Simulative Analysis of Adaptive Network Monitoring Methodologies for Attack Detection

Falko Dressler, *Member, IEEE*, Isabel Dietrich

***Abstract* — Due to the rapid growth of network traffic and available bandwidth, scalable network monitoring has become a major issue in the last years. Additionally, the importance of correct and complete monitoring data increased, e.g. for attack detection or accounting issues. Based on bio-inspired methodologies, we propose an adaptive mechanism for preventing single systems of being overloaded while providing as much monitoring information as possible for post-processing. Using a simulation model, we verified the applicability of our approach.**

***Keywords* — Modeling and Simulation, Performance Analysis, Scalable Network Monitoring, Feedback Loops, Bio-inspired Networking**

I. Introduction

MONITORING of high-speed communication networks has become a major research area during the last years. Driven by the needs of network security installations but also by the demands of accounting and charging systems, network monitoring methods and techniques have been standardized by several organizations, first of all by the IETF (Internet Engineering Task Force). In order to cope with the steadily increasing amount of data and the very high bandwidths in nowadays backbone networks, the reduction of monitor data is a key issue for successful monitoring solutions.

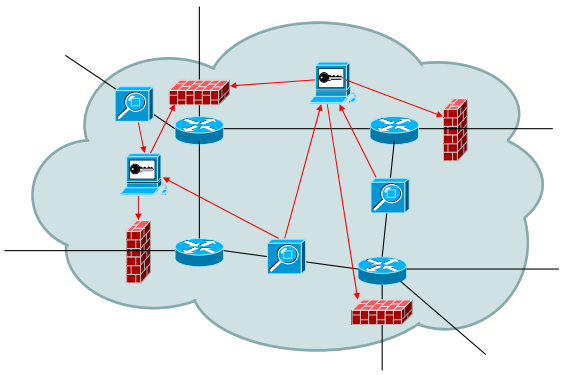

Fig 1. Network security scenario using monitoring probes, attack detection and firewall systems

Part of this research was supported by the European Commission funded project DIADEM Firewall.

F. Dressler and I. Dietrich are with the Department of Computer Sciences, University of Erlangen, Germany (phone: +49 9131 85-27914; fax: +49 9131 85-27409; e-mail: dressler@informatik.uni-erlangen.de).

In Fig 1, a simple scenario is depicted. Multiple monitoring probes are employed to obtain information about the ongoing network traffic. This information is forwarded to adequate intrusion detection systems, which in turn analyze the data and close the loop by configuring firewall systems to counteract the identified attacks. Similar scenarios can be thought of for accounting or traffic engineering purposes.

In this paper, we analyze the scalability of such monitoring approaches using appropriate simulation models. Additionally, we propose an adaptive mechanism for reducing the amount of monitoring data, or, to be more precise, to adjust the load in the entire monitoring system. The primary goal is to monitor as much as possible in order to achieve more accurate results. A threshold is given by the processing capacity of all involved systems, therefore, an upper bound is defined. Additionally, this threshold depends on the kind of the data to be processed.

We address this issue by using two separate feedback loops as inspired by similar solutions found in nature. The research area of bio-inspired networking has already shown application scenarios for new and unconventional solutions [8]. Most biological methods for self-organization depend on two mechanisms: a positive feedback loop for amplification and a negative feedback loop for regulation [4, 5]. Using similar mechanisms, we built an adaptive feedback mechanism for a self-organized network monitoring scenario, or, in other words, an immune system for the Internet.

In section II, the state-of-the-art of network monitoring is summarized followed by a description of our proposed adaptive control mechanism in section III. The simulation model and the accomplished simulations are discussed in section IV. Some conclusions summarize the paper.

II. Network Monitoring

Because available bandwidths grow much faster than the processing speed of the monitoring probes and subsequent analyzers, solutions have been developed that allow reducing the processing requirements at the analysis. The primary idea behind all these concepts is to split the monitoring and the subsequent analysis into two independent tasks. This became possible because not all packet information is required for network analysis (whether for accounting or security reasons).

The first concept developed in this context is the netflow accounting. The key idea is to store information

about netflows and the corresponding statistics instead of individual packets. Thereby, a netflow is defined as a unidirectional connection between two end systems as defined by the IP 5-tupel (protocol, source IP address, destination IP address, source port, destination port). Doing this, a single measurement data set contains information of one up to several thousands of individual packets. For the transmission of the monitoring data to an analyzing system, a special protocol was developed called netflow.vX, where X stands for the version number. Netflow.v9 is the latest version and standardized by the IETF [1]. Its successor is IPFIX (IP flow information export). It was developed by the corresponding IPFIX working group at the IETF and provides sufficient information for a distributed deployment [2, 10]. First implementations are available that support netflow.v9 as well as IPFIX, e.g. nProbe [3] and Vermont [6].

Even if this methodology works well under normal conditions (usual connections consist of about 7.7 packets per flow [9]), there is a major problem during DDoS attacks. Usual attacks are using forged IP addresses, different ones for each attack packet, which results in the creation of individual flows per packet. Thus, in such an attack situation, netflow accounting or IPFIX do not scale well, i.e. they overflow the connection between the monitoring probe and the intrusion detection system (regardless of the computational expense at the analysis). To cope with this problem, recently an aggregation mechanism was introduced [7] that allows to aggregate individual flows into so called meta-flows. This aggregation mechanism allows a free scaling of the amount of monitoring data and provides the basic functionality to build adaptive self-optimizing netflow accounting solutions. Nevertheless, even more functionality is required to cope with the growing amount of network traffic. In the next section, we discuss an approach for adaptively configuring all involved components in the network security scenario.

III. Adaptive Re-Configuration

In the context of this section, we discuss the possibility of an adaptive re-configuration of the monitoring environment dependent on the current situation as discovered by monitoring the network. The final goal is to show the practicability of a self-organizing monitoring architecture that still fulfills its role to collect as much as possible packets concentrating on the really necessary data sets. First, the problem is described including the primary objectives. Secondly, a basic model is provided which fulfills the requirements and builds a basis for the complete simulation model.

A. Problem Description and Objectives

The most important issue is to prevent the attack detection system from being overloaded by the monitoring probes in order to prevent the detection system from becoming a target itself and to increase the availability of the overall system. Even though each subsystem can perform attack detection autonomously, an overloaded single system might miss important packets that build a primary attack accompanied by a large amount of meaningless packets. To achieve this goal, an autonomous behavior of each sub-system is considered. In this paper we concentrate on adaptive re-configuration. Self-optimization is an important requirement for autonomous systems in general. In this context we understand self-optimization as the ability to optimize the detection quality. This can be achieved by exchanging information about already identified attacks or suspicious network connections and also by statistically forwarding parts of collected data packets and network statistics to neighboring probes. Additionally, the autonomously working entities must be capable to adapt to a changing environment. This adaptation, typically realized by re-configuration of runtime parameters, comprises of changes in the resource management and in the configuration of tasks and processes. Re-configuration is also required in the case of resource shortages. The attack detection must be modified by selecting algorithms and parameters which require less memory, while typically resulting in a lower detection rate.

B. Model and Solution

In Fig 2, a model is shown that represents the considered architecture including all necessary components. While focusing on the monitoring part, we want to reduce, or more precisely, to adapt the rate of packets sent to the attack detection system. This can be done in at least three ways as shown in the following.

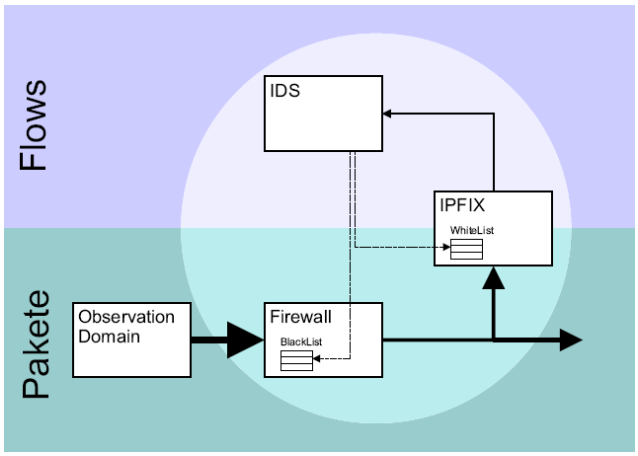

Fig 2. Basic model for adaptive re-configuration including the main components and the control and data flows

Compression/encoding – Monitored packet data can be encoded in a way reducing the number of transmitted bytes to the absolute minimum. Netflow accounting / IPFIX can help at this place, especially if aggregation mechanisms are employed. Nevertheless, during a DDoS attack, nearly every packet represents a separate flow and, therefore, the speedup is questionable in this case. Therefore, methodology based approaches are required.

Blacklists/whitelists – Usually, blacklists represent hosts involved in an attack and whitelists represent

legitimate traffic. Blacklists in packet filtering systems, i.e. firewalls, represent a functionality having two advantages. First, the packets are prevented from reaching the systems under attack and, secondly, these packets no longer reach the monitoring system. Therefore, the data rate being sent the monitoring systems to the attack detection systems is reduced. Additionally, whitelists can be used at the monitoring probes to reduce the amount of data transmitted to the analyzing systems.

Feedback – Finally, a methodological approach can be developed that is based on the parameterization being adapted to the current situation in the network. The attack detection system can communicate its current load to the monitoring probes. These can adapt a number of parameters, usually timeouts, based on the number of packets received from the network, the number of packets reported to the detection systems, and the current load of the analyzers.

C. Methodology

As mentioned in section I, bio-inspired methodologies are used to create appropriate feedback loops for adapting the parameters in the monitoring environment depending on the current load. Usually, two kinds of feedback loops are used in combination: positive feedback for short-term amplification and negative feedback for long-term regulation. Both loops are depicted in Fig 2. The intrusion detection reports detected attacks to the firewall that in turn is blocking this traffic and reduces the number of packets to be monitored. Additionally, the IDS reports legitimate traffic to the monitor. This monitor stops reporting the packets belonging to these flows and, therefore, reduces the number of packets to be analyzed. Obviously, both configurations cannot be permanent. Sources sending legitimate traffic might begin to send attack packets at any time. Also, attackers might be "corrected" and should not be starved by our firewalls.

The adaptation is done using the following formulas for calculating appropriate timeouts. TO_{black} corresponds to the firewall system and TO_{white} to the monitoring probe.

$$TO_{black} = C_1 \underbrace{\frac{\lambda_{bi}}{\lambda}}_{t_1} + C_2 \left(\underbrace{\frac{\lambda_b}{\lambda}}_{t_2} + \underbrace{\frac{\lambda}{\lambda_w}}_{t_3} + \underbrace{\frac{\lambda}{\lambda_{IDS}}}_{t_4} \right) \quad (1)$$

$$TO_{white} = C_3 \left(\underbrace{\frac{\lambda}{\lambda_{IDS}}}_{t_5} + \underbrace{\frac{\lambda}{\lambda_b}}_{t_6} \right) \quad (2)$$

The input parameters are summarized in table 1.

TABLE 1: INPUT PARAMETERS.

λ	Arrival rate
λ_b	Arrival rate of "black" packets
λ_{bi}	Arrival rate of "black" packets belonging to the i-th flow
λ_{bw}	Arrival rate of "white" packets
λ_{IDS}	Maximum capacity of the IDS

The single terms are discussed in the following. In principle, all terms belonging to one timeout are summed up and scaled by a constant. In our experiments we tried to find appropriate values for these constants. In a next step, the constants themselves can be adapted to the current scenario.

t_1: ratio of the i-th attack flow to the overall attack rate. Used for penalizing previously discovered attack flows. This term must be scaled separately using C_1 because it is usually very small.

t_2: Similar to t_1 but defined the ratio of arriving attack traffic to the overall throughout. The larger it is, the more aggressive the attack.

t_3: This term describes the safety of the arriving traffic: the larger the amount of "white" packets, the smaller the requirement for large timeouts at the firewall.

t_4: This term is a measure for the overload of the attack detection system.

t_5: The same as t_4 but used at the monitor.

t_6: similar to t_3 but defining the risk of arriving packets.

IV. MODELING AND SIMULATIONS

In order to show the potentials of this feedback, we executed a set of simulations showing the reduction of packet data that is to be received and processed by the attack detection systems. For a reasonable simulation, we decided to use "trace-based" input modeling. We accumulated several traces, e.g. in front of out workgroup server or at the Internet gateway our university. The data presented in this paper rely to the utilization seen at the university border gateway. The simulation model is depicted in Fig 3.

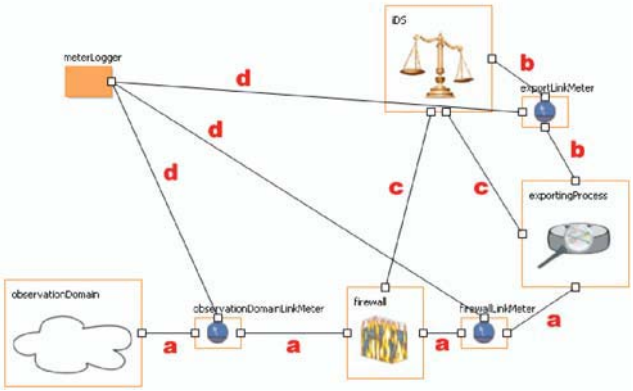

Fig 3. Simulation model including message types on the communication channels: a) IP packets, b) IPFIX, c) feedback, d) measurement information

This model was implemented using AnyLogic, a well-established discrete-event simulator. The primary goal was to find adequate values for the scaling factors C_1, C_2, and C_3. In the following, some of our simulation results are presented. Additionally, we compared the (over-)load of the attack detection system using the proposed adaptation to an unmodified system. Due to lack of space in this paper, we can present only some selected results for a simulation using the adaptation. The parameters for this

simulation were $C_1=9*10^8$, $C_2=236$, $C_3=120$.

As already mentioned, we monitored the traffic at the border gateway of our university. The utilization is quite constant (58.6kpps ± 1.8kpps) as shown in Fig 4 left. Due to few attacks, the firewall removes only few flows (around 0.5%). Therefore, the packet rate arriving at the monitor is quite the same but shaped (58.2kpps ± 1.8kpps, Fig 4 right).

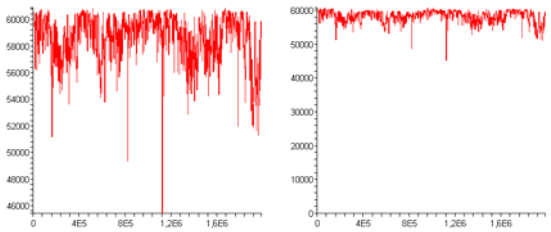

Fig 4. Input ratio (left) and input ratio at the monitor (right)

Even though the number of discarded packets at the firewall is very small, this is primarily a result of the small timeout of each entry. The IDS is not overloading, therefore, it may analyze detected attacks over and over again. The behavior of the firewall is depicted in Fig 5. In steady-state, around 6000 blacklist entries exist with an average timeout of 260s.

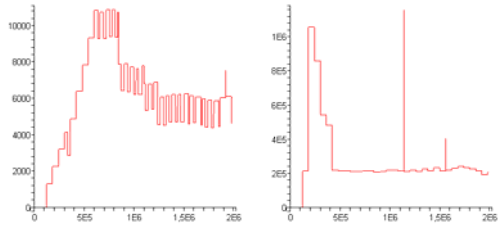

Fig 5. Number of blacklist entries (left) and timeout TO_{black} (right)

Due to the key ideas of netflow accounting, the summarization into flows, the number of exported data sets is quite low (Fig 6). Nevertheless, there is an enormous advantage compared to mechanism without adaptation. In this case, the timeouts have to be predefined and can only correspond to an assumed behavior of the network.

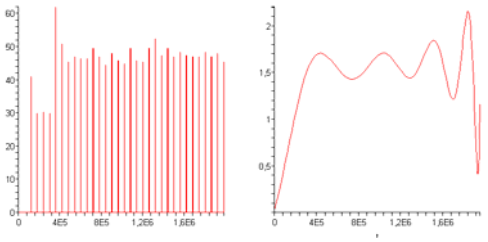

Fig 6. Number of exported data sets (left) and averaged (right)

Finally, the numbers of whitelist entries and the corresponding timeouts have to be examined. As shown in Fig 7, in steady-state, the number of whitelist entries is around 60000 and the average timeout is 320s. The number corresponds to the detection quality of legitimate traffic which is about ten times higher then for attacks. The timeout oscillates around the requested value.

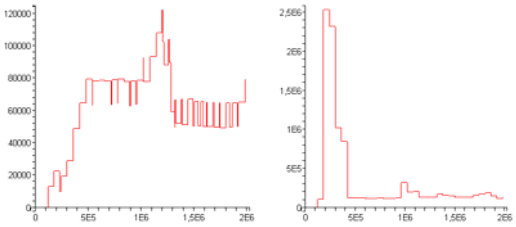

Fig 7. Number of whitelist entries (left) and timeout TO_{white} (right)

V. Conclusion

In conclusion it can be said that we were able to demonstrate the applicability of the proposed adaptive mechanism for self-organization and adaptation in network monitoring environments. Especially the capabilities of the adaptation to reflect the local needs of an analyzing system in combination to the observation of the environment, i.e. the arrival rates at each sub-system make this approach useful for most monitoring scenarios. In a next step, we will evaluate the algorithm in an experimental setup to compare this evaluation to the simulation results.

Acknowledgment

This work is part of a collaborative research project conducted together with Prof. Georg Carle and Gerhard Münz from University of Tübingen, Germany. We also thank our students, especially Alexander Lochschmied, for contributing to this work.

References

[1] B. Claise, "Cisco Systems NetFlow Services Export Version 9," RFC 3954, October 2004.

[2] B. Claise, "IPFIX Protocol Specification," Internet-Draft, draft-ietf-ipfix-protocol-08.txt, February 2005.

[3] L. Deri, "nProbe: an Open Source NetFlow Probe for Gigabit Networks," Proceedings of TERENA Networking Conference (TNC 2003), Zagreb, Croatia, May 2003.

[4] F. Dressler, "Bio-inspired mechanisms for efficient and adaptive network security mechanisms," Proceedings of Dagstuhl Seminar 04411 on Service Management and Self-Organization in IP-based Networks, Schloss Dagstuhl, Wadern, Germany, October 2004.

[5] F. Dressler, "Efficient and Scalable Communication in Autonomous Networking using Bio-inspired Mechanisms - An Overview," *Informatica - An International Journal of Computing and Informatics*, vol. 29 (2), July 2005.

[6] F. Dressler and G. Carle, "HISTORY - High Speed Network Monitoring and Analysis," Proceedings of 24th IEEE Conference on Computer Communications (IEEE INFOCOM 2005), Miami, FL, USA, March 2005.

[7] F. Dressler, C. Sommer, and G. Münz, "IPFIX Aggregation," Internet-Draft, draft-dressler-ipfix-aggregation-01.txt, July 2005.

[8] B. Krüger and F. Dressler, "Molecular Processes as a Basis for Autonomous Networking," *IPSI Transactions on Advances Research: Issues in Computer Science and Engineering*, vol. 1 (1), pp. 43-50, January 2005.

[9] T.-H. Lee, W.-K. Wu, and T.-Y. W. Huang, "Scalable Packet Digesting Schemes for IP Traceback," Proceedings of IEEE International Conference on Communications, Paris, France, June 2004.

[10] J. Quittek, S. Bryant, and J. Meyer, "Information Model for IP Flow Information Export," Internet-Draft, draft-ietf-ipfix-info-06.txt, February 2005.