AmpPot: Monitoring and Defending Against Amplification DDoS Attacks

Lukas Krämer†, Johannes Krupp†, Daisuke Makita‡⋄, Tomomi Nishizoe‡,
Takashi Koide‡, Katsunari Yoshioka‡, Christian Rossow(✉)†

† CISPA, Saarland University, Germany
‡ Yokohama National University, Japan
⋄ National Institute of Information and Communications Technology, Japan

Abstract. The recent amplification DDoS attacks have swamped victims with huge loads of undesired traffic, sometimes even exceeding hundreds of Gbps attack bandwidth. We analyze these amplification attacks in more detail. First, we inspect the reconnaissance step, i.e., how both researchers and attackers scan for amplifiers that are open for abuse. Second, we design AmpPot, a novel honeypot that tracks amplification attacks. We deploy 21 honeypots to reveal previously-undocumented insights about the attacks. We find that the vast majority of attacks are short-lived and most victims are attacked only once. Furthermore, 96% of the attacks stem from single sources, which is also confirmed by our detailed analysis of four popular Linux-based DDoS botnets.

1 Introduction

Distributed denial-of-service (DDoS) attacks have threatened critical Internet infrastructures for many years [1–3]. Recently, in particular amplification DDoS attacks [4] have gained increasing popularity. In such amplification attacks, an attacker abuses so called *amplifiers* (or *reflectors*) to exhaust the bandwidth of a victim. Instead of directing the attack traffic to the victim directly, the adversary sends requests to reflectors and spoofs the source IP address, so that the reflectors' responses are directed to the victim. An attacker may abuse any public server that is vulnerable to reflection attacks, such as open DNS resolvers or NTP servers. Worse, these protocols are known to amplify the bandwidth significantly, easily allowing an attacker to launch Gbps-scale attacks with a much smaller uplink. In fact, amplification attacks have caused the largest DDoS attack volume ever observed, e.g., against Spamhaus in 03/2013 (≈ 300 Gpbs) and OVH in 02/2014 (≈ 400 Gbps).

The rise of amplification attacks raises many research questions. How frequent are such attacks, and whom do they target? Are individual sources spoofing traffic to trigger attack traffic, or do distributed botnets cause the DDoS attacks? Which software do adversaries use to launch the attacks, and how do they identify amplifiers? Can network-based filtering methods be used to detect amplification attacks? All these questions help to improve our understanding of the threat, to learn attack motivations, and to devise effective countermeasures.

In this paper, we will close this gap by studying in-the-wild activities of attackers preparing and launching amplification DDoS attacks. We first leverage a `/16` IPv4 darknet to identify scans for amplifiers, revealing that over 5,000 hosts scanned for DDoS-related services. We observe the scans over time, and monitor a sudden increase of scans caused by whitehats in early 2014. Further analyses reveal that scans are widely distributed, and large parts of the scans rely on Zmap [5] for their reconnaissance. We then perform a longitudinal study of amplification attacks. To this end, we introduce AmpPot, a novel open-source honeypot specifically designed to monitor amplification attacks. AmpPot can mimic services that are known to be vulnerable to amplification attacks, such as DNS and NTP. To make them attractive to attackers, our honeypots send back legitimate responses. Attackers, in turn, will abuse these honeypots as amplifiers, which allows us to observe ongoing attacks, their victims, and the DDoS techniques. To prevent damage caused by our honeypots, we limit the response rate. This way, while attackers can still find these rate-limited honeypots, the honeypots stop replying in the face of attacks. We deployed 21 globally-distributed AmpPot instances, which observed more than 1.5 million attacks between Feb. and May 2015. Analyzing the attacks more closely, we find that more than 96% of the attacks stem from single sources, such as booter services. We show that most attacks are relatively short-lived, and victims are rarely attacked multiple times—giving interesting insights into the motivation behind the attacks. We conclude that amplification DDoS attacks are a global problem, with most victims being located in the US (32%) and China (14%).
To foster attack mitigation, we further devise reactive countermeasures against amplification attacks. First, we provide a live feed of amplification attacks. Second, we derive and present a list of domains that are abused in DNS-based amplification attacks. Finally, to study the root cause of amplification attacks, we analyze the new trend of Linux-based DDoS botnets. We inspect over 200 DDoS malware samples and classify most of them into four families. We manually reverse-engineer these samples to analyze their attack techniques, revealing amplification capabilities in all families. In an attempt to map attacks to DDoS botnets, we fingerprint the traffic of these families and link it to the attacks observed at the honeypots. This analysis reveals little overlap, showing that DDoS botnets are not the main source of amplification attacks.

To summarize, the contributions of this paper are as follows:

1. We design AmpPot, a novel honeypot to capture amplification DDoS attacks. We evaluate various response modes and, based on our collected attacks, devise best practices for deploying such honeypots.
2. We leverage a `/16` darknet and the data collected by 21 AmpPot instances to shed light on the current state of in-the-wild amplification attacks. We use these results to derive honeypot-assisted defense mechanisms.
3. We analyze the recent threat of Linux-based DDoS bots. We show that these bots offer amplification DDoS capabilities, but using traffic fingerprinting, we also reveal that their overall share in the amplification attacks is negligible.

2 AmpPot

This section starts with background information on amplification DDoS attacks. We then describe AmpPot, our novel honeypot that monitors amplification DDoS attacks.

2.1 Background

Amplification DDoS attacks aim to congest the network bandwidth of attack targets [4]. Attackers use two main techniques to launch amplification attacks. First, they abuse UDP-based Internet services that reflect traffic. For example, attackers may abuse open DNS resolvers to trigger responses to DNS lookups. By choosing particular DNS queries, attackers can even ensure that the responses are much larger than the requests—therefore triggering traffic amplification. Second, attackers spoof the source IP address of the traffic so that the responses flood a victim, instead of going back to the attacker. Such attacks inherently require *amplifiers*, i.e., hosts offering services that are vulnerable to amplification DDoS. Rossow documented 14 UDP-based protocols that can be abused for DDoS attacks, such as DNS, NTP or SNMP [4]. For many of these protocols, adversaries simply use Internet-wide scans to identify millions of amplifiers. Once discovered, attackers will abuse a subset of the discovered amplifiers as part of their attacks.

2.2 Honeypot Design

In the following, we will describe AmpPot, which acts as fake amplifier. Based on the above observations, we can use this honeypot to i) monitor reconnaissance steps performed by potential attackers, and ii) monitor amplification attacks. AmpPot mimics services having amplification attack vectors by listening on UDP ports that are likely to be abused. In particular, AmpPot supports all protocols that are said to be vulnerable [4]: QOTD (17), CharGen (19), DNS (53), NTP (123), NetBIOS (137), SNMP (161) and SSDP (1900), plus MSSQL (1434) and SIP (5060/5061). To serve these protocols, AmpPot listens on the according ports for incoming UDP packets.

Modes: Whenever AmpPot receives a request, it will respond. We use three "modes" that influence the type of response we send back:

- **Emulated**: In this mode, we use protocol-specific parsers. If a request is valid, we reply with a response, which is randomly chosen from a pre-generated set of protocol-specific responses. For a few protocols such as DNS, which requires dynamically-generated responses that are specific to the request (e.g., the queried domain name), we recursively resolve the requested resource before responding.
- **Proxied**: The proxy mode turns AmpPot into a proxy that forwards requests to internal servers that actually operate the vulnerable protocol. The responses, in turn, are sent back to the client. While this mode requires configuring servers (such as a DNS resolver, or NTP time server), it has the advantage that no emulation is needed.

- **Agnostic**: Finally, when run in the agnostic mode, AmpPot responds regardless of the validity of the request. In fact, even the response is invalid: AmpPot replies with a large response that contains random bytes (either 100x the size of the requests, or with the maximum MTU). This mode assumes that the attacker does not really care about the validity of the responses, but instead just aims to find hosts that send back large replies.

Section 4 will compare these three modes in terms of their effectiveness.

Responses: AmpPot is most attractive for attackers if its responses result in amplification. To be attractive, we carefully designed protocol-specific responses (emulated mode) or configured servers that send back attractive payloads (proxy mode). For example, for DNS we resolve the request that the client sent, and respond with the entire response, in particular also following the EDNS extensions to support large payloads. Furthermore, we trigger responses that are both vulnerable to NTP's `monlist` request and many other amplifying responses (e.g., version info). We gained this knowledge by a) inspecting known vulnerability reports, b) passively observing requests targeting a darknet (see Section 3), and c) scanning the Internet to find typical large responses. Except for the agnostic mode, we made sure that popular client software for each protocol can successfully parse the responses.

Rate Limiting: By mimicking services that have amplification vulnerabilities, AmpPot runs a risk of becoming involved in actual DDoS attacks. On the other hand, in order to attract attackers, the honeypots need to respond as if they were vulnerable. We have thus added a rate-limiting mechanism to AmpPot that helps to distinguish between scans (to which we would like to reply) and attacks (in which we do not want to participate). In particular, we block a client IP address (and its corresponding `/24` network), if the client sends more than 10 requests per minute. Once a network is blocked, no requests from this network range will be answered. After an hour, we re-evaluate the blacklist and remove a network from the blacklist when it has ceased sending requests. In our later deployment of the honeypots, we received only four emails from attack victims, which we responsibly answered. After our clarification, none of the victims claimed that we caused damage.

Data Collection: One of the core components of AmpPot is data collection. We collect data in two ways: raw requests and filtered data. Raw requests are simply recorded as `.pcap` files. However, as the raw data becomes large and difficult to handle quickly, we also record a filtered dataset. For this, each honeypot records the first 100 requests per source IP address and stores them in a *sqlite* database. The relational database eases analysis and data sharing.

Tool Sharing: AmpPot is implemented in Python and follows a modular design. We will share AmpPot with trusted parties and make it accessible to fellow researchers, assuming that we can use the derived data as input for the attack portal. Please contact Christian Rossow to obtain access to the source code.

2.3 Honeypot Deployment

We deployed 21 AmpPot instances to collect attack information. Table 1 summarizes our farm: eleven emulated, seven proxying and three agnostic honeypots. The emulated honeypots are scattered across countries, whereas the other honeypots are all located at Japanese ISPs.
In an attempt to make the honeypots popular, we tried to host the honeypots at ISPs providing static IP addresses. In a few cases, the honeypots have semi-dynamic IP addresses. That is, the addresses change every 3–10 weeks on average, as indicated by the braces in the *IP Addr.* column. Most honeypots were deployed in 2014 and have been continuously operated since then.

HP	Type	Location	Deployed	IP Addr.	Services
E01	Emulated	Australia	2014-11-14	Static	9
E02	Emulated	Brazil	2014-11-14	Static	9
E03	Emulated	US West	2014-11-14	Static	9
E04	Emulated	Ireland	2014-11-14	Static	9
E05	Emulated	Japan	2014-11-14	Static	9
E06	Emulated	Singapore	2014-11-14	Static	9
E07	Emulated	US West	2014-11-14	Static	9
E08	Emulated	US East	2014-11-14	Static	9
E09	Emulated	Greece	2014-12-10	Static	9
E10	Emulated	Iceland	2014-12-10	Static	9
E11	Emulated	Netherlands	2014-12-10	Static	9
P01	Proxy	Japan	2012-10-07	Dyn. (27d)	6
P02	Proxy	Japan	2013-05-13	Dyn. (22d)	1
P03	Proxy	Japan	2014-05-13	Dyn. (71d)	6
P04	Proxy	Japan	2014-05-13	Dyn. (33d)	6
P05	Proxy	Japan	2014-05-10	Static	6
P06	Proxy	Japan	2014-05-10	Static	6
P07	Proxy	Japan	2014-05-10	Static	6
A01	Agnostic F	Japan	2014-10-14	Dyn. (51d)	7
A02	Agnostic F	Japan	2014-10-24	Static	any
A03	Agnostic M	Japan	2014-11-23	Static	any

Table 1: Overview of honeypot deployments.

The honeypots support a variety of protocols. The proxy honeypots support CharGen, QOTD, DNS, NTP, SNMP and SSDP. In a continuous effort to support more protocols, we gradually added SNMP and SSDP after an initial deployment with the remaining subset of four protocols only. P02 support DNS only. The emulated honeypots support three additional protocols (NetBIOS, MSSQL, SIP). Finally, two of the agnostic honeypots listen on all UDP ports with varying response strategy settings. *Agnostic F* denotes that the honeypot always replied with 1472 bytes UDP payload. In contrast, *Agnostic M* multiplies the length of the request payload by 100 to create a response that is relative in length to the request. Either way, the responses contained random UDP payload that is not valid for the scanned protocol. Section 4 will analyze the effects of the varying settings of the agnostic honeypots.

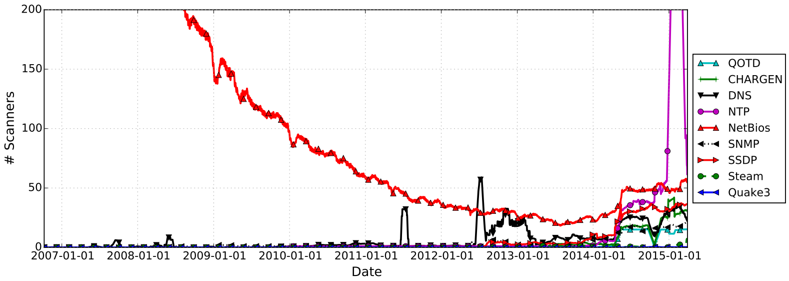
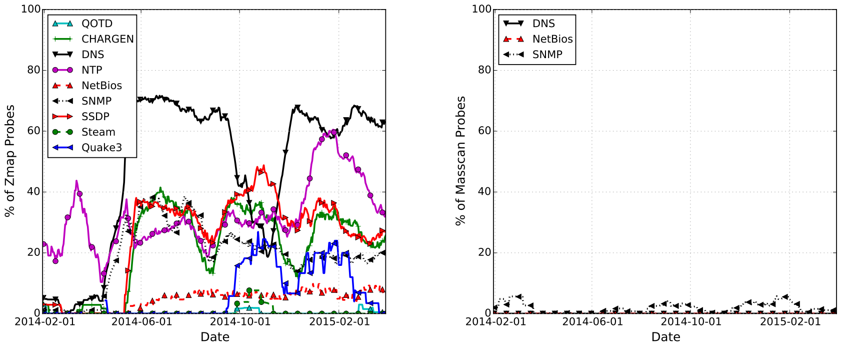


Fig. 1: Number of scanners per protocol and day.

3 Amplification Reconnaissance

Before analyzing the amplification attacks in more detail, we first want to understand how amplifiers are found. To launch effective amplification DDoS attacks, attackers have to actively search for amplifiers on the Internet. For many services, the easiest way to find amplifiers is an Internet-wide scan. Identifying scanners is also important in the later step of analyzing traffic at our honeypots (Section 4), to avoid falsely flagging scans as attacks. Therefore, in this section, we analyze scans performed for amplification reconnaissance. To grasp the trends and characteristics of the reconnaissance activities, we analyze a *darknet*, i.e., traffic observed at unused IPv4 addresses. By definition, a darknet has no hosts in its network, meaning that all traffic can be regarded as backscatter communication or scan traffic. In this paper, we analyze traffic of a `/16` darknet (i.e., 65,536 successive unused IPs) that is operated by NICTER [6].

Past 10 Years' Scans To grasp the overall trend of the reconnaissance, we investigated the hosts that scanned the DDoS-related protocols listed in [4] for the past 10 years. To this end, we first had to drop traffic that is not related to scans. In a best-effort approach, we only consider traffic from hosts that scanned at least 64 addresses of the darknet on the same port in a day. We defined these hosts as *scanners*.
Figure 1 shows the number of scanners from Sep. 2006 to Mar. 2015. The graph plots a 30-day moving average to smooth daily fluctuations. Before 2012, the number of scanners is small, with the notable exception of NetBIOS scans. In 2012, scanning for DNS became more popular, peaking at 55 hosts per day. In 2014, the number of scanners for all protocols increased dramatically, possibly an effect of the public release of amplification vulnerabilities in Feb. 2014 [4]. As we will show in the following paragraph, most of the new scanners come from security organizations (such as `ShadowServer.org`, Team Cymru, and Mauch's OpenNTPProject and the like). The popularity of NetBIOS constantly decreased although the negative trend similarly stopped in 2014. We speculate that most NetBIOS scanners are actually not related to amplification attacks, but are name lookups done by regular Windows-based systems that are



(a) Zmap

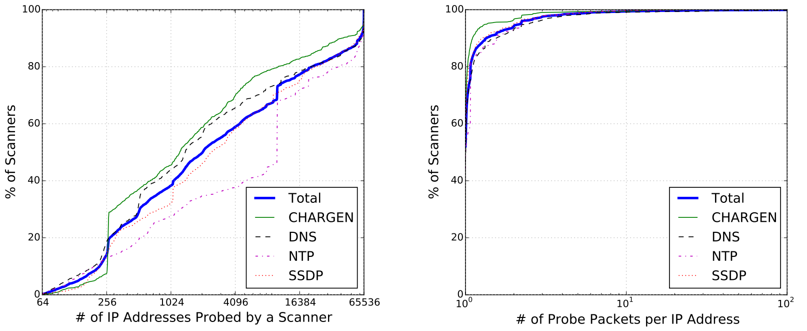
(b) Masscan

Fig. 2: Percentage of ZMap and Masscan probes

directly connected to the Internet (i.e., not NATed). The obvious spike of NTP scanners at the end of 2014 is caused by a heavily-distributed scan by a single security company.

Attribution of Scanners Next, we aim to measure a) which scanning tool was used and b) which organization performed the scans.
Both whitehats and adversaries can use off-the-shelf scanning tools, e.g., open-source scanners such as ZMap [5] and Masscan [7], to find amplifiers on the Internet. To measure the use of these scanners, we estimate the incoming packets generated by these scanners based on traffic fingerprints. In ZMap, the identification field of the IP header is hardcoded to 54321, which we use as ZMap's fingerprint. In Masscan, the ID in the IP header is derived by XORing the destination address, the destination port of the UDP header and the ID field of the application header (such as the DNS message ID). Figure 2 shows the percentage of probes identified by these fingerprints from Jan. 2014 to Mar. 2015. While Masscan is not frequently seen, Zmap's popularity increased since Apr. 2014 and holds a share of up to 60% of all scan probes.
Furthermore, we examined the scanning sources using Reverse DNS and WHOIS information. We found that about 70% of the scanning hosts using Zmap are hosted by universities and security organizations. We cannot determine the motivation and origin of the other scanning hosts, and found sources spread among many countries globally.

Scanners' Characteristics Next, we aim to understand the scanning behaviors in more detail. We conduct statistical analyses to analyze the reconnaissance activities, focusing on the top 4 services that are abused for amplification most frequently: CharGen, DNS, NTP and SSDP. Using the methodology defined above, we identified 5,269 scanners in the 27-month period from Jan. 2013 to Mar. 2015. We then analyzed the scanning activities in detail:



(a) Scan Coverage (b) Probe Packets per IP Address

Fig. 3: CDFs of scan coverage (left) and probe packets per IP address (right).

- **Scan Coverage**: We analyze how complete the scans are (i.e., the percentage of the darknet that is covered by a scanner). Figure 3a illustrates the CDF of the scan coverage per protocol. The coverage by the scanners, regardless of the protocol, is surprisingly low. About 64% of the scanners probed less than 10% of the darknet and only 10% of the scanners cover more than the 90% of the darknet. Therefore, we checked the low-coverage scanners and found that some scanners conducted distributed scanning. For instance, a security company conducted scans using about 240 hosts in the same /24 network. Each scanner scanned only for about 260 hosts of our darknet, but as a whole the scanners covered 97% of our darknet.
- **Scan Probes**: We count how many packets the scanners send per destination IP address. Figure 3b shows the CDF of the number of probes per IP address, scanner and day. About the 94% of the scanners send less than two packets per IP address on average. Surprisingly, 5.7% of the scanners send multiple (i.e., two or more than two) identical packets for the same service, presumably to mitigate packet loss. Identifying such scanners is also important to clean up our dataset of potentials attacks (cf. Section 4).
- **Scan Ports**: Finally, we analyzed how many services each scanner searches for. We found that 90% of the scanners search for a single protocol only (i.e., one port); just a few scanners send probes for multiple services. The most popular service was DNS (36%), followed by the equally-popular other three protocols (each 20%–22%).

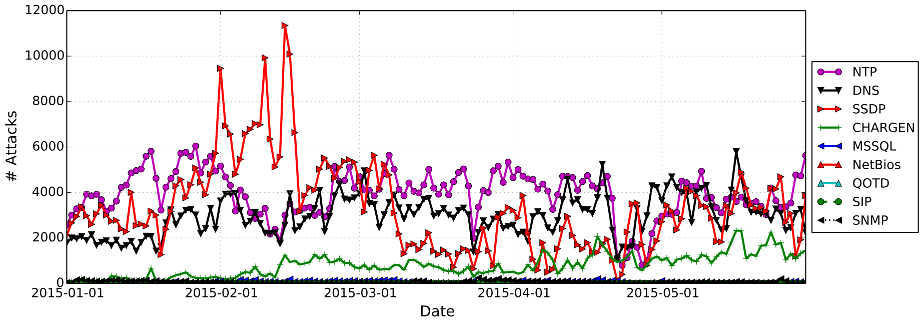


Fig. 4: Number of attacks per protocol and day.

4 Amplification Attacks

After shedding light on the reconnaissance part, we will now turn to the actual amplification attacks. We define an attack, then give an overview of attacks, and finally analyze the attacks in more detail.

Attack Definition When considering traffic at the honeypots, we have to separate actual attacks from random packets such as scans or backscatter before further analyses. To do so, we filter on those sources that sent at least 100 *consecutive* requests to our honeypots, whereas consecutive means that there was no gap of an hour or more between two packets. This conservative threshold discards most scanners, while it clearly also captures all powerful attacks. We chose this threshold given the lack of ground truth of labeled data on attacks/backscatter/scanners. We further discard all hosts that have been identified as scanners to obtain a dataset that consists purely of attacks.
We aggregate attacks based on the source IP address (i.e., the attack victim) and destination port (i.e., the protocol being abused). We group attacks seen by multiple honeypots into one combined attack, as long as the source IP address and the abused protocol match. If an attack pauses for an hour, and then resumes, we separate the traffic into two attacks.

Attack Overview Figure 4 summarizes the attacks our honeypots monitored over the period from Jan. 2015 to May 2015. In these five months, we monitored 1,535,322 amplification attacks. The graph shows that some protocols are clearly more popular than others. In fact, QOTD, MSSQL, NetBIOS and SNMP attacks sum up to less than 0.3% of all attacks. Most popular are NTP (37.0%), DNS (28.5%), SSDP (27.3%) and CharGen (7.0%), with a combined share of over 99%. The graph also shows that attacks are relatively constant over time.

Honeypot Convergence We next assess the completeness of our data by measuring whether the observed attacks converge. In other words,

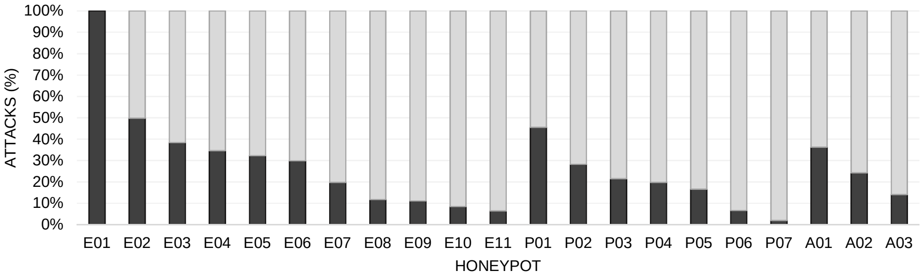


Fig. 5: Ratio of new (dark gray) vs. known (light gray) attacks per honeypot.

did we deploy sufficiently many honeypots to detect all attacks? For this, we measure how many previously-unknown attacks an additional honeypot observes. We focus only on those six protocols that all honeypots (with the exception of P02) support. Figure 5 shows the convergence graph, ordered by honeypot mode and honeypot name. The lower dark part of the bars indicates the percentage of attacks that a honeypot observed and were not yet been observed by the prior honeypots (i.e., the ratio of new attacks). For example, consider the eleven emulated honeypots. While the first honeypots (E01–E07) contributed many new attacks, the ratio of new attacks converges to small percentages at the later honeypots (E08–E11). This shows that—per mode—we had enough honeypots to cover most attacks out there.

Comparing the data across the honeypot modes reveals further interesting insights. First, the proxied honeypots (P01–P07) contributed many new attacks, showing that the protocol emulation was good, but not complete. Similarly, the agnostic honeypots discovered new attacks that the other honeypots had not seen, most of which abused protocols that neither of the other honeypots supported. For example, the agnostic honeypots attracted 9600 attacks abusing TeamSpeak servers, which offer about 5-fold amplification. In addition, we observed 9700 attacks abusing Quake game servers. We noted a few attacks against other protocols (including Sun RPC, ASF-RMCP, UT game servers, and more), but none of them was abused frequently.

Finally, we aim to answer the question of which honeypot mode was most effective. To this end, we drew convergence graphs with swapped orders of the honeypot modes (figures omitted for brevity), one with proxied honeypots first, and one with agnostic honeypots first. The share of attacks that are missed by the agnostic honeypots is significant, meaning that not all attackers blindly accept any large response. While this shows that agnostic honeypots alone are not sufficient for complete analysis, they are still helpful to capture new attacks—not only those abusing previously unseen protocols. We speculate that some attackers may favor the agnostic responses, as they are sometimes even larger than proxied or emulated responses. A good rule of thumb is to run agnostic honeypots in parallel to others. Similarly, the proxied honeypots missed attacks, particularly for unsupported protocols. But even for supported protocols, the proxied honeypots missed a significant proportion of attacks that the emulated honeypots did see—possibly as the number of proxied honeypots with static IP addresses was too low to converge towards a complete set of attacks. Summarizing, we cannot conclude that proxied honeypots are ultimately the best choice.

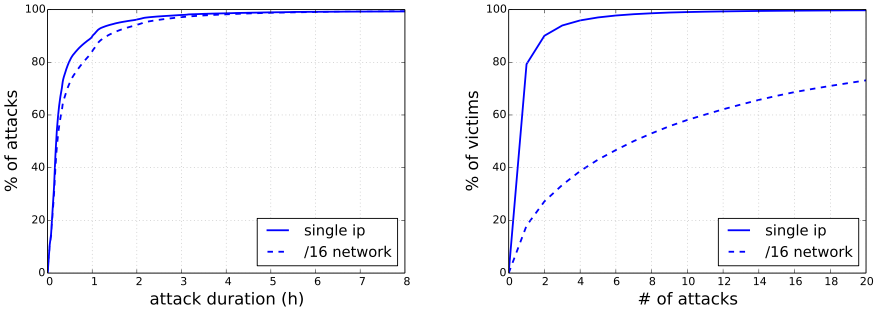
Deployment to Abuse Next, we analyze the time span between deploying a honeypot and the time it gets abused. In fact, all honeypots were already abused within 24 hours after deployment. However, the number of initial attacks was quite low, and we saw an increasing number of attacks as days passed after deployment. On average, the attacks observed at the honeypots reach a steady level after five days. While this may seem short, note that amplifiers in general are ephemeral in nature, and attackers constantly need to refresh their set of amplifiers. With the exception of NTP, Kührer et al. have shown that 42%–53% of the amplifiers vanish after one week due to IP address churn [8].

Attack Sources Due to IP address spoofing, it is not straightforward to attribute the attack traffic back to its true origin. Instead, the honeypots reveal the attack victim (i.e., based on the source IP address). However, we still aim to address an important question: Are amplification DDoS attacks caused by single sources (such as booter services), or do multiple hosts cause an attack (such as DDoS botnets)?

We aim to approach the analysis by leveraging the Time-To-Live (TTL) field in the IP header. Generally, the TTL field is decremented by every hop that forwards an IP packet. For the following analysis, we leverage the fact that our honeypots would observe varying TTL values for an attack if multiple attack sources are used. In contrast, if there is a single source, we would see a fixed (or at most a few) TTL values, assuming that the route from attacker to amplifier does not frequently change, and assuming that the initial TTL value is not randomized.

Therefore, we measured for which attacks the majority of honeypots saw at most two distinct TTL values. We use this small conservative threshold and a majority vote to counter potential route changes for individual $(attacker, honeypot)$ pairs. Using this method, we find that 96.3% of the attacks stem from a single source. This is an important observation, indicating that booter services cause more attacks than DDoS botnets. For the other 3.7%, we cannot tell with certainty if they stem from DDoS botnets. Unfortunately, an attacker may fool us by randomizing the initial TTL value. In other words, even if we see multiple TTL values, this could be caused by a single source. Still, our analysis gives a lower bound, showing that the vast majority of attacks are not distributed.

Attack Duration and Repetition Our honeypots also reveal how long a victim is being attacked. For these analyses, we were interested in the victim, rather than the protocol used to attack the victim. Therefore, we have grouped the attacks by source IP address, and regarded attacks abusing multiple protocols towards the same victim as a single combined



(a) Attack duration

(b) Number of attacks per victim

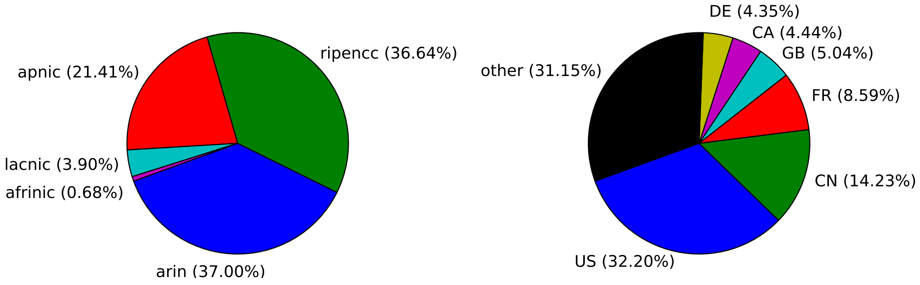
Fig. 6: CDFs of attack duration (left) and attacks per victim IP (right).

attack. Figure 6a shows the cumulative distribution of attack durations, i.e., the time between the first and the last packet monitored in an attack on a particular victim. Similar to the observations of DDoS botnets [9], amplification attacks also seem to be short-lived: 62% of the attacks are shorter than 15 minutes, and 90% of the attacks last at most one hour. Only 1.4% of the attacks last longer than 4 hours. This shows that attackers quickly move on to attack other victims. This is also in line with observations done on booter services [10], indicating that many clients run attacks in parallel. This is also confirmed by the high number of concurrent attacks: on average, we monitored 125.7 simultaneous attacks abusing our honeypots.

We further investigated *how often* a victim (i.e., an IP address) was attacked, as shown in Figure 6b. 79% of the victims were attacked only once; a further 11% were attacked twice. 0.81% of the victims were attacked more than 10 times. This may be counter-intuitive, especially as anecdotes claim that extortion is the main motivation for DDoS attacks. However, the vast majority of attacks are one-off operations, showing that in many cases the extortion—if any—is a non-persistent threat.

These observations may be biased due to our fine-grained definition of a victim, so we have repeated the measurements with a looser definition of an attack victim. Instead of measuring the attacks per IP address, we measured the attacks per victim network, aggregating per `/16` (i.e., class B) network. Figure 6 includes this comparison (dashed lines). Interestingly, while the number of attacks per network significantly increases, the attack duration does not. Following basic intuition, entire networks indeed attract more attacks than single IP addresses. However, the individual attacks are likely not linked to each other, as otherwise one would expect to see ongoing and consecutive attacks targeting the same network. Instead, the time span between two attacks (i.e., the time between two attacks during which there was no attack) is 9.6 days on average.

Victim Analysis In an attempt to understand the motivation of the attacks, we inspected the targets of the amplification attacks. To this



(a) Attacks by RIR (b) Attacks by countries

Fig. 7: Geolocation of victims and their share of the overall attacks.

end, we resolved GeoIP data for all attacked IP addresses, queried their reverse DNS record, and mapped the IP addresses to autonomous systems (ASes). Figures 7b and 7a show the distribution of countries and Regional Internet Registry (RIR) of the victims, respectively. Victims that belong to ARIN and RIPE each attract 37% of the attacks and APNIC attracts another 21%. Providers in Central- and South America (LACNIC) or Africa (AfriNIC) face relatively few attacks.

When looking at countries, the US stands out, hosting one-third of the victims. There are also many victims in China (14%) and France (8.6%), whereas all other victims form a long-tail distribution of 192 affected countries. Of those, 28 countries faced over 1000 attacks, showing the wide geographical distribution and global threat of amplification attacks.

In addition, the destination port may reveal what type of service is attacked. 35% of the attacks target UDP port 80, possibly to pass misconfigured firewalls that allow port 80 in general (i.e., not only TCP). However, in a few cases, the attacks are actually directed at UDP-based services. In descending order, 2.6% of the attacks target Xbox Live, 2.0% DNS servers, 0.9% Minecraft game servers and 0.6% each Steam game servers and TeamSpeak VoIP servers. The majority of attacks is scattered in a long-tail distribution over other ports. The less popular services include MSSQL servers, NTP servers, MMORPG servers, and further VoIP systems. All remaining requests seem to randomize the source port.

Request Entropy In an attempt to understand the attack techniques, we next inspected how much variety we see in the request payloads. That is, we measure how many different request payloads (i.e., *excluding* UDP and IP headers) we observe per protocol. The less adversaries vary their requests, the easier it will be to filter their requests (see Section 5).

Figure 8 shows a CDF of the request variety among all honeypots and attacks. The request variety is quite low for most protocols. For CharGen and QOTD, 99.66% and 99.34% of the requests are one byte long, with a low variety in that byte. But requests for protocols with more complex request structures and types (such as NTP) also did not show high variety and typically only varied in their length (not in type or content). In fact,

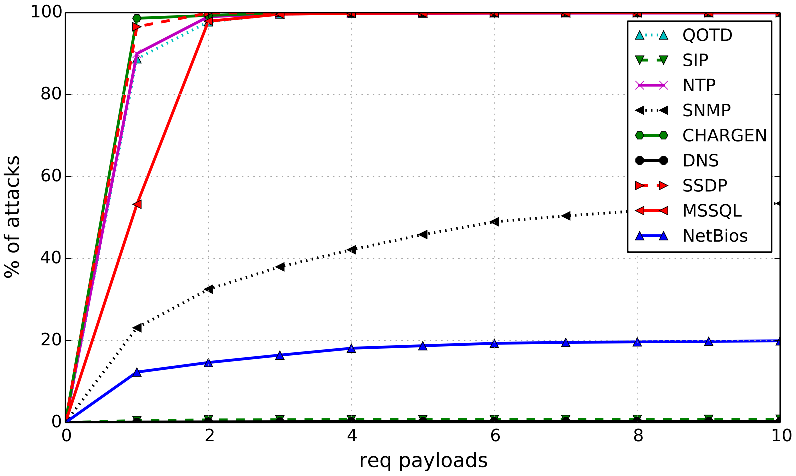


Fig. 8: CDF of the number of UDP payloads over all attacks.

for five of the protocols (CharGen, QOTD, NTP, MSSQL and SSDP), the two most popular request payloads per protocol caused more than 98% of the attacks.

DNS is at the other end of the scale, attributed to the fact that adversaries a) change the DNS headers (such as the DNS message ID), and b) change the domain name being queried. Similarly, the SNMP requests varied, as the attackers a) varied the Object Identifiers (OIDs) and transaction IDs, and b) also varied the request type (`getBulkRequest` and `GetRequest` being most prominent). But NetBIOS (randomized 2-byte-wide transaction ID) and SIP (random session ID) requests have also shown a higher variety. In any of these protocols, detecting the traffic towards the amplifiers is not as trivial as incorporating the most popular request payloads into network-based filters. However, one can still define payload signatures over static, non-randomized packet headers.

Finally, we also measure the request entropy within individual attacks. Following the observation from above, most attacks based on protocols with low request variety were caused by only a single request payload. However, even for DNS, which offers a high request variety, most attacks use very few different payloads. For example, 45.3% of the DNS-based attacks used a single request payload, and more than 80% of the DNS attacks had at most 3 request payloads. Interestingly, though, NetBIOS, SNMP and SIP still have a high request entropy (less than 30% of the attacks have only a single payload). This shows that one cannot conclude that the request payloads are static within individual attacks.

5 Honeypot-Assisted Defenses

Honeypots are powerful as early-warning systems. In this section, we describe how we can leverage our honeypots to create valuable inputs for both proactive and reactive DDoS defenses.

5.1 Real-time Attack Monitoring

We have published a live feed of attacks based on the data obtained by our honeypots. We use a web portal[1] to share information about incidents (such as attack start and end times) with registered service providers and trusted individuals. Providers can use the attack information to inform their customers or to filter attack traffic based on IP and port information. We chose to require registration only to prevent attackers misusing the data (e.g., to fingerprint or to evade our honeypots).

To test the usefulness of the honeypot data to detect DDoS attacks, we have cooperated with a large Japanese ISP. We compared the honeypot-based DDoS detection with a flow-based detection system that ISP had already deployed [11]. To this end, our honeypots generated an alert as soon as a potential victim (i.e., a single IP address) sent more than ten packets in a ten-second interval (i.e., more than 1 pps on average). In contrast, the ISP's detection mechanism raises an alert based on a threshold of packets per destination for DDoS-related ports.

We compared both systems in the time span from August 1st to November 30th, 2014. Our honeypots detected 75 potential attacks towards the ISP's networks. Of these, the ISP detected 56 alerts. 43 of the 56 alerts (77%) were detected first by our honeypot, and then by the ISP, with an average delay of 39 seconds. The honeypots detected 13 attacks later than the ISP, indicating that in these the attackers rotated the set of amplifiers. We hypothesize that deploying more honeypots would further improve the reaction times.

5.2 DNS Abuse Domain List

DNS stands out when it comes to amplification attacks. Most other protocols can be filtered, as they have little benign use on the Internet (e.g., CharGen, QOTD, or even typical LAN protocols such as NetBIOS and SSDP). In addition, unlike for management protocols like SNMP, DNS communication may involve different and distant endpoints (e.g., authoritative name servers). Finally, bad DNS filters would cause malfunctions for Internet users.

In any case, as we have seen in Section 4, DNS remains one of the typical attack protocols. Thus, to support better filtering, we derive a list of domains that have been abused for amplification attacks. Table 2 lists the 10 most popular attack domains since Feb. 2015, in descending order of the number of requests recorded at the honeypots. In the following, we will draw four observations from the table.

[1] Note that we intentionally do not publish the address of the web portal, as the portal contains potentially sensitive information. If you are interested in obtaining access, please request an account from Christian Rossow via email.

FQDN	**Type**	**First Seen**	**Last Seen**	**Days**	**Victims**
067.cz	ANY	2015-02-21	2015-05-31	47	15804
mg1.pw	ANY	2015-04-12	2015-05-31	49	71357
isc.org	ANY	2015-02-01	2015-05-31	106	12228
psg.com	ANY	2015-04-05	2015-05-28	51	25986
vizit.spb.ru	ANY	2015-03-23	2015-05-31	45	99543
mg1.pw	A	2015-04-12	2015-05-31	36	1606
pidarastik.ru	ANY	2015-02-11	2015-05-31	80	14431
dhs.gov	ANY	2015-02-24	2015-05-03	64	41857
r3a.es	ANY	2015-03-25	2015-04-16	23	23943
ironmen-style.ru	ANY	2015-04-22	2015-05-29	15	120595

Table 2: DNS domains, ordered by the number of requests seen at the honeypot.

First, in some cases it is not possible to use this list as a blacklist to filter bad traffic, as benign domains are also in the dataset (e.g., `isc.org` or `dhs.gov`). This is largely due to domains that deploy `RRSIG` or `DNSKEY` resource records, which are required for DNSSEC. For example, an `ANY` request for `isc.org` results in an approximately 1500-byte response, containing two `DNSKEY` and four `RRSIG` records—both of which are required for DNSSEC. The problem of potential false positives may be resolved by combining the request type (e.g., `ANY`) with the requested domain. However, without evaluating this further, our suggestion is not to blindly use the abuse list as input for filters.

Second, attackers register domains only for the purpose of DDoS attacks. For example, according to web archives, `mg1.pw` never hosted real content. Furthermore, Passive DNS analysis (using `dnsdb.info`) shows that the domain was first ever used only 30 hours before we noticed its abuse.

Third, each domain keeps being abused for a long time: The average time for abuse in the top 10 domains is 8 weeks (i.e., the time span from the first to the last attack). Even when considering all attack domains, the time span is still 7.5 weeks. We presume that attackers keep abusing the same domain to limit the overhead for registering new domains and setting up authoritative name servers. Defenders can use these insights on domains that are popular in amplification attacks, for example, to aid existing detection mechanisms (see Section 5). Similarly, collecting evidence on attacks via honeypots can help law enforcement to take down purely malicious domains.

	IptabLes	**XorDoS**	**BackdoorA/M**	**BillGates**
DNS Amplification	✓	✓	✓	✓
• fixed query type	`A`	`A`	`ANY`	
• EDNS payload size		4096	4096	8192
• DNSSEC OK				✓
• random domain	✓			✓
SYN	✓	✓	✓	✓
SYN+ACK		✓		
SYN with payload			✓	
ICMP PING				✓
Auth NS			✓	✓
Generic TCP				✓
Generic UDP				✓
HTTP GET			✓	✓
TCP connections			✓	✓

Table 3: Attack capabilites

6 DDoS Bot Analysis

We now turn our analysis to explore a potential source of amplification attacks: DDoS botnets. Botnets span multiple hosts that are instructed by the botmaster and have already been known to launch DDoS attacks in general [9, 12]. The TTL analyses in Section 4 have already indicated that the majority of attacks stem from a single source. In this section, we seek to test this hypothesis by analyzing DDoS botnets in more detail. Recently, adversaries have started to compromise insecure or vulnerable embedded devices, leading to a sudden increase in Linux-based DDoS bots. Embedded devices that are directly connected to the Internet (e.g., home routers) are an attractive platform to launch amplification attacks, as they are not filtered by firewalls or NAT gateways. Therefore, we will study these botnets in particular detail.

Analysis Methodology: We analyzed a set of Linux-based bots that are known to provide DDoS services. We obtained these binaries using Telnet- and SSH-based honeypots and via keyword searches on VirusTotal. Our dataset of Linux-based DDoS bots consists of 247 binaries that we collected between Jan. 2014 and May 2015. Our particular interest is to understand the DDoS attack functionality of the bots. To this end, we dynamically analyzed the samples in a sandbox and traced the C&C communication. Furthermore, using static analysis, we classified our samples into families, assigning names to the families based on characteristic strings.

Analyzed Families: Our analysis has revealed four popular Linux-enabled and DDoS-capable bot families. While Windows-based DDoS

bots are well-explored, to the best of our knowledge, we are the first to inspect Linux-based malware of this type. An overview of the attack capabilities of each family is given in Table 3. The two families *IptabLes* and *XorDoS* have only limited attack capabilites, whereas *BackdoorA/M* and *BillGates* offer a wide range of different attacks.
All four families abuse DNS for amplification attacks. Interestingly, *IptabLes* and *XorDoS* support only `A` lookups, which were less prominent in the attack domains (cf. Section 5). Three out of four families use EDNS to expand the maximum UDP-based response size to at least 4096 bytes, and *BillGates* bots specifically also ask for DNSSEC records. None of the families supports any amplification protocols other than DNS. However, *BillGates* also features generic UDP and TCP attacks, where headers, flags and the payload are taken as input via a C&C command. *BillGates* could thus be used for amplification attacks with any protocol.
For completeness, note that all bots also support non-amplification attacks. These span ICMP and TCP SYN floods, HTTP GET floods or TCP connection exhaustion. Two bots also support DNS-based "random domain" attacks, in which the bots randomize the FQDN of the lookup request. These requests are not abusing amplification, but presumably aim to flood authoritative name servers instead.

Attack Fingerprints: Seeing the potential for amplification attacks, we wondered how large the impact of these botnets is in the attacks we are observing with AmpPot. We derived attack fingerprints for the botnets by identifying artifacts in their attack traffic. Much to our surprise, this was possible for all families, as the malware authors re-used randomly generated values for various header fields. *IptabLes* sets the UDP source port to a value that is derived from the IP packet ID. *BillGates* uses the same value for DNS message ID and IP packet ID, and randomizes the TTL to five initial values. Similarly, *XorDos* equalizes both DNS message ID and IP packet ID, and also derives the source port from these values. Finally, *BackdoorA/M* uses a specific source port range and randomly draws an initial TTL value from four distinct groups.
We then searched for DNS-based amplification attacks that satisfy these filters at our honeypots. The above-mentioned filters were simple enough so that we could use SQL queries to search for matching packets in our attack database. For each attack, we computed the ratio of the number of packets that matched the filters compared to all packets belonging to this attack. While we found individual packets to match, presumably caused by accidental value pairs that just happen to match our filters, the ratio of "attributed" packets per attack never exceeded 1%. This indicates that these DDoS bots are not frequently used in amplification attacks, although they remain a lingering threat.

7 Discussion

This section raises a few aspects left over for discussions about the implementation and deployment of AmpPot.

Ethics: With AmpPot, we provide valuable insights into amplification attacks that otherwise could not be observed on such large scale. Unfortunately, we face a dilemma, as these insights can only be revealed if AmpPot participates in the attacks to some extent. To minimize the harm by AmpPot, we have included a rate limiting mechanism. Still, this leaves a small number of attack packets. Content-based classifiers to distinguish between scan and attack traffic are unfeasible, as attackers typically use the same kind of requests for both activities.
Seeing this risk, we considered to clearly mark AmpPot's responses as such, e.g., by embedding an info text explaining the traffic. However, first, this would enable attackers to trivially detect AmpPot deployments. Second, attack victims usually do not inspect the payload of each and every attack packet, but rely on flow-based information instead. Looking at the flows, however, would hide any note that we add to the responses. Summarizing, we concluded that an effective rate-limiting module is the most reliable and practical option to prevent abuse of the honeypots. Each AmpPot deployment can configure the rate-limiting thresholds on their own, possibly resorting to an overly conservative threshold (e.g., only a single request is answered per IP address and hour).

Rate Limiting: In our experiments, we chose an arbitrary rate-limiting threshold that seemed reasonable to us. However, choosing the threshold may have consequences on the number of scanners that discover AmpPot. In future work, we plan to evaluate varying rate-limiting thresholds and their effects on the attacks that are observed subsequently.
Furthermore, our current rate-limiting implementation treats all protocols equal. This may be unsuitable for protocols that have comparatively chatty responses, such as the `monlist` reply in the NTP protocol implementation, which consists of dozens of response packets. An alternative might be to include dynamic thresholds for rate-limiting, which vary depending on the response size and aggressiveness of the requests.

Honeypot Detection: Although we have not witnessed concrete attempts of doing so, an attacker can identify AmpPot instances to exclude them from any attacks she launches. AmpPot offers services on many UDP ports, and as such can be identified relatively easily. However, detection becomes more tricky if the honeypot is configured to listen on a single UDP port only. Still, an attacker may inspect artifacts, such as the response payloads, or observations of dropped requests due to rate-limiting. We leave it open to future work to explore how we could increase the stealthiness of AmpPot.

8 Related Work

This section summarizes related work, which we group by topic.

DDoS Works on reflective DDoS attacks date back to early observations by Paxson in 2001 [3]. But while theoretically known, amplification attacks have not played a big role until recently. Instead, research analyzing the DDoS threats focused on analyzing DDoS attacks in general. Büscher and Holz monitored the C&C servers of DDoS botnets to analyze the attacks and their targets and documented TCP- and HTTP-based attacks [12]. Similarly, Welzel et al. tracked commands of two DDoS botnet families and monitored whether victims of DDoS botnets were actually affected by the attacks [9]. Thus, DDoS botnets are a well-explored area, although none of the existing analyses inspected Linux-based bots.

With the recent increase of amplification attacks, which we believe is an orthogonal problem to DDoS botnets, researchers started to explore the new threat. Rossow provided an overview of 14 protocols that are vulnerable to amplification attacks [4]. As a follow-up, Kührer et al. have shed light on the amplifiers landscape, revealing their fingerprints and observing their lifetime [8]. These works inspect concrete amplification vulnerabilities in protocols, propose defense mechanisms and survey amplifiers, while giving only anecdotal evidence on actual attacks.

Others devoted their research to particular amplification protocols. Czyz et al. explored NTP in great detail, exploring all amplification vulnerabilities and inspecting attack victims based on artifacts in the NTP `monlist` feature [13]. Van Rijswijk-Deij et al. analyzed how DNS (and in particular DNSSEC) can be abused for amplification attacks [14], observations many of which confirm the trend of DNS attacks we observe. Our work adds to this in that we give insights on how the protocols actually are abused in DDoS attacks.

Closest to our work, researchers inspected *booters*, which are services that offer DDoS attacks on a pay-per-use basis. Karami and McCoy were the first to monitor such booter services, studying the adversarial DDoS-As-a-Service concept [15]. They observed booters launching amplification attacks, however, but did not reveal more details. Similarly, Santanna et al. analyze the databases and payment of 15 booters [10]. In contrast to using honeypots, analyzing booter services is a forensic challenge, and requires gaining access to the booter systems (or obtaining an image thereof). Our dataset is more complete in that we monitor attacks regardless of the specifics of particular booter services. Thus, the scale of our recorded dataset exceeds other observations by orders of magnitude.

Honeypots Honeypots have been used in many other contexts [16], such as for collecting malware [17], creating automated network signatures [18], or finding malicious websites [19]. To the best of our knowledge, AmpPot is the first honeypot to track amplification DDoS attacks. The idea of using vulnerable services to observe DDoS attacks was already known [4], whereas—due to the short deployment time of these "baits"—the dataset under analysis spanned only eight attacks. We revise this result by longitudinal, broad deployment of 21 honeypots, revealing that honeypots actually *are* useful to gain attack intelligence.

Scan Analysis Durumeric et al. have also analyzed a darknet to explore scanning behaviors [20]. They inspected scans for NTP, and we extend their analyses by considering all amplification-related ports.

9 Conclusion

Amplification attacks continue to be a dangerous threat to millions of users. We have shown that one can passively monitor attacks, and insights into these attacks can help to derive helpful countermeasures. However, our research has also identified new research directions, such as trying to understand the attackers' motives and actual origins. We have shown that DDoS botnets are likely not the main source for amplification attacks, shifting focus to other potential attack sources such as booter services. AmpPot assists in analyzing the amplification threats in more detail, and the web portal can help operational security operators to become informed about or to defend against attacks.

References

1. Mirkovic, J., Reiher, P.: A Taxonomy of DDoS Attack and DDoS Defense Mechanisms. In: ACM SIGCOMM Computer Communication Review. Volume 34. (2004) 39–53
2. Specht, S.M., Lee, R.B.: Distributed Denial of Service: Taxonomies of Attacks, Tools, and Countermeasures. In: Proceedings of the International Conference on Parallel and Distributed Computing (and Communications) Systems (ISCA PDCS), San Francisco, CA (2004)
3. Paxson, V.: An Analysis of Using Reflectors for Distributed Denial-of-Service Attacks. In: Computer Communication Review. (2001)
4. Rossow, C.: Amplification Hell: Revisiting Network Protocols for DDoS Abuse. In: Proceedings of the 2014 Network and Distributed System Security (NDSS) Symposium. (2014)
5. Durumeric, Z., Wustrow, E., Halderman, J.A.: ZMap: Fast Internet-wide Scanning and Its Security Applications. In: Proceedings of the 22nd USENIX Security Symposium, Washington, D.C., USA (2013)
6. NICTER. (`http://www.nicter.jp/`)
7. Graham, R.D.: MASSCAN: Mass IP port scanner. `https://github.com/robertdavidgraham/masscan` (2014)
8. Kührer, M., Hupperich, T., Rossow, C., Holz, T.: Exit from Hell? Reducing the Impact of Amplification DDoS Attacks. In: Proceedings of the 23rd USENIX Security Symposium. (2014)
9. Welzel, A., Rossow, C., Bos, H.: On Measuring the Impact of DDoS Botnets. In: Proceedings of the 7th European Workshop on Systems Security (EuroSec). (2014)
10. Santanna, J., Durban, R., Sperotto, A., Pras, A.: Inside Booters: An Analysis on Operational Databases. In: 14th IFIP/IEEE International Symposium on Integrated Network Management (IM). (2015)

11. Urakawa, J., Sawaya, Y., Yamada, A., Kubota, A., Makita, D., Yoshioka, K., Matsumoto, T.: An Early Scale Estimation of DRDoS Attack Monitoring Honeypot Traffic. In: Proceedings of the 32nd Symposium on Cryptography and Information Security. (2015)
12. Büscher, A., Holz, T.: Tracking DDoS Attacks: Insights into the Business of Disrupting the Web. In: Proceedings of the 5th USENIX LEET, San Jose, CA, USA (2012)
13. Czyz, J., Kallitsis, M., Gharaibeh, M., Papadopoulos, C., Bailey, M., Karir, M.: Taming the 800 Pound Gorilla: The Rise and Decline of NTP DDoS Attacks. In: Proceedings of the 2014 Conference on Internet Measurement Conference, ACM (2014) 435–448
14. van Rijswijk-Deij, R., Sperotto, A., Pras, A.: DNSSEC and its potential for DDoS attacks - a comprehensive measurement study. In: Proceedings of the Internet Measurement Conference 2014, Vancouver, BC, Canada, ACM Press (2014)
15. Karami, M., McCoy, D.: Understanding the Emerging Threat of DDoS-as-a-Service. In: Presented as part of the 6th USENIX Workshop on Large-Scale Exploits and Emergent Threats. (2013)
16. Provos, N., Holz, T.: Virtual Honeypots: From Botnet Tracking to Intrusion Detection. Pearson Education (2007)
17. Baecher, P., Koetter, M., Holz, T., Dornseif, M., Freiling, F.: The Nepenthes Platform: An Efficient Approach to Collect Malware. In: Recent Advances in Intrusion Detection, Springer (2006) 165–184
18. Kreibich, C., Crowcroft, J.: Honeycomb: Creating Intrusion Detection Signatures Using Honeypots. ACM SIGCOMM Computer Communication Review **34** (2004) 51–56
19. Nazario, J.: PhoneyC: A Virtual Client Honeypot. In: Proceedings of USENIX Workshop on Large-scale Exploits and Emergent Threats (LEET). (2009)
20. Durumeric, Z., Bailey, M., Halderman, J.A.: An Internet-Wide View of Internet-Wide Scanning. In: USENIX Security Symposium. (2014)