Performance Influence of Security Function Chain Ordering

Lukas Iffländer
University of Würzburg
Germany
lukas.ifflaender@uni-wuerzburg.de

Nicolas Fella
University of Würzburg
Germany
nicolas.fella@stud-mail.uni-wuerzburg.de

ABSTRACT

In modern days security systems often reach their performance peak and limit the protected application. Utilizing the available resources for security more efficiently is becoming more critical. In this paper, we introduce the claim, that no static security function chain is optimal in every situation. First experiments prove our claim.

CCS CONCEPTS

• **Networks** → **Network control algorithms**; **Network performance analysis**; **Network security**; *Middle boxes / network appliances*; • **Security and privacy** → *Intrusion detection systems*; *Virtualization and security*;

KEYWORDS

intrusion detection, DDoS defense, firewall, network function virtualization, adaptive networking

ACM Reference Format:
Lukas Iffländer and Nicolas Fella. 2019. Performance Influence of Security Function Chain Ordering. In *Tenth ACM/SPEC International Conference on Performance Engineering Companion (ICPE '19 Companion), April 7–11, 2019, Mumbai, India.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3302541.3311965

1 INTRODUCTION

With the end of Moore's Law [4] security systems face new challenges. While bot networks expand and attacks become more aggressive from day to day, the systems can not keep up. The processing power of a single CPU or server no longer scales as fast as the potential threat.

Networks are vulnerable to attacks in various ways. Common types are "HTTP Flood" or "SYN Flood." Dedicated Security Appliances (SAs) exist to defend against every type of network attack. An Intrusion Detection System (IDS) protects against attacks targeted at known vulnerabilities in deployed software. At the same time, a firewall can protect against HTTP flood attacks on blocked ports.

In our previous work [3] we introduced the idea of self-aware security function chain reordering. The basic concept is to change the order of a Security Service Function Chain (SSFC) depending on the incoming attacks. Since SAs drop packets deemed malicious thereby lowering the load on subsequent SAs.

ICPE '19 Companion, April 7–11, 2019, Mumbai, India

ACM ISBN 978-1-4503-6286-3/19/04.
https://doi.org/10.1145/3302541.3311965

When we presented this idea a frequent inquiry was whether this approach is needed at all or if there might be a static solution that works in any case. Thus, we decided to perform a performance evaluation of the performance influence of the SSFC order. In the following, we present some first results, that confirm the possibilities from the dynamic adaptation of the order.

2 MEASUREMENT SETUP

2.1 Testbed Architecture

For the evaluation, we use a testbed with multiple servers and SDN switches. Traffic generator, receiver, DDoS Mitigation System, Intrusion Detection System, Firewall and SDN Controller each run on one server. Each server connects to the management network. Furthermore, each of these servers has two Ten Gigabit Ethernet ports (Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection (rev 01)) to connect the servers. Four HPE 5130 24G 4SFP+ EI SDN-switches connect the servers. Each server is running Ubuntu 18.04 Bionic Beaver.

Traffic Generator: The traffic generator is making use of both 10 Gb interfaces. HTTP-Load-Generator [5] generates benign packets. IDS floods are generated using Cisco's Trex[1]. Trex uses Intel DPDK to create high-volume loads. DPDK binds the whole interface to the program. Therefore, the HTTP traffic needs to be sent on a different interface. Both interfaces connect to the same switch, so from there on the packets will be treated equally by the network. BoNeSi2[2] generates HTTP floods. BoNeSi can create high-volume HTTP floods by emulating spoofed IP addresses.

Intrusion Detection System: The IDS host is running Snort[3] in version 2.9.7. Snort is a common, open-source IDS developed by Cisco. It is the base of Cisco's commercial IDS solutions. One 10 Gb processes incoming and the other outgoing traffic. For the measurements, we used the standard Snort Community signatures extended by several rules.

Firewall: Like the IDS the Firewall used one interface for incoming and one for outgoing traffic. A Linux bridge connects them. The packet filtering is accomplished using netfilter/iptables rules.

Target: The target server is running a default Apache 2 installation on port 80.

SDN Controller: Ryu[4] is used as a SDN controller.

[1] https://trex-tgn.cisco.com/
[2] https://github.com/markus-go/bonesi
[3] https://www.snort.org
[4] https://osrg.github.io/ryu/

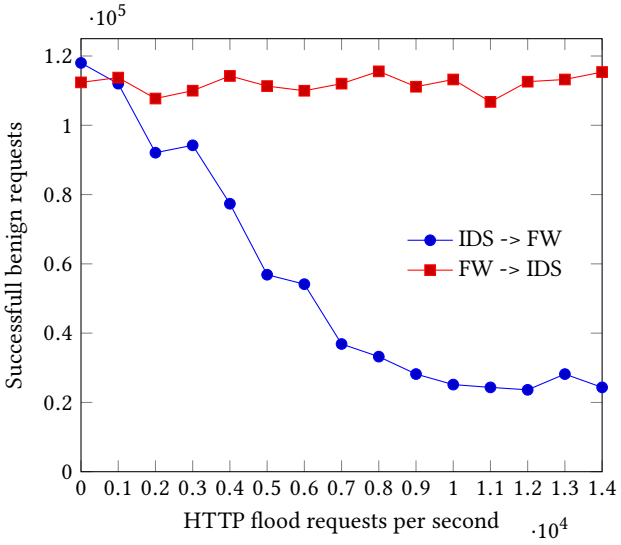

Figure 1: Performance during HTTP-Flood

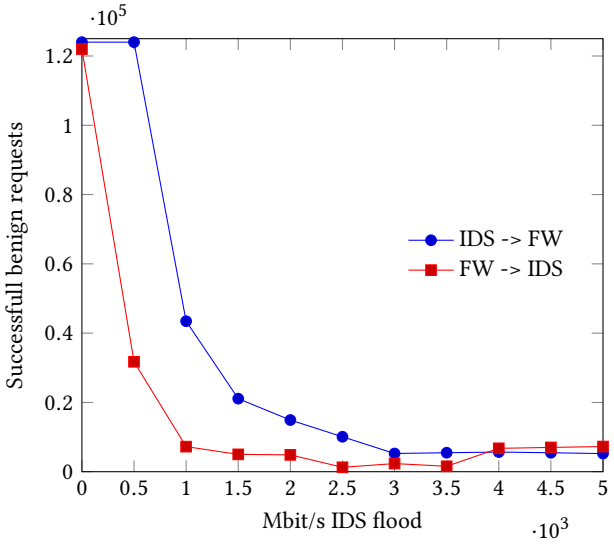

Figure 2: Performance during IDS-Flood

2.2 Measurement Methodology

HTTP-Load-Generator returns the number of successful requests. Thus, we use this number to assert the performance of the security appliance. The more requests the security system can handle successfully, the more efficient it is.

We put an SSFC consisting of two devices under stress. The first component is the firewall as mentioned earlier and the other the IDS. After evaluating the performance without an attack, we perform an HTTP flood and an IDS flood. For both, we evaluate the performance with the IDS placed at the first position and in a second experiment with the reversed order. We perform 2 000 benign requests per second over 60 seconds.

3 EVALUATION

3.1 Baseline

At first, we test the capabilities of the target, the firewall, and the IDS under benign load. The target shows to be able to handle up to 16 000 requests per second. The addition of the firewall does not affect the performance. Adding the IDS lowers the performance to around 3 000 requests served successfully.

3.2 HTTP Flood

Figure 1 shows that for the HTTP flood the firewall at first position has a significant advantage. While at no or low attack load the performance is similar for both orders, the performance for the IDS at first position drops significantly with increased load.

3.3 IDS Flood

Here both systems lose performance, once an attack starts. Nevertheless, the IDS at first position reduces the loss of performance. For example, at 1 500 Mbit/s flood throughput, the IDS at first position still has four times as many requests handled than the firewall at first position.

3.4 Summary

The results show that depending on the current attack different orders are optimal. The order that is optimal for the HTTP-Flood lacks performance when facing an IDS flood. Thus, there is no optimal static order.

4 CONCLUSION AND OUTLOOK

These first results show that our claim for the need of adaptive SSFC reordering is well-founded. In future work, we will analyze further combinations and the effect of the order on CPU and RAM load. Also, we will incorporate our SDN based DDoS-Defense [1] and IDS-Optimization [2].

5 ACKNOWLEDGMENTS

This work was funded by the German Research Foundation (DFG) under grant No. (KO 3445/16-1).

REFERENCES

[1] Lukas Iffländer, Stefan Geißler, Jürgen Walter, Lukas Beierlieb, and Samuel Kounev. 2018. Addressing Shortcomings of Existing DDoS Protection Software Using Software-Defined Networking. In *Proceedings of the 9th Symposium on Software Performance 2018 (SSP'18)*.

[2] Lukas Iffländer, Jonathan Stoll, Nishant Rawtani, Veronika Lesch, Klaus-Dieter Lange, and Samuel Kounev. 2019. Performance Oriented Dynamic By-passing for Intrusion Detection Systems. In *Proceedings of the 2019 ACM/SPEC International Conference on Performance Engineering (ICPE '19)*. ACM, New York, NY, USA, 8.

[3] Lukas Iffländer, Jürgen Walter, Simon Eismann, and Samuel Kounev. 2018. The Vision of Self-aware Reordering of Security Network Function Chains. In *Companion of the 2018 ACM/SPEC International Conference on Performance Engineering - ICPE '18 (ICPE '18)*. ACM Press, New York, NY, USA, 1–4. https://doi.org/10.1145/3185768.3186309

[4] Thomas N. Theis and H.-S. Philip Wong. 2017. The End of Moore's Law: A New Beginning for Information Technology. *Computing in Science & Engineering* 19, 2 (mar 2017), 41–50. https://doi.org/10.1109/mcse.2017.29

[5] Jóakim von Kistowski, Maximilian Deffner, and Samuel Kounev. 2018. Run-Time Prediction of Power Consumption for Component Deployments. In *Proceedings of the 15th IEEE International Conference on Autonomic Computing (ICAC 2018)*. IEEE. https://doi.org/10.1109/icac.2018.00025