How Media Reports Trigger Copycats: An Analysis of the Brewing of the Largest Packet Storm to Date

Vincent Ghiëtte
TU Delft, Cyber Security Group
Delft, Netherlands
v.d.h.ghiette@tudelft.nl

Christian Doerr
TU Delft, Cyber Security Group
Delft, Netherlands
c.doerr@tudelft.nl

ABSTRACT

In late February 2018, news spread through the mainstream media about a massive distributed denial-of-service attack on the popular software collaboration website github.com. Estimated at a rate of 1.3 Terrabit per second, this massive packet flood was the largest DDoS attack by volume to date, surpassing previous records set by the first IoT-based DDoS attacks in 2017.

In this paper, we analyze the behavior of the actors scanning and probing the Internet for presence of exploitable memcached servers that were the root cause of this attack, both before and after the media coverage. We find that the attacks of late February were preceeded by a large scale reconnaissance action a month before, and that the attacks were the result of an extended evolution of methods to find a suitable attack strategy. Furthermore, we see that the coverage about the massive DDoS attack actually triggered another wave of DDoS attacks, resulting in the large influx of new, previously unseen users who seem to be leveraging ready-made tools.

CCS CONCEPTS

• **Networks** → **Denial-of-service attacks**;

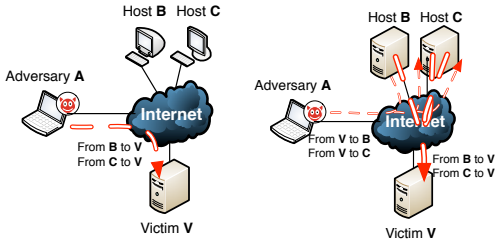
KEYWORDS

denial-of-service attacks, memcached, threat intelligence

ACM Reference Format:
Vincent Ghiëtte and Christian Doerr. 2018. How Media Reports Trigger Copycats: An Analysis of the Brewing of the Largest Packet Storm to Date. In *WTMC '18: Workshop on Traffic Measurements for Cybersecurity, August 20, 2018, Budapest, Hungary*. ACM, New York, NY, USA, Article 4, 6 pages. https://doi.org/10.1145/3229598.3229606

1 INTRODUCTION

The past twelve months changed the perspective on denial-of-service attacks. Over years, until 2017, the magnitude and method of large scale distributed DoS attacks remained relatively static, with noteable large incidents typically reaching a maximum volume of 300 - 400 Gigabit per seconds, abusing well-known vectors such as DNS or NTP amplification. The proliferation of Internet-of-Things (IoT) devices, which frequently contain improperly secured remote access and administration capabilities, began to change these established practices. The first widely distributed IoT-based attack in August 2017 overshadowed past records by about a factor of two, only to be eclipsed by another IoT-based DDoS a few weeks later allegedly breaking the Terrabit/second mark. In contrast to previous DoS campaigns, these IoT-based attacks leveraged the power of millions of small connected devices that together could generate a large packet flood. In late February 2018, this record was again eclipsed, when various DDoS mitigation providers reported attacks that began to abuse the memcached service to reach combined packet floods of initially 1.3 Tbps, and later 1.7 Tbps.



(a) Resource Depletion (b) Amplification

Figure 1: DoS attacks may be classified in resource depletion and volumetric attacks. For economics, these packet floods are typically realized using amplification attack vectors.

Denial-of-service attacks may be classified in two types as shown in figure 1. In resource depletion attacks, adversaries overwhelm a victim by initiating many connection attempts that will dry up a limited but critical resource for service delivery. This ranges from buffer space in the operating system network stack for IP packet reassembly to a maximum number of available connections, which is exploited in the most common type of resource depletion attacks on the Internet, TCP SYN floods. In the second type, volumetric attacks, the adversary aims to generate a very large traffic volume to saturate the victim's uplink. As this would otherwise come at a great expenditure for the adversary, this is normally done through exploitation of a third-party service, from where he can request some data. By spoofing the source IP address of the request to that of the victim, the response is then delivered to the victim. When exploiting Internet services with a high difference between request and response size, these so-called amplification attacks may be generated using comparatively small effort. As the memcached service is meant for caching media files, it features a very high

WTMC '18, August 20, 2018, Budapest, Hungary

ACM ISBN 978-1-4503-5910-8/18/08…$15.00
https://doi.org/10.1145/3229598.3229606

amplification ratio and thus lets adversaries initiate a very effective DDoS using comparatively low effort.

In order to prepare and counteract coming attacks, it is necessary to better understand the ecosystem of actors behind DDoS attacks, assess their capabilities and trace their techniques and tactics. This so-called threat intelligence can hence be used to plan and place DDoS migitation techniques, design more effective countermeasures and predict future adversarial capabilities and techniques. In this context, this paper makes three contributions:

- We find that the memcached amplification DDoS attacks are preceeded by a systematic search for vulnerable servers weeks before the attacks, at least for several actors.
- We show that there are subtle differences in the attack packets in terms of packet header and payload, which can be used to fingerprint actors and trace their evolution over time.
- We demonstrate that the bulk of the hosts engaged in testing for memcached services can be attributed to copycat behavior. Entering the scene after the media coverage of the attacks, they utilize a proof-of-concept implementation to realize their objective.

The remainder of this paper is structured as follows: Section 2 provides an overview of previous studies that investigated past DDoS incidents. Section 3 describes the purpose and setup of the memcached system, which was abused in this specific incident. Section 4 briefly outlines the data set and data acquisition for our investigation. Section 5 presents our findings on the dynamics of memcached probing activities. Section 6 summarizes our work.

2 RELATED WORK

As we have discussed in the introduction, the current memcached attacks are actually the result of a long evolution of DDoS activities. The idea of creating packet floods through amplifying reflectors has actually been known for a long time, and was originally described by Paxson [5] based on the DNS and Gnutella services. While there exist a number of countermeasures such as deactivation of UDP for high amplification services or network wide enforcement of BCP38, misaligned economic incentives and speed requirements have deterred these initiatives in practice.

In 2014 Rossow published [7] a detailed work presenting different DDoS amplification attack vectors. He explains which UDP protocols can be used to perform an attack, as well as the expected amplification factor. At that time, the highest measured amplification was based on the network time protocol (NTP) at an amplification factor of 4670. Like our study, Rossow also leverages two darknets of size /17 and /23, to monitor the scanning activity for vulnerable protocols in the wild. While he identified highly lucrative services for abuse, he did not come across evidence that all of them are actually being tested for, especially the NTP protocol which has the highest amplification factor. Thomas et al. ran a comparable study [10] and tracked scanning activity for various candidate services on a /28 subnet for a duration of 3 years after the publication of [7]. They do see an uptake of the services Rossow described as attractive targets, and further note that specific services are not continuously targeted but appear in traffic surges.

Until now, very little information is available how actors engaged in DDoS attacks find exploitable targets and use them in an attack. One noteable study is the work of Santanna et al. [8], who have analyzed the ecosystem of booter services. They find that there is cut throat competition between different services, but when they ordered attacks on themselves for investigative purposes, most booters did not exceed an overall volume of 7 Gbps. Santanna et al. report that actors running a booter service need to be highly competitive, and thus would need to take any advantage over other services if they can perform more powerful attacks with less infrastructure. Thus, if a DDoS service can obtain more amplification it would mean that it could make bigger attacks happen or use less resources to perform more "small" attacks.

Recently Krupp et al. [2] presented a honeypot system for amplification attacks. Their amppot service offers protocols which are prone to DDoS amplification attacks, and tracks which actors exploit certain services to get a better understanding of the DDoS dynamic, allowing them to attribute the attacks to booter services.

While the memcached system is neither new nor free of known exploiteable vulnerabilities [4, 9], this service has not been picked up until now by actors for distributed denial-of-service attacks at production scale until the events in Feburary 2018. The following is the first study investigating the behavior of the actors exploiting this new service for amplification attacks.

3 THE MEMCACHED SYSTEM

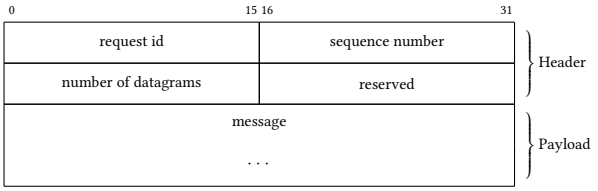
The goal of the memcached software [1] is to speed up data requests a client would normally make to a database. Instead of for example performing a lookup for an image embedded on a web page at an SQL server which would then be fetched from comparatively slow disk, the client could instead request the file from a memcached server which would serve it directly from fast system memory, and only if the file is not available fall back to the slower primary data source. If the available cache at a server runs out, the least recently used item is displaced from memory.

Clients find and request a resource from the memcached server based on a hashed identifier. For better scalability, multiple memcached servers can each be assigned a part of the cached data, but these servers do not communicate with each other, for example to distribute and reallocate shards. Memcached essentially provides an opportunitistic (distributed) key-value store. For security reasons, these setups should be placed within a separated, trusted network, as clients can interact with the caches and obtain data in an unauthorized way, a threat vector which was outlined by [9].

The memcached server fulfills all requirements to be an interesting target for amplification abuse, as it is meant to deliver media files at fast pace to speed up Internet sites. Common to many protocols designed for fast turnaround, requests to a memcached server can be issued by UDP[1] in addition to TCP, which reduces latency through omission of the transport layer handshake and was suggested by the developers as a preferred mode according to the protocol manual [6] as the number of TCP connections can cause scaling difficulties in large installations. This however allows the service to be misused by an adversary by spoofing the requesting IP address to that of the victim. An abuse of memcached is furthermore very attractive, as the service can feature a high amplification ratio, potentially delivering large files in response to a small query.

[1] Until version 1.5.6, released in response to the incident described in this paper.

While the UDP protocol of memcached is similar to the TCP protocol (the main difference is a simplified header in UDP), we will in the following only briefly outline the UDP version as it is the one most easily abused. Knowledge of the format of memcached requests is important to understand the fingerprinting techniques we will introduce in the section 5. The figure below displays the general format of the memcached protocol message [6]:



The header of the protocol consists of four 16-bit words. The request id is an identifier the client uses to match query and responses, which is necessary due to the connection-less nature of UDP. The specification states that the request id should be randomly selected by the client at the beginning of a session and then increased by one for each request. An initial value randomization is however not obliged by the protocol so the client is free to always start at 0. The sequence number is to enable larger responses that exceed the length of a single UDP packet. Stating a number between 0 and $2^{16} - 1$, it marks the position of the current UDP in the response or query. The third word lists the overall number of datagrams in the request or response. The reserved field should always be 0.

The header is followed by a payload field, which contains either (part of) the requested resource or a control command sent to the server. The command `status` will return the status of the server, while `version` fetches the memcached server version. The request `get a b c d e f` will retrieve files associated with any of the keys *a*, *b*, *c*, *d*, *e* or *f*. The end of a response or query should always end in a `\r\n`, it is the symbol used to indicate the end of the message.

4 DATA COLLECTION

Data source for this study is a network telescope of three partially populated /16 networks. All data directed at the unused IP addresses from these ranges is logged, which provides an insight into port scanning, probing of available systems, as well as information about current attacks on the Internet from backscatter. As the monitored ranges contain no clients, the log files are entirely void of any user traffic. This means that this monitoring method creates a very clean data set of only adversarial traffic, and simultaneously eliminates possible issues about the privacy of users.

The dataset used for this study spans from 14th of August 2017 until 1st of April 2018. Over this time frame, we have recorded 31,278,705 packets directed to memcached ports at 130,000 monitored IP addresses, which originated from a total of 38,383 IP addresses. In order to track the availability of memcached services and the response of the ecosystem to the attacks, we relied on two UDP 11211 port sweeps by Rapid7 made available through the scans.io platform. As the scans.io data did not extend beyond the initial attacks, we performed one additional port sweep, sending one packet querying the version number to potential memcached servers to verify whether they are still present or a new instance has spun up. For this procedure we tested the IP addresses in random order to minimize the impact on an AS at any given time. While a single packet is unlikely to cause issues and this was a one-time measurement, an opt-out protocol was in place for network owners to deregister from measurements in the future.

5 DYNAMICS OF MEMCACHED ACTORS

After the discussion of the memcached protocol and our data collection procedures, this section will describe the behavior of actors searching for and trying to exploit memcached servers for denial-of-service attacks. Being a spoofed amplification vector, the aggregate traffic of the denial-of-service attacks themselves would be visible only at the victims' sites or at a DDoS mitigation provider. [3] provides a brief write-up of the specifics of the attacks.

In order to execute the DDoS, the perpetrators would however first need to know a large number of memcached servers to exploit. Adversaries would typically accomplish this by scanning the Internet for machines responding to memcached's well-known service port TCP/UDP 11211, and in the following we will report about these reconnaissance activities and exploitation attempts across some 130,000 addresses we continuously monitor for this and other attack traffic. As an adversary is trawling through the Internet to find vulnerable services, we can assume the source IP address of the connection request to be authentic, and either belong to the adversary or to a host he has compromised as otherwise the perpetrator would not be able to collect any responses from exploitable targets. This allows us to quantify who and how many actors are performing reconnaissance for this service and the techniques they use during their search.

5.1 A DDoS Attack Triggering a Scan for DDoS

While news about these attacks was first covered in the media on Feb 27th, the attacks exploiting memcached servers at a large and systematic scale did not start there. Figure 2 shows a historical account of memcached protocol traffic directed towards our monitored IP ranges between August 2017 and April 2018 aggregated in hourly intervals, on the top for the total number of packets, on the bottom the total number of unique IP addresses involved per hour, the red line in both plots shows the date of the first news release.

As we can clearly see in the graphs, actors have scanned and attempted to exploit memcached servers already for months before the emergence of the incident, by sending requests that would trigger responses with a high amplification factor. Although a similar volume of exploitation attempts was recorded at least twice before, four days before the media coverage the total amount of memcached attack traffic massively increased and remained at that level for the duration of an entire month. When looking at the volume of adversary IP addresses over these 8 months, an entirely different picture emerges. Aside from one peak in activity one month prior to the attacks, few people have ever probed our IP ranges for evidence of unsecured memcached servers. With the public discussion of the attacks this however changes drastically, and the overall volume of attacking IP addresses increased by an order of magnitude and remained a continuous activity until the end of the study.

We hence see that following the February 27th public news break, a lot of actors jumped on the band wagon. In essence, the media reports thus triggered a massive influx of new and previously

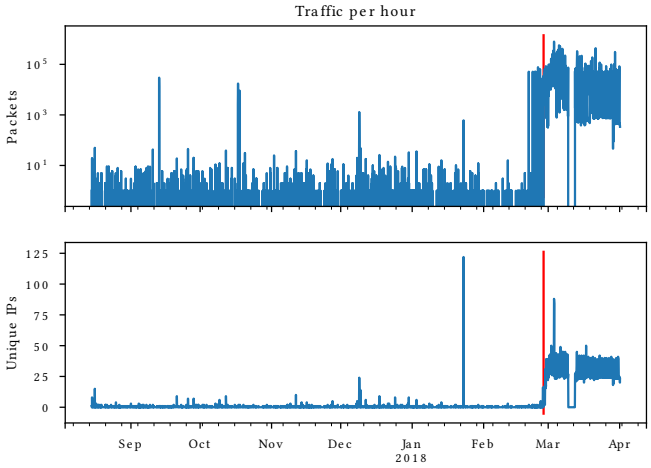


Figure 2: Aggregated traffic directed at TCP and UDP ports 11211 between August 2017 and April 2018. The top part shows the number of packets, the bottom part the total number of unique IPs involved in the reconnaissance.

unseen actors, copycats who are then scanning the Internet and trying to find memcached servers they can exploit themselves.

5.2 Reconnaissance and Attack Campaigns

From figure 2 we see that already long before the February attacks, there were continuous attempts to find exploitable memcached servers. These tests originated from only a handful of IP addresses, what is even more astonishing is that we find that adversaries frequently require extensive trials and iterations to send memcached commands that are syntactically correct and would trigger valid, useful responses for an attack. Few IPs actually start out with the right commands to use.

When comparing the top and bottom plots in figure 2, we furthermore see that the reconnaissance campaigns prior to the massive DDoS were quite different in nature. We can observe two intense traffic peaks as early as September and October 2017, similar in size to the overall probing activities after the public reports, which were generated by a single IP address located in China. Based on the test progression through our IP ranges, the two scan campaigns appear to target the entire Internet, and from the beginning show a good understanding of the memcached protocol.

The tallest spike in the bottom plot embodies actually an actor with entirely different characteristics. As we can see in the figure, the surge in unique IP addresses involved in a scan is not followed by an equal surge in attack traffic. In order to stay undetected from intrusion detection systems which typically flag IP addresses as malicious if their traffic exceeds a predefined threshold, this adversary is distributing the scan for memcached instances over a total of 125 IP addresses. In this reconnaissance, each address would only send five probe packets in total, before the next IP address will continue the scan within the next second. The IP addresses are distributed over multiple autonomous systems and countries with tight coordination, which indicates some level of sophistication and determinism on behalf of the attacker. Surprisingly though, this sophistication does not extend to the specification of the memcached protocol, as each of the involved hosts malforms the application protocol packet in exactly the same way. Besides these two adversaries, most of the pre-incident scanning for memcached remains unsystematic, and the bulk of the activity is only triggered with the influx of additional actors after the media coverage.

5.3 Attack Strategies

If an adversary would like to test the presence of a service at a TCP port, he could negotiate a connection and test whether the service would respond to a query as expected. More commonly however, an attacker would only go through the first part of the negotiation and assert the presence having received a response to the initial TCP SYN. For UDP, there is no such negotiation that would provide this information, thus in order to test whether a service in question is actually running at a given port, it is necessary to send a valid application protocol message in UDP.

As adversaries attempt to exploit the UDP version of the memcached protocol as it allows response redirection, our IP ranges hence do not just receive connection attempts, but actual application layer data from the remote parties. We have briefly mentioned above that packets frequently appear malformed and not every type of command is suited to elicit a usable response from an exposed memcached server, these kinds of artifacts allow us to understand the strategy each party pursues.

While the 34,000 IP addresses we have identified as testing for memcached sent a total of about 16,000 different commands, there are a number of clear favorites people are trying. Overall, 10 commands account for 97% of the total traffic by packets. Figure 3 shows an evolution of the activity of these 10 most popular commands over time, aggregated by the total number of unique IP addresses that use them. As we can see in the graph in brown, the command `stats`, which returns a listing of memcached's system status variables, is predominantly used throughout the reconnaissance activities, but only dominates in the wild *after* the reporting of the incident in the media. The other main commands (by volume) originate only from a handful of IP addresses, which again underlines that reconnaissance for memcached is primarily driven by a small subgroup.

When we look at the evolution of the main command `stats`, we immediately notice two main surges in activity. On the first day of the media coverage, somebody implemented and posted a proof of concept (PoC) on pastebin (pastebin.com/GZcekqLz), which tests for the presence of a server using a fixed UDP payload by requesting statistics from the memcached server. The software can further be identified in the data as it scans through IP ranges in a distinctive way. A few days later, while the attacks continued and an increasing number of media outlets were reporting, another proof of concept appeared on pastebin (pastebin.com/ZiUeinae), which used the same UDP payload but has a different target scanning behavior, and was in contrast to the first PoC not suitable for convenient scanning of a large number of IP addresses. Curiously though, when we traced which articles linked to the two pastebin buckets, we found that media articles actually more often referred to the second, less capable PoC.

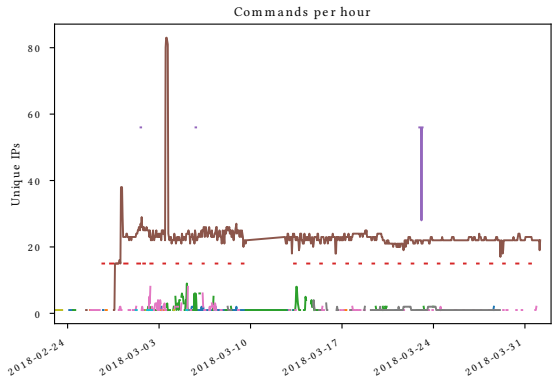


Figure 3: Activity of the 10 most popular memcached commands over time by the total number of remote IP addresses.

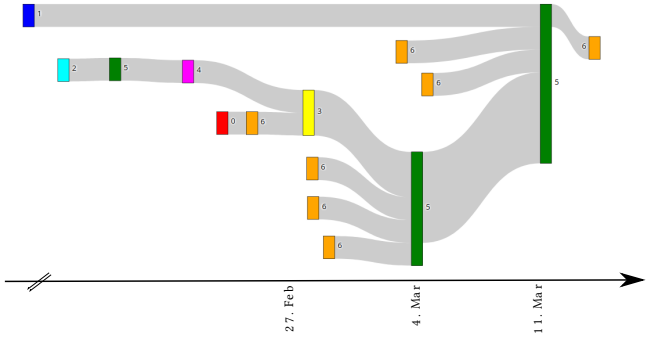


Figure 4: Exemplary command conversation of a set of IP addresses over time. Each strain symbolizes the progression of an IP, the color the command in use at a given moment.

The publication of both PoC results in clear visible spikes in reconnaissance activity in the Internet. We see that after the wide media coverage, new actors seem to adopt the tool that was linked in the reports, a clear characteristic of copycat or script kiddie behavior. Very shortly afterwards however, much of the second surge disappears and a core group remains that continue their scanning activity until the end of the study one month later.

Horizontally across the graph, we can see a line of dots reappearing in regular intervals. This small group can be attributed to shadowserver by IP ownership and rDNS lookup. This set is characterized by a separate command, as the scanners make a mistake in the memcached payload. As discussed in section 3, a memcached command must be terminated by `\r\n`, this entire group however sends an additional `\0`, the null character, at the end of each command. This mistake can most easily happen when the tool scanning for memcached was implemented in C, and designed to append the fixed payload string to the socket data stream. Similar artifacts can however be found across many source IP addresses and many types of implementations leave distinctive fingerprints which allow clustering of sources and partial attribution to specific tools. In figure 3, we can identify two faint purple dots in the first and second week as well as a peak in the third week of March. These activities, which again stand out by unique implementation characteristics, point back to the Rapid7 scan for vulnerable memcached instances, which is one of the datasets used for this study.

5.4 Evolution of Adversaries

These mistakes in how to format memcached packets however do give us the opportunity to consolidate individual scans to a likely common origin, especially when the campaign appears synchronized in time and behavior. It furthermore provides us with a leverage to investigate how actors evolve their understanding of memcached over time and consequently also adapt their tooling.

Indeed, we see many people change their attack techniques over the observation period. Frequently, we detect that the tooling initially contains some mistakes in the protocol, for example header fields are incorrectly set, a wrong command is sent or improperly terminated and formatted. A very noticeable evolution that appeared in late March belongs to an actor who mistakenly sends malformed packets and instead of originating a memcached query from a client, attempts to connect a memcached server to a memcached server. As the protocol is designed as a client-server exchange, this however does not work and the activity soon after stops.

As discussed in section 3, there are some header fields which can be arbitrarily set by the client while others require a very specific content. When participating in large scale scanning for vulnerable systems, it is much more economical to inject a pre-made payload towards many destinations than freshly crafting a new one for each trial. In result, we see for example the same request id reappearing from single origins to a large number of IP addresses in our range. Furthermore, some actors pick valid but unusual or invalid content for the remaining fields, which in combination with the commands are so characteristic to the actors that it allows us to fingerprint sources and their evolution over time. While as stated above most actors do not send the correct command at first, we find that they adapt in time and converge to the same set of valid and productive commands. This process is surprisingly quick, most sources manage these adaptation steps with a turnaround time of just a few hours. It is however astonishing to see that a large number of sources perform large scale port sweeps with the wrong command and rescan with each command update, rather than first test locally for the right configuration and then farming the scan out.

Figure 4 shows such a command evolution exemplarily for 8 IP addresses, which replace their strategies over the course of several days but ultimately converge to the same behavior. This progression of commands is visualized in figure 5 for the entire adversary space, but due to the data volume and tens of thousands of transitions is aggregated to the percentage of overall traffic volume a particular command is used on a given day. The color of each bar denominates a specific command. As we see in the graph, the commands used in the exploitation traffic go through several rounds of evolution, with clear transitioning moments at the public disclosure and the first PoC until the bulk of traffic ultimately arrives at the same solution.

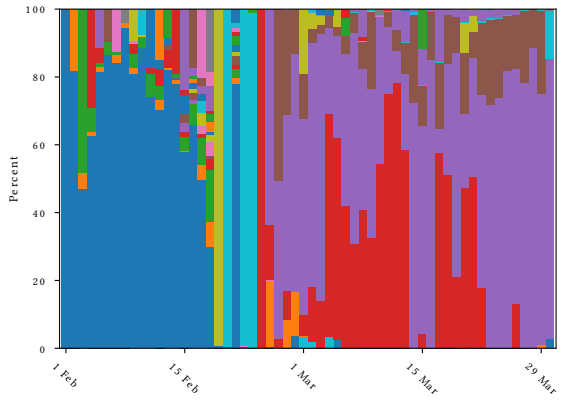


Figure 5: Transition and consolidation of used commands during Feburary and March, shown in percentage of packets per day. The color indicates the command in use.

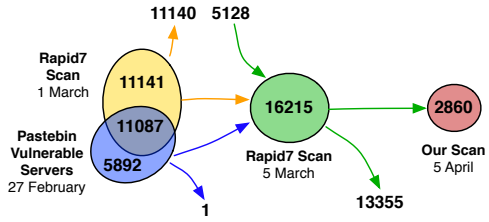


Figure 6: Reconfiguration of exploitable memcached servers in public datasets in response to the DDoS attacks.

5.5 Reconfiguration of memcached Servers

As the abuse of a memcached server also creates an annoyance for the operator of the server, one would expect that after such a major incident and given the expectation that similar attacks are soon going to follow, those who operate an insecure server would change their setup. In fact, with the emergence of the attacks, memcached released version 1.5.6 which by default disables its UDP service, thus removing the easy exploitability of the system.

In order to test this hypothesis, we are comparing two datasets from Rapid7 provided to scans.io, which list IP addresses of open UDP 11211 ports on 1st and 5th of March 2018. This enumerates the landscape just after the emergence of memcached abuse, once before and once after the surge of activity from the widely available PoC. To track the long term change in either reconfiguring their setups or operators updating to 1.5.6, we redid the scan exactly one month later on the 5th of April.

Figure 6 shows how the total volume of memcached servers changes over time throughout the lifespan of the attack. The two public data sets at the beginning of the DDoS campaigns, one public scan by the security firm Rapid7 indicated in yellow and a list of vulnerable servers provided together on Pastebin with the PoC indicated in blue, list a total of 28120 servers open for exploitation. At a second scan 4 days later, about half of the targets from the yellow group had vanished, while from the pastebin list only one had been closed. Surprisingly, 5128 new exploitable memcached servers appeared on the Internet within this timeframe. Within a month after the incident, 83% of the exposed servers had been closed down, deactivated the UDP socket or had been updated.

Those which remain, seem to be heavily geographically clustered. The top 5 of locations by country is dominated by China, where a whopping 37% of vulnerable instances can be found, followed by 12% in the US, 6% in Russia, 3% in Vietnam, and 2% in Brazil. Also when normalizing this finding by the total number of registered IPs in these countries or the number of inhabitants, this geographic bias remains. While it seems that the IPs originally identified as vulnerable do react within a month's timeframe, it is troublesome that actually additional, new vulnerable hosts are introduced over time. This is especially curious as with the memcached version that was released on Feb 27th, the vulnerability is by default switched off, which would indicate either a deliberate reactivation or that new installations draw from outdated software repositories.

6 CONCLUSION

Before a service can be abused for amplification attacks, an adversary must first know about the presence and location of such services. In this paper, we have analyzed the reconnaissance activities on the Internet for memcached servers, which were used to launch the largest DDoS attack to date, 7 months prior to the attacks and in the month thereafter. We find that the attacks are preceeded by some systematic reconnaissance before and actors frequently make mistakes in this process and evolve their techniques for the final goal. We also find that the media coverage about the attacks resulted in a major surge in new actors into the ecosystem, copycats or skript kiddies who adopt ready-made tools and attempt to compile lists of vulnerable machines on their own. Within a month of the attacks, we see that the vast majority of originally vulnerable services are removed or reconfigured from the Internet, we unfortunately also find that even after the attacks new exploitable machines are still added to the Internet. As the latest memcached release has removed the default option for UDP, it should thus be a matter of time before the installation base is sufficiently small to deter adversaries from this attack vector.

REFERENCES

[1] Brad Fitzpatrick. 2004. Distributed caching with memcached. In *Linux Journal.*
[2] Johannes Krupp, Mohammad Karami, Christian Rossow, Damon McCoy, and Michael Backes. 2017. Linking Amplification DDoS Attacks to Booter Services. In *RAID.*
[3] Marek Majkowski. 2018. *Memcrashed - Major amplification attacks from UDP port 11211.* Technical Report. Cloudflare.
[4] Ivan Novikov. 2014. The New Page of Injections Book: Memcached Injections. In *Black hat 2014.*
[5] Vern Paxson. 2001. An analysis of using reflectors for distributed denial-of-service attacks. *ACM SIGCOMM Computer Communication Review* (2001).
[6] Memcached protocol. 2017. https://github.com/memcached/memcached/blob/master/doc/protocol.txt. (2017).
[7] Christian Rossow. 2014. Amplification Hell: Revisiting Network Protocols for DDoS Abuse. In *NDSS.*
[8] José Jair Santanna, Roland van Rijswijk-Deij, Rick Hofstede, Anna Sperotto, Mark Wierbosch, Lisandro Zambenedetti Granville, and Aiko Pras. 2015. Booters—An analysis of DDoS-as-a-service attacks. In *International Symposium on Integrated Network Management (IM).*
[9] Sensepost. 2010. Cache on Delivery. In *Black Hat.*
[10] Daniel R Thomas, Richard Clayton, and Alastair R Beresford. 2017. 1000 days of UDP amplification DDoS attacks. In *Symposium on Electronic Crime Research.*