A System for in-Network Anomaly Detection

Thomas Gamer

Institut für Telematik, Universität Karlsruhe (TH), Germany

Abstract. Today, the Internet is used by companies frequently since it simplifies daily work, speeds up communication, and saves money. But the more popular the Internet gets the more it suffers from various challenges like DDoS attacks. This work, therefore, proposes an anomaly-based system that is able to detect adverse events caused by such challenges. The detection of network anomalies, in contrary to signature-based systems, ensures that previously unknown adverse events can be detected, too. Furthermore, the proposed system is designed for deployment within the network to allow a detection of adverse events as fast as possible, i.e., not only at the victim's edge of the network. To achieve such an in-network anomaly detection the system is designed hierarchically and applies refinement of detection granularity.

1 Introduction

Today's networks are threatened by challenges that appear with increasing frequency and comprise various kinds of attacks as well as unintended network problems. Challenges currently threatening networks include attacks like denial-of-service (DDoS) attacks [1] and worm propagations [2]. Furthermore, unintended network problems due to misconfigured nodes or flash-crowd events [3] pose a threat to today's networks, too. An automatic detection of adverse events caused by such challenges is still a problem for network operators. Additionally, autonomic networking will be a topic in the near future. Such networks need a mechanism to detect adverse events and apply suitable countermeasures autonomously.

With DDoS attacks an attacker does not exploit a weakness of the victim's operating system or application but aims to overload resources like link capacity or memory by flooding the system with more traffic than it can process. The attack traffic is generated by many slave systems which the attacker has compromised before. The attacker only has to coordinate all these slave systems to start the attack nearly at the same time against a single victim. Internet worms on the other hand exploit security holes in operating systems or applications to infiltrate a system. Afterwards, they start to propagate themselves to as many other systems as possible. One side effect of this propagation is the increasing bandwidth consumption since more and more worm instances try to propagate themselves to other systems. Today's countermeasures to worms are signature-based detection systems scanning for well-known worms.

An early detection of adverse events caused by such challenges allows a fast reaction and, thus, ensures a suitable protection of the network, the victims, and the network's resources. This requires a detection system within the network. Programmable networks enable a router to flexibly set up new services on that router, i.e., within the network. Therefore, programmable networks are suitable to achieve an in-network deployment of a detection system for adverse events. Such a detection system, however, has to face some difficulties, too. One of those is the fact that – in the worst case – the detection takes place within high-speed networks. This means that, though an on-line analysis is performed by the detection system, a negative impact on that router's forwarding performance must be avoided. Therefore, we propose the usage of a hierarchical detection system that applies refinement, i.e., detection granularity and analysis effort are adapted to the current stage of the detection. Such a system, therefore, works resource-saving and ensures that – even if it is built completely in software – there is no affection of a router's forwarding performance.

Furthermore, we propose to use an anomaly-based detection since various challenges, e.g. DDoS attacks, cannot be detected by a signature-based system due to their usage of protocol-conform packets. An anomaly-based detection system, however, analyzes traffic behavior and, therefore, can detect such challenges as well as previously unknown adverse events. Lastly, a signature-based system is only applicable in high-speed environments if it uses special-purpose hardware since it has to inspect each packet deeply.

If a DDoS attack, for example, is running error messages are generated by routers close to the victim as soon as the victim is not reachable anymore. Such changes of the traffic can be detected by combining various anomalies. In case of worm propagations an anomaly-based detection system can collect hints on such an adverse event e.g. by analyzing the ratio of error messages due to closed ports to the total number of connection requests. Such error messages are generated by scanned systems that are not vulnerable to this specific worm.

This paper details on a system for in-network anomaly detection. It is organized as follows: section 2 presents a short introduction to packet selection mechanisms. Section 3 details on the main characteristics of the detection system – a hierarchical architecture and refinement. Furthermore, architecture details are given for an example scenario. An evaluation of this example scenario then is described in section 4 and finally, section 5 gives a short summary.

1.1 Related Work

There are some existing approaches for DDoS attack detection that use special-purpose hardware: [4] uses network processors to perform a deep packet inspection of all observed packets in a backbone network. [5] uses special-purpose hardware to add a timestamp to each packet and then, does the analysis off-line.

Other anomaly-based approaches either cannot be applied in high-speed networks due to their high resource consumption or perform only a very coarse-grained detection without further refinement. The pushback mechanism [6], for example, is activated as soon as congestion occurs on a router. In this case a

flooding attack is assumed and a rate limiter is installed for the highest bandwidth aggregate of dropped packets. This approach has several disadvantages: an attack can be detected not until congestion occurs on a router and hence a detection is only possible at the edge of the network. Furthermore, no further verification is done if the rate limited aggregates really belong to an attack. Sterne et al. [7] detects stochastic anomalies by using a threshold-based DDoS detection mechanism on active networking nodes but no further refinement is done if an attack has been detected. Bro [8] is an open source network intrusion detection system that applies refinement. But – unlike our approach – the refinement has a different scope. Bro is an event-driven system and consists of three parts: the packet capture, the policy-neutral event engine, and the policy layer. A problem of this approach is that Bro creates lots of state by deep packet inspection and semantic analysis. Finally, the MVP architecture of Cisco Systems [9] uses refinement for detection of DDoS attacks, too, but this refinement is not very flexible and is only done in two steps, i.e., multiple stages are not possible for refinement.

2 Packet Selection

A packet selection mechanism is used especially in high-speed environments to reduce the number of packets that have to be inspected by a specific application, e.g. measurement or intrusion detection. The IETF working group PSAMP [10] proposed two types of packet selectors: filtering and sampling. *Filtering* is used if only a particular subset of packets is of interest. Filtering schemes are always deterministic and are based on packet content or router state. Therefore, filtering schemes are not suitable for a detection of adverse events. Any attacker who knows the filtering rules can adapt his challenge in a way that his packets are not selected by the detection system. This makes bypassing of the detection system easy. In contrast to filtering, *sampling* is used to infer knowledge about an observed packet stream without inspecting all packets. Therefore, only a representative subset of packets is selected which enables an estimation of properties of the total traffic. Sampling methods are either nondeterministic or do not depend on packet content or router state. The sampling methods are further grouped into two categories: random sampling and systematic sampling.

Systematic count based sampling is an example for a systematic sampling method. This sampling method is deterministic but independent of packet content and router state. For this method a *sampling interval* is defined consisting of a *selection interval* and a *non-selection interval*. A periodic trigger defines the beginning of a sampling interval. The unit of the intervals is count based. An example of this sampling method with a sampling interval of 5 packets, a selection interval of 2 packets, and a non-selection interval of 3 packets is shown in figure 1.

A sampling mechanism effectively reduces the number of packets that are inspected but it also introduces estimation errors. Thus, the parameters of the

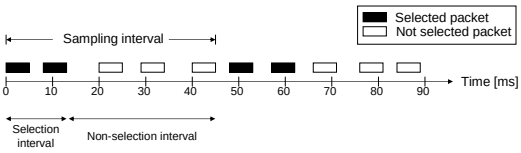

Fig. 1. Example of packet selection with systematic count based sampling

applied sampling mechanism have to be chosen in such a way that the error caused by packet selection is restricted to a predefined tolerance level.

3 Architecture

The detection system is designed hierarchical, anomaly-based, and flexible. Due to the fact that an anomaly-based detection is performed – i.e. only traffic behavior is analyzed – a packet selection mechanism as described in section 2 can additionally be applied.

The hierarchical characteristic of the system allows to split the detection of adverse events into different stages: A basic stage that scans for *stochastic anomalies* is running all the time. Specialized stages are loaded on demand for a more detailed detection of adverse events. Thus, refinement of detection granularity is applied by the detection system, i.e., detection granularity is increased with each subsequently loaded detection stage (see fig. 2). The basic stage of the hierarchical detection system dedicates only low analysis effort – this stage does only a simple packet classification – in order to perform a coarse grained detection that scans for indications of an adverse event. Further stages then are loaded whenever an adverse event is assumed in the basic stage. These further stages analyze only a part of the whole packet stream due to the information about the assumed adverse event gathered by the basic stage. Therefore, the further stages are able to do a more fine grained detection by applying deeper packet inspection on the reduced packet stream. Thus, the detection system gathers more detailed information about the adverse event in each of the further stages by using a higher analysis effort. In this paper, the notion *packet stream* designates a link's total aggregated traffic whereas a set of packets with same characteristics, e.g., all TCP packets, is referred to as an *aggregate*.

In summary, the hierarchical architecture of the detection system and the application of refinement save resources by running a basic stage with low resource consumption all the time and by loading further stages not until a stochastic anomaly is detected in the basic stage.

In order to detect stochastic anomalies the basic stage divides the packet stream on the fly into intervals with a fixed length. Furthermore, aggregates of interest are defined for observation, for example all TCP or all UDP packets. Then, for each predefined aggregate the number of packets that belong to this aggregate is counted in every interval. To make the system self-adaptable to net-

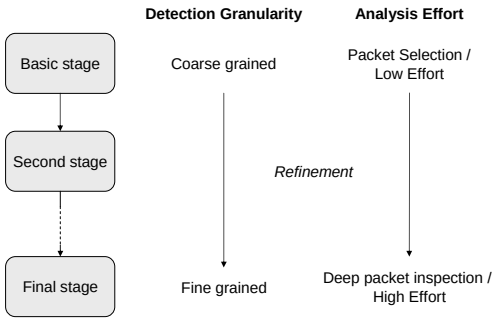

Fig. 2. Architecture of a hierarchical detection system using refinement

work load changes a dynamic *packet threshold* representing the average packet count in this aggregate for the last couple of intervals is calculated. At the end of each interval, for any aggregate a check is performed if the observed number of packets exceeds the packet threshold. To prevent the system from generating false positive indications and starting further stages for deeper inspections unnecessarily an *interval threshold* is defined. This interval threshold is necessary due to the self-similarity of internet traffic [11] which can cause normal traffic to exceed the packet threshold even though no adverse event is currently going on. Therefore, an indication only is generated if the packet threshold is exceeded in more consecutive intervals than the interval threshold. In addition to the detection of stochastic anomalies in the basic stage, the suspicious packet stream is scanned for further anomalies in specialized stages.

Flexibility of the detection system is ensured by usage of programmable networks. Since service modules are not tightly coupled to the packet forwarding but are loaded on demand, it is easy to update such modules or to add new service modules without a change to the rest of the system. Furthermore, the hierarchical architecture of the system allows the addition of new specialized stages. All characteristics described so far provide a flexible system for in-network anomaly detection that can be deployed in different environments like high-speed networks or small provider networks.

3.1 Small provider network

This section illustrates an exemplary architecture of the system for anomaly detection in case of a small provider network. In such a network detection of DDoS attacks and worm propagations is focused since these attacks are the most prevalent challenges. Therefore, the system scans for stochastic anomalies in the basic stage as described above. After detecting such a stochastic anomaly refinement is applied by loading two specialized consecutive stages (see fig. 3). The second stage uses a *distribution anomaly* to make a differentiation between DDoS attacks and worm propagations. This can be achieved by analyzing the distribution of packets into subnet prefixes based on destination addresses. Therefore,

the whole address space is divided into subnet prefixes based on the routing table of the node deploying the detection system. If large parts of the suspicious traffic – the number of packets by which the packet threshold was exceeded in the basic stage – are sent into exactly one subnet a DDoS attack is indicated since only one victim is currently attacked. If the suspicious traffic is equally distributed to all existing subnet prefixes a worm propagation is assumed since worms spread all over the internet. Based on the result of the second stage attack type specific protocol anomalies are scanned for in the third stage to identify either DDoS attacks or worm propagations in more detail. Currently, the anomalies used in our system for network anomaly detection offer no possibility to differentiate between DDoS attacks and legitimate traffic with the same characteristics, e.g. flash-crowd events [3].

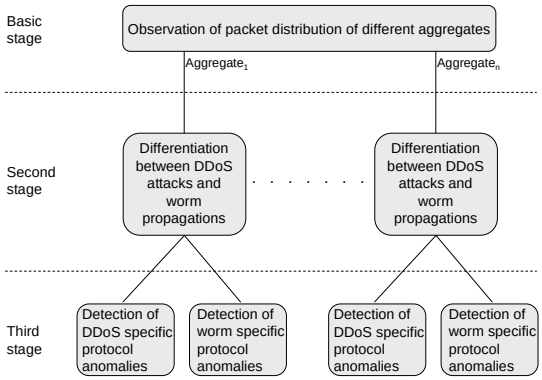

Fig. 3. Architecture of the detection system in a small provider network

In case of DDoS attacks the third stage of the detection system is mainly based on the fact that most of the existing DDoS attacks lead to a breach of symmetry between incoming and outgoing sub-aggregates which belong together by protocol definition. A TCP SYN flooding attack, for example, tries to exhaust a victim's open connection storage space by flooding the victim with TCP packets with SYN flag set. Due to the mass of connection requests the victim can only respond to a part of all requests by sending TCP packets with SYN and ACK flag set. All remaining requests are dropped and the victim sends no response at all if storage space is already exhausted. This leads to an asymmetry between incoming TCP packets with SYN flag set and outgoing TCP packets with SYN and ACK flag set which can be used to detect this kind of DDoS attack.

4 Evaluation

A prototype of the proposed detection system was implemented on a programmable platform. The basic stage of the detection system is the only service module

loaded at system startup. If this stage detects a stochastic anomaly in any aggregate, specialized service modules for further stages are loaded dynamically.

A network trace of real traffic with an average data rate of about 3 Mbit/s was used as background traffic of a simulation. Additionally, self-generated traffic was used representing a TCP SYN flooding attack with a packet rate of about 15 k packets per interval which corresponds to a data rate of about 0.8 Mbit/s. The average TCP traffic within the background traffic was about 1.7 Mbit/s. Due to the rather low bandwidth of background and attack traffic this evaluation is only a first step towards a small provider scenario but nevertheless, it shows that the mechanisms of the detection system work.

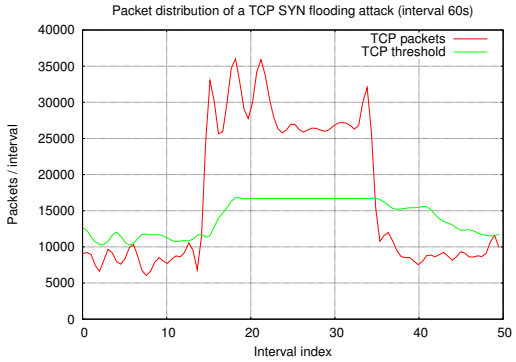

Fig. 4. Packet distribution of a TCP SYN flooding attack

The aggregated traffic – background and attack traffic – was analyzed by our detection system (see fig. 4). The red line shows the observed number of TCP packets per interval whereas the green line shows the packet threshold of the aggregate *TCP packets*. If an indication is generated by the basic stage, the threshold remains constant while the attack is running. We can clearly see that the simulated attack begins in interval 14. The exceeding of the packet threshold in more consecutive intervals than the interval threshold in the TCP aggregate results in loading further stages of the detection system in interval 18. Then, refinement is applied , i.e., the second stage analyzes only the suspicious TCP aggregate in more detail. It scans for a distribution anomaly which provides a differentiation between a DDoS attack and a worm propagation. In the simulation one specific subnet prefix could be detected which most of the traffic is sent to. Thus, the third stage is loaded that again applies refinement and analyzes only those packets of the suspicious aggregate for DDoS-specific protocol anomalies that are sent into the suspicious subnet prefix. In our simulation the third stage was able to detect an asymmetry between incoming TCP SYN packets and outgoing TCP SYN-ACK packets as described in section 3.1. Thus, the system correctly detected the TCP SYN flooding attack.

5 Conclusion and Outlook

In this paper a system for in-network anomaly detection is presented which is hierarchical and applies refinement of detection granularity. Therefore, the system is able to detect various adverse events in different environments, e.g. in small provider networks by scanning for stochastic anomalies, distribution anomalies, and protocol anomalies. A simulation of a TCP SYN flooding attack shows that our anomaly-based system is able to detect DDoS attacks.

In this paper, the evaluation was done only with low-bandwidth background traffic. Thus, future research has to address evaluations using background traffic with a higher bandwidth to simulate a more realistic small provider network. Furthermore, some work has to be done to achieve a differentiation between challenges like DDoS attacks and legitimate traffic with similar characteristics.

6 Acknowledgements

I would like to thank the supervisors of my diploma theses – Dr. Marcus Schöller, Dr. Roland Bless, and Prof. Dr. Martina Zitterbart – for very valuable discussions and their important feedback.

7 References

1. A. Hussain, J. Heidemann, and C. Papadopoulos. A framework for classifying denial of service attacks-extended. Technical Report, USC, 2003.
2. C. Shannon and D. Moore. The spread of the witty worm. IEEE Security and Privacy, 2(4):46 – 50, 2004.
3. I. Ari, B. Hong, E. L. Miller, S. A. Brandt and D. E. Long, "Managing Flash Crowds on the Internet", Proc. 11th IEEE/ACM Int. Symposium on MASCOTS, 2003.
4. L. Ruf, A. Wagner, K. Farkas, and B. Plattner. A detection and filter system for use against large-scale ddos attacks in the internet backbone. Proc. 6th Annual International Working Conference on Active Networking (IWAN), 2004.
5. D. Sass and S. Junghans, I2MP – An architecture for hardware supported high-precision traffic measurement, Proc. 13th GI/ITG Conference MMB. 2006.
6. J. Ioannidis and S. M. Bellovin. Implementing pushback: Router-based defense against DDoS attacks. Proc. of NDSS Symposium, 2002. The Internet Society.
7. D. Sterne, K. Djahandari, R. Balupari, W. L. Cholter, B. Babson, B. Wilson, P. Narasimhan, and A. Purtell. Active network based ddos defense. dance, 00:193, 2002.
8. Vern Paxson, Bro: A System for Detecting Networks Intruders in Real- Time, Computer Networks, 31 (23 – 24), 1999.
9. Cisco Systems, Defeating DDoS attacks, White Paper. 2005
10. N. G. Duffield. A framework for packet selection and reporting. Internet Draft, Work in Progress, Internet Engineering Task Force, January 2005.
11. K. Park and W.Willinger. Self-similar network traffic: An overview. In Self-Similar Network Traffic and Performance Evaluation. Wiley Interscience, 1999.