Improving MassDNS: Adding CNAME Resolution Output Information

Dimitar Vasilev, Patrick Sattler*, Johannes Zirngibl*
*Chair of Network Architectures and Services
School of Computation, Information and Technology, Technical University of Munich, Germany
Email: dimitar.vasilev@tum.de, sattler@net.in.tum.de, zirngibl@net.in.tum.de

***Abstract*—MassDNS is an open-source software for resolving domains on a large scale, that is used to capture and study the state of the Domain Name System (DNS). Currently, MassDNS combines in its output all resource records from all DNS responses. As a result, in addition to the address records, this output often contains CNAME records. This complicates the retrieval of IP addresses. We present a solution to this problem, that involves modifying MassDNS to perform the so-called CNAME resolution on each individual DNS response. This allows for more convenient IP address retrieval. CNAME resolution in this context simply means following all CNAME records and retrieving the IP addresses for a particular domain. This can be especially useful for studies, where only the resolved domains and their respective IP addresses are relevant. In addition, we use our new approach to evaluate our previous, post-processing approach for IP address extraction. As a consequence of performing the CNAME resolution on the entire output of MassDNS, our post-processing approach occasionally results in additional, unforeseen domain-to-address mappings. Our new approach prevents this while also performing better. Based on two scans with an input of around 1 Million domains, we find that our new approach takes around 23% less time for a complete scan. We also observed, that the post-processing approach introduced unexpected addresses to around 1% of all domains, which is relatively insignificant.**

***Index Terms*—massdns, dns, cname**

1. Introduction

Today, many projects and studies focus on capturing and analyzing the state of the Internet. They usually require huge datasets, containing various information about the participants of the World Wide Web, e.g. IP addresses, Geo IP data, ASN (Autonomous System Numbers). Here, at the Technical University of Munich, there are several ongoing studies on this matter, united under the name GINO [1] - The Global INternet Observatory. For some of these studies, we use MassDNS to perform frequent scans of a relatively large portion of the DNS namespace. In this paper, we focus on implementing the CNAME resolution output functionality directly in MassDNS. This will allow us to easily retrieve the resolved domains and their IP addresses in a separate file. In this process, each DNS response is considered separately from all other responses. In contrast, the post-processing program of our previous approach takes as a basis for performing the CNAME resolution the entire output of MassDNS, which can lead to unforeseen domain-to-address mappings.

There are a few significant advantages of embedding the CNAME resolution directly into MassDNS. Firstly, this process can be performed on each individual DNS response, which prevents the error that our previous approach makes. Secondly, we eliminate the need for executing the post-processing program, thus making the scanning process more straight forward. And thirdly, by offloading the job of following `CNAME` records to MassDNS, we do not affect its performance. As a result, the new scanning workflow is significantly faster than the previous one.

When modifying MassDNS, we also made sure not to interfere with the normal output of the program. This is important, as we want to store these output files in an archive. We have been collecting our scan results for over 5 years now and would therefore like to retain their format.

In the rest of this paper, we will adhere to the following structure: Section 2 gives an overview of some projects, similar to MassDNS. In Section 3, we introduce some core concepts of the Domain Name System, relevant for this paper. Section 4 describes our previous, post-processing approach for extracting IP addresses of domains using MassDNS, along with the one that we propose in this paper. Here, we also provide concrete examples to better illustrate their differences. Alternative possible solutions are considered. Section 5 focuses on the changes we made in MassDNS. In Section 6 we evaluate the differences between the two approaches in terms of performance and final outcome (the set of domain-address pairs they produced). We conclude the paper with Section 7, where we summarize our work and emphasize the most important points of our evaluation.

2. Related Work

Researchers have two ways for acquiring large amounts of DNS related data. They can perform DNS scans themselves, or, alternatively, they can use the datasets, provided by other projects, companies and tools. In the following, we will discuss these options with the main focus being how convenient they are for retrieving the IP addresses of resolved domains.

ZDNS is another fast DNS Lookup Tool, part of a collection of open-source internet measurement tools, called the ZMap [2] project. Just like MassDNS, ZDNS can output all resource records from all DNS responses, meaning this output would need some processing for the aforementioned purposes. According to [3] by Liz Izhikevich et al., the authors of ZDNS have opted for a

 doi: 10.2313/NET-2024-09-1_07

modular design for implementing the DNS query specific logic. This allows developers to implement custom behavior for performing lookups. If we were using ZDNS, custom module would be a possible way to implement the CNAME resolution output functionality. In this regard, ZDNS is pretty flexible.

OpenINTEL is also a well-known project for DNS measurement. Within this project, huge DNS scans are performed on a daily basis and the data is provided to researchers. Although not open-source, the overall design and implementation are presented in detail by van Rijswijk-Deij et al. in [4]. According to [4], their datasets "store all resource records included in the answer section of the DNS response, including all `DNSSEC` signatures, `CNAME` records and full `CNAME` expansions". Retrieving resolved IP addresses would therefore still require some explicit processing of the datasets OpenINTEL provides.

OpenINTEL and other similar projects have served as a basis for some of our previous studies ([5], [6]). These projects, however, do not offer any kind of control over the scanning process, which is why sometimes we prefer to perform the scanning ourselves and tailor it to our needs.

3. The Domain Name System

DNS is a distributed system for storing various information, assigned to names (domains). While the main goal is to provide a service for translating domains to host IP addresses, there is no restriction for this single application. For example DNS can be used with different internet protocol families, or to store mailbox data [7]. In this paper, we consider DNS only for the translation of domains to IPv4/v6 addresses. As explained in [7], the DNS consists of three major components: the *Resource Records*, the *Name Servers* and the *Resolvers*. **Resource Records(RR)** contain the data, that is associated with the names (domains). The RR-s, relevant for this paper, are described in Table 1. **Name Servers** are responsible for storing part of the Domain Name Space and making it available for others. **Resolvers** are programs, that communicate with Name Servers and extract information in response to client requests.

It is important to understand the purpose of `CNAME` records. Consider the two DNS responses, presented in Listing 1. In this example, we have resolved two domains: `blog.example.com` and `shop.example.com`. We see, that both domains are aliases for `example.com` and therefore have the same IP address. This configuration is convenient, because in case the IP address of `example.com` is changed, the resource records for `blog.example.com` and `shop.example.com` will still be resolved to the correct IP address. In general, `CNAME` records are useful, because usually one domain offers multiple services under different subdomains. At the same time, multiple domains can be hosted on the same machine or subnet and therefore need to be mapped to the same IP address. Here, the main takeaway is that to reach the address records for queried domains, often `CNAME` records must be followed. This applies to both the domain resolution process (which is handled by the resolver) and the output process (which we implement).

Listing 1: Example DNS Responses

```
;; Response for 'blog.example.com':
blog.example.com    CNAME    example.com
example.com         A        1.1.1.1

;; Response for 'shop.example.com':
blog.example.com    CNAME    example.com
example.com         A        1.1.1.1
```

TABLE 1: Record Types, [7], [8]

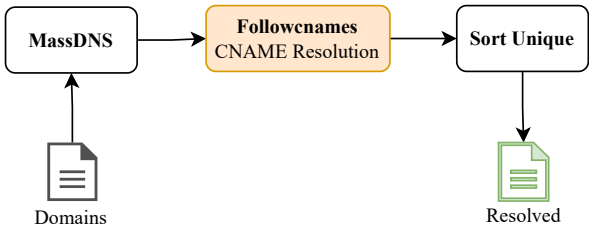
Record Type	Description
A	Address Record
AAAA	IPv6 Address Record
CNAME	Identifies the canonical name of an alias

4. Scan Workflows

In this section, we briefly present our previous scan workflow and the one that we propose. The concrete examples emphasize how exactly the two approaches differ from each another. In addition, we discuss other possible solutions and justify the decision to implement the desired functionality into MassDNS.

4.1. Post-Processing Scan Workflow

Figure 1: Post-processing Workflow Schematic



Until now, our process for resolving domains on a large scale consists of three steps, as visualized in Figure 1:

Step 1. With the right set of command line arguments, we pass a list of domains and resolvers to MassDNS and configure the output format and destination. Then, we execute MassDNS and get an output file, that contains all RR-s received from the resolvers.

Step 2. We pass the output file to our post-processing program, called Followcnames. This program performs the CNAME resolution. For each domain, it simply searches the entire output file of MassDNS for relevant RR-s. In case of `CNAME` records, all possible `CNAME` chains (sequences of `CNAME` records) are followed. In shortly we will show a concrete example to better explain this behavior.

Step 3. We use the Linux `sort` utility to remove all duplicating domain-address pairs. Duplicates can be introduced in the previous step, but in rare cases, DNS responses can also contain duplicating `A` or `AAAA` records. This way we keep our datasets clean and tidy.

Here is a real scenario, that we observed during our tests, in order to illustrate how the post-processing works. Suppose we want to resolve only two domains: `cookbook.openai.com` and

 doi: 10.2313/NET-2024-09-1_07

app.rifei.com.br. Listing 2 shows a simplified version of the MassDNS output, that we observed.

Listing 2: MassDNS Raw Output

```
cookbook.openai.com  CNAME cname.vercel-dns.com
cname.vercel-dns.com A     76.76.21.22
cname.vercel-dns.com A     76.76.21.164
app.rifei.com.br     CNAME cname.vercel-dns.com
cname.vercel-dns.com A     76.76.21.142
cname.vercel-dns.com A     76.76.21.241
```

We can see, that two different domains are aliases for the same domain, which in turn has four different IPv4 addresses. Two of them were received for the first domain and the other two for the second domain. Now, when Followcnames tries to extract all addresses for the domain cookbook.openai.com, it will follow the CNAME record to cname.vercel-dns.com and then find all 4 different A records of cname.vercel-dns.com. The same goes for the second resolved domain, app.rifei.com.br. As a result all 4 IP addresses will be assigned to both domains. Listing 3 visualizes this result.

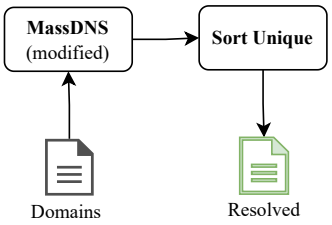
Listing 3: Post-processing Result

```
76.76.21.22, cookbook.openai.com
76.76.21.142,cookbook.openai.com
76.76.21.164,cookbook.openai.com
76.76.21.241,cookbook.openai.com
76.76.21.22, app.rifei.com.br
76.76.21.142,app.rifei.com.br
76.76.21.164,app.rifei.com.br
76.76.21.241,app.rifei.com.br
```

In cases like this, we say that the post-processing program (performing CNAME resolution on the entire MassDNS output) has affected the two resolved domains by assigning them additional IP addresses.

4.2. New Scan Workflow

Figure 2: New Workflow Schematic



Taking advantage of the improvement we made to MassDNS, a single scan with the new workflow, as shown in Figure 2, only includes **Step 1** and **Step 3** from our previous approach. Since we embedded the CNAME resolution into MassDNS, executing the post-processing program Followcnames is unnecessary. In addition, by following the CNAME records in each DNS response separately, we expect to get different results. For the same example of resolving cookbook.openai.com and app.rifei.com.br, we get the set of domain-address pairs, shown in Listing 4. We see that it is impossible for this approach to mix address records from different DNS responses, which is exactly what we aimed for.

Listing 4: Expected Result

```
76.76.21.22, cookbook.openai.com
76.76.21.164,cookbook.openai.com
76.76.21.142,app.rifei.com.br
76.76.21.241,app.rifei.com.br
```

4.3. Alternative Solutions

MassDNS supports different output formats, e.g. it can store the resource records from each DNS response in json format. Again, as a post-processing step, a simple script could parse this json output and extract the IP addresses associated with each domain from the respective DNS responses. However, judging by our Followcnames program, we expect similar performance from analogous post-processing scripts, which as we later confirm is not optimal. In addition, we must consider one important constraint when evaluating different possible solutions. As mentioned, we want to preserve the standard output of MassDNS, because over time we have stored a lot of historical data in the same format. With this requirement in mind, we see how modifying MassDNS and tuning it to our needs is the better solution and also aligns with our requirements for higher performance and efficiency.

In the rest of this paper, we call the approach that we propose the direct or the new approach, and the one that we used until now the post-processing or the previous one.

5. Implementation

Implementing the desired functionality into MassDNS does not require any major changes. As we do not want to replace the standard output of MassDNS, we first add a new command line option, called "--ip-outfile". It is used for specifying the file for the extracted pairs of domains and IP addresses.

We embed the address extraction process in the do_read function, in the main.c file. This function processes the responses of all DNS queries. If the query was successful, MassDNS parses the received DNS response and writes to the standard output file in the requested format. Immediately after that, we perform few additional steps. First, we check if A or AAAA records were requested. If this is the case, the DNS response is parsed once again. We follow the CNAME records, if there are any. Eventually, we reach the address records and write them in a separate file, together with the originally queried domain.

We do not expect this tweak to have any performance impact on MassDNS and confirm this in our evaluation. Network communication is rather slow, even when compared to tasks typically considered slow, such as IO operations on a persistent storage device. We suspect that despite the fact that MassDNS is designed with a high concurrency in mind, a significant portion of the total execution time is spent in waiting, whereas just a small fraction in processing the DNS responses. This explains how no slowdown would accumulate even when millions of domains are resolved.

TABLE 2: Evaluation of the Effect of Followcnames

Test	Total Domains	Resolved Domains	Affected Domains	Falsely Resolved Domains
#1	997 382	953 393 (95.6%)	6566 (0.7%)	100
#2	997 382	954 507 (95.7%)	13 002 (1.4%)	104

(a) Domains

Test	Total Pairs	Added Pairs
#1	1 572 013	17598(1.1%)
#2	1 581 746	25799(1.6%)

(b) Domain-Address Pairs

6. Evaluation

In this section we show if the changes we made in MassDNS negatively affect its performance. Afterwards, we compare the two approaches for address extraction with respect to the final result. For this we use real data from two MassDNS scans.

To conduct our tests, we need a set of domains, ideally a large sample, representative for the most frequently used domains in the domain name space. There are several top lists available for this purpose. We opt for the one, provided by CrUX [9] - the Chrome User eXperience Report. This list consists of around one million domains. This size is suitable for our tests, as it is large enough to be considered large-scale, but small enough for it to be feasible to compare the results.

6.1. Speed and Efficiency

For the given input, we did not observe any significant difference between the normal and the modified version of MassDNS. Here are the specific time measurements:

1) **MassDNS** - *1 Min. 40 Sec.* (both workflows)
2) **Followcnames** - *50 Sec.* (previous workflow)
3) **Sort Unique** - *1 Min. 10 Sec.* (both workflows)

However, since we skip the execution of Followcnames in our new workflow, the total compute time is reduced by around 23%. Based on rough calculations, we found that this improvement can save up to 3 hours of compute time within one of our typical scan days, depending on how well our solution scales.

6.2. Output Comparison

In Section 4 we showed that under certain circumstances our post-processing program can artificially introduce unexpected domain-to-address mappings. Given that we have used this scan workflow in the past, and considering it remains the only option for retrieving domain-address pairs from our archived scans, we want to estimate the magnitude of the error, that it produces. For this purpose we take advantage of the fact, that our modification of MassDNS does not affect its standard output. We proceed as follows: we run the improved MassDNS version with the aforementioned CrUX list as an input. Then, we pass the standard output file through the post-processing step. Effectively we just perform the old and the new approach simultaneously, in a single scan. We could also just conduct two scans with the two different workflows separately, but the scans tend not to be exactly reproducible, which could render the comparison invalid or misleading.

The data in Table 2 illustrates the measurable effect of the Followcnames program on the final outcome. In the following, we explain each statistic.

Resolved Domains. This is just a control statistic and shows the ratio of successfully resolved domains over all domains. Lower values should raise suspicion. This can indicate e.g. poor quality of the top list, some kind of error in the configuration or even in the implementation of MassDNS. However, we observe that around 95% of all domains were resolved, which is quite reasonable.

Affected Domains. As previously mentioned, affected are all domains, that received additional IP addresses as a result of the post-processing step Followcnames. We observed, that unforeseen addresses were assigned to around 1% of all queried domains. We have calculated, that each affected domain received on average between 2 and 3 additional addresses, but this value goes up to 56 for some domains.

Falsely Resolved Domains. The Followcnames program was able to find in the MassDNS output IP addresses for around 100 domains that our modification did not report as resolved at all. Upon closer inspection however, we found that all of these domains do exist and can be resolved, so we did not observe any non-existent domains to appear as resolved as a result of the post-processing.

Total and Additional Pairs. Under 2% of the all extracted domain-to-address pairs were artificially introduced by Followcnames.

Even though the amount of affected domains is relatively small, their presence still raises the question of how such differences can occur. Logically if two domains are aliases for the same, third domain, they should always be resolved to the same addresses. However, this is not always true, as we have seen in the example with `cookbook.openai.com` and `app.rifei.com.br` in Section 4. There can be several reasons for this. As an example, although it is unlikely, misconfigured Name Servers or invalid caches could cause such issues. Alternatively, Name Servers often store different information depending on their geolocation, which is a neat way of employing DNS for load balancing. Therefore, by communicating with different Name Servers, a resolver can receive different IP addresses for the same domain.

In the end, whether the differences we observed are significant depends on the context in which the resolved addresses are used. Theoretically, depending on the order in which different Name Servers are queried, a resolver can also receive the additional IP addresses, that our post-processing finds. This is why we believe that our previous use of the post-processing approach should not raise any concerns.

7. Conclusion

For several of our studies here, at the Technical University of Munich, we use MassDNS to perform huge DNS scans on a daily basis. However, the standard output

of MassDNS is not convenient for our studies, as it contains the raw DNS responses. This is why our current scanning workflow includes some post-processing on the output of MassDNS. Essentially we perform the CNAME resolution on the entire output file. In this paper we present a new way to perform the CNAME resolution on a single DNS response level, in order to retrieve the resolved domains and their IP addresses. We embedded this functionality directly into MassDNS.

In the evaluation phase, we used our previous approach to assess the new one with respect to speed and efficiency. In addition, we used the new scan workflow to evaluate the impact of the post-processing procedure in our previous workflow. We show that the performance of MassDNS is not affected by our modifications whatsoever. Furthermore, by omitting the execution of the post-processing program Followcnames, we can save up to 3 hours of compute time within one of our typical scan days, which is a major improvement. Based on two large-scale scans with an input of approximately 1 million domains, we found that around 1% of all domains had received additional addresses as a result of the post-processing procedure. Some of them had up to 56 additional addresses. Moreover, during the CNAME resolution, the post-processing procedure was able to find addresses in the MassDNS output for around 100 domains that were not resolved during the scan. However, we believe that with the right combination of Name Servers, a resolver can also reach those IP addresses, that we considered as unexpected (those artificially introduced by Followcnames). For this reason, we do not label the previous approach as strictly invalid, nor do we reject its application to this point.

References

[1] "Global INternet Observatory," https://www.net.in.tum.de/projects/gino/, 2016, [Online; accessed 02-April-2024].

[2] "ZMap Project," https://zmap.io, 2022, [Online; accessed 04-April-2024].

[3] L. Izhikevich, G. Akiwate, B. Berger, S. Drakontaidis, A. Ascheman, P. Pearce, D. Adrian, and Z. Durumeric, "Zdns: a fast dns toolkit for internet measurement," in *Proceedings of the 22nd ACM Internet Measurement Conference*, ser. IMC '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 33–43. [Online]. Available: https://doi.org/10.1145/3517745.3561434

[4] R. van Rijswijk-Deij, M. Jonker, A. Sperotto, and A. Pras, "A high-performance, scalable infrastructure for large-scale active dns measurements," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 6, pp. 1877–1888, 2016.

[5] J. Zirngibl, S. Deusch, P. Sattler, J. Aulbach, G. Carle, and M. Jonker, "Domain Parking: Largely Present, Rarely Considered!" in *Proc. Network Traffic Measurement and Analysis Conference (TMA) 2022*, Jun. 2022.

[6] O. Gasser, Q. Scheitle, S. Gebhard, and G. Carle, "Scanning the IPv6 Internet: Towards a Comprehensive Hitlist," in *Proc. 8th Int. Workshop on Traffic Monitoring and Analysis*, Louvain-la-Neuve, Belgium, Apr. 2016. [Online]. Available: https://net.in.tum.de/pub/ipv6-hitlist/

[7] "Domain names - concepts and facilities," RFC 1034, Nov. 1987, [Accessed 02-April-2024]. [Online]. Available: https://www.rfc-editor.org/info/rfc1034

[8] V. Ksinant, C. Huitema, D. S. Thomson, and M. Souissi, "DNS Extensions to Support IP Version 6," RFC 3596, Oct. 2003, [Accessed 02-April-2024]. [Online]. Available: https://www.rfc-editor.org/info/rfc3596

[9] "CrUX," https://developer.chrome.com/docs/crux, [Online; accessed 04-April-2024].