Design and Implementation of Tor Traffic Collection System Using Multiple Virtual Machines

Hyun-Jae Choi, Hyun-Soo Kim, and Dong-Myung Shin

***Abstract*—In order to detect copyright infringement that contents are shared illegally on Tor network, traffic has been collected and analyzed efficiently. A Tor traffic collection system has been designed and implemented using multiple virtual machines. A number of virtual machines and Mini PC's are used as clients to connect the Tor network, and collection and refinement processes in the traffic collection server have automated through script-based test client software. The client PCs that make up the system collect and refine traffic through the C & C server, and share and store the collected traffic using the CIFS protocol. Through this system, only Tor network traffic and necessary field data can be stored, and the performance of recognizing and refining Tor traffic only is achieved more the 95%. The collected traffic will be used for research such as 'Traffic Pattern Analysis' and 'Traffic Fingerprinting'.**

***Index Terms*—Tor network, virtual machine, traffic collection, refining.**

I. Introduction

A. The Concept of Tor Network

Tor(The Onion Routing) is a technology designed to use the Internet without exposing the IP address of the user's origin and destination. Currently, it is communicating by encrypting and transmitting IP and packets over the thousands of unspecified relay nodes. Unlike a normal Web site, on Tor network it provides services called 'Hidden Service' to users. It uses '.onion' address and only can be accessed by using Tor browser. When connecting to a Web site using Tor browser, it relays at least three relay nodes (Entry Node, Relay Node, Exit Node). When accessing the Hidden Service, it relays at least six relay nodes then accesses corresponding destination [1]. When accessing the hidden service, users can know IP address of three relay nodes passing through the first, but can not know the IP address of three relay nodes passing through the second. Likewise, the IP address of the first relay node connected to the server can be known from hidden service, but the IP address of the three relay nodes connected to the user to access the hidden service can not be known. For this reason, the directory server's role is connecting to three relay nodes passing through the second, and the point at which the three nodes meet is called as a "Rendezvous Node". Since the nodes can encrypt the received data with their own key and can not know that both connection and server pass through which nodes, Tor network is guaranteed to be anonymous and difficult to trace [2]-[4].

Tor network is estimated to be connected to an average of hundreds of thousands of users per day [5]. Tor network, which acts as a network, is very dynamic and easy for anyone to participate in and provides resources to other users. Thus, network delays due to network layer and capacity distribution problems caused by Tor's use of clients. Some studies suggest that when you connect to a website or hidden service using Tor network, it takes about four seconds to wait [6]. As a result, some users are tolerant of Tor's low performance due to national censorship, but most users are dissatisfied with this low performance [7].

B. Problems and Research Contents

Since Tor network can hide identity of users or service provider, it is being exploited for illegal transactions such as drugs and weapons, and its usage rate is rapidly increasing. In addition to illegal transaction, such as drugs and weapons, the piracy and distribution of content, which is an issue recently, is also increasing. Since using the Tor network which is an anonymous network leads to these problems, copyright infringement cases are occurring. So, it is one of problem that being new channel to illegally distribute contents. In the future, the copyright infringement case, which is an illegal sharing contents on Tor network, become serious. Tor network needs technology to detect copyright infringement. As a technique for detecting copyright infringement occurring on the Tor network, traffic fingerprinting technology and Tor network traffic analysis may be used [8]-[11]. In addition to these technologies, it will be possible to detect copyright infringement by analyzing the hidden services itself. Detecting copyright infringement using these techniques requires a technique for efficiently collecting Tor network traffic. The collected traffic is likely to contain information from the Tor network as well as other traffic. If other traffic is included, inaccurate results can occur when that traffic is analyzed. Therefore, a technology that extracts and refines traffic related with only the Tor network from collected traffic is needed. And this technology requires high performance.

In this paper, the purpose is collecting a large amount of Tor network traffic efficiently for the prior study to detect copyright infringement. First, using module for collection and the system configuration for collecting Tor traffic and the database design for saving traffic is explained. Second, utility traffic filtering method developed for automation collection system is explained. Third, this concludes with performance evaluation and conclusion. Finally, this paper concludes by briefly introducing a plan for conducting research to detect copyright infringement.

Manuscript received June 29, 2019; revised September 26, 2019.

Hyun-Jae Choi, Hyun-Soo Kim, and Dong-Myung Shin are with the LSware, Korea (e-mail: esther@lsware.co.kr, hskim94@lsware.co.kr, ronald@lsware.co.kr).

doi: 10.18178/ijke.2019.5.2.120

II. Tor Traffic Collection Process

A. Tor Traffic Collection System

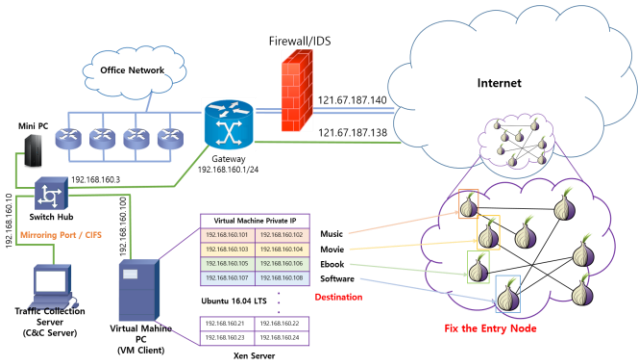

Fig. 1. Tor network traffic collection system configuration.

Fig. 1 shows schematic diagram of equipment prepared to collect Tor traffic.

In order to prevent traffic to be blocked from firewalls, IDS, and IPS, a separate Tor network traffic collection network was constructed, and a virtual machine PC was installed with Xen Server v7.5 OS. 20 virtual machines operating independently as Ubuntu v16.04 TLS were installed and were used as client to connect to Tor network. Also, same OS on the 5 Mini PC was installed to solve the problem that can be occurred with virtual machines and was used as client. Each virtual machine and Mini PC is assigned a private IP within our established network, and connect to the Tor network. At that time, first entry node of connecting Tor network is classified by category in consideration of bandwidth and speed.

Traffic collection server collectively controls the virtual machines in the virtual PC and Mini PCs, and sends commands such as node setting and traffic collection to the virtual machines connected to the Tor network to collect traffic. First method to collect traffic is that each virtual machines collect traffic and convey it to traffic collection server. Second method is that collect whole traffic of all virtual machines by connecting the traffic collection server to the switch mirroring port.

B. Database Design for Storage Traffic

Tor traffic data collected using traffic collection system was refined and stored only for necessary fields to be used for future feature point research [12]. Fig. 2 shows the structure of data table which can store refined traffic data. The pcap files stored in traffic collection server through virtual machines were converted to csv data only necessary fields and stored in PostgreSQL database.

PostgreSQL is one of the powerful RDBMS (object-relational database system) that uses and extends SQL language. Also, PostgreSQL is open source program. This PostgreSQL is known for its proven architecture, reliability, data integrity, enhanced feature set, scalability, and commitment from the open source community. PostgreSQL includes many features to help administrators protect data integrity, build fault-tolerant environments, and manage data at any size. In addition, using its own GUI program called PGADMIN, anyone can easily search and visualize database structure or database contents search without making complicated query.

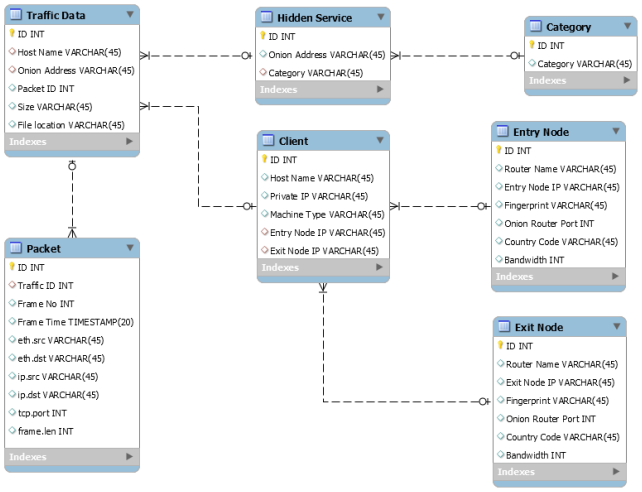

Fig. 2. Tor network traffic data table definition.

The Traffic Data table shows summary information of traffic data based on specific virtual machine or hidden service.

The Packet table can store and search time of collectable traffic, MAC, IP, Port, packet length when connect one time to specific hidden service.

The Hidden Service table stores address of hidden service which distribute illegal contents on Tor network. The Category table can search hidden services that distribute illegal contents by type. Finally, the Client table stores each virtual machine's information. And which node and the node information is used by a particular virtual machine can be inquired.

C. Using Software Module for Traffic Collection

Tor packages (torify, torsocks), wget, Tshark module were installed on virtual machine and were used for traffic collection.

Torify module, the most basically and simple Tor wrapper in the system, called torsocks using specific configure file(change Entry Node modifying /etc/tor/torrc file).

Torsock module uploaded libtorsocks.so library on bash using FD_PRELOAD environment variable. And the bash conducted encryption communication using Tor. In this paper's traffic collection process, a method that connect to Tor network calling torsocks instead of Tor Browser was used because Tor Browser consume lots of client's resource.

Wget module was a software that fetches contents from web server and supports download using HTTP, HTTPS, FTP protocol.

Tshark module was a software that filters copy of packet and sent it to kernel subsystem every time when network driver received network packet and stores libpcap file format.

D. Utility Development for Traffic Collection and Refinement

Tshark module was installed on every virtual machine, Mini PC, and traffic collection server to collect Tor network traffic. Also, to collect traffic with stable speed and delay fix high bandwidth node refer to Tor Network Status(https://torstatus,.blutmagie.de) as Entry Node [13]. (modify /etc/tor/torrc file) Nodes have been configured in every client, the Tshark module starts collecting packet.

Tshark module sets the IP and ORPort of Entry Node in CaptureFilter like Fig. 3 as a filtering option before collecting packet because Tshark module perform packet collection and refinement simultaneously [14].

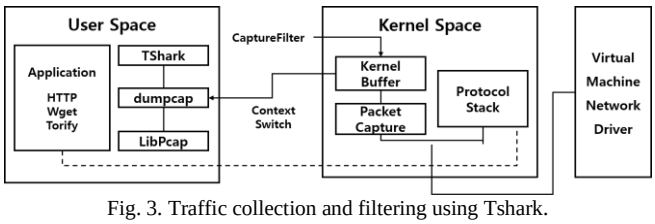

Fig. 3. Traffic collection and filtering using Tshark.

CaptureFilter filters and sends packet to the dumpcap according to the filtering option set at the time of collection, then this dumpcap stores received packet in the virtual machine storage space in libpcap file format. Collecting and refining all the network traffic of virtual machine without filtering option gives a lot of CPU load because it has to copy all the packets from the kernel space to the dumpcap of the user space (context switching) when collecting [15]. So, we copy only Tor network traffic from kernel space to user space using CaptureFilter without separating collection process and filtering process specially. In this paper, collection and refinement process is developed as one shell script. And this shell script operate at virtual machine and Mini PC.

III. Collection System Automation and Performance Evaluation

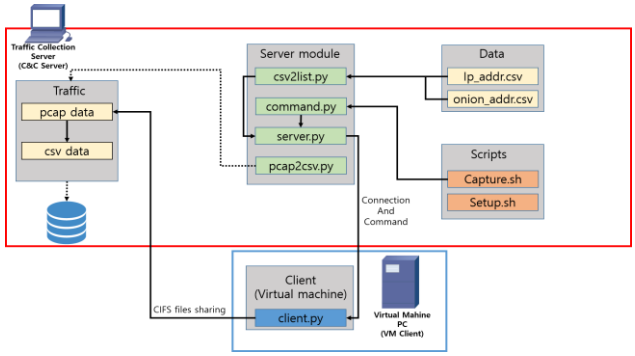

Fig. 4. Script-based test client software configuration diagram.

In this paper, Collecting server was developed that looks like C&C (Command &Control) collectively manage and command multiple virtual machine and Mini PCs connected to the Tor network. Fig. 4 shows operating process and configuration.

Using client program that establish traffic collecting server between more than 20 Virtual machines and Mini PCs connect and grant access and understand command can do many things. First, it can change torrc file which allows to change Entry Node and Exit Node. Second, using torify module and wget module can connect Tor network and download web page. Third, the Tshark module can collect and refine Tor network traffic at the same time as the Tor network connection and downloading web site process. Finally, traffic collection server program was developed that can covert from the pcap formatted file stored in Tor network traffic to csv formatted time series data, and converted csv formatted data was saved to PostgreSQL database. The role of each program is as follows.

1) setup.sh

Control the torrc file with a script that can change the EntryNode and ExitNode(Specify the fingerprint value that identifies the Node).

2) ip_addr.csv

Enter the internal IP address of the virtual machine to perform the operation.

3) onion_addr.csv

Enter the hidden service (Onion address) on the Tor network to collect traffic.

4) client.py

Run client.py on each virtual machine and Mini PC, and waits for it to perform the command operation from the traffic collection server. (Listening state)

5) command.py

When the traffic collection server enters the server.py, the virtual machines execute the following:

capture.sh connects to the Tor network through the torify module and downloads the hidden service page via wget. At the same time, the Tshark module is used to execute the Tor network traffic collection and refinement script. When the traffic collection is completed in the virtual machine, the traffic is stored in the traffic collection server through the CIFS share.

6) server.py

Run server.py on the traffic collection server and instruct the virtual machines that run client.py to operate. It sends the command written in command.py to the virtual machines entered in 'ip.addr.csv' targeting the hidden service entered in 'onion_addr.csv' to collect traffic.

7) pcap2csv.py

Convert the traffic file (.pcap) stored in the traffic collection server to csv format data.

Through the test client software based on script, the collection process was able to be automated. In a single time, more than 10,000 packets were able to be stored on average. Then more than 95% performance of recognizing and refining Tor traffic only could be achieved.

IV. Conclusion and Future Plan

Since the identity of user or service provider is hidden, Tor network is used to illegal trade of drug or weapons, even though circulation of illegal distribution of contents, which is increasing the problem of copyright infringement due to illegal copy distribution.

In this paper, Collection process automation has been shown through Tor traffic collection system using multiple virtual machine, saving only Tor network traffic with high refinement rate, and storing necessary field data to database.

As a future plan, first, a hidden service execution server will be developed for test. A server that can execute both web service and hidden service in the Google Cloud Platform VM instance will be constructed. Then difference of traffic will be analyzed and method to infer the hidden service's content through traffic of web service will be studied. The reason of constructing server in the Google Cloud Platform VM instance is that it can use different public IP than the virtual machines IP in the traffic collection system. Difference of traffic will be collected according to Relay Node which is

changed by IP changing because the platform itself can flexibly change the public IP.

Second, it is a consideration of the difference of the traffic according to fixing Entry Node and using module and the time of collection. In the previous study, selecting Entry Node was fixed and wget module was used and traffic was collected until the contents of the connected page were all loaded. However, users who use real Tor Browser never fix Entry Node and use wget module. So Study about difference of traffic between outgoing previous collecting traffic and traffic collected by using Tor Browser will be conducted [16].

Third, the wget and torigy-modules are currently used to collect traffic. But the actual Tor Browser Bundle will be used to collect traffic and compare the collected traffic. The reason of collecting traffic using Tor Browser Bundle, there may be differences between the traffic collected using torify modules and the traffic collected using the Tor Browser Bundle. To do this research, the Tor Browser will be needed to be manipulated and traffic about specific hidden service will be collected.

Finally, research about traffic pattern analysis, feature point analysis using machine learning, traffic active fingerprinting, and passive fingerprinting using collected traffic will be conducted.

ACKNOWLEDGEMENT

This study was conducted as a result of the study of the copyright technology development project of the Ministry of Culture, Sports and Tourism and the Copyright Commission of Korea in 2019.

REFERENCES

[1] D. Roger, M. Nick, and S. Paul, "Tor: The second-generation onion router," in *Proc. the 13th Conf. on USENIX Security Symposium*, 2004, pp. 1-17, vol. 13.

[2] N. Evans, R. Dingledine, and C. Grothoff, "A practical congestion attack on Tor using long paths," in *Proc. USENIX Security Symposium*, 2009.

[3] O. Lasse and P. Syverson, "Locating hidden servers," in *Proc. IEEE Symposium on Security and Privacy (S&P'06), IEEE*, 2006.

[4] M. Damon, *et al.*, "Shining light in dark places: Understanding the Tor network," in *Proc. International Symposium on Privacy Enhancing Technologies Symposium*, Berlin, Heidelberg: Springer, 2008.

[5] M. Steven and R. N. Watson, "Metrics for security and performance in low-latency anonymity systems," in *Proc. International Symposium on Privacy Enhancing Technologies Symposium*, Berlin, Heidelberg: Springer, 2008.

[6] W. Rolf, D. Herrmann, and H. Federrath, "Performance comparison of low-latency anonymisation services from a user perspective," in *Proc. International Workshop on Privacy Enhancing Technologies*, Berlin, Heidelberg: Springer, 2007.

[7] K. Stefan, "Low latency anonymous communication–How long are users willing to wait?" in *Proc. International Conf. on Emerging Trends in Information and Communication Security*, Berlin, Heidelberg: Springer, 2006.

[8] T. Wang and G. Ian, "Improved website fingerprinting on tor," in *Proc. the 12th ACM Workshop on Workshop on Privacy in the Electronic Society*, ACM, 2013.

[9] M. Steven and G. Danezis, "Low-cost traffic analysis of Tor," in *Proc. 2005 IEEE Symposium on Security and Privacy (S&P'05)*, IEEE, 2005.

[10] H. Andrew, "Fingerprinting websites using traffic analysis," in *Proc. International Workshop on Privacy Enhancing Technologies*, Berlin, Heidelberg: Springer, 2002.

[11] P. Andriy, *et al.*, "Website fingerprinting in onion routing based anonymization networks," in *Proc. the 10th Annual ACM Workshop on Privacy in the Electronic Society*, ACM, 2011.

[12] B. Adam, U. Möller, and A. Stiglic, "Traffic analysis attacks and trade-offs in anonymity providing systems," in *Proc. International Workshop on Information Hiding*, Berlin, Heidelberg: Springer, 2001.

[13] F. Cangialosi, D. Levin, and N. Spring, "Ting: Measuring and exploiting latencies between all Tor nodes," in *Proc. the 2015 Internet Measurement Conf.*, 2015.

[14] Y. Gilad and A. Herzberg, "Spying in the dark: TCP and Tor traffic analysis," in *Proc. the 12th Privacy Enhancing Technologies Symposium (PETS 2012)*, July 2012.

[15] Y. Sun, A. Edmundson, L. Vanbever, O. Li, and J. Rexford, "RAPTOR: routing attacks on privacy in Tor," in *Proc. the 24th USENIX Security Symposium*, August 2015.

Hyun-Jae Choi received his master degree from the Sungkyunkwan University in 2018. He is currently working as a research engineer at LSware's research and development group. His interest researches are network security, system security, block chain, vulnerability analysis.

Hyun-Soo Kim received his bachelor degree from the Dankook University in 2019. He is currently working as a research engineer at LSware's research and development group. His interest researches are software testing, network, software engineering, artificial intelligence.

Dong-Myung Shin received his doctoral degree from the Daejeon University in 2003. From 2011 to 2006, he was worked as a research engineer at the Korea Information Security Agency. Then He was worked as the head of copyright engineering team of the Korea Copyright Commision from 2006 to 2014. He was worked as the team leader of the security certification team in Korea Smart Grid Project. He is currently working as a CTO at LSware's research and development group. His interest researches are opensource license, smart gird certification and security, system and network security, SW vulnerable analysis and authentication.