International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 1 (2018) pp. 483-489
© Research India Publications. http://www.ripublication.com

A Survey Paper on Botnet Attacks and Defenses in Software Defined Networking

Rahmadani Hadianto[1] and Tito Waluyo Purboyo[2]

[1]Research Scholar, Faculty of Electrical Engineering, Telkom University, Bandung, Indonesia.
[1]Orchid : 0000-0001-8641-7103
[2]Lecturer, Faculty of Electrical Engineering, Telkom University, Bandung, Indonesia.
[2]Orchid : 0000-0001-9817-3185

Abstract

This paper present review of Software Defined Networking (SDN) security and potential threat, especially against botnet Attack. A botnet is a collection of computer controlled by hackers to run various network threats. In other hand SDN system work in a centralized configuration, control, and operation with separated control and data plane. With this system, botmaster can insert and deploy their infection through SDN control plane as an unauthorized computer. This problem considered as Integrity on CIA triad (Confidentiality, Integrity, and Availability) that is used for SDN security performance evaluation. Integrity in CIA triad means a condition where information is kept accurate and consistent unless authorized changes are made. At the end of this paper, we explain a future research to handle botnet attack.

Keywords: Software Define Networking, Botnet, Integrity

INTRODUCTION

SDN [27][28][29] as a new technology in network development plays a role in a centralized configuration, control, and operation network. It can empower network architectures [], cost efficiency, and give the opportunity to new network application/function by update software system. SDN has not been well recognized by the security community yet [1]. As an evidence of this statement, while there are more and more research papers in top networking venues and several new SDN focused conferences created recently in 2017, there is still less attention from a security researcher [1].

On the other side, SDN has potential to be attacked along with technological development. Botnet attack is one type of attack that became the main threat to the traditional network that has a chance to become a dangerous threat in SDN. The reason why botnet is chosen as one of various attack because of its impact on disruption for integrity. Integrity is one of CIA (Confidentiality, Integrity, Availability) Triad component that used as security parameter for network security. Integrity in CIA triad is a condition where information is kept accurate and consistent unless authorize changes are made.

This paper arranged as follows, section II described SDN Feature next in Section III, we provide SDN potential attack to each architecture that vulnerable to be attacked. In Section IV, various botnet classification will be described. In Section V, SDN defense against Botnet will be explained. At last on Section VI Future research direction are identified.

SDN FEATURE

SDN provide benefit with four feature that divided as follow :

1) Dynamic Flow Control

   SDN has the ability to dynamically control data flow on the network, with this feature SDN data flow can be controlled efficiently. Malicious network packets flow can be divided from authorized flows without setting up new middleboxes and it can be done with network device only. PBS (Programmable BYOD Security) [3] is an example of this feature, it grants access operator for establishing and set up virtual Software Define Network.

2) Integrate Control with Extensive Visibility

   Supervise and manage the overall network to get all status of network and data flow. NetSecVisor [4] has the ability to utilize security devices and affect SDN function to be virtualized.

3) Network with Programmability

   Network function on SDN control plane can be programmed by Application Programming Interfaces (APIs) [5]. This feature makes network security development become easier by the setup program for security application. One of this feature examples is FRESCO [6].

4) Simple Data Package

   Not like as in traditional network, SDN divided control and data plane. This feature makes SDN data plane become simpler and open opportunity for adding data plane extension for security function. OFX (Open Flow Extension Framework) [7] is an example of this feature, OFX has the ability to enable SDN security application with existing OpenFlow.

Potential Attack on SDN

Based on [2] there are several potential security problems in the SDN grouped into several points as shown in Figure 1 and described in each of the following points:

1) Unauthorized Access

   Access, in this case, is referred to access control. There is some possibilities attacker disguised as controller or application from a 3rd party. An attacker could get access to network resources and control the network operation.

2) Data Leakage

Packet handling on SDN has several potential actions such as forwarding, drop, and send a packet to the controller [2]. With this system, there is possibility attacker to determine the action to the specified packet by determining time process for packet arrived because packet forwarding from packet handling to the controller is longer than others.

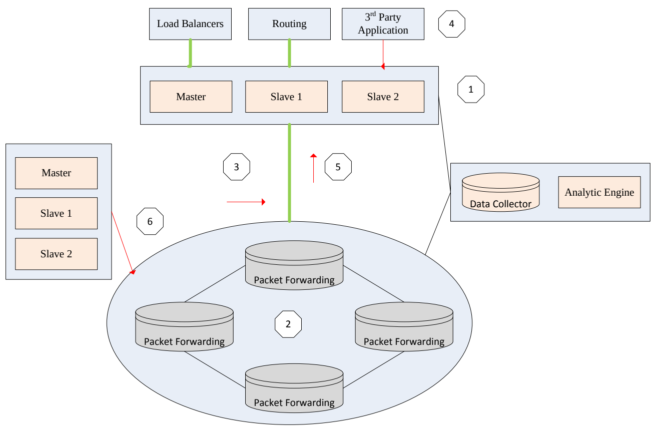

Figure 1. Potential Attack on SDN [2]

3) Data Leakage

Packet handling on SDN has several potential actions such as forwarding, drop, and send a packet to the controller [2]. With this system, there is possibility attacker to determine the action to the specified packet by determining time process for packet arrived because packet forwarding from packet handling to the controller is longer than others.

4) Data Modification

The attacker has a chance to modify data whenever they can hijack controller for the entire system. If this condition happens, an attacker can modify an inject flow rules in network devices that allow the packet to be controlled for attacker advantage.

5) Malicious Application

Because of SDN open for 3rd party application to the architecture [9], some application could be exploited by an attacker and drive into the unsafe state.

6) Denial of Services

The main weakness of SDN is in separated control and data plane. It is because in the communication path between controller and network device attacker could spam and flood and caused unavailable services on the network [8].

7) Configuration Issues

SDN as the programmable network has weakness in security because it can come to be vulnerable especially for data or control communication.

8) System Level SDN Security

Its important for an operator to know the mode and switch activated in connection disruption, forwarding pattern during failures, the effect on flow entries and pattern of the controller when reestablishing the connection.

Botnet Overview

As defined before, Botnet is a collection of the computer that controlled by hackers. Botnets are divided into 3 main cycles in the way its phase of infection, C & C communication phase, and phase of attack. [14][16][30]. Differences botnet than other types of attacks is the existence of C&C that work in giving orders from botmaster to bot. Bots always hide while looking for an unattended target, when bot find the target they will report to the botmaster[10].

Architecture Botnet

Botmaster manages their C&C (Command and Control) to communicate with their bot indirectly in purpose to hide the C&C architecture. There are three topologies on Botnet attack can be seen in figure 2, 3 topologies on Botnet described as follow:

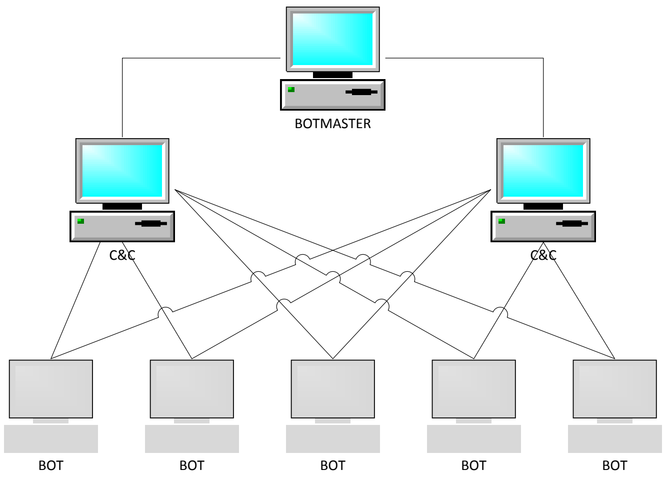

Figure 2. Centralized C&C Server [10]

1) Centralized C&C Server

This is the oldest type of botnet topology, Centralized C&C server topology provides low latency, unknown, and real-time communication to botmaster with a central point of delivering a message from botmaster to contact their bot. One main point of C&C has responsibility for exchanging instruction and information between Botmaster and Bot [10]. The weakness of this topology is whenever C&C server has been tracked then all of the botnet systems will be useless. Several examples of this topology can be found on AgoBot, RBot, and SDBot.

a) Botnet IRC

Figure 3 shows the Botnet attack scheme through IRC, IRC is a text message sent over the internet [11]. The protocol works based on the model of the client-server, that is used on computers in a distributed network. The advantages of using IRC protocol on botnet are:

1) Has a low latency on the communication side
2) Real-time communication is done covertly
3) Have the ability to work in groups and in pairs
4) Easy to set up
5) Simple command instruction, basic command consists of, commands to connect to the server, post messages in the channel.
6) Communication run flexibly

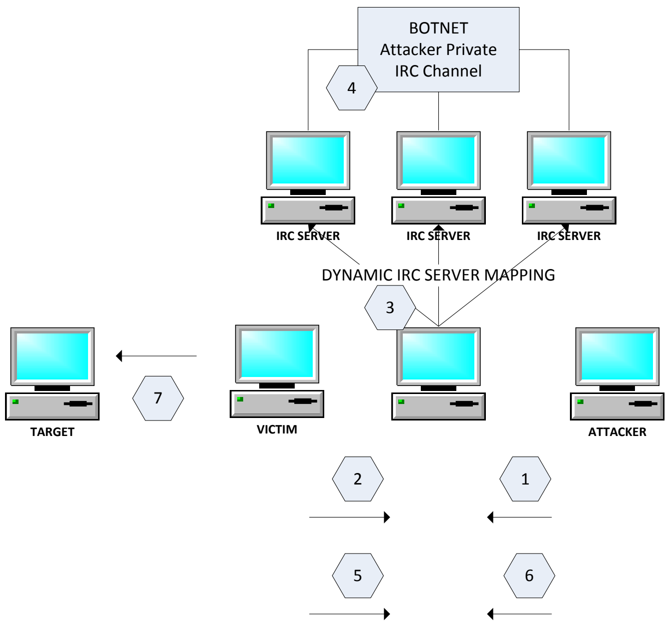

Figure 3. BOTNET IRC[12]

Therefore, the IRC protocol is a popular protocol used in botnet communication. Mechanism of Botnet IRC infection and guide process [12] described as follow :

- Botmaster scans and tries to inject infection through IRC server.
- Once the target is infected and the bot is installed, the bot id is randomly assigned as a channel for the botmaster.
- The Bot makes a Request on the DNS server and mappings the IP address on the IRC server dynamically.
- Next, Bot joins private IRC channel and standby for receive command to attack.
- Botmaster sends launch attack command into private IRC, bot executes the command.
- After that Botmaster enters IRC server by entering password authentication. When Botmaster is accepted, it will launch command that has been scheduled.
- At the end of the process, the bot will get command and launch it.

b) Botnet HTTP

Since the use of IRC Botnets has been recognized, researchers are beginning to focus on tracking the whereabouts of IRC botnets.Therefore, the attacker initiated the use of HTTP Botnet, in this way the botnet becomes hard to find. This is because the botnet uses the HTTP protocol to hide the existence of its bot in web traffic, so botnets can easily bypass the firewall using a port-based filtering mechanism and avoid detection from IDS.

Under normal circumstances, the firewall will block both incoming traffic and outbound traffic to unwanted ports which include IRC ports. Several examples of HTTP botnet are Rustock, Bobax, and ClickBot.

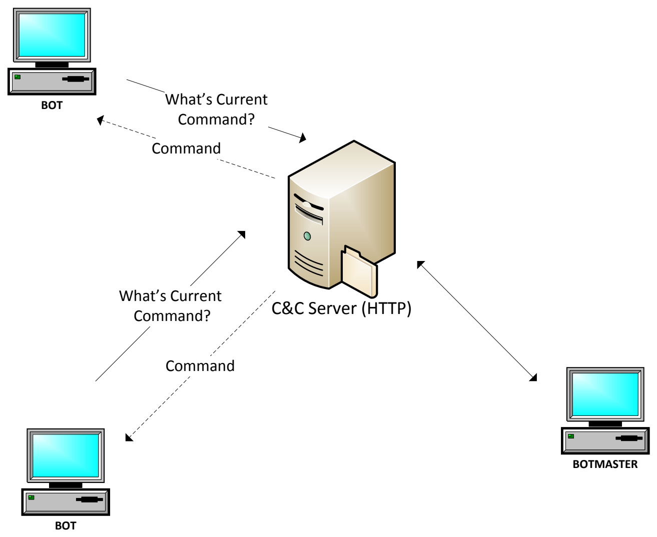

Figure 4. HTTP Botnet [10]

Since the use of IRC Botnets has been recognized, researchers are beginning to focus on tracking the whereabouts of IRC botnets.Therefore, the attacker initiated the use of HTTP Botnet as shown in figure 4, in this way the botnet becomes hard to find. This is because the botnet uses the HTTP protocol to hide the existence of its bot in web traffic, so botnets can easily bypass the firewall using a port-based filtering mechanism and avoid detection from IDS.

Under normal circumstances, the firewall will block both incoming traffic and outbound traffic to unwanted ports which include IRC ports. Several examples of HTTP botnet are Rustock, Bobax, and ClickBot.

2) Hybrid

The next development of P2P botnet is Hybrid P2P botnet [22], bot on this type is classified 2 categories as follows:

a) Servant Bots - In the first category referred to as servant bots, this is because the bot acts as a client and server, which has static, routable IP addresses and is easily accessible from the rest of the internet.

b) Client Bots - Bots in the second category act as clients because they do not accept incoming connections [28]. This category contains the remaining bots as follows:

1) Bots with dynamic IP addressing properties

2) Bots with fixed route IP address

3) Bots that work behind a firewall, they cannot be connected to the global internet.

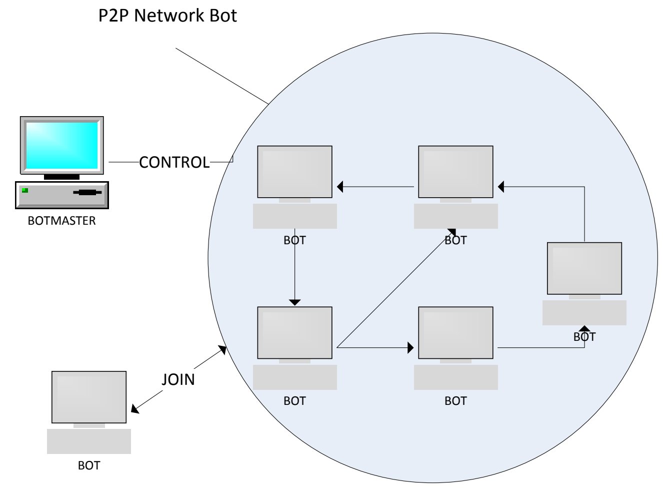

Figure 5. Peer to Peer C&C Server [10]

3) Decentralized / Peer to Peer C&C Server

The next development on botnet topology is decentralized C&C, this development is done because the centralized botnet has been recognized and resolved by the researcher. In decentralized botnets, attackers use peer to peer communication systems as a pattern of C&C, which in turn provides an advantage in avoiding network failures [30]. Every bot in this topology has limited size and periodically every bot connects to a neighbor to receive a command from botmaster. So Botmaster in this topology only needs to connect with one bot to send command.

Figure 5 shows the absence of a centralized point in communication, each bot keeping in touch with other bots. On the other hand, the bot also acts as a client and server in forwarding information. The newly infected bot must know the other botnet connected to it. If the bot in the botnet is offline, other bots can continue operations under the botmaster command.

The first known botnet using the P2P protocol in its communication was the Slapper worm that appeared in 2003 [13]. Other notable examples of P2P Botnets are Sinit [15], Nugache in 2005 [17], and Storm Worm in 2007.

P2P botnet aims to eliminate or hide the existence of a central point of failure or major weakness in a centralized model.

here are some models of botnet P2P and its features and characteristics :

1. Slapper

Slapper allows routing commands for different nodes. Use public key / private cryptography key for command authentication. The botmaster marks the command with a private key, and only the node that has the corresponding public key can accept commands[13]. There are two main points of weakness in this type of botnet, among others are:

a) This botnet type maintains a list of known botnets. So when one bot is caught, it can reveal the entire botnet[13].

b) Botnet mechanism is quite complicated, resulting in high traffic. This provides a detectable botnet opportunity based on network flow data.

2. Sinit

This type of botnet uses random probing to find communication with other botnets. This gives a weakness, that is easily detected due to extensive probing traffic[18].

3. Nugache

The weakness of this type of botnet lies in the dependence on the botmaster on sending a list of 22 IP addresses during the bootstrap process[19].

4. Phatbot

The use of the Gnutella cache server for the bootstrap process has the effect of easily disabling the botnet. In addition, P2P protocols are used redundant and disadvantageous[20].

5. Storm Worm

This botnet type utilizes Overnet P2P Protocol in controlling bots. Overnet Protocol implements hash of distributed tables based on the Kademlia algorithm described in the paper [21]. Based on Grizzard et al [22] paper analysis of network data traces, the communication protocol for a trojan.

This botnet type can be divided into 5 stages of work, described as follows:

a) Connect with Overnet-Bots. Each Bot starts with hard-coded binary files that are included in IP addresses of P2P based on Botnet nodes.

b) Find and download a second injection URL that uses hard-coded keys to search and download URLs in the Overnet network.

c) Decrypt The second injection, the bot uses a hard-coded key to decrypt the encrypted URL. Currently, Bot has a URL address of the second injection that can be activated.

d) Download the second injection, Bot downloads the second injection from the web server using the decrypted URL. This can either be an infected file or an addition to a list of P2P nodes

e) The second injection execution, Bot executing from the second injection, allows for the scheduling of future upgrades within the P2P network.

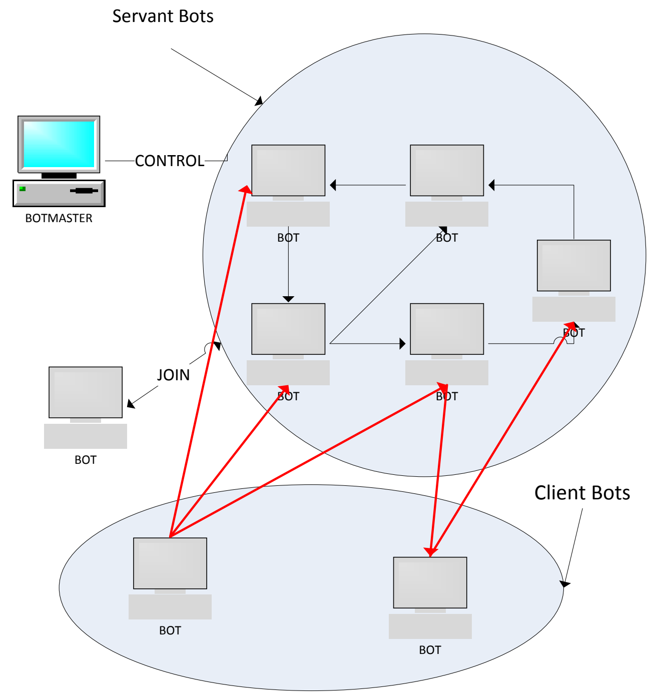

Figure 6. Hybrid P2P Botnet [10]

4) Hybrid

The next development of P2P botnet is Hybrid P2P botnet [22], bot on this type is classified 2 categories as follows:

c) Servant Bots - In the first category referred to as servant bots, this is because the bot acts as a client and server, which has static, routable IP addresses and is easily accessible from the rest of the internet.

d) Client Bots - Bots in the second category act as clients because they do not accept incoming connections [28]. This category contains the remaining bots as follows:

4) Bots with dynamic IP addressing properties

5) Bots with fixed route IP address

6) Bots that work behind a firewall, they cannot be connected to the global internet.

In Figure 6 Hybrid P2P Botnet [22] shown some features as follows:

a) Only the IP address of Servant Bots as a candidate on the peer list. This ensures peer list on each bot has a long lifetime.

b) Each Servant bot independently determines the incoming port and generates an encryption key on incoming connections. This makes it harder to detect the existence of botnets based on network flow.

c) Botmaster enters commands through any bot in the botnet. All bots are periodically associated with servant bots on the peer list that aims to receive commands from the botmaster. When the bot takes a new command, the bot will immediately forward the command to the entire servant bot in its peer list.

d) Botmaster uses the host sensor to monitor the entire botnet, the host's IP address always changes.

SDN Defense Mechanism against Botnets

Several previous studies on botnet mitigation on the traditional tissue can be found in the paper [24][25][26]. Generally, detection methods that have been done before on traditional network are based on the habit and pattern of network traffic. Paper [23] discusses the detection of P2P botnets divided into 5 main components as shown in Figure 8 with the following explanation:

1. Traffic Flow and Feature Extractor Module

At this stage, the network traffic from different hosts is classified with the aim of obtaining information obtained from network flow.

2. P2P Application Detection Module

   This module has the main purpose of obtaining the vector feature. In the process this module is divided into 4 sections as follows:

   1. Building Detection Model and Training set
   2. Feature Selection
   3. Classifier
   4. Traffic Analysis

3. Report to OpenFlow Controller

   If the classification result of the detection agent matches the class on the P2P botnet, the detection agent will inform the rule arbitrator to adjust the flow entries in the data link bridges.

4. Flow Rule Modify on OpenFlow Switch

   Once the Rule Arbitrator receives a RESTful HTTP request sent from the detection agent, the flow table will be modified by a RESTful HTTP request in the data-link bridges.

5. Drop, Forward, or Redirect Packet

Figure 8 discusses the sequence diagram in the bot detection scheme, following the work stages in the diagram:

1. Initialize Flow Tables
2. Send a registering packet
3. Send Packet-in
4. Add new flow rules to mirror
5. Generate network traffic from P2P botnet
6. Mirror Packets
7. Report result via Restful API
8. Add new flow rules to manage network traffic
9. Generate Network traffic from P2P botnet

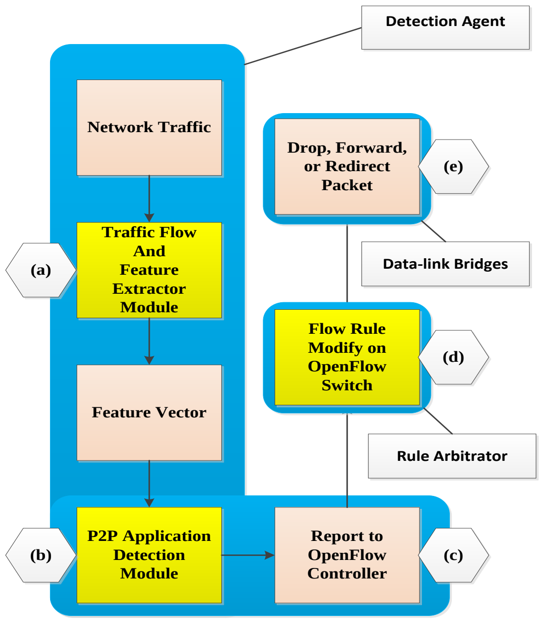
Detection Agent
Network Traffic
Traffic Flow And Feature Extractor Module
Feature Vector
P2P Application Detection Module
Drop, Forward, or Redirect Packet
Data-link Bridges
Flow Rule Modify on OpenFlow Switch
Rule Arbitrator
Report to OpenFlow Controller
(a)
(b)
(c)
(d)
(e)

Figure 7. Botnet detection flow diagram of SDN [23]

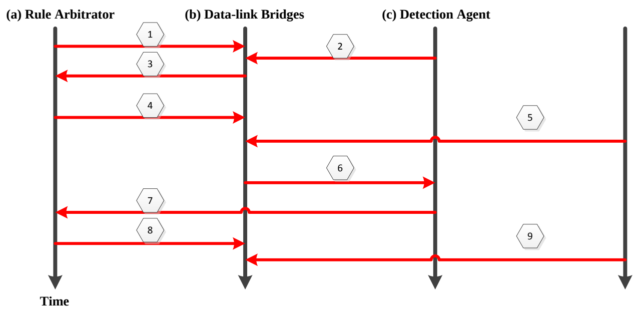

Figure 8. Sequence Diagram of Bot Detection [23]

CONCLUSION AND FUTURE WORK

The SDN architecture that separates the functions of the control plane and data plane provides advantages on various sides, the ease of altering the functioning of the network system by programming the network-based functions of the network, the cost efficiency of development and network enhancement, and providing opportunities for adaptation of new technology developments to SDN.

However, the development of research on the SDN security system is still low at the moment, as evidenced by the lack of research that addresses the defense aspects of the variation of attacks on SDN network security. Based on the review paper it can be found that there are some aspects that need to be of concern to the SDN security system, especially in Botnet attacks. With this problem we need to develop various security methods in handling attack variation, this paper is devoted to variations of a Botnet attack. One of the most potential security methods in handling botnets is the handling of botnets with the honeypot system. Where honeypot has proven able to cope and learn botnet attack on the traditional network.

REFERENCES

[1] S. Shin, L. Xu, S. Hong and G. Gu, "Enhancing Network Security through Software Defined Network (SDN)"IEEE, 2016.

[2] S. Scott-Hayward, S.Natarajan, and S. Sezer , “A Survey of Security in Software Define Network” IEEE Communication Surveys & Tutorials, Vol.18, 2016.

[3] S. Hong, R. Baykov, L. Xu, S. Nadimpali, and G. Gu, “Towards SDN-Defined Programmable BYOD (Bring Your Own Device) Security” NDSS’16, 2016.

[4] S. Shin, Wang, and G. Gu, “First Step Toward Network Security Virtualization: From Concept to Prototype” IEEE Transaction on Information Forensics and Security, 2015.

[5] M. Mitchiner, “Software-Defined Networking and Network Programmability: Use Cases for Defense and Intelligence Communities” CISCO White Paper, 2014.

[6] S. Shin, P. Porras, V. Yegneswaran, M. Fong, G. Gu, and M. Tyson, “FRESCO: Modular Composable Security Services for Software Defined Networks”

Proceedings of the 20th Annual Network and Distributed System Security Symposium (NDSS'13), 2013.

[7] J. Sonchack, A. J. Aviv, E. Keller, and J. M. Smith, "Enabling Practical Software Defined Networking Security Application with OFX" NSDI'16, 2016.

[8] S. Sezer *et al.*, "Are we ready for SDN? Implementation challenges for software-defined networks," *IEEE Communication. Magazine.*, vol. 51, no. 7, pp. 36–43, Jul. 2013

[9] D. Drutskoy, E. Keller, and J. Rexford, "Scalable Network Virtualization in Software Defined Networks" IEEE Internet Computing, Vol. 17, No. 2, 2013.

[10] Hossein, R. Zeidanloo, Azizah, A. Munaf, "Botnet Command and Control Mechanism" 2nd International Conference on Computer and Electrical Engineering, 2009.

[11] Z. Chi, Z. Zhao, "Detecting and Blocking Malicious Traffic Caused by IRC Protocol Based Botnets" IEEE, IFIP International Conference on Network and Parallel Computing Workshop, 2007.

[12] R. Puri, "Bots &; Botnet: An Overview" Research on Topics in Information Security, 2003.

[13] I. Arce, E. Levy, "An Analysis of The Slapper Worm" IEEE Security and Privacy Magazine, 1(1):82-87, 2003.

[14] J. Leonard, S. Xu, R. Sandhu, 2009. A Framework for Understanding Botnets. *International Conference on Availability, Reliability, and Security (ARES 2009)* 917-922.

[15] R. E. Overill, J. A. M. Silomon, "A Complexity Based Forensic Analysis of the Trojan Horse Defence" IEEE, Sixth International Conference on Availability, Reliability, and Security, 2011.

[16] S. Silva, R. Silva, R. Pinto, R. Salles, 2013. Botnets: A survey. *Computer Networks*, 57 (2): 378-403.

[17] S. Stover, D. Dittrich, J. Hernandez, and S. Dietrich. Analysis of the storm and nugache trojans: P2P is here. *;login:*, 32(6), 2007.

[18] A. Karasaridis, B. Rexroad, and D. Hoein, "Widescale botnet detection and characterization," Proc. of Hot Topics in Understanding Botnets (HotBots'07), April 2007

[19] D. Diitrich, S. Dietrich, "P2P as Botnet Command and Control: A Deeper Insight" IEEE, Malicious and Unwanted Software 3rd Conference, 2008.

[20] P. Wang , S. Sparks, C. Zou. An advanced hybrid peer-to-peer botnet. Hotbots'07 workshop program, School of Electrical Engineering and Computer Science, University of Central Florida, 2007.

[21] P. Maymounkov, D. Mazières, "Kademlia: A peer-to-peer information system based on the XOR metric," 1st International Workshop on Peer-to-Peer Systems, March 2002, pp. 53-62.

[22] J.B. Grizzard, V. Sharma, and C. Nunnery. "Peer-to-Peer Botnets: Overview and case study," Proc. of the 1st Workshop on Hot Topics in Understanding Botnets (HotBots 2007). 2007.

[23] Shang-Chiuan Su, Yi-Ren Chen, Shi-Chun Tsai, and Yi-Bing Lin, "Detecting P2P Botnet in Software Defined Networks"Security and Communication Networks, 2017.

[24] Z. Abaid, M. Rezvani, S. Jha, 2014. MalwareMonitor: An SDN-based Framework for Securing Large Networks. *Proceedings of the 2014 CoNEXT on Student Workshop.* Pages 40-42.

[25] M. Stevanovic, J. Pedersen, 2014. An efficient flow-based botnet detection using supervised machine learning. *International Conference on Computing, Networking, and Communications (ICNC)* 797 – 801.

[26] S. Shin, Z. Xu, and G. Gu, 2012. EFFORT: Efficient and Effective Bot Malware Detection. *Proceedings of the 31th Annual IEEE Conference on Computer Communications (INFOCOM 12).* 2846-2850.

[27] F. Hu, Q. Hao and K. Bao, 2014. A Survey on Software-Defined Network and OpenFlow: From Concept to Implementation. *IEEE Communications Surveys & Tutorials.* 16(4), 2181–2206.

[28] Y. Jarraya, T. Madi and M. Debbabi, 2014. A Survey and a Layered Taxonomy of Software-Defined Networking. *IEEE Communications Surveys & Tutorials.* 16(4), 1955-1980.

[29] D. Kreutz, F. Ramos, P. Verissimo, C. Rothenberg, S. Azodolmolky and S. Uhlig, 20115. Software-Defined Networking: A Comprehensive Survey. em Proceedings of the IEEE. 14-76.

[30] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, Jon Zolla, Urs Holzle, Stephen Stuart, ¨ and Amin Vahdat. B4: Experience with a Globally-deployed Software Defined Wan. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, 2013.