DoS Amplification Attacks – Protocol-Agnostic Detection of Service Abuse in Amplifier Networks

Timm Böttger, Lothar Braun, Oliver Gasser, Felix Von Eye, Helmut Reiser, Georg Carle

DoS Amplification Attacks – Protocol-Agnostic Detection of Service Abuse in Amplifier Networks

Timm Böttger[1], Lothar Braun[1], Oliver Gasser[1], Felix von Eye[2], Helmut Reiser[2], and Georg Carle[1]

[1] Technische Universität München, Munich, Germany
{boettget,braun,gasser,carle}@net.in.tum.de
[2] Leibniz Supercomputing Centre, Munich, Germany
{voneye,reiser}@lrz.de

Abstract. For many years Distributed Denial-of-Service attacks have been known to be a threat to Internet services. Recently a configuration flaw in NTP daemons led to attacks with traffic rates of several hundred Gbit/s. For those attacks a third party, the *amplifier*, is used to significantly increase the volume of traffic reflected to the victim. Recent research revealed more UDP-based protocols that are vulnerable to amplification attacks. Detecting such attacks from an abused amplifier network's point of view has only rarely been investigated.
In this work we identify novel properties which characterize amplification attacks and allow to identify the illegitimate use of arbitrary services. Their suitability for amplification attack detection is evaluated in large high-speed research networks. We prove that our approach is fully capable of detecting attacks that were already seen in the wild as well as capable of detecting attacks we conducted ourselves exploiting newly discovered vulnerabilities.

1 Introduction

Denial-of-Service attacks aim at making services unavailable to their intended users. Attackers can use different methods to consume bandwidth or deplete other resources of the victim. One method to exhaust bandwidth is called Distributed Reflection Denial-of-Service (DRDoS) attack: an attacker sends forged requests to several servers with the victim's spoofed source address. In response the servers will send replies to the victim. If these replies are significantly larger than the requests the attack is called an *amplification attack*.

Recent research has shown that at least 14 UDP-based protocols are vulnerable to such attacks [10]. Reports show that current amplification attacks can result in more than 100 Gbit/s of bandwidth consumption [11]. The spam blocklist provider Spamhaus was attacked by a DNS amplification attack in March 2013 with an unprecedented traffic rate of up to 300 Gbit/s [8].

Researchers proposed many different mechanisms to identify and protect victims of DRDoS attacks. Furthermore commercial products for this purpose exist. One provider of such products is CloudFlare [1], who successfully mitigated the

above mentioned attack against Spamhaus. In contrast the development of approaches to detect actual service abuse received less attention. Operators of amplifier networks (i.e. networks in which services are abused as amplifiers) are in a good position to take effective countermeasures if they are aware of that their services are used in an attack. However, to enable service operators to employ countermeasures, they first must know that their services are abused as amplifiers. Unfortunately detecting amplification attacks on the border of an amplifier network is more challenging, because illegitimate incoming requests might look the same as legitimate requests.

In this paper we present a novel method to detect service abuse of arbitrary UDP-based protocols in amplifier networks. Our method leverages knowledge on amplification attacks to distinguish legitimate client requests from spoofed attack requests. We evaluate our method with measurements in a large-scale university network. Furthermore, we inject our own attacks into the network for protocols that are known to be exploitable, but for which we did not observe any real world attacks yet.

The remainder of the paper is structured as follows: In Sect. 2 we discuss related work on amplification attacks and DRDoS detection. The following Sects. 3 and 4 present our detection mechanism: we start with already known yet still important prerequisites in Sect. 3 and continue with describing our new approach in Sect. 4. We evaluate the approach in Sect. 5, followed by a discussion on the approach's limitations in Sect. 6. Finally, we conclude the paper in Sect. 7.

2 Related Work

Denial-of-Service attacks have been an active research topic for many years. Specht and Lee provide a taxonomy of attacks, tools and countermeasures and give a good overview of different DoS attack types [12]. Among other attacks, the authors discuss amplification as one way to generate large amounts of attack traffic. Certain protocols, e.g. DNS or SNMP, have long been known to be vulnerable to amplification attacks: Several studies analyze amplification attacks based on the DNS protocol and researchers proposed different methods to identify attack victims [4], [9], [14]. The majority of these studies aims at finding attacks on the border of the victim's network or at detecting attacks which target specific application layer protocols.

Other protocols have recently been identified to be vulnerable to amplification attacks. Rossow revisits a number of UDP-based applications and identifies 14 of them to be vulnerable [10]. He describes how these protocols can be used to conduct attacks and analyzes their possible impact. Even though the list of vulnerable protocols is impressive and contains protocols that have not been known to be vulnerable, it is also known that further protocols such as SIP are vulnerable as well [2]. Based on such findings, one can presume that other protocols could be vulnerable to amplification attacks as well. Therefore we see the need for an amplification attack detection mechanism that works independently from specific protocols. The detection method presented in this paper is

protocol-agnostic and thus differs from previous approaches focusing mostly on individual protocols.

Rossow presents a first detection approach for amplifier networks that is based on NetFlow data [10]. He compares the amount of request and response data sent between a client and a server and reports an attack if a certain threshold is exceeded. The approach is restricted to network protocols that are known to be vulnerable and operate on a fixed UDP port. At the same time he also acknowledges that protocols which exhibit a download-like behavior, e.g. the also vulnerable BitTorrent protocol, will lead to false positives.

Rossow also discusses other approaches for detecting amplification attacks and compares them to his own proposal. Since our work relies on the same considerations as his work, all his considerations apply to our approach as well. We therefore refer the reader to the paper by Rossow [10] for further discussion and comparison with other related work.

3 Important Prerequisites

Some important prerequisites needed for our detection approach to work were already formalized and described by Rossow in [10]:

To identify attacks the communication between a server and a client has to be modeled. In certain protocols, e.g. DNS, the client uses a new port for each request message. The communication between a single client and server can therefore result in multiple UDP flows. To aggregate such a set of flows Rossow proposes to use a so-called *pairflow* for each server/client pair:

$$pairflow := < C_{IP}, S_{IP}, S_{port}, B_{2s}, B_{2c}, t > \tag{1}$$

In a pairflow C_{IP} matches the client IP, S_{IP} and S_{port} are the server's IP and port. Furthermore the payload bytes sent to the server (B_{2s}) and to the client (B_{2c}) are assessed. The duration t of the pairflow is recorded for calculating average rates. To identify the server in a communication flow a fixed set of 14 well-known UDP server ports is used.

In addition Rossow defines the so-called *bandwith amplification factor* (BAF) to characterize the amount of traffic exchanged between client and server. The BAF is calculated per pairflow as:

$$BAF = \frac{len(UDP\ payload)\ amplifier\ to\ victim}{len(UDP\ payload)\ attacker\ to\ amplifier} \tag{2}$$

Communication between a server and a client with at least a 10 kBit/s data exchange rate, a BAF larger than five and a server that sends more than 10 MB of payload is classified to be an amplification attack.

4 Detection Approach

Our detection approach exploits characteristics of attack traffic to distinguish it from legitimate traffic. For modeling the communication relationship between

server and client we rely on the foundations laid by Rossow as explained in the previous section. We also stick to his thresholds, i.e. classifying a pairflow as an attack if it exhibits a BAF of five and more than 10MB of traffic are sent towards the victim. These thresholds are reasonable as amplification attacks are characterized by an amplifier which sends a lot more traffic than it receives. Hence we expect pairflows corresponding to an amplification attack to exhibit a (relatively) large BAF. The threshold of 10MB is probably large enough to not be easily reached with simple requests but at the same time should be small enough such that amplification attacks certainly reach it.

In contrast to Rossow we want to provide a protocol-agnostic approach that does not depend on a fixed set of well-known UDP server ports. To build a pairflow, however, we need to identify the client and server roles of the communication. As these roles can not be reliably determined we assume that the servers are within our network. This simplification is reasonable because we want to detect amplifiers within the monitored network. However, this might lead to internal clients being treated as servers, potentially resulting in false positives.

As opposed to Rossow we are not working with NetFlow data, thus we chose to apply ten minutes active/inactive timeouts to each pairflow.

4.1 Characteristic Properties of an Amplification Attack

Even though the bandwidth and BAF criteria are surely fulfilled by every amplification attack relying only on these two criteria is, as we will show later in Sect. 5, not sufficient because they are also fulfilled by legitimate service usage (e.g. Peer-to-Peer or VPN traffic). Hence more criteria are needed to prevent false positive alarms from being generated.

To derive further criteria it is beneficial to discuss certain aspects of an amplification attack in more detail: To conduct an amplification attack the attacker sends requests to an amplifier service, which she expects to be answered with responses larger than the requests. These responses are in turn sent to the victim. In order to accomplish this task, the attacker must use the IP address of the victim as source address for her requests.[3] If the attacker is not located on the same broadcast domain as the amplifier or the victim, which is the common case we focus on, then the attacker will not see any response packets from the amplifying service.

Therefore an attacker can neither establish shared state with the amplifying service through the request packets, nor can she be sure that her requests produce the desired response. As a consequence she cannot send arbitrary requests to the amplifying service, but only those that do not require shared state. She

[3] It seems reasonable to assume that nowadays filter mechanisms to mitigate IP spoofing are widely deployed. Unfortunately the Spoofer Project reports that roughly 40% of all AS' worldwide allow (at least partially) for using spoofed sender IP addresses [13]. Furthermore to effectively prevent IP spoofing all AS' must filter their traffic, because the attacker needs to find only one AS allowing spoofed IP addresses. Hence we must IP spoofing expect to happen.

furthermore is interested in sending requests where she is reasonably sure to produce a large response. As the possible requests an adversary would use are limited, we expect highly similar messages from the attacker during a single attack. In turn the responses generated by the amplifier are also expected to share similarities.

If an attacker is successful with provoking the amplifier to generate messages, then the victim will receive many unsolicited messages, i.e. messages that it did not request and hence not expects. A reasonable network stack should react to such unsolicited messages by sending ICMP port unreachable messages. Therefore in the early stages of an attack, when the resources are not yet depleted, such ICMP messages sent by the victim might be observed.

4.2 Improved Amplification Attack Detection Criteria

Based on the previous considerations we propose to use the following additional criteria for amplification attack detection:

Request and response packet size similarity: The attacker wants to obtain a large amplification factor while at the same time she is, as argued above, restricted in the requests she can send. Thus she is likely to only use a very small set of different requests, for which she verified in advance that they will generate large responses. A simple attacker might even stick to using only the one request which yields the highest amplification factor. In conclusion the attacker will only use a few different short requests, so we expect the sizes of the request messages to be very similar.

The amplifying service on the other hand cannot rely on shared state with the attacker, therefore, if not returning random information, the responses to the same request are expected to be similar. Likewise, as the attacker only uses a few different request messages, the amplifying service can only generate a few different responses. Thus we also expect the sizes of the response messages to be similar between all responses belonging to a single attack. To measure this similarity we assess the packet's payload sizes in both directions of the communication.

Request and response payload similarity: We already justified that an attacker will only rely on a very restricted set of requests. For a single attack we therefore expect the payloads of the requests sent by the attacker to be very similar. The responses from the amplifier are expected to exhibit the same characteristic as these responses are generated by only a small set of different requests.

In order to assess the similarity of the messages, we apply the deflate compression as provided by the zlib library [16] to the payloads and use the ratio of compressed and uncompressed size for a similarity estimation. The deflate algorithm uses both Huffmann coding and LZ77 compression to create the compressed data. To save resources on the monitoring system we sample 100 packets per direction of a pairflow after this pairflow reached the BAF threshold and

then apply the compression to this sample only. Using the payloads we calculate a *similarity factor* (SF) per traffic direction as

$$SF = 1 - \frac{len(deflate(concatenated\ UDP\ payload))}{len(concatenated\ UDP\ payload)}. \quad (3)$$

A similarity factor close to one indicates good compressible and hence similar payloads, a similarity factor close to zero indicates rarely compressible and hence unsimilar payloads. This calculation is performed separately for each direction.

Unsolicited Messages: The messages from the amplifying service that the victim receives are unsolicited messages. A network stack should react on these messages with ICMP port unreachable messages if no service is running on the port that receives the UDP frame. As a further attack indicator we count the number of ICMP port unreachable messages for any possible victim.

IP Spoofing: We use the IP header field of incoming requests to determine the path length between the sender of the request and the amplifier service. Initial TTL field values are set by the operating system and differ between the OS[4]. Each IP router on the way decrements the TTL by one. We record the TTL of the incoming requests and calculate the IP path length by comparing the value to the nearest known initial value for different operating systems. If we receive an ICMP reply from the victim, we also extract the path length in the same way. We can use the difference between those values to check whether the path length of the request from the attacker and the path length of the victim differ. If we do not receive any ICMP messages, we try to obtain the path length ourselves by performing trace routes to the victim.

Other Criteria: Surely there will be further criteria that can be used to distinguish legitimate and attack traffic. One possible criterion might be the client's inter-arrival times. For an attack we would expect very small, almost similar inter-arrival times, whereas for an interactive session we would expect higher inter-arrival times with a higher variance. However, so far we restrict our attention to the four criteria mentioned above and leave further criteria for future research.

5 Evaluation

We implemented our method as a module of an Intrusion Detection System (IDS) and evaluated it on real traffic traces. Sect. 5.1 describes our measurement setup in a large university network. As only a small subset of all vulnerable protocols is currently used in real-world attacks, we additionally conducted attacks ourselves to also evaluate our approach on the other protocols, which we describe

[4] Linux and many BSD variants use an initial TTL of 64, Windows networking stacks have an initial value of 128 and some Unix variants start with a TTL of 255.

in Sect. 5.2. Our measurements in a high-speed research network are explained in Sects. 5.3, 5.4 and 5.5. We begin by briefly presenting the individual measurement runs, continue with using a subset of one run for deriving detection thresholds and finish by applying these thresholds to all three measurement runs.

5.1 Measurement Setup

We conducted the traffic measurements in the Munich Scientific Network (MWN) which is operated by the Leibniz Supercomputing Centre (LRZ). This network infrastructure connects the different sites of the Munich universities, many student residence halls, the Bavarian Academy of Science and Humanities, the Bavarian State Library, Max Planck and Fraunhofer Society institutes and various museums. In the course of one month the LRZ handles more than 1200 TByte of inbound and 730 TByte of outbound traffic. On average the measured link transmitted 2.6 GBit/s of incoming and approximately 1.5 GBit/s of outgoing traffic.

The traffic measurements were conducted on a dated commodity server running a Linux 3.2 kernel. It employs a 3.2 GHz Intel Core i7 CPU with four cores with hyperthreading and 12 GB RAM. The machine is connected to a monitoring session at the LRZ's border gateway router via an Intel 10 GE network card that is based on the 82598EB chipset. The card is driven by PF_RING and *Direct NIC Access* (DNA) [6], which is a zero-copy solution that allows the network card to directly copy packets into the userspace application without any CPU overhead. The capturing was configured to pass only UDP and ICMP traffic to the user space application, because we expect IP address spoofing to happen only on UDP.

5.2 Generated Attack Traffic

During our initial investigation, we realized that the only real-world attacks in the network were abusing DNS and NTP services. However, recent prior research showed that additional protocols are vulnerable [10]. In order to evaluate whether our approach is suitable to detect those cases, we created our own attacks on known vulnerable protocols. We searched for freely available attack tools for amplification attacks and found several that supported attacks on SNMP, DNS and NTP. None of the other vulnerable protocols were supported by these tools, so we added support for these protocols. The functional extensions are implemented to exploit the vulnerabilities as outlined in [2] and [10] .

Some of the vulnerable protocols could easily be exploited: NTP, DNS, SNMPv2, Chargen and SIP have implementations that are simple to exploit if the service is provided to the open Internet. For some protocols we had to alter the standard configurations, as per default the services are configured securely.

Other services posed more difficulties: The legacy Quote-of-the-Day (QOTD) service's exploitability strongly depends on the actual implementation and the size of the returned quotes. Implementations following the recommendations of

the RFC [7] only send quotes with 512 or less characters. Therefore they can only produce low bandwidth attacks. Nevertheless exploiting the protocol is possible.

For aMule, Quake3 and Steam we are able to confirm their vulnerability. However, all of them include hard-coded rate-limits which effectively prohibit the generation of significant attack traffic. Thus we could not include them in our evaluation. Similarly many BitTorrent clients employ rate-limiting for their vulnerable DHT protocol. However, when intially writing this paper we discovered that the Mainline DHT plugin of Vuze [15] was vulnerable. More recent versions seem not to be vulnerable to our attacks any longer.

We created attacks using services we deployed in the monitored amplifier network. Both attacker and victim networks were placed outside the monitored networks. This setup allowed us to inject the attack traffic and simultaneously monitor it as part of our live traffic measurements. Our attacks lead to amplification factors (BAFs) ranging from roughly five (BitTorrent) and ten (SIP) up to 2,500 (NTP). Using the NTP protocol we generated 500MB per attack, using BitTorrent we generated roughly 100MB and using SIP only 30MB.

5.3 The Measurement Runs

For the final evaluation of our detection approach we conducted three measurement runs. The first one took place from June 7 until June 13, 2014 and lasted for 144 hours. The second run lasted for 96 hours from September 26 until September 30, 2014. The last run was performed from September 30 until October 1, 2014 and captured another 24 hours. For each run we logged all the pairflows exceeding the BAF-thresholds mentioned in the beginning of Sect. 4. The information in Table 1 thus refers to all logged pairflows only. We only conducted own attacks during measurement run #1.

Table 1. Measurement Runs

	Run #1	Run #2	Run #3
Duration (in h)	144	96	24
Total Bytes Sent	7,340.66 GB	3,425.62 GB	734.67 GB
Total Packets Sent	6,589,456,476	3,208,724,852	674,865,692
Total Pairflows Reported	77,693	45,747	10,974
Unique Server-Port-Client Triples	22,428	14,567	4,058
Unique Server-Port Pairs	3,324	1,682	504
Unique Servers	530	309	204

5.4 Deriving Detection Thresholds

If we want to use the criteria explained above to detect amplification attacks, we must first define detection thresholds. For that we extracted a sample set

of pairflows, which we manually classified as attacks or legitimate traffic. This training set was chosen as a subset of the first measurement by only considering an 8 hour timeframe. It was chosen in such a way that it includes most of the attacks we conducted ourselves. The training set covered roughly 5% of the traffic captured during the first measurement run. Table 2 contains further details about the size of the training set.

Table 2. Training Set

	Training Set	Run #1	Share
Duration (in h)	8	144	5.55%
Total Bytes Sent	380.19 GB	7,340.66 GB	5.18%
Total Packets Sent	365,076,992	6,589,456,476	5.54%
Total Pairflows Reported	4,883	77,693	6.28%
Unique Server-Port-Client Triples	1,573	22,428	7.01%
Unique Server-Port Pairs	348	3,324	10.47%
Unique Servers	146	530	27.54%

In the following we will separately deal with our attacks using the Quote-of-the-Day (QOTD) protocol. They are unique in the sense that for the same request the server can reply with an arbitrary quote. This will significantly interfere with our detection criteria as we assumed that for a reasonable service the replies to the same request will be the similar. But as QOTD is the only[5] service exhibiting this behavior network operators will be able to compensate for attacks using this one special service. Nevertheless dealing with QOTD services even today is relevant as identifying a thousand exploitable QOTD services in the Internet took Rossow less than four minutes on average [10]. It is true that also for other attacks the attacker can try to evade the detection by using different requests. However, to achieve a significant impact she will still have to stick to a set of requests yielding a high amplification factor. Thus by just sampling more packets per pairflow the similarity can still be detected and this problem can be remediated. But so far we did not observe any attacks with varying request patterns.

We begin with how the similarity factors differ between attack and legitimate traffic. Please not that in the following each pairflow is only displayed once. Hence multiple attacks from the same attacker towards the same victim using the same protocol only result in one data point plotted. According to Fig. 1 the similarity factors already seem to provide a measure to distinguish the two classes of traffic: As we claimed above for attacks we observe high similarity factors whereas for legitimate traffic the similarity factors are significantly lower. As expected the

[5] Every other service exhibiting this behavior can be seen as QOTD service with a very broad set of quotes.

QOTD-attacks exhibit a significantly lower similarity factor as the other attacks. Based on our training set we choose to require similarity factors of 0.75 or more in each direction to classify a pairflow as an attack. This choice ensures that all attacks are captured while at the same time the vast majority of legitimate pairflows is not captured. The SF-values less than zero are artifacts caused by our packet capture method. They stem from pairflows from which due to active timeouts only a few packets were sampled. This lead to an increased size after compression.

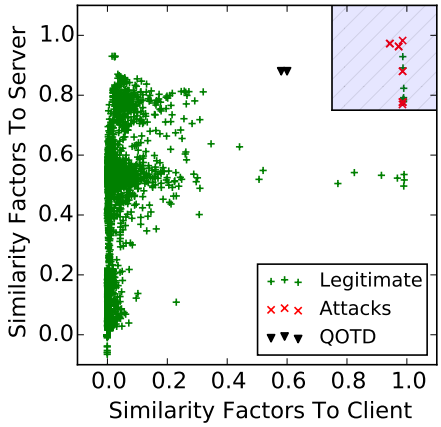

Fig. 1. Similarity Factors

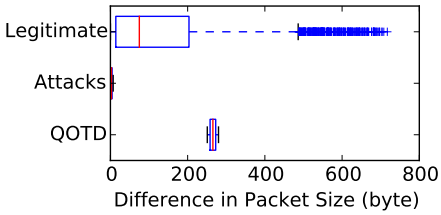

Fig. 2. Packet Sizes To Client

As second criterion we want to evaluate the difference in the sizes of the packets. We calculated the difference between the average packet size and the minimal resp. maximal packet size and took the smaller of the two differences. As Fig. 2 and 3 indicate these differences are also different for attack and non-attack traffic. In accordance with these figures the remainder of this paper requires a difference of 25 or less bytes of the packet sizes in either direction to client or direction to the server for an amplification attack to happen.

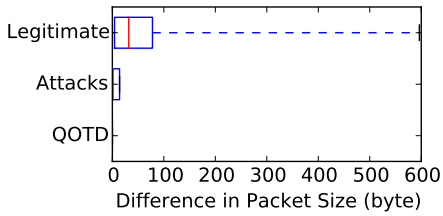

Fig. 3. Packet Sizes To Server

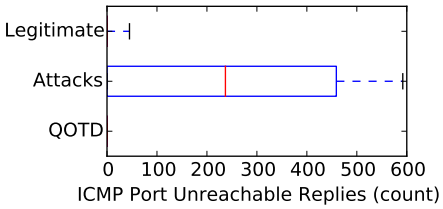

Fig. 4. ICMP Port Unreachable Replies

We also tried exploiting the ICMP unreachable replies. As shown in Fig. 4 it is true that for attack traffic we observe more ICMP unreachable replies as for legitimate traffic. But even in our small subset of mostly controlled attacks the number of ICMP unreachable replies varies largely. As we cannot assure that for every attack ICMP unreachable replies are present, we will not further use this criterion for our attack detection. However as their presence still is a strong indicator for an undesired behavior resp. an attack, the generated alarms should be enriched with the number of observed ICMP unreachable replies.

The same is true for the path length detection. We could only obtain path length information for a minority (roughly 20%) of all pairflows. Thus we cannot rely on mismatches in path length for the actual attack detection, but for a pairflow classified as attack the path length information can be used to harden the detection result.

As the MWN connects different types of users like universities, student halls or research institutions, the traffic we observe is a representative cross-section of different network types. We therefore believe that for other networks similar thresholds as the ones we derived here can be used.

5.5 Live Measurement Evaluation

To evaluate the decisiveness of our new criteria and thresholds we applied them to all pairflows exceeding the BAF-thresholds. After applying them we manually verified all pairflows that were marked as an attack. For the pairflows classified as legitimate traffic we only verified that we did not miss an attack using one of the well-known vulnerable protocols.

We grouped alerts by the triple of server, port and client, hence a long-lasting attack resulting in several pairflows is counted only once. Table 3 summarizes the detection results and proves that our approach is capable of very precisely distinguishing legitimate from attack traffic. It detected all attacks that took place and at the same time produced only very few false positive alarms.

Table 3. Detection Summary

	Run #1	Run #2	Run #3
BAF identified services	3,324	1,682	504
BAF identified alarms	22,428	14,567	4,058
True positive alarms	277	30	18
False positives alarms	3	9	0
True negative alarms	22,149	14,534	4,041
False negatives alarms	0	0	0

For the true positives we encountered some attacks (roughly five per run) using the SIP-protocol, which were similar to our own attacks. Thus we classify

them as amplification attacks, while we cannot distinguish them from enumeration attacks for sure. In any case, administrators should be informed about them.

All the false positive alarms were mainly raised due to highly similar payload content. For all of them we determined the used application layer protocol with nDPI [5] resulting in six alarms for BitTorrent, one for Skype and two for unknown protocols. In all cases we could manually verify the similarity of the payloads due to the presence of repeating byte patterns. For Skype and BitTorrent we cannot explain what caused the similarity. For the unknown protocol a lot of null-bytes were observed which were probably used for payload padding.

This evaluation further proves that our additional criteria are necessary. When omitting them and only relying on the BAF-criteria from the beginning of Sect. 4, all pairflows that our approach classified as true negative alarms would be classified as amplification attacks. Thus applying only the BAF-criteria to all server ports without additional checks leads to a large amount of false positive alarms.

6 Detection Evasion and Limitations

In the following we will discuss evasion strategies and limitations of our approach.

Evading Detection: Our detection approach imposes assumptions on the attacker's behavior which can be used by an attacker to evade the detection. First of all we require a certain BAF and amount of traffic to be sent. An attacker can clearly evade our detection by generating less traffic. However, by doing so, she reduces the impact of the attack, which is desirable from our point of view. When reducing the amount of traffic sent below our detection rate, the impact of this attack is very low and hence neglectable. To overcome this an adversary could employ several amplifiers and forging requests such that each amplifier does not send more than 10 MB of traffic in ten minutes. However, to achieve a significant impact many amplifiers must be used as for this scenario each single amplifier may not exceed an average outgoing traffic rate of 136 kBit/s.

Instead of reducing the amount of attack traffic, an attacker can try to adapt his request packet lengths and payload entropy. She has two ways to achieve this goal: Firstly, she can send garbage messages to the amplifying service that are not legitimate messages. Since we have a generic protocol-independent approach, we cannot detect this. However, the attacker will reduce his amplification factor if she sends such messages. Secondly, she can try to employ different types of messages in his attack, which still result in large response messages. In general, however, this decreases the amplification factor as typically only a few requests yield high amplification ratios. This can be further dealt with by sampling more packets per pairflow to get a better estimate of the message similarity. Figure 1 indicates that there is a large gap between attacks and legitimate traffic when evaluating the similarity factors. Hence lowering the detection threshold should allow for detecting even attacks with a varying request message scheme while at

the same time only very few additional false positive alarms are raised. Evading the detection of our approach would therefore reduce the impact that an attacker can have with his amplification attack.

Limitations of the approach: We rely on estimating the entropy of the communication. If an attacker succeeds in generating encrypted amplification traffic, this criterion will fail as encrypted traffic looks rather random. However, we argue that generating encrypted amplification traffic is not easily achievable. Setting up encryption requires holding state which in case of an amplification attack as explained above is not possible.

The approach is designed for networks that can be monitored at a single point, in the simplest case for networks having only one uplink. If a network is connected through multiple uplinks our approach can still be applied if the traffic running through the uplinks is consolidated in a suitable way at single monitoring points. This might be achieved by consolidating all traffic at one monitoring point or at multiple monitoring points by applying a suitable splitting scheme. Nevertheless monitoring a network with multiple uplinks is a more general problem set which is out of the scope for this paper.

7 Conclusion

Distributed Reflection Denial-of-Service attacks are responsible for significant disruptions in the Internet. Recent research mainly has focused on detection of DRDoS-attacks on the edge of the victim's network. The potential countermeasures against such attacks that service operators in amplifier networks can employ remained unused, as detection of such attacks was hardly possible. In this paper we presented a novel approach to successfully solve this shortcoming.

As detection base we reused ideas from an already existing detection approach. Our key contributions are two novel detection criteria which allow for distinguishing between legitimate and attack traffic for any arbitrary application protocol. We showed that our protocol-agnostic approach enhances the detection process by not only defending against attacks on static port numbers, but also to thwart novel DRDoS attacks. Our practical evaluation in a large scientific network revealed that with our approach we were able to detect real attacks as well as artificial attacks that used new vulnerabilities.

In comparison to other mitigation strategies, like e.g. BCP 38 [3], our approach is applicable in the amplifier network, where the BCP 38 approach focuses on filtering in the attacker's network. Patching or disabling affected services also is a possible solution, however simply patching or disabling might not always be possible. With our approach network operators can at least detect ongoing amplification attacks. Additionaly our method only requires modest hardware; we used a dated commodity server.

In the future the detection scheme can be improved by changing it to a feedback-driven approach using machine-learning capabilities. We are confident that the criteria we developed in this paper will be suitable features for such

an approach. Additionally measures to detect IP spoofing will surely help to strengthen the detection results.

Acknowledgement. This work has been supported by the German Federal Ministry of Education and Research (BMBF) under support code 01BY1203C, project *Peeroskop*, and 16BP12304, EUREKA project *SASER*, and by the European Commission under the FP7 project *EINS*, grant number 288021.

References

1. CloudFlare. https://www.cloudflare.com/, last accessed: December 2014
2. Fatih Özavci: VOIP Wars: Return of the SIP. DEFCON 21 - http://www.defcon.org/images/defcon-21/dc-21-presentations/Ozavci/DEFCON-21-Ozavci-VoIP-Wars-Return-of-the-SIP-Updated.pdf (Aug 2013), last accessed: December 2014
3. Ferguson, P., Senie, D.: Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing. RFC 2827 (Best Current Practice) (May 2000), http://www.ietf.org/rfc/rfc2827.txt, updated by RFC 3704
4. Kambourakis, G., Moschos, T., Geneiatakis, D., Gritzalis, S.: Detecting DNS Amplification Attacks. In: Critical Information Infrastructures Security (CRITICS 2007). Springer (2007)
5. nDPI-Homepage. http://www.ntop.org/products/ndpi/, last accessed: December 2014
6. Direct NIC Access – Gigabit and 10 Gigabit Ethernet Line-Rate Packet Capture and Injection. http://www.ntop.org/products/pf_ring/dna/, last accessed: December 2014
7. Postel, J.: Quote of the Day Protocol. RFC 865 (INTERNET STANDARD) (May 1983), http://www.ietf.org/rfc/rfc865.txt
8. Prince, M.: The DDoS That Almost Broke the Internet (Mar 2013), http://blog.cloudflare.com/the-ddos-that-almost-broke-the-internet, last accessed: December 2014
9. Rastegari, S., Saripan, M.I., Rasid, M.F.A.: Detection of Denial of Service Attacks against Domain Name System Using Neural Networks. International Journal of Computer Science Issues (IJCSI) 7(4) (2009)
10. Rossow, C.: Amplification Hell: Revisiting Network Protocols for DDoS Abuse. In: Proceedings of the 2014 Network and Distributed System Security (NDSS) Symposium. San Diego, CA (Feb 2014)
11. Soluk, K.: NTP ATTACKS: Welcome to The Hockey Stick Era. http://www.arbornetworks.com/asert/2014/02/ntp-attacks-welcome-to-the-hockey-stick-era/ (Feb 2014), last accessed: December 2014
12. Specht, S., Lee, R.: Distributed Denial of Service: Taxonomies of Attacks, Tool and Countermeasures. In: Proceedings of the ISCA 17th International Conference on Parallel and Distributed Computing Systems. San Francisco, CA (Sep 2002)
13. Spoofer Project: State of IP Spoofing. http://spoofer.cmand.org/summary.php, last accessed: December 2014
14. Sun, C., Liu, B., Shi, L.: Efficient and Low-Cost Hardware Defense Against DNS Amplification Attacks. In: IEEE Global Telecommunications Conference 2008. (GLOBECOM 2008). IEEE (2008)
15. Vuze homepage. http://www.vuze.com/, last accessed: December 2014
16. zlib Homepage. http://www.zlib.net/, last accessed: December 2014