Defense Against TCP Flooding Attack

Seungyong Yoon, Jintae Oh, Ikkyun Kim and Jongsoo Jang
Managed Security Research Team, Electronics and Telecommunications Research Institute,
161 Gajeong-dong, Yuseong-gu, Daejeon, 305-700, Republic of Korea

Keywords: DDoS, TCP Flooding Attack.

Abstract: This paper generally relates to a DDoS attack prevention method, more particularly, to a Transmission Control Protocol (TCP) flooding attack prevention method which defines several session states based on the type and direction of a packet, tracks the session state for each flow, and detects and responds to a flooding attack. An anti-DDoS system with a capacity of 20Gbps throughput, we call 'ALADDIN' system, was implemented in FPGA based reconfigurable hardware. The possibility of high-speed hardware implementation was shown in this paper. The system was tested using existing DDoS attack tools in 8Gbps of background traffic. According to the test results, TCP flooding attacks could be defended through our proposed method rapidly and accurately.

1 INTRODUCTION

Denial-of-Service (DoS) attack is undoubtedly a very serious problem in the Internet, whose impact has been well demonstrated in the computer network literature. According to the WWW Security FAQ (Stein, 2002) a DoS attack can be described as an attack designed to render a computer or network incapable of providing normal services. Distributed Denial of Service (DDoS) attack uses many computers to launch a coordinated DoS attack against one or more targets. Using client/server technology, the perpetrator is able to multiply the effectiveness of the DoS significantly by harnessing the resources of multiple unwitting accomplice computers, which serve as attack platforms (Christos, 2004).

This paper generally relates to a DDoS attack prevention method, more particularly, to a Transmission Control Protocol (TCP) flooding attack prevention method which defines several session states based on the type and direction of a packet, tracks the session state for each flow, and detects and responds to a flooding attack.

The rest of this paper is organized as follows. Section 2 describes the context of prior and ongoing research related to DDoS attack. And section 3 describes TCP flooding defense mechanism in detail. Implementation and experimental results are contained in section 4. And conclusion and future work are discussed in section 5.

2 RELATED WORK

A DDoS attack is divided into a network level attack and an application level attack (Xie, 2009). The network level attack designates a network layer attack such as TCP flooding, User Datagram Protocol (UDP) flooding, and Internet Control Message Protocol (ICMP) flooding. The application level attack designates an application layer attack such as Hypertext Transfer Protocol (HTTP) flooding, Session Initiation Protocol (SIP) flooding, Domain Name Server (DNS) flooding and so on. Since the attack properties of the two types of attacks are different from each other, the detection and response methods are different from each other.

Most of existing DDoS prevention techniques use a simple method of measuring the amount of traffic volume, such as Bit per Second (BPS) or Packet per Second (PPS), and blocking packets for a predetermined time if the amount of traffic volume is greater than a threshold (Talpade, 1998); (Huang, 2001). Further, Intrusion Detection System/Intrusion Prevention System (IDS/IPS) products use a method of applying string patterns, which mainly appear in a DDoS attack tool, to detection rules, performing a pattern matching function, and instantly blocking a corresponding packet when the packet is detected

DOI: 10.5220/0004119604160420
In *Proceedings of the International Conference on Security and Cryptography* (SECRYPT-2012), pages 416-420
ISBN: 978-989-8565-24-2

(NFR, 2007); (Snort, 2012). However, since there are limits on simple pattern matching, attempts at effective response have been recently made by providing priority queues combined with Quality of Service (QoS) or applying a rate limiting scheme (Mirkovic, 2002); (Kargl, 2001); (Garg, 2002).

However, such existing DDoS prevention techniques perform detection and response based on the basically simple amount of traffic volume and string patterns, so that there are limits on realizing rapid and accurate prevention in an actual DDoS attack situation.

3 DEFENSE MECHANISM

3.1 Basic Architecture

The goal of this paper is to provide a TCP flooding attack prevention method. Figure 1 shows the basic architecture.

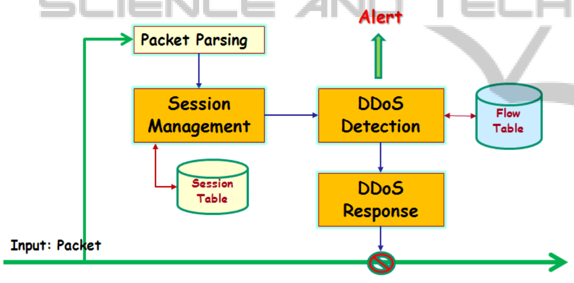

Figure 1: Basic architecture.

Once the packet is inputted, a packet parser extracts necessary information for attack detection from the packet. The TCP flooding attack prevention system is composed of a session management module, a DDoS detection module, and a DDoS response module.

The session management module defines several session states based on the types and directions of respective packets which are received at an intermediate stage between a client and a server. In a session establishment and termination process, packets are exchanged at respective steps. Here, the session state may be variously defined based on the types and directions of the respective packets so that the system can detect and response various types of TCP flooding attacks. The direction of each of the packets may be determined in such a way as to compare source IP address with destination IP address. The session management module may differently define the session state based on the direction of a corresponding packet. The session state will be described next chapter in detail.

The DDoS detection module tracks session state for each flow, detects a TCP flooding attack, and identifies the type of the TCP flooding attack. In order to detect a TCP flooding attack, basically, session state for each flow should be tracked by the attack detection module. The flow may be defined as a set of packet streams having common properties. In this paper, flow is defined as a set of packet streams, having the same <destination IP> or the same <destination IP, destination port>. If specific session state is tracked for each flow, TCP flooding can be detected and identified for each attack type.

The DDoS response module responds to the TCP flooding attack, detected using the attack detection module based on the type of the flooding attack. That is, when the DDoS detection module tracks session state for each flow and then detects a TCP flooding for each type thereof, the DDoS response module appropriately responds according to each attack type.

3.2 Session State Definition

The conventional state transition for the establishment and termination of a TCP session between a client and a server may include three steps, that is, a session establishment step using 3-way handshaking, a data request and transmission step, and a session termination step using 4-way handshaking. State transition which is appropriate for the client and the server is realized in such a way as to exchange a connection request (SYN) packet, an acknowledgement (ACK) packet, and a termination (FIN) packet.

In this paper, we used the session table architecture of (Seungyong Yoon, 2008) and newly defined several session states. The new session state, which is different from session state defined based on the conventional TCP state transition (Gordon Mckinney, 2002), are defined in order to detect a DDoS attack at the intermediate stage between a client and a server.

Figure 2 shows a new transition diagram of the session state defined in this paper. According to the packet type (SYN, SYN+ACK, ACK, FIN, …) and direction (Client to Server, Server to Client), each session state transit the next state appropriately. The last ACK packet of the TCP 3-way handshaking process is always transmitted from the client to the server. Therefore, if the session state is differently set based on the size of two IP addresses, the direction of a packet obtained after the session is created can be easily known, that is, whether the

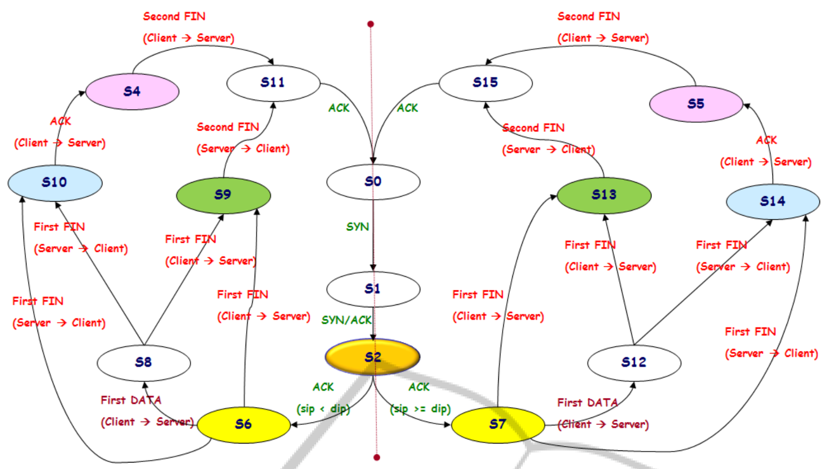

Figure 2: State transition diagram.

packet is headed for the server from the client or the packet is headed for the client from the server, can be easily ascertained.

3.3 Attack Type and Defense

DDoS attacks at the network level may be divided into TCP flooding, UDP flooding, and ICMP flooding based on the type of a protocol. Further, TCP flooding attack may be divided into various types such as SYN flooding, ACK flooding, FIN flooding, and so on. Our proposed method provide a solution of detecting and responding to 6 types of TCP flooding attacks such as SYN flooding, flag flooding, open flooding, connection flooding, FIN-WAIT-1 flooding, and FIN-WAIT-2 flooding.

3.3.1 TCP SYN Flooding

The SYN flooding attack is the most representative DDoS attack in which a large amount of SYN packets is transmitted for the purpose of new session request. In the SYN flooding attack, a large number of SYN packets are transmitted to a server but no ACK packet is transmitted in response to SYN+ACK packets transmitted from the server. Therefore, the server is full of sessions having SYN packet received state, that is, "SYN-RCVD" state. The backlog queue of the server is exhausted and no more TCP session can be created, so that the server cannot accept SYN packets from users trying to connect to the server. During the SYN flooding, the ACK packet is not transmitted so that session state in which the client receives the SYN+ACK packet, that is, the session state "S2", rapidly increases. In the case of a normal session, the ACK packet is instantly transmitted and the session state is changed to "S6/S7" state, so that the number of sessions state "S2" is maintained in constant range for a periodic time. Generally, the number of "S2" state is maintained to be lower than threshold. Therefore, if the number of session state "S2" measured for each flow is equal to or larger than the threshold, it could be detected that SYN flooding has occurred.

3.3.2 TCP FLAG Flooding

In the flag flooding attack, a large amount of packets in which a session connection is not established, that is, out-of-state packets, is transmitted. While a TCP 3-way handshaking process used to create normal session connection is not performed, arbitrary TCP flags are generated and a large amount of abnormal packets are transmitted to the server. When a packet other than a SYN packet, for example, an ACK packet, a FIN packet, or a PSH+ACK packet, is received and session state detection reveals that the packet does not correspond to relevant session state, the corresponding packet is regarded as an abnormal packet since the packet is an out-of-state packet. If the number of such abnormal packets, measured for each flow, is equal to or larger than a threshold, it is determined to be the flag flooding.

3.3.3 TCP Open Flooding

After a session connection is created, a session connection is continuously made while data packet is not transmitted. In a normal case, a process of creating a TCP session connection, transmitting data, and then terminating the session is performed. If the open flooding attack is continued, a session connection is continuously performed without performing the process of terminating the session, so that the number of session connections of the server reaches the maximum limit and the server cannot provide a new session connection any more.

In the case of normal session connection, after session connection is created, data transmission is instantly performed, so that the session state is changed to the "S8" or "S12" state, thereby maintaining the number of sessions in which the session state is "S6" or "S7" within a constant range for a periodic time. Therefore, the open flooding could be detected when the number of session state "S6" or "S7" measured for each flow is equal to or larger than a threshold.

3.3.4 TCP Connection Flooding

After a TCP session connection is created, a FIN packet or an RST packet is instantly transmitted while data packet is not transmitted, so that the session is terminated, thereby adding to the load of the server. The connection flooding attack could be detected when the number of sessions, in which the FIN packet or the RST packet is received in the session state "S6" or "S7", is equal to or larger than a threshold.

3.3.5 TCP FIN-WAIT-1/FIN-WAIT-2 Flooding

The FIN-WAIT-1 flooding attack is performed in such a way that an ACK packet is not transmitted in response to a first FIN packet received from the server in order to terminate a session, thereby exhausting the resource of the server. In the case of normal session termination, after the first FIN packet is received, the ACK packet is instantly transmitted and the session state is changed to "S4" or "S5" state, so that the number of sessions of the session state "S10" or "S14" is maintained within a constant range for a periodic time. Therefore, if the number of session state "S10" or "S14" measured for each flow is equal to or larger than a threshold, the FIN-WAIT-1 flooding attack could be detected. The FIN-WAIT-2 flooding attack is very similar to FIN-WAIT-1 flooding attack. Server transmits the first FIN packet used to terminate a session to the client. If the client does not transmit a second FIN packet after receiving the first FIN packet and transmitting an ACK packet, the server is full of sessions of "FIN-WAIT-2" state. Therefore, resources are exhausted, so that the server cannot provide normal service until the sessions of "FIN-WAIT-2" state have timed out.

4 IMPLEMENTATION AND EXPERIMENTS

Our proposed method was implemented on a 20Gbps Anti-DDoS system named 'ALADDIN' (Advanced Layer-free DDoS Defense Infrastructure). The developed system prototype and security board is shown in Figure 3.

The board was configured using two 10Gbps fiber optic interfaces with a MAC layer. It has one load balance chip to distribute the input packets to the DDoS detection engines. The detection engines each process 5 Gbps of data per second. The detection engines were implemented as a daughter board type. The load balancer and detection engines were implemented using a grade-1 speed Vertex-5 Xilinx FPGA. Each engine had 4 external SRAMs for session, flow, and ACL tables. The chips were implemented using the verilog HDL language. The interfaces between the load balance and detection engines were implemented using 64-bit bus widths and a 100Mhz clock speed.

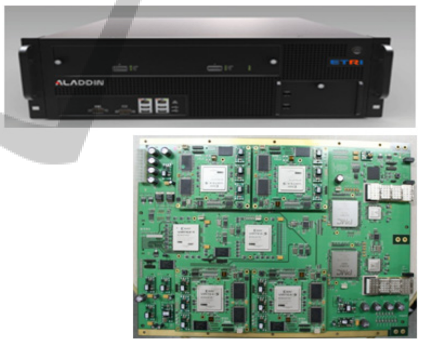

Figure 3: ALADDIN system prototype.

The detection engine processes packets using 32-bit bus widths internally. The engine was synthesized successfully at a clock speed of 178Mhz. The maximum performance of the detection chip is 5.69Gbps. A Synplify pro 9.4 synthesis tool (Synplicity, 2012) was used for hardware synthesis. A Modelsim PE 6.4 simulator (Model, 2012) was used for the logic simulation. An Xilinx ISE 10.1 (Xilinx, 2012) was used for mapping logic and routing the resources in the FPGA.

Our system was tested in a local DDoS testing network using various DDoS tools and test equipment including IXIA (Ixiacom, 2012). Our test environment was configured with two racks, 23 servers, several layer-2 gigabit switches, a 10 gigabit switch with a 24-gigabit interface and two 10Gbit interfaces, and two routers. First, the system was tested using real DDoS attack tools including Netbot Attacker (Han, 2009) and Netkill (Netkill, 2000) under 8 Gbps of background traffic. Second, the system was tested using zombie codes that invoked the 7.7 DDoS attack (Hauri, 2009) in Korea on the 7th of July, 2009.

Table 1 shows the results of test. The test was performed application level attack as well as network level attack including UDP and ICMP flooding attack. In this paper, we only show the test results of TCP Flooding attack.

Table 1: The test results of DDoS attack.

Attack Tool	Attack Menu/Protocol	Result
Netbot Attacker	[01]SYN Flood	TCP SYN Flooding detected.
	[05]TCP Flood	Not detected.
	[06]TCP Multi-Connect	TCP Connection Flooding detected.
	[11]Route Attack	TCP Connection Flooding detected.
	[12]Smart Auto Attack	TCP Connection Flooding detected.
	[13]SYN+UDP Flood	TCP SYN Flooding detected.
	[14]ICMP+TCP Flood	TCP Connection Flooding detected.
	[15]UDP+TCP Connect	TCP Open Flooding detected.
7.7 DDoS	TCP	TCP SYN Flooding detected (spoofed/non-spoofed)
	TCP	TCP Flag (ACK) Flooding detected (spoofed/non-spoofed)
Netkill	TCP	TCP Fin-Wait-1 Flooding detected.

5 CONCLUSIONS

TCP Flooding attack can easily overwhelm a server with big amounts of traffic. Most existing DDoS prevention techniques perform detection and response based on the basically simple amount of traffic or string patterns, so that there are limits on realizing rapid and accurate prevention in an actual DDoS attack situation. Our defense mechanism provide a TCP flooding attack prevention method which defines several session states based on the types and direction of a packet, tracks the session state for each flow, and detects and responds to a TCP flooding attack. A 20Gbps anti-DDoS system, we call 'ALADDIN' system, was implemented with our proposed mechanism. The possibility of high-speed hardware implementation was shown in this paper. The system was tested using existing DDoS attack tools in 8Gbps of background traffic. The system detected TCP flooding attacks during our test. A test in a real network will be prepared in the near future.

REFERENCES

L. D. Stein, J. N. Stewart, 2002. The World Wide WebSecurity FAQ, version 3.1.2, In <*http://www.w3.org/Security/Faq*>.

Christos Douligeris, Aikaterini Mitrokotsa, 2004. DDoS attacks and defense mechanisms; classification and state-of-the art, In *the International Journal of Computer and Telecommunications Networking,* Vol.44, Issue 5.

Yi Xie and Shun-Zheng Yu, 2009. Monitoring the Application-Layer DDoS Attacks for Popular Websites, In *IEEE/ACM Transactions on Networking*, Vol.17, No 1.

R. R. Talpade, G. Kim, S. Khurana, 1998. NOMAD: Trafficbased network monitoring framework for anomaly detection, In *Proceedings of the Fourth IEEE Symposium on Computers and Communications*.

Y. Huang, J. M. Pullen, 2001. Countering Denial of Service attacks using congestion triggered packet sampling and filtering, In *Proceedings of the 10th International Conference on Computer Communiations and Networks.*

NFR Security, 2007. NFR Network Intrusion Detection, In <*http://www.nfr.com*>.

Snort, 2012, The Open Source Network Intrusion Detection System, In <*http://www.snort.org*>.

J. Mirkovic, G. Prier, P. Reiher, 2002. Attacking DDoS at the source, In *Proceedings of ICNP 2002*, pp. 312–321.

F. Kargl, J. Maier, M. Weber, 2001. Protecting web servers from Distributed Denial of Service attacks, In *Proceedings of the Tenth International Conference on World Wide Web*, pp. 514–524.

A. Garg, A. L. N. Reddy, 2002. Mitigating Denial of service Attacks using QoS regulation, In *Proceedings of the Tenth IEEE International Workshop on Quality of Service*, pp. 45–53.

Gordon McKinney, 2002. TCP/IP State Transition Diagram, In *RFC793*.

Seungyong Yoon, Byoungkoo Kim, Jintae Oh, and Jongsoo Jang, 2008. H/W based Stateful Packet Inspection using a Novel Session Architecture, In *International Journal of Computers*, Vol.2, Issue 3.

Synplicity, 2012. In <*http://www.synplicity.com*>

Model, 2012. In <*http://www.model.com*>

Xilinx, 2012. In <*http://www.xilinx.com*>

Ixiacom, 2012. In <*http://www.ixiacom.com*>

K. Han, E. Im, 2009. A Study on the Analysis of Netbot and Design of Detection Framework, In *Proceedings of JWIS.*

Netkill, 2000. Generic remote DoS attack tool, In <*http://www.securiteam.com/tools/5QR0B000AU.html*

Hauri, 2009. 7.7 DDos Virus Report. In <*http://www.maxoverpro.org/77DDoS.pdf*>