Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 86 (2016) 437 – 440

Procedia Computer Science

2016 International Electrical Engineering Congress, iEECON2016, 2-4 March 2016, Chiang Mai, Thailand

# A Security Analysis of a Hybrid Mechanism to Defend DDoS Attacks in SDN

Saksit Jantila and Kornchawal Chaipah*

*Department of Computer Engineering, Faculty of Engineering, Khon Kaen University, Khon Kaen 40002, Thailand*

**Abstract**

Software Defined Network (SDN) is a new network architecture based on centralized management that configures a network in real time through a controller. In this paper, we analyze the vulnerability of an SDN security system in the midst of a DDoS attack. We regard an existing security mechanism, which employs a trust value and entropy computed by client's access behaviors, as a security mechanism of a controller. We analyze this security system using the STRIDE threat model. In addition, suggestions when designing a secure application for an SDN will be discussed in this paper.


Peer-review under responsibility of the Organizing Committee of iEECON2016

*Keywords:* Sofware Defined Network (SDN); STRIDE; DDoS;

## 1. Introduction

A software defined network (SDN) is a new network architecture that increases efficiency for configuration in real time. SDN separates the control plane from the data plane, making it easier to manage the network using software. The two planes communicate with the OpenFlow protocol[1]. The control plane contains information related to routing traffic, network topology, etc., all of which are controlled by a controller. The data plane forwards traffic following configuration in a controller.

* Corresponding author. Tel.: +66-43-362160.
*E-mail address:* kornchawal@kku.ac.th


Peer-review under responsibility of the Organizing Committee of iEECON2016
doi:10.1016/j.procs.2016.05.072

DDoS attack – an invasion by attacker(s) to interrupt legitimate users from getting a service – uses a large amount of compromised users (botnet) to exhaust a victim's resources such as CPU, memory, bandwidth, database, and socket. It is challenging to distinguish legitimate users from compromised users because they produce seemingly similar traffic patterns. Attacking entities in an SDN would be considered an application level attack due to the SDN architecture.

In this paper, we analyze the vulnerability of an SDN system that adopts a trust value and an entropy concept in the midst of a DDoS attack on the application layer using the STRIDE threat model[2]. Although the SDN architecture has security mechanisms such as OpenFlow Random Host Mutation[3] and Resonance[4], they do not directly deal with DDoS attacks. We examine a security mechanism dealing with DDoS attacks in a traditional network called a hybrid approach to counter application layer DDoS attacks[5] proposed by S. R. Devi and P. Yogesh. The mechanism employs a trust and entropy values to differentiate attackers from legitimate users. We have not found any work that analyze and adapt such mechanism for an SDN. Therefore, we investigate how effective it could be applied as a security mechanism against DDoS attacks in SDN.

The rest of the paper is organized as follows: We explain related work in the next section. Then the concept and design of a security mechanism using trust values and entropy are discussed. Next, we analyze the security in SDN using STRIDE and finally conclude the paper.

## 2. Related Work

### *2.1. Security Applications for SDN*

OpenFlow Random Host Mutation[3], proposed by J.H. Jafarian et al, frequently changes an IP address of a host in a network to ensure that an attacker cannot identify the vulnerable victim easily. Traditionally, the network configuration is static, therefore, not adaptive to varied situations. Resonance[4], proposed by A. Nayak et al, employs a dynamic access control in the network with OpenFlow that is a potential solution to protect the network and resources against being tampered with data, information disclosure and DoS. These mechanisms, while designed for SDN, nevertheless, do not directly focus on combatting DDoS attacks.

### *2.2. Security Applications for Traditional Network*

The defense mechanisms against the application level DDoS attacks are classified into[6]: 1) a destination-based mechanism which is managed on a server to handle malicious requests, and 2) a hybrid mechanism which is a collaboration between a client and a server to detect and respond to attacks[5,7,8,9]. CAPTCHA[8], proposed by L. Ahn et al, generates a puzzle for each client who solves the puzzle and can get access to a server. This approach has high accuracy in distinguishing botnet from legitimate users, but it is annoying for most users and causing delays. Another hybrid mechanism, proposed by S. R. Devi and P. Yogesh[5], employs the trust value and entropy information derived from clients' access behaviors. It allows only clients with valid trust values and entropy deviations to get access to a server. Although the approach is appropriate in the traditional network, it has not been examined for use in SDN.

### *2.3. STRIDE*

STRIDE[2] is a threat model for identifying design flaws in a security reliant mechanism on an application level. The security mechanism we propose is planned to work on an application level. We, therefore, analyze our mechanism using the STRIDE model. STRIDE consists of abbreviations of security threats, each of which an application must be able to handle. S stands for Spoofing identity; T stands for Tampering with data; R stands for Repudiation; I stands for Information disclosure; D stands for Denial of service; and E stands for Elevation of privilege.

## 3. Security Mechanism Concept & Design

The best way to prevent a DDoS attack is to authorize each client before allowing them to access a server. We examine a security mechanism proposed by S. R. Devi and P. Yogesh[5]. The mechanism aims to distinguish legitimate users from attackers based on a client's access behavior. Each client is assigned a trust value by a server based on its access behaviors such as the rate of access. Every time a client starts a session, the trust value will be checked and updated in the server. A client with a valid trust value will be given the priority over other requests. A new client is assigned with the default trust value which should be higher than those of attackers and lower than those of legitimate users. Besides computing a trust value, a server also performs the entropy computation for a client per session. The entropy deviation value measures the changes of randomness of requests at an interval. The entropy deviation value is the second filter for identifying an attacker who initially acts like a legitimate user to gain a high trust value, but misbehaves later.

When applied for SDN, this mechanism operates in a controller instead of a server to reduce the load of a web server and to gather controlling functions at a controller for easy management. The SDN system in Fig. 1.a. consists of four components: a web server, a controller, an SDN switch, and clients. In order for a client to contact the web server, it needs to go through a switch and a controller. In our proposed mechanism, the web server sends client's behavior information to a controller after a session ends. The controller computes and updates a trust value and entropy deviation of the behavior and stores them for future use. We allow access to the stored trust values only for connections via internal data flow; i.e. connections from entities within the privilege boundary. Therefore, attackers outside the privilege boundary cannot tamper with the trust values.

Fig. 1.b. shows how the proposed mechanism works in a controller. When a session arrives, the controller checks the client's trust values calculated from the past sessions. If the trust value is valid, the controller continues with checking the deviation of entropy; otherwise, the session is dropped. If the deviation exceeds the threshold, then the session is dropped immediately and the lowest trust value is assigned to the client. If the deviation of entropy value is within the threshold, the session is forwarded to the web server. After the session ends, the web server sends the client's access behavior to the controller. The controller updates the trust value of the client, computes the deviation of entropy, and stores these values to use the next time the client opens a new session.

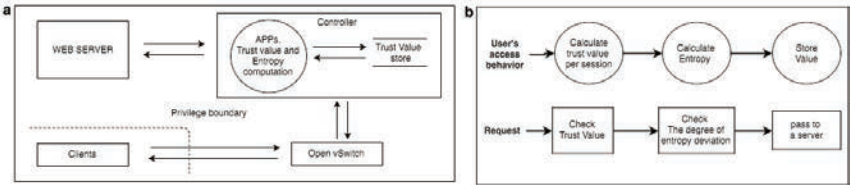


Fig. 1. (a) An SDN system with a web server; (b) A trust value mechanism in a controller.

## 4. Security Analysis

Each entity in an SDN is associated with different threats. In this section, we analyze how each entity combats with different threats using the proposed mechanism and the SDN properties. The security analysis is discussed based on the characters of STRIDE as follows: First, Spoofing identity (S) is a masquerade of an attacker's IP Address. There is no explicit defense of spoofing in SDN just yet. However, communication between a controller and a switch is secured by Transport Layer Security (TLS), supported by the OpenFlow protocol. Therefore, it is unlikely that an attacker could spoof a controller's or a switch's IP addresses.

Tampering (T) with data and Information disclosure (I) are associated with the data store. We consider providing security for the trust value store to prevent an attacker from modifying or disclosing the information without permission. As mentioned before, we only allow entities in the privilege boundary to access the data. If IP addresses of such entities cannot be spoofed, then the data should be safe. However, we may have additional security by employing authentication and authorization mechanisms. Moreover, backing up should be applied to prevent data

loss. Next, Repudiation (R) is a serious threat as a result of the system lacking the access authentication. A controller is secured against this threat because whoever wants to obtain the service must install its own information into a flow table via a switch. Hence, the repudiation problem is mitigated.

Elevation of Privilege (E) happens when an attacker modifies its privilege to access a web server and a controller without a permission. The solution is to use authentication and authorization mechanisms. In general, a web server is secured from the elevation of privilege by secure shell access, in which a server uses public/private key pairs for authentication instead of using passwords. The security of trust values stored in a controller is also viable by encryption and authentication. Lastly, Denial of Service (D) is a fatal threat that is our biggest concern. When a server faces high resource usage where the number of requests is great during a very short period, a DoS attack occurs. In general, an administrator can shutdown processes or resources in a server. However, we recommend using our proposed trust-based mechanism in a controller to prevent and mitigate an effect of a DDoS attack.

Table 1 summarizes the security analysis. We can see that for a server and a controller in SDN, which are likely the targets for an attack, all threats can be prevented or mitigated. This is done through authentication (both public/private keys and passwords), encryption, and using a trust value based mechanism.

Table 1. Summary of the security analysis where √ denotes the threat can be mitigated, O denotes the threat is ignored or not significant, and x denotes the threat cannot be mitigated.

| Entity | Controller | Server | Switch | Client |
|---|---|---|---|---|
| S | √ | √ | √ | O |
| T | √ | √ | O | O |
| R | √ | √ | √ | O |
| I | √ | √ | √ | √ |
| D | √ | √ | O | O |
| E | √ | √ | O | O |

## 5. Conclusion

In this paper, we propose adapting a hybrid mechanism against DDoS attacks from the traditional network for SDN. The mechanism relies on trust values and entropy based on clients' access behaviors. We identify threats to this application and suggest use of existing SDN's and our proposed mechanisms to prevent and mitigate the threats. Moreover, authentication, encryption, and the use of public/private keys play important roles in keeping entities in SDN safe from attackers. In the future, we plan to implement and simulate the proposed mechanism in a virtual SDN to assess the mechanism's effectiveness and efficiency, and to identify any flaws we might have overlooked. We hope that the results of our experiment offer better understanding of providing security in an SDN.

## References

1. OpenFlow Switch Specification version 1.5 [Internet]. Open Networking Foundation. 2016 [cited 30 March 2016]. Available from: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.5.0.noipr.pdf
2. Hernan S et al. Uncover Security Design Flaws Using the STRIDE Approach [Internet]. [cited 30 March 2016]. Available from: http://msdn.microsoft.com/en-gb/magazine/cc163519.aspx
3. Jafarian J, Al-Shaer E, Duan Q. OpenFlow Random Host Mutation: Transparent Moving Target Defense using Software Defined Networking. HotSDN. 2012. p. 127-132.
4. Ankur N, Alex R, Nick F, Russ C. Resonance: Dynamic Access Control for Enterprise Networks. WREN. 2009. p. 11-18.
5. Renuka Devi s. A Hybrid Approach to Counter Application Layer DDOS Attacks. International Journal on Cryptography and Information Security. 2012;2(2):45-52.
6. Saman Z, James J, David T. A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks. IEEE Communication Surveys & Tutorial. 2013. p. 2046-2069.
7. Jie Y, Chengfang F, Liming L, Zhoujun L. A Lightweight Mechanism to Mitigate Application Layer DDoS Attacks. in Institute for Computer Science, Social-Informatics and Telecommunications Engineering. 2009. p. 175-191.
8. Luis A, Manuel B, Nicholas H, John L. CAPTCHA: Using Hard AI Problems for Security, in EUROCRYPT. 2003. p. 294-311.
9. Walfish M, Vutukuru M, Balakrishnan H, Karger D, Shenker S. DDoS defense by offense, SIGCOMM Computer Communications. 2006. p. 303-314.