ANALYSIS AND DETECTION OF SLOWCOMM AND SLOW NEXT DOS ATTACKS

Marek Sikora
Doctoral Degree Programme (4.), FEEC BUT
E-mail: marek.sikora@vutbr.cz

Supervised by: Vaclav Zeman
E-mail: zeman@feec.vutbr.cz

Abstract: This paper describes the proposal of Slowcomm and Slow Next attack models and their implementation. The attack generator is used to discover the vulnerabilities of several Internet services in an experimental network. Based on the described characteristics of attacks, these attacks' detection methods are investigated and implemented as an intrusion detection system (IDS).

Keywords: Slow DoS attacks, Slowcomm, Slow Next, attack generator, IDS, intrusion detection system, detection signatures.

1 INTRODUCTION

This paper is focused on the slow DoS (Denial of Service) attacks that are independent of the application layer protocol. DoS attacks are trying to prevent legitimate users from accessing a specific service. DoS attacks have gradually evolved from primitive lower-layer flood attacks to sophisticated application-layer attacks. Nowadays, attackers most often attack user-end applications using unsecured vulnerabilities. Due to lower-layer protocols' valid use, effective application attack detection is one of the leading research challenges [1].

1.1 RELATED WORKS

Almost no scientific works deal more deeply with the Slowcomm and Slow Next attacks, so this paper article is based primarily on [2] and [3], which present a novel of these attacks. However, several related works deal with the categorization of slow DoS attacks [1], general detection techniques [4], or summarize methods of prevention and mitigation [5].

1.2 CONTRIBUTION

Due to the absence of a freely available Slowcomm and Slow Next attack generator, this paper presents its implementation of an attack generator. Following current research, this article proposes and implements specific detection signatures for these attacks. Improving network systems security is the primary goal of this paper.

2 SLOW DOS ATTACKS

Slow DoS attacks are characterized by low traffic and similarity to the traffic of legitimate users with a slow Internet connection [1]. Therefore, they can be performed from only one computer, mobile phone, smart or IoT device. The slow DoS attacks strategy consists of establishing the maximum number of connections to the server and keeping them open as long as possible. In this way, the server's input queues will be flooded, which leads to the dropping requests of legitimate users.

2.1 Slowcomm

The Slowcomm attack mechanism is universal and can be used to exploit various application protocols. The attacker establishes the maximum number of concurrent connections to the target server and sends an incomplete valid request. Instead of sending another piece of data, the attacker waits for the maximum time allowed for the connection not to be closed and then sends the next incomplete portion of the request. In this way, the attacker prolongs the transmission of the entire request for as long as possible. The average data transfer rate is usually less than 1 KB/s [2]. Based on the server configuration, the server can close the connection if it does not reach the allowed values, such as the processing time or the data rate. However, the attacker monitors the state of the connection, tries to adapt to the server's response, and re-establish the closed connections to re-occupy the victim's free resources. Thus, a legitimate user does not have the opportunity to establish communication with the server. This attack is practically very similar to the Slowloris attack. However, compared to it, Slowcomm is more sophisticated - it needs less bandwidth; it can attack more application protocols and respond to connection closures [2]. The course of the attack against the Hypertext Transfer Protocol (HTTP) server is shown in Figure 1.

2.2 Slow Next

Similar to Slowcomm, this attack can be applied to various application protocols. It aims to establish a maximum connection with the target server and keep them occupied as long as possible. The attacker sends valid requests to the server, and the server then legitimately responds. The attack exploits the *next* parameter, which is the time between the end of the server response and the beginning of the next

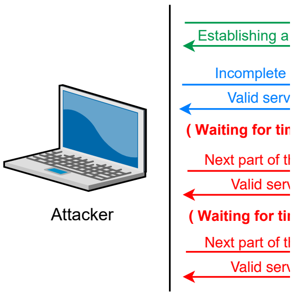

Figure 1: Slowco

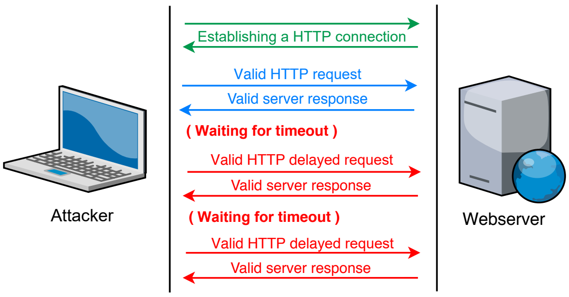

Figure 2: Slow Next attack model.

3 ATTACK GENERATOR

The generator is implemented as a Python 3.0 script, and it uses the Sockets library to connect to the target system. In this paper, the generator is used to attack HTTP, File Transfer Protocol (FTP), and Secure Shell (SSH) protocols.

Mandatory parameters for launching an attack are attack selection, target IP address, target port, and the number of connections the generator should establish. It is also possible to specify the data (payload) that the requests will contain, the frequency of requests, and the number of threads, including the initialization speed (only for Slow Next).

4 ATTACK DETECTION

Based on the obtained attack samples, it was possible to propose Slowcomm and Slow Next attacks detection signatures and implement an intrusion detection system (IDS). This detector was created in Python using the Scapy and LibPcap libraries.

Detection is based on the interception of communications and finding suspicious connections that match the attack's signature. Suspicious connections are stored in a database for subsequent evaluation of the situation. Each signature has a set of defined network communication thresholds for attack detection. Setting these thresholds is key to detection accuracy. Accuracy also depends on the volume of data collected. Because both attacks simulate a legitimate user with a slow connection, they are not detectable at the first occurrence. Detection thresholds are set based on values a normal user cannot reach, but an attack does. Both attacks are detected when more than 20 connections from one IP address are established. To detect Slowcomm attacks, network communication must contain 20 or more unfinished requests at the same time and a time limit of 10 seconds for processing all requests of each connection. To detect Slow Next attacks, network communication must contain more than two seconds between requests within one connection, the spacing between requests is constant with a tolerance of 0.1 seconds, requests size is constant with a tolerance of 10 characters. If any of the conditions are met, network traffic corresponds to the behavior of a Slowcomm or Slow Next attack.

5 TESTING

The test environment consists of four virtual machines - an attacker, a legitimate client, and two servers. In all tested scenarios, the attacker is trying to cause a denial of service to the user.

The targets of the attack were chosen: vsftpd (FTP server), OpenSSH (SSH server), Microsoft IIS, Apache, Nginx, and lighttpd (web servers). All servers were left in the default configuration. The Apache server had active modules `reqtimeout` and `mpm_prefork`.

A summary of the results of vulnerability testing is shown in Table 1. More detailed information on the tests is described in chapters 5.1 and 5.2. Detailed test results are also shown in Figure 3 and 4. Web server availability is expressed by a green line that takes only two values: maximum – the server is available; minimum – the server is unavailable. The attacker's established Transmission Control Protocol (TCP) connections are expressed by the orange line.

Table 1: Discovered vulnerabilities.

Server	Slowcomm	Slow Next
Apache 2.4.29	✓	✓
Nginx 1.14.0	✓	✗
MS IIS 10.0	✗	✗
lighttpd 1.4.45	✓	✓
vsftpd 3.0.3	✓	✓
OpenSSH	✓	✗

5.1 SLOWCOMM ATTACK

The Slowcomm attack on the Apache server was configured to open 300 concurrent connections and send another piece of data every 7 seconds. The web server handles approximately 225 connections, so it is vulnerable to this attack. Even if the reqtimeout module terminates the connection after 20 to 40 seconds, the attack generator immediately re-establishes the connection. The course of this attack scenario is shown in Figure 3.

When testing the vulnerabilities of other web servers, a vulnerability was also found for Nginx and lighttpd servers. The only Microsoft IIS that uses asynchronously processed thread groups was completely immune to the attack [6]. The Nginx server, which has a different event-driven architecture than Apache, was able to handle many more connections with minimal memory requirements [7], but after establishing 800 connections, it began terminating all connections. This paradoxically led to so much server load that it made the website unavailable.

The FTP attack scenario contained the initial data payload `USER <name>`, representing an incomplete FTP message. The following pieces of data always contained one random character. When the attacker established 2000 concurrent connections, the legitimate user could no longer establish a session with the server.

The attack on SSH revealed independence of request content. Data only needs to be incomplete or invalid. Compared to other tested services, SSH requires more computing capacity to establish communication. Establishing 12 connections was sufficient to perform a DoS attack.

5.2 Slow Next attack

The Slow Next attack on the Apache server was configured to open 700 connections and send portions of data every 4 seconds. After establishing approximately 680 connections, the DoS attack was successful. The web server was flooded, and a legitimate user could not connect to the server. The course of this attack is shown in Figure 4. The difference against the Slowcomm attack could be observed in the data rate. Slow Next generates an average data flow of 20 kB/s, thanks to the transmission of valid requests and responses.

When deploying the attack on other Web servers, only the lighttpd vulnerability was discovered. Microsoft IIS and Nginx have not been affected.

The Slow Next attack on FTP was similar to the Slowcomm attack. The server became unavailable when the attacker established around 2000 connections. The Slow Next attack on SSH failed. SSH requires only valid communication. When an SSH message is repeated, the server terminates the communication. Slow Next could not be used for the username entry phase either, as the user has only three attempts and a time limit of 60 seconds. Also, the client must first be prompted by the server to enter his login.

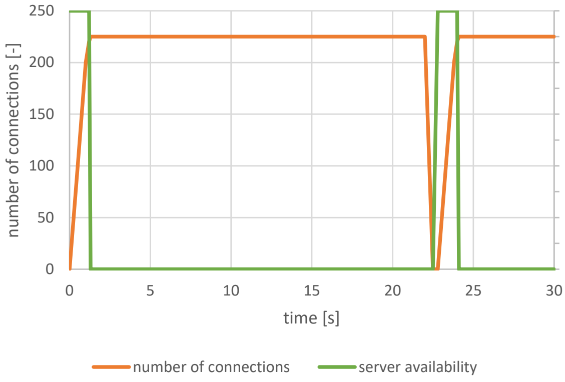

Figure 3: Slowcomm attack on Apache server.

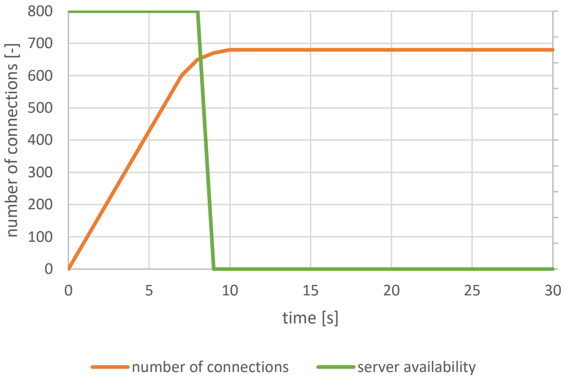

Figure 4: Slow Next attack on Apache server.

5.3 Detection

In the case of non-distributed attack scenarios, the detector detected an attack immediately when 20 established connections with invalid content from one source IP address are exceeded. Detection times differed depending on attack type and settings. The attacks against HTTP were detected approximately 0.2 seconds after launch. In the case of a distributed DoS simulation, each attacker's computer established a single connection. Attacking connections initially appears to be legitimate users, but their behavior over time meets the criteria for classifying an attack. In the HTTP scenario, each Slowcomm attack connection was detected 10 seconds after launch when the allowed time to complete the request was exceeded. Slow Next attack was detected approximately 60 seconds after launch, once enough client requests have been accumulated and their distribution has been analyzed.

6 CONCLUSION

The Slowcomm and Slow Next attack generator proved the vulnerability of the tested HTTP, FTP and SSH servers. Based on the obtained attack samples, IDS was implemented and tested in a laboratory network. In all test scenarios, the attack detection was successful. However, the speed of a distributed attack detection is highly dependent on the appropriate setting of the allowable parameters of network traffic. A detection delay may vary. Future work may focus on improving the accuracy of detection and analysis of the occurrence of false positives in the network with real traffic.

7 ACKNOWLEDGMENT

The research was supported by the Technology Agency of the Czech Republic under grant no. FW01010474.

REFERENCES

[1] Cambiaso, E., Papaleo, G., Chiola, G., Aiello, M.: Slow DoS attacks: definition and categorisation, International Journal of Trust Management in Computing and Communications, 2013, DOI: 10.1504/IJTMCC.2013.056440

[2] Cambiaso, E., Papaleo, G., Aiello, M.: Slowcomm: Design, development and performance evaluation of a new slow DoS attack, Journal of Information Security and Applications, 2017, DOI: 10.1016/j.jisa.2017.05.005

[3] Cambiaso, E., Papaleo, G., Chiola, G., Aiello, M.: Designing and Modeling the Slow Next DoS Attack, International Joint Conference. Cham: Springer International Publishing, 2015, pages 249–259, Advances in Intelligent Systems and Computing, DOI: 10.1007/978-3-319-19713-5_22

[4] Aiello, M., Cambiaso, E., Scaglione, S., Papaleo, G.,: A similarity based approach for application DoS attacks detection, 2013 IEEE Symposium on Computers and Communications (ISCC), Split, Croatia, 2013, DOI: 10.1109/ISCC.2013.6754984

[5] Mahjabin, T., Xiao, Y., Sun, G., Jiang, W.: A survey of distributed denial-of-service attack, prevention, and mitigation techniques, International Journal of Distributed Sensor Networks, 2017, DOI: 10.1177/1550147717741463

[6] Tanwar, S.: Hands-On Parallel Programming with C# 8 and .NET Core 3 [online], Birmingham: Packt Publishing, 2019, p. 251, [cited 2020-12-09], ISBN 978-1-78913-241-0, URL: <https://bit.ly/3duOZkj>

[7] Garrett, O.: Inside NGINX: How We Designed for Performance & Scale, NGINX [online], Seattle: F5 Networks, 2015 [cited 2020-12-09], URL: <https://www.nginx.com/blog/inside-nginx-how-we-designed-for-performance-scale/>