Taxonomy and Adversarial Strategies of Random Subdomain Attacks

Harm Griffioen and Christian Doerr
Delft University of Technology, Cybersecurity Group
Van Mourik Broekmanweg 6, Delft, The Netherlands
{h.j.griffioen, c.doerr}@tudelft.nl

***Abstract*—Ever since the introduction of the domain name system (DNS), attacks on the DNS ecosystem have been a steady companion. Over time, targets and techniques have shifted, and in the recent past a new type of attack on the DNS has emerged. In this paper we report on the DNS random subdomain attack, querying floods of non-existent subdomains, intended to cause a denial-of-service on DNS servers. Based on five major attacks in 2018 obtained through backscatter measurements in a large network telescope, we show the techniques pursued by adversaries, and develop a taxonomy of strategies of this attack.**

Keywords*—*cyber threat intelligence, random subdomain attack, DNS, DDoS

I. Introduction

The domain name system (DNS) is one of the fundamental services in today's IP networks. Translating human-readable domain names to Internet Protocol addresses used for forwarding data, a failure of the DNS will let resources effectively disappear for the typical user, and malicious interfering with the records stored there – so-called DNS poisoning – can transparently redirect unsuspecting users to malicious servers. Given this criticality, the DNS ecosystem has been subject of attacks ever since its inception, and we have seen in the recent past numerous occasions of cyber criminals using them of large-scale compromises [**?**].

While the basic functioning of DNS is largely unchanged since its introduction in 1983, over the years many types of attacks on the domain name system have emerged that target each of the individual components using a variety of attack vectors. Aside from attack that target the integrity of records such as classic DNS poisoning or the Kaminsky attack, we have also seen assaults on availability aimed to bring down authoritative DNS providers and recursive resolvers [**?**], root domain servers [**?**], or selected top-level domain zones [**?**].

In the very recent past, this portfolio of attacks was complemented by a new attack vector that targets authoritative DNS servers through the massive distributed querying of random subdomain attacks. Sometimes dubbed the DNS water torture attack, a lookup of a randomly chosen subdomain will in near certainty lead to a cache miss in DNS servers and thus force a query to the authoritative server responsible for serving a domain. If this attack is executed from multiple end points, the flood of queries will soon overwhelm this endpoint and effectively remove a particular domain from the Internet. This new type of attack gained some visibility when source code triggering it was discovered in a forensic analysis of the Mirai Internet-of-Things malware in 2016 [**?**], since then a number of actors have picked up this particular technique.

Until now, very little is known how these attacks are executed and exactly using which means. This paper intends to address this gap. Based on backscatter measurements through a large network telescope, we observed the emergence of random subdomain attacks over a period of three years and will in this paper make the following two contributions:

- We show the spectrum of different types of random subdomain attacks in use in the wild today based on five exemplary case studies, and use this to develop a taxonomy of random subdomain attacks. For these types of attacks, we quantify how much they occur based on 575 random subdomain attacks identified in 2018.
- We are able to show in contrast to previous reports random subdomain do not only target authoritative DNS servers, but we find evidence that many of them are used to attack the recursive resolvers of Internet Service Providers.

The remainder of this paper is structured as follows: Section II review the still sparse prior work on this topic. In Section III we summarize the functionality of the domain name system, briefly outline previous attacks on the DNS and place this new type of attack into context. In Section IV, we provide an overview of our data acquisition methodology. Section V describes adversarial strategies observed in major attacks observed in 2018, and places them into a taxonomy. Section VI summarizes our findings and concludes our work.

II. Related work

As the attack vector using random subdomains is comparatively recent, it has only begun to be considered within the academic literature. Luo et al. [**?**] state that random subdomain attacks are not researched as much as Domain Generation Algorithms, while random subdomain attacks are more abundant. This shows that random subdomain attacks are abundant, and are able to cause serious problems for DNS services. Most notably was the Mirai attack on Dyn DNS [**?**], putting the attack in the spotlight as the biggest DDoS attack

at the time. In the work by Takeuchi et al. [**?**], a method was introduced able to identify these random subdomain attacks.

Alonso et al. [**?**] conclude random subdomain attack to first impair a recursive resolver, before hitting an authoritative server. Therefore, the target of a random subdomain attack could be one of two: the recursive resolver, or the authoritative nameserver. Current research has not differentiated between the two, and there is to our knowledge no research on the targets of random subdomain attacks. To address this gap, this paper will analyze random subdomain attacks for commonalities, and based on this create a taxonomy of this attack vector.

III. Attacks on the DNS ecosystem

In order to make resources in IP-based networks accessible using easily rememberable names, the domain name system realizes a lookup that maps domain names to network addresses and vice versa. Here, we will briefly outline the key features of the DNS, and after a summary of existing attacks describe the alternative vector that random subdomain attacks introduce.

Suppose a client would like to visit the website of amazon.com. In order to know by which webserver the site is served, it would send a DNS lookup to its recursive resolver, which is typically provided by the Internet Service Provider that facilitates the Internet connection. As shown in Figure 1, the recursive resolver check its local cache whether an answer already exists and could be immediately be returned. In case of no record, the recursive resolver would proceed to explore the entire hierarchy, first querying the root DNS for the location of the nameserver responsible for the .com top-level domain (TLD), and query this nameserver for the address of the nameserver responsible for queries for the second-level domain amazon.com. This so-called authoritative domain name server is then in the position to answer the inquiry for "www.amazon.com", and will return the desired record to the recursive resolver. In practice, authority for a subdomain could be delegated further to different nameservers, and in order to avoid querying the entire hierarchy the address of the TLD nameserver would be cached with a long time-to-live, so that the evaluation would begin either at the TLD DNS or directly at the authoritative DNS server.

There have been numerous attacks over the years on the DNS ecosystem. As with other critical infrastructures, it is vital to protect DNS servers from attacks on integrity and availability. Each of these aspects has been targeted over the years, with a variety of different attacks enabling attackers to direct people to malicious websites or deny people access to certain DNS records. In this paper, we will focus on an attack that targets the availability of certain key parts of the ecosystem. In the past, we have seen denial-of-service attacks on the root DNS servers (for example, the attacks on the K root in 2016), as well as attacks on the servers hosting the root zones [**?**]. As businesses are increasingly moving to service providers also for serving DNS, successful attacks on authoritative DNS providers nowadays will have major impact, as we have seen with the outage of the Dyn DNS in 2016. The attack impaired service availability for many websites, including popular websites like Netflix, Twitter and Spotify,

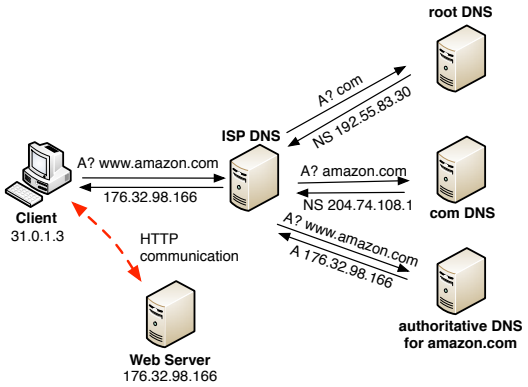


Fig. 1: For a lookup the recursive DNS iteratively queries root, TLD, and authoritative DNS server unless the entry is cached.

for several hours [**?**]. As DNS is a protocol that generates comparatively large responses to small requests, this high amplification ratio has also been exploited to generate traffic floods as part of reflection and amplification attacks, where many requests are sent to resolvers on behalf of a spoofed client and the volume of answers then overflows the victim.

The Random Subdomain Attack

Similar to the attacks above is the random subdomain attack, which creates a denial-of-service against the authoritative DNS server. While past DDoS attacks would for have used traffic floods to impair the authoritative DNS, this attack causes a resource exhaustion within the DNS protocol itself.

The attack is rather simple in nature. As shown in Figure 2, in this type of attack an adversary is sending queries to a recursive DNS server which are unlikely to be currently cached. The easiest and effective method is by means of random, non-existent subdomains under the attacked domain, which require the resolver to reach out to the authoritative nameserver. This server responds that the domain is non-existant (an `NXDomain`), only to be served immediately after the request for another random subdomain within its zone. The attack is difficult to mitigate as the requesting recursive resolvers are legitimate DNS servers, so if the authoritative DNS server would block these IPs the domain would no longer be serviced also to legitimate clients. The requests for random lookups eventually exceed the limited resources of the authoritative DNS, which stops answering also legitimate lookups and all domains under its administrative control will become unreachable if their cached entries will time out. The recursive DNS will indicate the outage of the authoritative DNS to the requesting client as a `ServFail` response. Note that the random subdomain attack is sometimes also referred to as a "DNS water torture attack" after one of its first reports in [**?**]. As the naming is not motivated from the attack mechanics and also highly non-intuitive, we will throughout this paper use the descriptive label of "random subdomain attack".

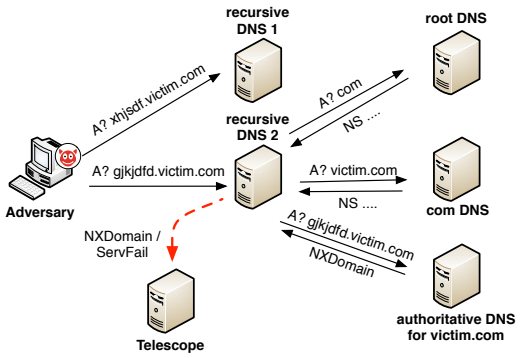


Fig. 2: Backscatter from random subdomain attacks provides insights into which part of the DNS ecosystem is attacked.

IV. Observing Attacks through Backscatter

As we have seen above, in the random subdomain attack the adversary circumvents the caching properties of the domain name system by repeatedly requesting random subdomains that have not been queried before, and thus force the recursive resolver to reach out to the authoritative DNS server to retrieve it. As the record of the server being responsible for a particular zone will be stored at the recursive DNS, subsequent client queries will directly go out to the authoritative DNS, thereby the impact of the attack will accumulate at this endpoint as it is the entity where all requests converge.

In order to increase the force of the attack the adversary will likely request records via multiple recursive resolvers (in the figure for readability only the lookups of server 2 are shown), here even the same query could be replayed as records are cached per server. Furthermore, in order to veil their identity, attackers are unlikely to use their own IP address in the request packet, but spoof the network address of a third party where the response is then delivered to. This is possible as DNS mainly relies on UDP transport layer protocol to reduce latency, but the handshake-less UDP cannot provide IP spoofing protection. While address spoofing hampers attribution, it also makes the attack much more difficult to mitigate, as the recursive DNS resolver could not simply block individual IP addresses where the requests come from. The optimal strategy for the adversary is to randomly spoof a source IP for each DNS lookup, thereby distributing the attack and the responses over the entire Internet. As the responses trickle in to all hosts whose IP addresses were spoofed as part of the attack, this reflected backscatter allows an insight into which domains were attacked using which recursive DNS servers as shown in Figure 2. As the response contains the relevant features of the original query, this backscatter further allows insights into the way the attack packet was crafted.

For our study, we built on the backscatter traffic collected by a large-scale telescope. For this study, we used the reflections sent to approximately 65,000 unused IP addresses spread across two partially utilized /16 networks, which as shown in [**?**] provides a statistically accurate view of attack characteristics and volumes of Internet attacks. Over a period of 9 months, we collected a total of 12.9 million unsolicited DNS response packets. In this paper, we report only on a subset of these attacks, namely responses that indicated a random subdomain in progress. We selected these as incoming responses for subdomains (i.e., patterns in the form of ????.domain.TLD) with NXDomain or ServFail status codes, thus, the subdomain was reported as non-existent by the authoritative DNS or the recursive DNS reported the authoritative DNS to be not responding, in other words the attack was successful. Random subdomain attacks are still relatively infrequent, and we report only 2 percent of all DNS response packets to be part of such attacks.

V. A Taxonomy and Adversarial Strategies in Random Subdomain Attacks

After we have discussed the conceptual idea behind the attack and placed it into context, we will report here key findings from our investigation of this vector in 2018, and a taxonomy of attack strategies. For this, we have analyzed 575 attacks, but will start the discussion by discussing a small number of attacks in depth.

A. Adversarial Attack Strategies

From our analysis, we have found multiple different attack infrastructures used to perform the attacks observed in 2018. While the infrastructure is generally different, the structure of the attack is often times not. For five major attacks, we have looked at the attack strategy, the recursive resolvers and authoritative DNS servers hit, as well as the concrete implementation of the attack packets, such as the randomization of the subdomain, source port, the transaction ID in the DNS header and the requesting IP address. All of these fields are set by the adversary, and not changed by DNS infrastructures. As these fields are chosen by the adversary, methods of selecting these fields provide identifying information. Therefore, the strategy used to select these fields can be used to identify whether an attack came from the same source or tool. The attacks and some of their properties can be found in Table I.

While the random subdomain attack is designed to flood authoritative DNS servers with traffic, they also put a strain on the recursive resolvers. Of the five major attacks, one attack likely targeted an ISP's DNS resolver due to two reasons: first, in this case the attack was not distributed but all attack packets were directed at this single resolver thus the adversary deliberately let go of the additional "fire power" that was leveraged in other attacks, and second the zone file for the domain was hosted by a major DNS provider in a multicast configuration, thus the lookup traffic of the single recursive resolver had not been enough to impair the availability of these servers. As the authoritative server was well provisioned, in this attack we have observed the resolver to fail after 352 randomized queries. Extrapolating over the entire Internet based on the technique in [**?**], 23,258,752 queries were needed to successfully undermine the availability of the DNS resolver, and given that the typical DNS request was observed in the order of 75 - 85 bytes per packets, we can compute that this recursive DNS resolver failed after only 15 Mbps of sustained random subdomain lookups.

TABLE I: Characteristics of surveyed subdomain attacks. Only the most noteable fields are shown in this table.

Domain	Subdomain	Transaction ID	Target DNS	Type
anever.cn	8 random characters	Randomized	Authoritative	(b)
ddoscc.cn	8 random letters	Randomized	Authoritative	(a)
hntl.cn	8 random characters	Reused	Authoritative	(a)
tenih.tw	mk[9-10 numbers]	Reused	Recursive	(c)
ttququ.net	random length	Static per IP	Authoritative	(b)

Unlike the attack above, the other attacks most likely targeted authoritative DNS servers, as these attacks used multiple DNS resolvers for their queries, or the response of the authoritative DNS server was sent to our telescope directly, without a recursive resolver as intermediary. From this, a distinction can be made between attacks where a resolver was queried directly, and attacks where the resolver was impersonated by the attacker. In the latter case, we observe a response coming directly from the authoritative DNS. Two of the five attacks queried DNS resolvers which were used as a proxy to forward the query to the authoritative DNS. While the goal of the attacks seems to be to impair the availability of an authoritative DNS, there is no indication that the attacks were actually successful as the servers continued to return NXDomain instead of ServFail responses.

In some attacks, packets are replayed several times. As UDP is a stateless protocol, the DNS resolver receiving a replayed packet will just resolve this. However, as the DNS resolver already received this query, it will most likely have it cached, mitigating part of the attack traffic to the authoritative DNS. While this would hold true when the attack is executed using a single DNS resolver, we notice that these attacks involve the help of multiple DNS resolvers. By using multiple DNS resolvers, the attackers that replay packets circumvent caching on a single server, as the DNS resolvers will all query the authoritative server. When using multiple servers, adversaries first start by starting the attack traffic using one resolver, and then adding additional resources during the progression of the attack. Additional servers are generally added a few seconds later, with the majority of the servers only added after 30 seconds.

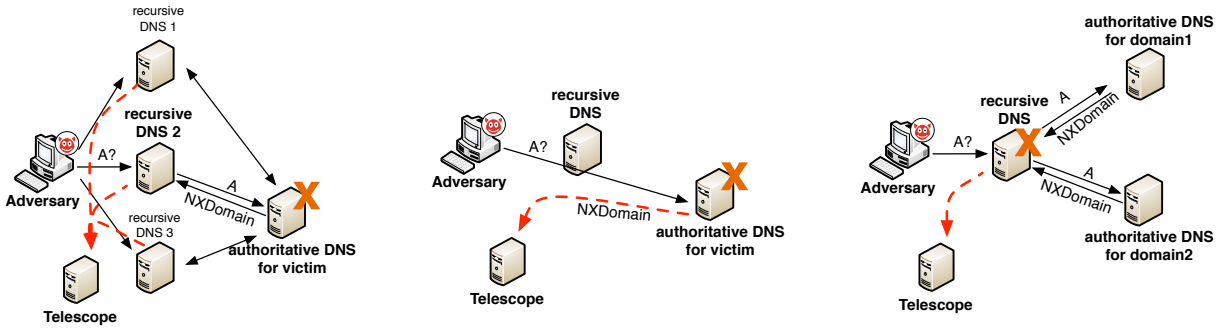
The results above shows that there are different attack techniques used to perform random subdomain attacks. This distinction also becomes apparent when looking at the strategies used by adversaries to craft the packets used in the attack. As this is a random subdomain attack, an adversary has to choose the subdomain used for querying to the DNS resolver. In five major attacks analyzed, four ways of constructing a random subdomain have been identified. Three of the five attacks use generated subdomains of fixed length. From these three attacks, two are identical in the chosen subdomains, as they both consist of 8 characters, which are either a number or a lowercase letter. The other attack using subdomains of fixed length also uses 8 characters, but never uses a number. The remaining two attacks differ greatly, as they are both do not use fixed length subdomains. One of the two attacks always constructs the subdomain with 'mk' followed by 9 or 10 numbers. Numbers used by this attack do not have any order or structure. The final attack does not only randomly generate the characters for the subdomain, but also randomizes the length of the queried subdomain.

Attacks also differ in the port allocation of the attack. Similar to the subdomain, a source port can be chosen by the attacker. If the adversary injects the random subdomain requests at the application layer (instead of for example bypassing the networking stack) these would typically originate from the same source port. Most attacks randomize this port in a way which would indicate a customized application (instead of a regular OS socket), but the used techniques differ in various ways. One of the attacks randomize the destination port only when they also randomize the IP address. This behavior is not shared by other attacks, as the other attacks randomize the packet each time a new packet is created. While they all randomize, one of the attacks only randomizes the fields in a certain range, whereas three of the attacks randomize in the entire port range (0 - 65535). Even when the source port is randomized, the transaction ID is often times not. In DNS, the transaction ID is used to identify which response corresponds to which request. To be able to identify a DNS response, it should have a distinct query ID on the client system. In three of the five attacks however, this is not the case. In one attack, transaction IDs are reused over 20 times, spanning multiple different subdomain queries. So even though the applications are customized and bypass the network stack, most adversaries deem it unnecessary to randomize the transaction ID.

B. A Taxonomy of Random Subdomain Attacks

Based on the concrete attacks observed during 2018, we find that random subdomain attacks are used in practice in a variety of flavors. Figure 3 places these instances into an attack taxonomy. The previously reported and most logical setup of such an attack is the use of recursive DNS resolvers as proxies querying the authoritative DNS servers as shown in Figure 3(a), primarily as attack packets would be scaled out and the victim would have little opportunity to blacklist the recursive DNS as it would otherwise deny service to legitimate clients served by them. This form follows the tradition of classic TCP SYN floods where an adversary injects massive amounts of request packets from randomly spoofed sources. While whitelisting "known" DNS and dropping other traffic would be an option that would incur no collateral damage, combating this attack flavor is more easy for the victim, yet requires a significant firewall ruleset. As shown in Figure 3(c), random subdomain attacks not only target authoritative servers but can also be employed to overwhelm the recursive DNS as well, by exhausting the maximum number of open, unanswered queries that may be in flight at any given moment. By triggering extensive lookups, the server is thus unable to recursively query the DNS hierarchy for other client requests, thus crippling the domain name resolution for all clients configured to use this particular resolver.

The five attacks from above showed the spectrum of strategies adversaries utilize in random subdomain attacks, but we have quantitatively analyzed a total of 575 random subdomain attacks that were recorded by our telescope in 2018. From the observed attacks, we see that the most occurring attack strategy



(a) Subdomain attack on authoritative DNS (b) Subdomain attack with resolver bypass (c) Subdomain attack targeting resolver

Fig. 3: Random subdomain attacks manifest themselves in practice in three different flavors, and aside from targeting the authoritative DNS are also used in practice to take down recursive DNS resolvers.

is (a), as 64% of attacks were aimed at multiple resolvers for the same domain name. As this is the original use of the random subdomain attack, where the resolvers cannot cache the responses due to the changing subdomains, it is expected to observe this flavor the most. Attacks proxied using resolvers are hard to block as that would mean that the legitimate users of that resolver would also be denied access to that particular domain. This also holds true for direct attacks as in (b), which is second most occurring and has been observed in 28% of the cases. Upside of this attack is that it is totally infeasible to block on an IP level, whereas an authoritative server could opt to deny service to a resolver when under attack, accepting the collateral damage in the process. As the attack is harder to block when performed using this strategy, it can be assumed that adversaries start to shift towards this type of random subdomain attack. Attacks targeting a single resolver as in (c) have been observed in only 8% of the cases. All random subdomain attacks that were recorded by the telescope could be placed into the categories of the taxonomy, without any of the attacks indicating a different attack scenario. As most of the attacks are low in volume, the received number of packets does not provide an accurate estimation how many attacks were actually successful. Additionally, no indication has been found that attacks observed were conducted by a single person or tool, as the behavior in which random fields were chosen differed greatly between attacks. This is based on analysis on the fields such as source ports, transaction IDs and the subdomains, which as stated before, adversaries are free to choose. In terms of source ports, 73% of the analyzed attacks do randomization, with the other 27% not randomizing this port. Surprisingly, the opposite is true for the transaction ID where 63% of observed attacks sometimes reuse transaction IDs, the other 37% always pick a unique one per try. For subdomains there is a dominant randomization technique, as 57% generates a subdomain of 8 random characters. Solely the subdomain randomization does not provide enough evidence to indicate collaboration, as the other fields are not common among most of the attacks.

VI. Conclusion

In this paper, we have analyzed a recent new attack vector on the DNS, the random subdomain attack, and investigate adversarial methods using 575 attacks in the wild. Our empirical evidence furthermore shows three main flavors of random subdomain attacks in the wild, and we are able to show that in contrast to previous reports this new type of attack is not only targets authoritative DNS servers but is also used to take down recursive resolvers through resource exhaustion. While the attack is used to take down recursive resolvers, the most popular use of this attack is to attack the authoritative DNS server. Recently emerged, random subdomain attacks are still comparatively low volume at this moment, but as much less traffic is needed to successfully attack a nameserver we can expect them to gain wider popularity in the future.

References

[1] Cloudflare, "Bgp leaks and cryptocurrencies," 2018.

[2] S. Hilton, "Dyn analysis summary of friday october 21 attack," 2016.

[3] G. Moura, R. d. O. Schmidt, J. Heidemann, W. B. de Vries, M. Muller, L. Wei, and C. Hesselman, "Anycast vs. ddos: Evaluating the november 2015 root dns event," in *Internet Measurement Conference*, 2016.

[4] A. Gonsalves, "China ddos attack shows not all tld servers equally secure," *CSO*, 2013.

[5] NN, "Investigating mirai," tech. rep., A10Networks, 2016.

[6] X. Luo, L. Wang, Z. Xu, K. Chen, J. Yang, and T. Tian, "A large scale analysis of dns water torture attack," in *International Conference on Computer Science and Artificial Intelligence*, 2018.

[7] Y. Takeuchi, T. Yoshida, R. Kobayashi, M. Kato, and H. Kishimoto, "Detection of the dns water torture attack by analyzing features of the subdomain name," *CSAI*, 2016.

[8] R. Alonso, R. Monroy, and L. A. Trejo, "Mining ip to domain name interactions to detect dns flood attacks on recursive dns servers," *Sensors*, 2016.

[9] T. Degroote, "Water torture: A slow drip dns ddos attack," tech. rep., Secure64, 2014.

[10] N. Blenn, V. Ghiette, and C. Doerr, "Quantifying the spectrum of denial-of-service attacks through internet backscatter," in *International Conference on Availability, Reliability and Security (ARES)*, 2017.