJae Lee**, HyoTaek Lim**

*Department of Ubiquitous IT, Dongseo University, Busan, South Korea

**Division of Computer and Information Engineering, Dongseo University, Busan, South Korea

feei_@naver.com, hjlee@dongseo.ac.kr, htlim@dongseo.ac.kr

Abstract— with the recent development of telecommunication technology, a new technology called IoT (Internet of Things) has attracted attention. The IoT is a technology that connects objects to objects using the internet and it is expected that the age in which a large number of things form a huge network will come. As wireless network technology develops day by day, wireless LAN is being used in the frequency of a 5G network, which is over 2.4G. However, mobile market operators are commercializing using 5G networks. As wireless network technology develops, wireless devices can be connected quickly and more reliably. However, it is still vulnerable to network attacks due to the nature of wireless. Network forensics is needed to analyze and defend against these attacks. Network Forensics refers to a technology that analysis all actions are taken on the network and analysis and responds to attacks through packet analysis. In this paper, we have used software (Wireshark) and hardware (Observer) to collect and analyze the packets. Moreover, to check the performance of our work, we have compared and analyzed each packet collection using Wireshark and Observer respectively.

Keywords— *Wireshark, Observer, Network forensic, Packet analysis*

I. INTRODUCTION

The 4th industrial revolution has started where IoT (Internet of Things) and 5G telecommunication are introduced and commercialized. IoT refers to a technology that connects an object and object through the Internet. According to Gartner, it expects to use 11.7 billion endpoints in 2019 and reach 13.7 billion endpoints, in 2020 which is an increase of up to 17 percent [1]. In the future, almost everything is expected to be connected to the Internet.

Because IoT devices can be connected and communicate with each other's through the Internet, it is important to find traces, but since telecommunication is devices, further analysis is required.

Network forensics is an analysis of all actions taken in the network, and many of them are forensics through packet analysis. For instance, protocols, traffic, data, etc. can be detected by attacks and vulnerabilities.

However, there are many types of analysis tools for network forensics, most of the research and analysts use Wireshark, which is a tool that is easy to use and has sufficient functions for analysis.

Currently, the protocols used to access websites are HTTP and HTTPS. HTTP is a short abbreviation of Hypertext Transfer Protocol and HTTPS is a short abbreviation of HyperText Transfer Protocol Secure. However, many websites still use the Http protocol.

In this paper, we have analyzed the Wireshark in the previous studies, and further analyze the packets through the Observer and then compare the two analysis.

II. PACKET CAPTURE ANALYZER AND TOOLS

The protocol analyzer can capture packets passing through the network and analyze them in detail. It is mainly used in combination with software or/and hardware. The software analyzer can be installed and used on a PC or laptop. In the case of a hardware analyzer, it refers to separate hard software or a combination of software and hardware. The hardware analyzer is portable and comes with everything you need, and the software analyzer can be operated by installing software on your PC or laptop. In addition, the capture and storage of packets enable the capture and storage of data packets up to the capacity limits of the storage device, and the interpretation/transformation of the protocol allows various major protocols to be analyzed and interpreted in various forms. To show in various forms, real-time monitoring and analysis of the network enable the monitoring of network traffic and the diagnosis and analysis of network problems.

A. Viavi Observer

Figure 1 shows the Viavi Observer Analyzer. The Observer Analyzer is a program developed to collect and analyses packets stored in Viavi's Observer Gigastor in real time. Triggers and alarms provide real-time warning options such as e-mail or alarms when packets exceed a set level.



Figure 1. Viavi Observer [2]

B. Wireshark

Wireshark is one of the most commonly used network protocol analyzers. You can capture data from your network in real time and read the data from the packet that you have already captured. It can run on Windows, Linux, macOS, Solaris, FreeBSD, NetBSD, and many other platforms, including Ethernet, IEEE 802.11, PPP, and loopback, and captures network traffic with a capture library supported by the operating system.

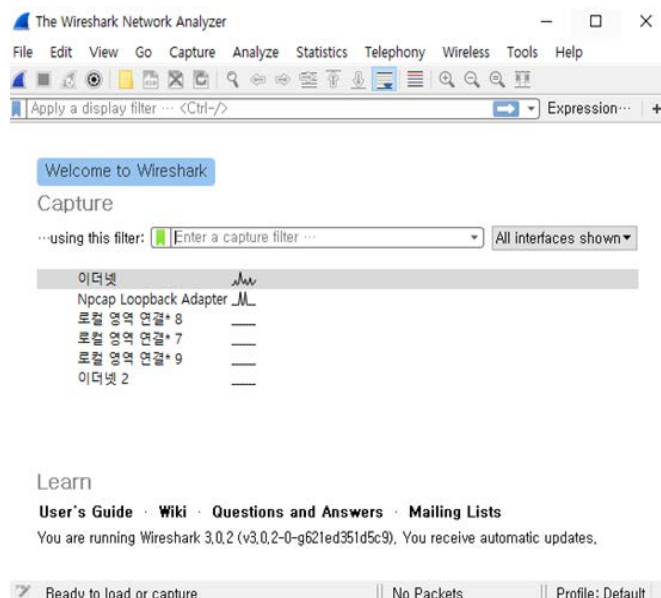


Figure 2. Wireshark Main Screen

C. Packet Capture

Packets capture is divided into three phases: packet capture, protocol parsing, and search/analysis. Packet capture is the process of collecting packet traffic on a network. Protocol parsing is the process of parsing packet binary data into individual network protocols. Fields search and visualization can be visualized using programs and processes that scrutinize or aggregate data. Sniffing technology needs to be set to a Promiscuous Mode state that allows LAN cards to capture all packets that flow over the network. In a switched environment, other segment data can also be sniped by utilizing security attack techniques such as switch Jamming, ARP Redirect, and ICMP Redirect. To snip computers connected to other ports in a switched environment, we use port mirroring, hubbing-out, network tap, and ARP cache poisoning technologies.

III. TREND OF NETWORK FORENSIC RESEARCH USING WIRESHARK

A. Comparative Study of two Most Popular Packet Sniffing Tools- Tcpdump and Wiresharkage Layout[3]

In this research, packet capture and comparative analysis are performed using Tcpdump and Wireshark. During the analysis, both memory and processor has been used. The memory usage of the tool and the processor usage are measured at the same time.

As shown in Figure 3 and Figure 4, the power used is similarly measured, but memory measurement showed the Tcpdump used less memory than Wireshark. First, the virtual size (VIRT) is measured at 620-640Mb with a Tcpdump of 27-29Mb, while Wireshark is measured at more than 20 times the Tcpdump. In packet capture, Wireshark is able to capture more packets.

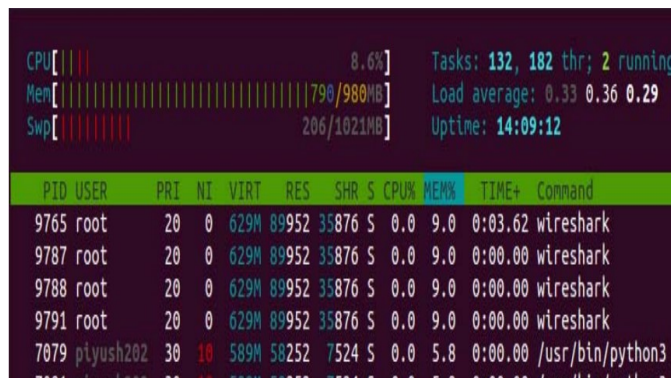


Figure 3. Memory used by Wireshark

From the above we can summarize that Tcpdump is better for memory and processor usage, but Wireshark is better for captured packets and capture speed.

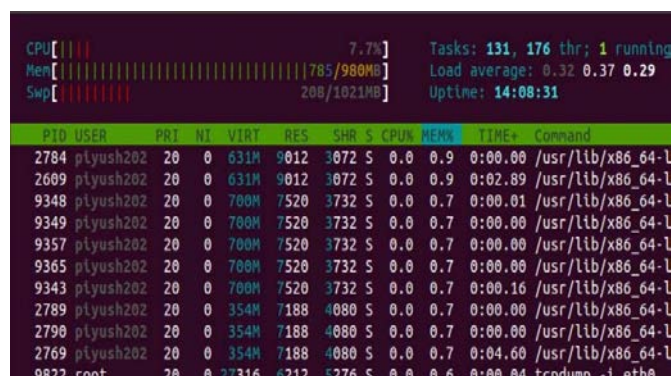


Figure 4. Memory used by Tcpdump

B. Analyzing the webmail using Wireshark[4]

In this study, we used Wireshark to capture and analyze packets when sending webmail from Http to https. Figure 5 shows the monitor and observe by Http filter.

No.	Time	Source	Destination	Protocol	Length	Info
163	0.970855	192.168.0.109	192.168.0.101	HTTP	463	HTTP/1.1 200 OK (image/x-icon)
165	0.975835	192.168.0.101	192.168.0.109	HTTP	553	GET / HTTP/1.1
169	0.977186	192.168.0.109	192.168.0.101	HTTP	1514	Continuation
170	0.977192	192.168.0.109	192.168.0.101	HTTP	821	Continuation
171	0.977461	192.168.0.109	192.168.0.101	HTTP	1514	Continuation
173	0.977751	192.168.0.109	192.168.0.101	HTTP	1514	Continuation
174	0.977736	192.168.0.109	192.168.0.101	HTTP	63	Continuation
183	1.017081	192.168.0.101	192.168.0.109	HTTP	476	GET /folder/tree/file.gif?action=SkinFile HTTP/1.1
184	1.017094	192.168.0.101	192.168.0.109	HTTP	477	GET /folder/tree/login.png?action=SkinFile HTTP/1.1
188	1.019175	192.168.0.109	192.168.0.101	HTTP	1856	HTTP/1.1 200 OK (GIF89a)
190	1.019480	192.168.0.109	192.168.0.101	HTTP	958	HTTP/1.1 200 OK (PNG)
193	1.034783	192.168.0.101	192.168.0.109	HTTP	454	GET /action=Subdir HTTP/1.1
196	1.074043	192.168.0.109	192.168.0.101	HTTP	113	HTTP/1.1 200 OK (text/plain)
198	1.086713	192.168.0.101	192.168.0.109	HTTP	517	GET /folder/tree/empty.gif?action=SkinFile HTTP/1.1
201	1.124664	192.168.0.109	192.168.0.101	HTTP	295	HTTP/1.1 200 OK (GIF89a)
203	2.481882	192.168.0.101	192.168.0.109	HTTP	536	GET /sample.pdf HTTP/1.1
224	2.506513	192.168.0.109	192.168.0.101	HTTP	973	HTTP/1.1 200 OK (application/pdf)

Figure 5. Filter HTTP

As shown in Figure 5, when the webmail is sent through Http, the contents are shown as it is, and when sending the webmail through https, it is encrypted and it is impossible to check the packet.

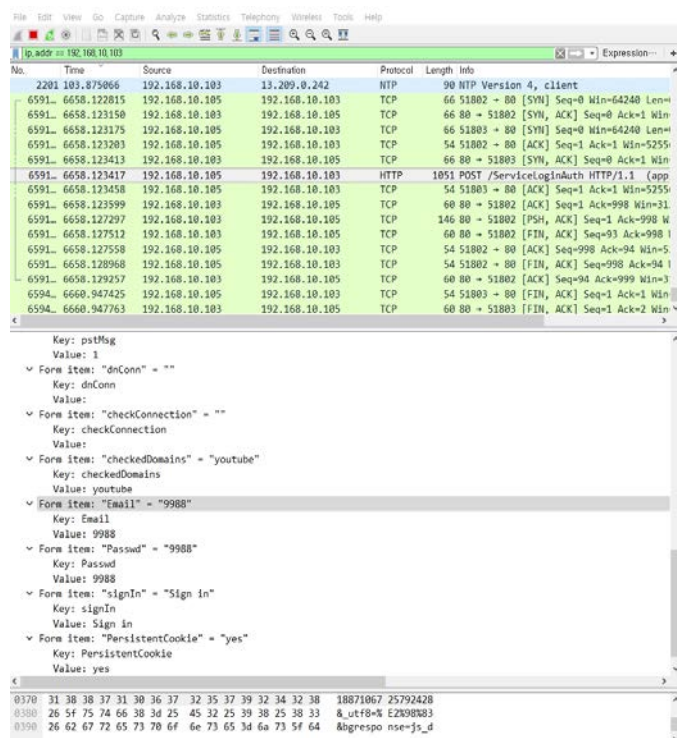


Figure 6. Find Information [5]

Using the test website, the information that can be checked by the packet, (for instance, user ID, password, and mail contents) are shown in figure 6.

IV. COLLECTS AND ANALYZES NETWORK ATTACKS THROUGH THE OBSERVER ANALYZER

To test network traffic capture and analysis, network packets are collected and analyzed using VIAVI's Ntap network collector, a large-capacity observer gig store for packet capture, and an observer analyzer on a PC that will visualize the stored data.

A. UDP Flood Attack

In figure 7, UDP-flood attacks are a type of denial-of-service attack (DOS) that generates large amounts of UDP packets and sends them to any port number on the target system. The system which confirms that there are no applications that use ports sends ICMP Destination Unreachable packets to the source address of the packet sent by the attacker. As a result, the system consumes system resources to process a large number of UDP packets and send ICMP packets, resulting in a denial of service condition.

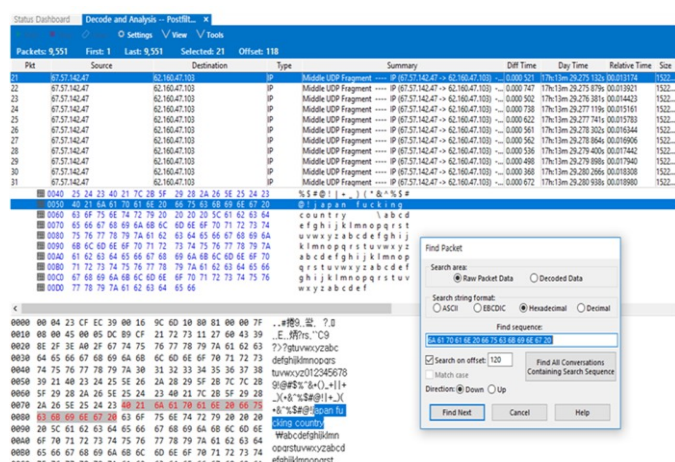


Figure 7. UDP Flood Attack

The server puts a certain amount of space in memory (RAM) to accept the client's connection. This is an attack that depletes memory of a server as one packet comes in from a particular IP. TCP SYN's could be repeated to confirm that they were trying to connect (Figure 8).

B. TCP SYN Flood Attack

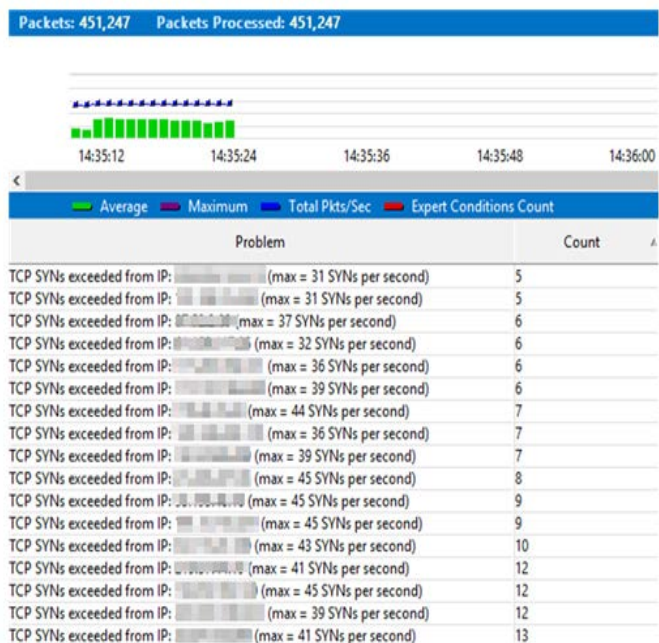


Figure 8. TCP SYN Flood Attack

C. Port Scan

Port scanning network attacks are used as preparation. An attacker performs a Port scan to find an open port on the target host. The attacker can determine which port is open by examining the ports of the selected host (Figure 9). The attacker then uses the open port of the Victim host to perform network attacks.

Pkt	Source	Destination	Type	Summary	Diff Time	Day Time	Relative Ti
1	10.1.0.2	10.1.0.1	IP	TCP SYN (2984 → 1) [RST] Seq=10.1.0.2 → 10.1.0.1	0.000000	10:10:10.15.131.196	0.000000
2	10.1.0.1	10.1.0.2	IP	TCP RST ACK (1 → 2984) Seq=10.1.0.1 → 10.1.0.2	0.000290	10:10:10.15.131.196	0.000290
3	10.1.0.2	10.1.0.1	IP	TCP SYN (2986 → 2) [RST] Seq=10.1.0.2 → 10.1.0.1	0.000463	10:10:10.15.131.196	0.000463
4	10.1.0.1	10.1.0.2	IP	TCP RST ACK (2 → 2986) Seq=10.1.0.1 → 10.1.0.2	0.000463	10:10:10.15.131.196	0.000463
5	10.1.0.2	10.1.0.1	IP	TCP SYN (2988 → 3) [RST] Seq=10.1.0.2 → 10.1.0.1	0.000812	10:10:10.15.131.196	0.000812
6	10.1.0.1	10.1.0.2	IP	TCP RST ACK (3 → 2988) Seq=10.1.0.1 → 10.1.0.2	0.000812	10:10:10.15.131.196	0.000812
7	10.1.0.2	10.1.0.1	IP	TCP SYN (2990 → 4) [RST] Seq=10.1.0.2 → 10.1.0.1	0.000907	10:10:10.15.131.196	0.000907
8	10.1.0.1	10.1.0.2	IP	TCP RST ACK (4 → 2990) Seq=10.1.0.1 → 10.1.0.2	0.000907	10:10:10.15.131.196	0.000907
9	10.1.0.2	10.1.0.1	IP	TCP SYN (2992 → 5) [RST] Seq=10.1.0.2 → 10.1.0.1	0.000907	10:10:10.15.131.196	0.000907
10	10.1.0.1	10.1.0.2	IP	TCP RST ACK (5 → 2992) Seq=10.1.0.1 → 10.1.0.2	0.000907	10:10:10.15.131.196	0.000907
11	10.1.0.2	10.1.0.1	IP	TCP SYN (2994 → 6) [RST] Seq=10.1.0.2 → 10.1.0.1	0.000907	10:10:10.15.131.196	0.000907
12	10.1.0.1	10.1.0.2	IP	TCP RST ACK (6 → 2994) Seq=10.1.0.1 → 10.1.0.2	0.000907	10:10:10.15.131.196	0.000907
13	10.1.0.2	10.1.0.1	IP	TCP SYN (2996 → 7) [RST] Seq=10.1.0.2 → 10.1.0.1	0.000907	10:10:10.15.131.196	0.000907
14	10.1.0.1	10.1.0.2	IP	TCP RST ACK (7 → 2996) Seq=10.1.0.1 → 10.1.0.2	0.000907	10:10:10.15.131.196	0.000907
15	10.1.0.2	10.1.0.1	IP	TCP SYN (2998 → 8) [RST] Seq=10.1.0.2 → 10.1.0.1	0.000907	10:10:10.15.131.196	0.000907
16	10.1.0.1	10.1.0.2	IP	TCP RST ACK (8 → 2998) Seq=10.1.0.1 → 10.1.0.2	0.000907	10:10:10.15.131.196	0.000907
17	10.1.0.2	10.1.0.1	IP	TCP SYN (2999 → 9) [RST] Seq=10.1.0.2 → 10.1.0.1	0.000907	10:10:10.15.131.196	0.000907
18	10.1.0.1	10.1.0.2	IP	TCP RST ACK (9 → 2999) Seq=10.1.0.1 → 10.1.0.2	0.000907	10:10:10.15.131.196	0.000907
19	10.1.0.2	10.1.0.1	IP	TCP SYN (3000 → 10) [RST] Seq=10.1.0.2 → 10.1.0.1	0.000907	10:10:10.15.131.196	0.000907
20	10.1.0.2	10.1.0.1	IP	TCP SYN (3001 → 11) [RST] Seq=10.1.0.2 → 10.1.0.1	0.000907	10:10:10.15.131.196	0.000907
21	10.1.0.2	10.1.0.1	IP	TCP SYN (3002 → 12) [RST] Seq=10.1.0.2 → 10.1.0.1	0.000907	10:10:10.15.131.196	0.000907

Figure 9. Port Scan Test

D. HTTP Reconstruct Stream

When restoring an image in HTTP, Wireshark cuts only the contents of the image file packet in the request part and the response part after performing the Follow TCP Stream, and then converts it to raw format, using the carving method HxD editor, or the HTTP where Export object should be check-in the object list. This enables carving through URL, content type, file size, and name. In contrast, the Observer uses the Reconstruct Stream function to display the results. (Figure 10, 11).

Station/Port	Server/Client Totals	Application	Status	Packets	Bytes	Content Type
172.16.1.125 (8)	184.25.108.105 (80)	HTTP	200	175	318	text/html
172.16.1.125 (8)	184.25.108.105 (80)	HTTP	200	37	52	text/html
172.16.1.125 (8)	23.111.9.204 (80)	HTTP	200	14	22	text/html
172.16.1.125 (8)	114.188.157.155 (80)	HTTP	200	20	24	text/html
172.16.1.125 (8)	68.232.45.201 (80)	HTTP	200	22	19	text/html
172.16.1.125 (8)	58.18.43.124 (80)	HTTP	200	8	11	text/html
172.16.1.125 (8)	104.16.89.20 (80)	HTTP	200	28	21	text/html
172.16.1.125 (8)	113.23.188.167 (80)	HTTP	200	13	13	text/html
172.16.1.125 (8)	64.233.188.154 (80)	HTTP	200	15	13	text/html
172.16.1.125 (8)	172.217.25.68 (80)	HTTP	200	15	13	text/html
172.16.1.125 (8)	216.58.197.131 (80)	HTTP	200	15	13	text/html

Figure 10. HTTP Reconstruct Steam (Observer)

File Name	File Size	Content Type	Download Link
ReconstructedFile.htm	8138 bytes	text/html	https://172.16.1.125/
ReconstructedFile.htm	12120 bytes	text/html	https://172.16.1.125/
ReconstructedFile.htm	12120 bytes	text/html	https://172.16.1.125/
ReconstructedFile.htm	49482 bytes	text/html	https://172.16.1.125/
ReconstructedFile.htm	429 bytes	text/html	https://172.16.1.125/
ReconstructedFile.htm	4176 bytes	text/html	https://172.16.1.125/
ReconstructedFile.htm	16292 bytes	text/html	https://172.16.1.125/
ReconstructedFile.htm	17891 bytes	text/html	https://172.16.1.125/
ReconstructedFile.htm	56006 bytes	text/html	https://172.16.1.125/
ReconstructedFile.htm	13844 bytes	text/html	https://172.16.1.125/
ReconstructedFile.htm	679957 bytes	image/png	https://172.16.1.125/
ReconstructedFile.htm	641355 bytes	image/png	https://172.16.1.125/
ReconstructedFile.htm	89938 bytes	image/png	https://172.16.1.125/
ReconstructedFile.htm	330756 bytes	image/png	https://172.16.1.125/

Figure 11. Reconstruct File Download

V. CONCLUSION

The Observer platform, operated by Viavi Corp., has greater effectiveness than Wireshark in environments where large amounts of data are used. Using the Viavi GigaStor, the Observer analyzer can capture and record data for a long period of time to restore TCP session information, such as HTTP webpage, IM, FTP/Telnet, excluding encrypted packets

using network forensics, so that it can support forensics by complying with the requirements of Sarbanes Oxley or HIPPA. The advantages of free wire shacks, scalability through plug-ins, and install and operate on almost any operating system, are used by many network managers for mobile use.

ACKNOWLEDGMENT

This work was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science, and Technology (grant number: NRF2016R1D1A1B01011908). And it was also supported by the Institute for Information and Communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No.2018-0-00245).

REFERENCES

- [1] E. U.K. "Gartner Says 5.8 Billion Enterprise and Automotive IoT Endpoints Will Be in Use in 2020" Gartner Company, August, 29, 2019. <https://www.gartner.com/en/newsroom/press-releases/2019-08-29-gartner-says-5-8-billion-enterprise-and-automotive-iot-endpoints-will-be-in-use-in-2020>
- [2] VIAVI Observer GigaStor, 2018. <https://www.viavisolutions.com/de-de/literature/observer-gigastor-de-broschure-de.pdf>
- [3] P. Goyal, A. Goyal, "Comparative Study of two Most Popular Packet Sniffing Tools-Tcpdump and Wireshark", International Conference on Computational Intelligence and Communication Networks 2017.
- [4] P. Navabud, Chin-Ling Chen, "Analyzing the Web mail using Wireshark", International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), 2018.
- [5] Naverblog. 2019. https://blog.naver.com/genesis_k453/221434886353



HyunHo Kim received his BS degrees in computer science from Dongseo University in 2013. MS degree in Department of Ubiquitous IT from Dongseo University in 2015. Currently, he is a Ph.D. candidate student in the Department of Information and Communication Engineering at Dongseo University, Korea. His research interests include Digital forensic, Information security, IoT security, and Network security.



Prof. HoonJae Lee received his BS, MS, and Ph.D. degrees in electronic engineering from Kyungpook National University, Daegu, Rep. of Korea, in 1985, 1987, and 1998, respectively. He is currently a professor in the Department of Information Communication Engineering at Dongseo University. From 1987 to 1998, he was a research associate at the Agency for Defense Development (ADD). His current research interests include Password Theory, Network Security, Side-Channel Attack, and Information Communication/Information Network.



Prof. HyoTaek Lim received his B.S. degree in computer science from Hongik University in 1988, the M.S. degree in computer science from POSTECH and the Ph.D. degree in computer science from Yonsei University in 1992 and 1997, respectively. From 1988 to 1994, he had worked for Electronics and Telecommunications Research Institute as a research staff. Since 1994, he has been with Dongseo University, Korea, where he is currently a professor in the Division of Computer Engineering. His research interests include ubiquitous and mobile networking, cloud computing, and storage area networks.