Learning-Based Detection of Malicious Hosts by Analyzing Non-Existent DNS Responses

Jawad Ahmed*,†, Hassan Habibi Gharakheili*, and Vijay Sivaraman*
*Electrical Engineering and Telecommunications, University of New South Wales, Sydney, Australia.
†The UNSW Institute for Cyber Security (IFCYBER), Sydney, Australia.
Emails: {j.ahmed, h.habibi, vijay}@unsw.edu.au

***Abstract*—DNS Water Torture attack is a type of DDoS attack on authoritative DNS servers and/or open resolvers, whereby the victim is bombarded with random non-existent domains (NXDs) DNS requests, exhausting their entire resources. A famous example of this attack was launched by Mirai botnet on Dyn DNS architecture in 2016. Researchers have proposed solutions to detect these attacks; however, they predominantly apply static thresholds to the count of NXD responses. This method can result in high false positives and needs to be customized to the traffic pattern of victim DNS servers, making it practically challenging for adoption at the source of potential attacks. This paper aims to detect possibly infected hosts of a university campus network that take part in this specific type of DNS-based attacks. Our contributions are threefold: (1) We analyze 120 days' worth of DNS traffic collected from the border of a large university campus network to draw insights into the characteristics of non-existent domain (NXD) responses from incoming DNS packets. We discuss how malicious NXDs differ from benign ones and highlight two attack scenarios based on their requested domain names; (2) We develop a method using multi-staged iForest models to detect malicious internal hosts based on the attributes of their DNS activity; (3) We evaluate the efficacy of our proposed method by applying it to live DNS data streams in our university campus network. We show how our models can detect infected hosts that generate high-volume and low-volume distributed non-existent DNS queries with more than 99% accuracy of correctly classifying legitimate hosts.**

***Index Terms*—DNS water torture attack, NXDs, DDoS**

I. Introduction

Over the last two decades, there has been tremendous growth in malicious activities exploiting DNS protocol as the number of network devices grows daily. DNS is a mission-critical service but open by design and rarely monitored by the firewall compared to email, FTP, or HTTP. Attack on DNS infrastructure of Dyn (providing DNS services to big companies such as AirBnB, Spotify, and Twitter) caused by a malware known as Mirai 2016 is a famous example to explain water torture attack in which thousands of vulnerable IoT devices took part in sending queries for random domains to the companies whose DNS infrastructure is operated by Dyn [1]. According to the FBI, the attackers used random subdomain attacks to target US-based state-level voter registration and information website in 2020 [2].

DNS works in such a way that if a query is being asked from the DNS authoritative name server or open resolver, it is an obligation on them to answer it even if the query is non-existent in their ecosystem. Non-existent domains are of two types: (a) benign: popular search engines and anti-viruses utilize random-looking domains to convey a one-time signal to their servers known as disposable domains (*e.g.,* `elb.amazonaws.com.cn`, `cloudfront.net`, and `avts.mcafee.com`). Benign domains may also contain typo mistakes. For example, a user accidentally writes "`googel.com`" instead of "`google.com`"; and (b) malicious: launch a type of DDoS attack, DNS water torture attack [3] also known as random subdomain attack by dynamically generating random strings as the prefix of a victim domain. The DNS Water Torture Attack is a type of DDoS attack on DNS servers. This attack affects both authoritative servers and open/recursive resolvers, but mainly it targets the former.

Cyber actors use bots (compromised devices) to send many randomly generated domain names on their victim servers. The queried domain names relate to the primary domain that is governed by its authoritative name server to return the IP address of that particular domain. During the attack, due to the high number of requests, the victim authoritative servers and/or the recursive resolvers may have slow response to the queries being asked or potentially become unavailable. Although the problem has been well understood over the last decade, it is mostly dealt with from the perspective of the victim server by identifying the malicious queries. We see this as an opportunity to detect potentially infected hosts of an enterprise that initiate non-existent queries to the outside world. It is important for enterprises to implement certain cyber hygiene practices that aid in boosting an organization's overall security posture as well as miniating their reputation on the Internet community by preventing their internal hosts from attacking others.

In this paper, we make the following three contributions: (1) We analyze 120 days' worth of DNS traffic collected from the border of a large university campus network to draw insights into the high volume of incoming Non-Existent Domains (NXD) responses and to identify the difference between two scenarios of water torture attack (attack on authoritative DNS server and/or open resolver); (2) Based on the behavioral attributes, we develop a multi-staged iForest model to classify the internal hosts (those receiving benign NX responses versus those taking part in water torture attack); and, (3) We evaluate the efficacy of our proposed approach on live DNS data with an accuracy of over 99% correctly classifying legitimate hosts.

978-1-6654-3540-6/22

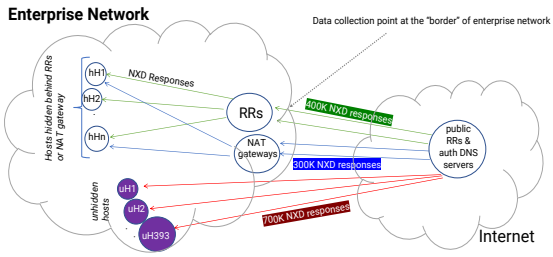


Fig. 1: Visual illustration of our data collection and various entities identified.

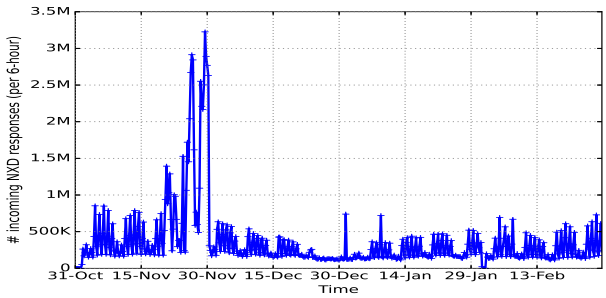


Fig. 2: Timetrace of incoming NX DNS responses.

II. Related Work

DNS traffic has been analyzed to identify malicious network activities [4]–[9]. Over the past decade, there has been an increasing number of works [10]–[13] on detecting malicious network activities mostly related to DNS exfiltration, DNS tunneling, and C&C communications [14]–[18].

Identifying Malicious Queries: To resolve the NXD attacks, specifically water torture attacks, researchers come up with significant countermeasures such as rate limiting and IP address blocking. However, it can severely affect the legitimate users since the malicious queries can forward through the open resolvers or ISP cache servers. Therefore, the IP address of the cache server will be blocked, and the legitimate users will not have access to the ISP cache server. Similarly, if the volume of queries exceeds the limit set in rate-limiting, it will block all queries, even from legitimate users. Researchers [19], [20] have also examined domain names to detect the malicious queries by focusing on NXD error responses only. Kazato et al. [19] predicted whether a domain name included random words via a score calculated by comparing bigrams of domain names of malicious domains and those of benign domain names.

Identifying Attack on DNS Servers: A group of researchers [20], [21] have identified NXD attacks on DNS servers by setting the threshold on the number of non-existent domains. The approaches can be fruitful for heavy volume attacks (such as bursty data) - moreover, choosing a threshold value would be challenging. However, this approach does not work efficiently for the lightweight and distributed NXDs, bypassing the threshold-based security systems.

Researchers have proposed a range of countermeasures to detect NXD attacks; however their proposed methods are too simple (threshold-based) and hence fall short when it comes to sophisticated attacks - with a different objective being the detection of victim primary domains. Instead, we develop a monitoring system at the source that can detect enterprise hosts that generate high volumes of NXD attacks and low and distributed NXD attacks.

III. Analyzing Two Variants of DNS Random Subdomain Attacks in Our Campus Network

In this section, we first analyze the prevalence of NX domains in our campus network. We then look at the two variants of water torture attack - the first case is when the victim is an authoritative name server, and the second case is when the victim is an open/recursive resolver. The study here considers data collected over four months from 31st Oct 2019 to 28th Feb 2020.

Benign incoming responses: Before exploring the anatomy of NXD attacks, we discuss the two possible cases of incoming benign NXDs responses in an enterprise network: (i) Typing mistakes, and (ii) Disposable domains used by antivirus tools (benign data exfiltration).

Illustration of Our Data Collection: Let us understand the anatomy of water torture attack by taking a closer look at our data from a day. As shown in Fig. 1, we show the visual illustration of our data collection for the incoming NX responses. As we collect the DNS data from the border router, the identity of some of the internal hosts gets hidden behind the UNSW recursive resolvers (RRs) and NAT gateways. Hence the scope of this work is limited to those unhidden internal hosts receiving NXD responses. Out of 1.4 M incoming NXDs on this day, 700K responses were destined to 393 hosts. The responses are sourced from either open resolvers and/or authoritative name servers. In what follows, we will highlight how internal enterprise hosts behave during attack scenarios.

A. Attack Scenarios

This section discusses the attack scenarios *i.e.,* attack on the authoritative name server of the victim domain, and an attack on the open resolver. We first plot in Fig. 2 the time trace of incoming NXD responses in our dataset. Each datapoint in this plot represents the number of NXD responses over a 6-hour window. We observe that the typical values of the number of incoming NXDs are less than 500K domains. These domains are mostly benign (either typos or disposable domains). However, some spikes can be seen in the plot highlighting some abnormal activities. We further analyzed those days with spikes in the count of incoming NXD responses and found out the two different scenarios of NXD attacks as described in the following subsection.

1) Attack on Authoritative DNS Servers: Analyzing the spike on 8th January 2020, we found out that most of the incoming NXD responses (350K) are destined to a regular host. Interestingly, the host displays unexpected (suspicious) behavior on 8th January 2020 *i.e.,* we see no NXD activity other than one massive spike within an hour on 8th January.

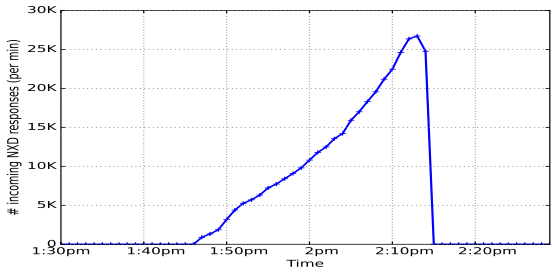


Fig. 3: Zoomed-in time trace of the suspicious host during the hour of interest (8th January 2020 1:30 - 2:30pm).

To better understand the behavior of that host in terms of incoming NXD responses, let us zoom into the host activity on a per-minute basis for those particular hours to see a fine-grained pattern of NXD arrivals (whether spread across hours uniformly or bursty). As shown in Fig. 3, we observe that the number of NXD responses starts to grow linearly at around 1:45pm, peaks at 26K and sharply becomes zero at 2:15pm. Analyzing their query names revealed that they all target a single primary domain name `ahrtv.cn` - this highlights an attack on the corresponding authoritative name server. Some FQDNs found in those queries contain random subdomains *e.g.,* "`3guc.ahrtv.cn`", and some contain dictionary words *e.g.,* "`disposal.ahrtv.cn`" in the subdomain part. In general, we found out that the volume of incoming NXDs from water torture attack is significantly much greater than disposable and benign NXD domains.In total, we found 350,695 NXD responses from "`ahrtv.cn`" out of which 350,581 responses use unique FQDNs (more than 99.99% of FQDNs occurred just once, which is a very unusual behavior).

2) Attack on Open Resolvers: We found out another form of NXD attacks in which an open resolver was the target. After observing unusual spikes during the last week of Nov 2019 (shown in Fig. 2), we focused on 26th Nov 2019 to better analyze the behavior of the involved host as well as top queried FQDNs and primary domains. We found out that an internal host of our campus made unusual queries to Google's public DNS resolver 8.8.8.8 - the query names were "`shu-Aspire-V3-572`" and "`mtrnlab5`" and hence not fully qualified since they did not conform to a standard structure. We observed that the volume of those non-standard queries goes to 400K NXDs per hour, highlighting a very abnormal behavior by this internal host.

B. Drawback of Using a Threshold for NXDs Attack Detection

In this section, we discuss the drawback of using threshold-based attack detection. The first point we want to make here is that various hosts behave differently on a campus network. Therefore, setting a threshold for hosts would be challenging, given the variation in the attack profile. Fig. 4 illustrates the count of incoming NXD responses during a day for three representative infected hosts, at three time granularities: hourly, minutely, and secondly. Let us first look at the hourly count of incoming NXDs across these possibly infected hosts. Fig. 4a depicts the time-trace for the infected H1. The behavior of this host is bursty – only became active at around 1pm, peaking at 250K incoming NXD responses during the day. Comparing H1 with another infected host in Fig. 4b which is super active during the whole day with 300K incoming NXD responses on average. Based on the above two infected hosts, one may choose a threshold of more than 200K NXD responses per hour. However, for our third representative infected host (shown in Fig. 4c), this threshold will not be triggered and this infected host (H3) will go undetected. Also, we test the thresholding method by changing the timescale to a per-minute basis. Fig. 4d provides the time trace of infected host 1 with incoming NXDs from 5K to 26K in a minute and comparing it with infected host 2 which ranges from 4K to 8K per minute. We can set a threshold of incoming NXDs to 5K (any hosts receiving more than 5K NXDs will be flagged as malicious). The infected host 3 will pass from this threshold criteria undetected as in Fig. 4f, we can see that incoming NXDs of infected host 3 range from 100 to 700 NXDs. Similarly, in Figures 4g-4i, we compare the infected hosts on a per-second basis where the infected host 1 and 2 have the peak value of 500 and 600 respectively, whereas the infected host 3 has a peak incoming NXDs count of 70.

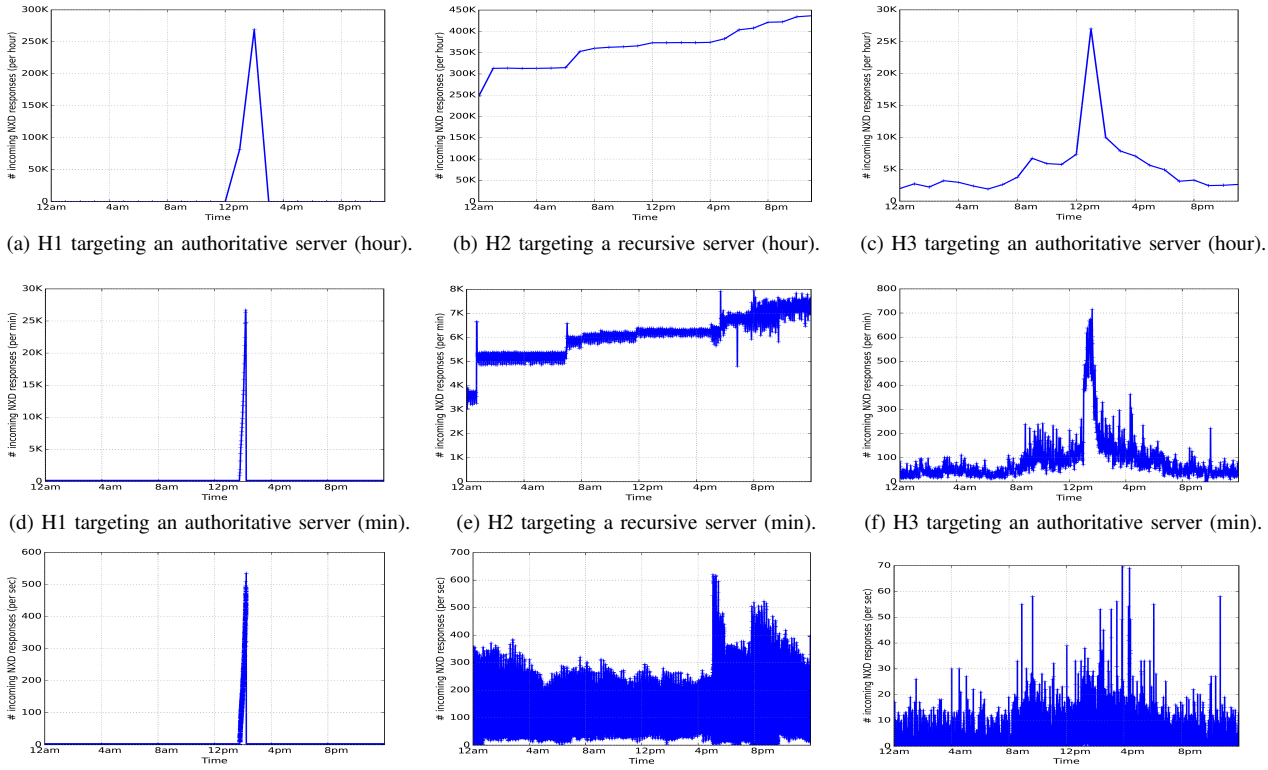
IV. Multi-Staged Machine Learning Architecture

In this section, we present the overall architecture of our method, and then we discuss the details of our multi-stage machine learning algorithm.

A. System Design

Fig. 5 shows the structure of our detection system. An incoming NXD DNS response triggers our system. First, we start monitoring the behavior of the internal host which receives the NXD response. We track the number of NXD responses, timestamp of NXD responses, and FQDNs of individual NXD responses. Upon receiving the NXD response destined to the internal host, we start tracking all the responses to that specific host to get the ratio of NXD responses versus all the other response types. We have devised a new mechanism that uses cascaded machine learning-based models. In stage-1, we use an iForest model to detect whether the incoming NX response is exfiltrated or not (the exfiltrated response represents the benign disposable domains generated by antivirus tools). We design our system to detect volumetric NXD attack as well as distributed NXD attack. We devised two approaches *i.e.,* fine-grained approach and coarse-grained approach. We then process the attributes on a per-second basis (fine-grained) and pass it to the stage 2 iForest model if the model identified that host as benign, we pass it to our coarse-grained model to detect the distributed NXD attack over time. For that, we process the attributes per 30 seconds interval (coarse-grained approach) and pass it to the iForest model to classify the host as malicious or benign.

Our multi-staged ML is based on "Isolation Forest (*iForest*)" [22]. In the first stage, we pass the FQDNs per host to the model that we used to detect DNS exfiltration in [16], [17]. However, our focus here is to detect the non-exfiltrated



(a) H1 targeting an authoritative server (hour). (b) H2 targeting a recursive server (hour). (c) H3 targeting an authoritative server (hour).

(d) H1 targeting an authoritative server (min). (e) H2 targeting a recursive server (min). (f) H3 targeting an authoritative server (min).

(g) H1 targeting an authoritative server (sec). (h) H2 targeting a recursive server (sec). (i) H3 targeting an authoritative server (sec).

Fig. 4: Time-trace of count of NXD responses for various infected hosts at various time granularity (per hour, per min and per sec): (a,d,g) infected H1, (b,e,h) infected H2, and (c,f,i) infected H3.

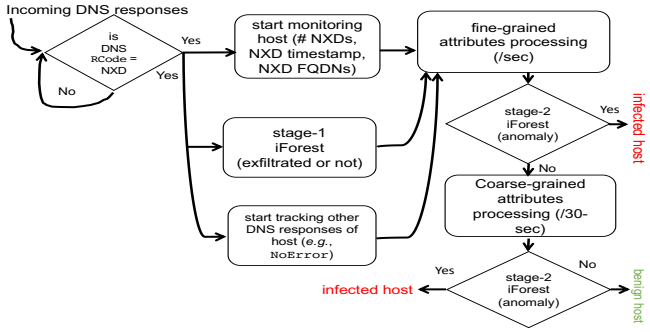


Fig. 5: Overview of our proposed scheme.

domains. The reason to choose non-exfiltrated domains is that the water torture attack queries do not conform to the attributes of exfiltrated domains. We have utilized our previously trained ML model from [16], [17] to extract an attribute *i.e.,* **fraction of non-exfiltrated domains**. We extract eight attributes (from the query name section of each incoming NX DNS response packet) that collectively have strong predictive power in determining whether the query name is exfiltrated or not (output of stage-1). The attributes include: (1) *total count of characters in FQDN*, (2) *total count of characters in sub-domain*, (3) *total count of uppercase characters*, (4) *total count of numerical characters*, (5) *entropy*, (6) *number of labels*, (7) *maximum label length*, and, (8) *average label length*. We compute the fraction of non-exfiltrated domains for each internal host as an attribute for our next stage ML-based model.

For the second stage, we train two iForest models: (a) one is fine-grained for detecting volumetric water torture attacks (in terms of volume of DNS requests), and (b) another is coarse-grained for detecting distributed water torture attacks. We consider the attributes discretely on a per-second basis (fine-grained) for each host, feeding the iForest model. We compute attributes discretely every 30 seconds for each host for the coarse-grained model. By analyzing NXD traffic from our campus network, we consider the following five attributes, including: (1) *ratio of NXD responses to other response types*, (2) *average inter-arrival time between NXD responses*, (3) *standard-deviation of inter-arrival time between NXD responses*, (4) *fraction of non-exfiltrated domains*, and (5) *average number of labels in query names of NXD responses*, for each internal host to infer by fine-grained and coarse-grained iForest models (stage-2).

We will now explain the motivation behind choosing the above attributes for our ML models. The **ratio of NXD response to other response types** is the most important attribute for us to distinguish a malicious host from a benign one. To explain this, let us take a look at Fig. 6 that shows the activity of all hosts of our campus network on a day. The stackplot depicts that the number of No Error responses is significantly higher than that of NXDs, with the peak value of more than 100K responses in a minute. In contrast, the NXD responses are less than 10K for overall hosts. We then plot in

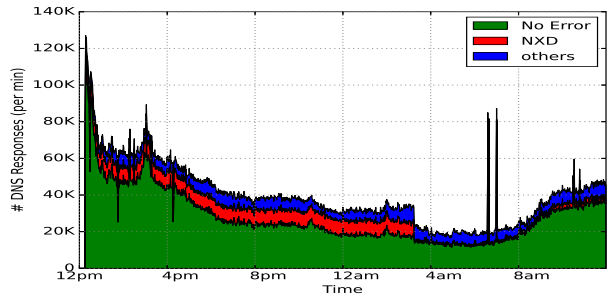


Fig. 6: Stack plot of number of incoming DNS responses in our campus network over a day (26th Nov 2019).

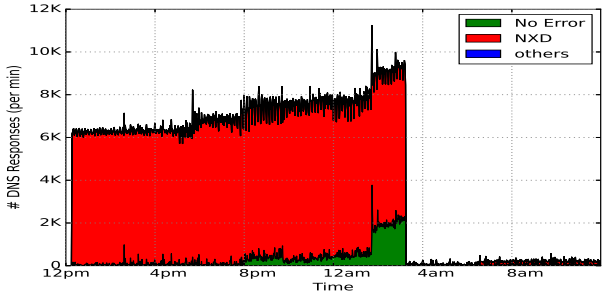


Fig. 7: Stack plot of number of incoming DNS responses of a suspicious host over a day (26th Nov 2019).

Fig. 7 the number of DNS responses (per min) for a suspicious host during that day. We can see that No Error responses peaked at 2K per minute, whereas NXD responses (shown in red) peaked at 6K. Therefore, the ratio of NXD to all the other responses becomes much greater than 1 from 12pm till 3am. We identify **average and standard deviation inter-arrival time between NXD responses** as other two main attributes for classification based on the analysis of benign and suspicious host as discussed in §III where we showed that benign NXDs are distributed in time and do not occur very often over a day as opposed to malicious NXDs involved in a water torture attack. Similarly, **fraction of non-exfiltrated domains** gives the percentage of domains used other than disposable domains and **average number of labels of NXD responses** captures whether a domain is a disposable domain or a water torture attack domain. We note that disposable domains are often long and contain more than 5 or 6 subdomains whereas the random NXD domains typically have one subdomain [23].

B. Model Training

In this subsection, we provide details of our ML model training used in each stage.

Stage-1 Model: For the stage-1 model, we train an iForest model with benign data from four days of our DNS dataset. For ground truth of benign domains, we use Majestic Million [24] that releases a free dataset of top 1M domains and updates it on a daily basis. Majestic ranks sites by the number of subnets linking to that site. For the benign training instances, we only use top 10,000 primary domains. We also include FQDNs for "`sophosxl.net`" domain which is not among the top 10K Majestic dataset. More details of our stage-1 model training and tuning can be found in [17].

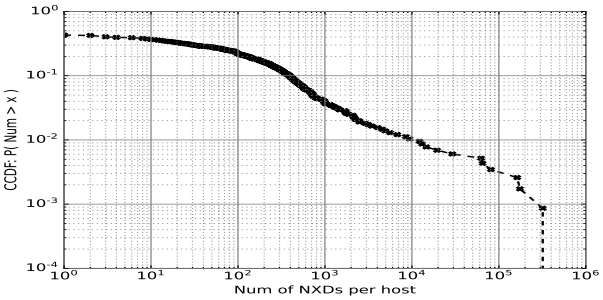


Fig. 8: CCDF of number of occurrences of NXDs per host.

TABLE I: Anomaly detection by fine-grained model.

Input	Output	Days 1-4	Days 5-7
Benign hosts	normal	99.6%	98.5%
	anomalous	0.4%	1.5%
Remaining hosts	normal	94.1%	93.3%
	anomalous	5.1%	6.7%

Stage-2 Model: To train the stage-2 model, we need a dataset of benign NXDs that is quite challenging as there is no public dataset from enterprise hosts. Therefore, we construct our own dataset. To populate benign domains, we start with 4 days' worth of NXD responses from our original DNS dataset. Fig. 8 shows the CCDF of the total count of NXD responses per host. Interestingly, more than 55% of hosts just receive one NXD response. We believed these hosts accidentally mistyped a domain name, resulting in an NXD response. Therefore, we assumed those hosts to be benign. Also, if a host just receives NXD responses for disposable domains (generated by antivirus tools), we assume it as benign too (although the host may or may not involve in other malicious activities, our primary focus is on NXD attack). Note that this threshold value can be configured by the network administrator based on the requirement for their network.

To achieve our objective of detecting NXD attacks at two levels of granularity (fine-grained and coarse-grained), we train two iForest models at stage-2, the first model is trained to detect the heavy volume of NXDs based on attributes computed on a per-second basis. Similarly, the second model is trained with attributes computer on a per 30-sec basis for a distributed NXD attack.

V. Performance Evaluation

In this section, we evaluate the efficacy of our scheme by cross-validating and testing the accuracy of the trained models for benign instances and quantifying their performance on a full campus traffic stream.

Performance of Fine-Grained Model: We evaluate the performance of our fine-grained ML model. Table I shows that benign hosts are classified as normal with an accuracy of more than 98% for both training and testing with false-positive rates of less than 2%. Our objective here is to detect anomalous hosts when the models are applied to remaining hosts. We find that 5.1% of the hosts are classified as anomalous. Further analysis revealed that this model captures all the

TABLE II: Anomaly detection by coarse-grained model.

Input	Output	Days 1-4	Days 5-7
Benign hosts	normal	99.4%	98.1%
	anomalous	0.6%	1.9%
Remaining hosts	normal	90.6%	90.1%
	anomalous	9.4%	9.9%

heavy volume NXD attacks with some false positives. Some of the hosts classified as malicious are NAT gateways and recursive resolvers due to their heavy activity of receiving NXD responses (actual end hosts are hidden behind NAT gateways and resolvers, and hence, the detection of those actual end hosts in this case is beyond the scope of this work).

Performance of Coarse-Grained Model: We next evaluate our coarse-grained model, which was trained by the iForest algorithm. Table II summarizes the results. It can be seen that trained internal hosts are correctly classified as benign with an accuracy of more than 99% during validation. Similarly, the benign hosts are correctly classified as benign with an accuracy of 98% with a false-positive rate of less than 2%. These results are similar to those of the fine-grained model. However, our intent of using a coarse-grained model was to detect distributed NX attacks. We can see here for the remaining hosts, the number of hosts classified as normal is decreased drastically to 90%, whereas the anomalous hosts are nearly 10%. When we analyzed the hosts classified as anomalous, we found out that some hosts were taking part in water torture attack with a very low volume of NXDs, but the requests were distributed in time.

Discussion: To better understand these findings, we have further analyzed hosts that are detected as anomalous in "remaining hosts" category. First, we look at the anomalous hosts for our fine-grained model. In total, we have 45 unique anomalous hosts. By analyzing these IP addresses, we found (by reverse lookup) that 8 of them are NAT gateways (the actual number of hosts are hidden behind these NAT gateways). Other 37 are all regular end hosts coming from 5 different subnets of size /24 of our university campus. Interestingly, out of these five subnets, 18 are from the same subnet, indicating that this particular subnet might be infected by malware.

Comparing these results with those of our coarse-grained model, we found that the anomalous hosts increased to 87, which shows that it also flagged hosts involved in distributed random subdomain attacks. Performing reverse lookup revealed that 11 of these 87 anomalous hosts are indeed NAT gateways, while remaining (76) are regular end-hosts sitting on 9 different subnets of size /24. Of those 76 hosts, 26 fall under a subnet during the entire week. Upon investigation, we found the subnet is the same that our fine-grained model flags. This shows some hosts are possibly involved in high volume random subdomain attack while other involved in distributed low-volume random subdomain attacks.

VI. Conclusion

Enterprise networks are a potential target of cyber-attackers, specifically those exploiting DNS to perform various attacks. We have developed a multi-stage machine learning-based solution to detect NXD attacks. First, by analyzing incoming NXD responses from DNS traffic of our campus network, we highlighted two types of random subdomain attacks on authoritative DNS servers and open resolvers. We developed a method using multi-staged iForest models that analyze host behavioral attributes to identify malicious internal hosts taking part in water torture attacks. Lastly, we evaluated the efficacy of our proposed approach on live DNS data from the network border of a large campus.

References

[1] (2016) Major cyber attack disrupts internet service across Europe and US. Accessed on 28.03.2021. [Online]. Available: https://bit.ly/3fP7LHN

[2] (2020) FBI Outlines Technique Behind DDoS Attacks on US Voter Registration Website. Accessed on 31.03.2021. [Online]. Available: https://bit.ly/3wC0II7

[3] (2014) Water Torture: A Slow Drip DNS DDoS Attack . Accessed on 20.03.2021. [Online]. Available: https://bit.ly/3291xdF

[4] I. Jawad *et al.*, "Identifying DNS Exfiltration based on Lexical Attributes of Query Name," in *Proc. IJCNN*, 2021, pp. 1–7.

[5] M. Lyu *et al.*, "Mapping an Enterprise Network by Analyzing DNS Traffic," in *Proc. Passive and Active Measurement (PAM)*, Puerto Varas, Chile, Mar 2019, pp. 129–144.

[6] M. Antonakakis *et al.*, "Detecting malware domains at the upper dns hierarchy." in *USENIX security symposium*, vol. 11, 2011, pp. 1–16.

[7] S. Schiavoni *et al.*, "Phoenix: DGA-based botnet tracking and intelligence," in *Proc. Springer DIMVA*, 2014, pp. 192–211.

[8] M. Lyu *et al.*, "Hierarchical Anomaly-Based Detection of Distributed DNS Attacks on Enterprise Networks," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 1031–1048, 2021.

[9] J. Ahmed *et al.*, "Automatic Detection of DGA-Enabled Malware Using SDN and Traffic Behavioral Modeling," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 4, pp. 2922–2939, 2022.

[10] S. Hao *et al.*, "An Internet-Wide View into DNS Lookup Patterns," *VeriSign Labs, School of Computer Science, Georgia Tech*, 2010.

[11] L. Bilge, E. Kirda, C. Kruegel, and M. Balduzzi, "Exposure: Finding malicious domains using passive dns analysis." in *Ndss*, 2011.

[12] M. Antonakakis *et al.*, "Building a Dynamic Reputation System for DNS," in *Proc. USENIX Security*, Washington, DC, Aug 2010.

[13] S. Hao *et al.*, "Monitoring the Initial DNS Behavior of Malicious Domains," in *Proc. ACM IMC*, Berlin, Germany, Oct 2011.

[14] C. J. Dietrich *et al.*, "On Botnets that use DNS for Command and Control," in *Proc. IEEE EC2ND, pages=9–16, year=2011,*.

[15] M. Feily *et al.*, "A survey of botnet and botnet detection," in *Proc. ACM ESIST*, Athens, Glyfada, Greece, Jun 2009, pp. 268–273.

[16] J. Ahmed *et al.*, "Real-Time Detection of DNS Exfiltration and Tunneling from Enterprise Networks," in *Proc. Integrated Network and Service Management (IM)*, Washington, DC, USA, Apr 2019, pp. 649–653.

[17] J. Ahmed *et al.*, "Monitoring Enterprise DNS Queries for Detecting Data Exfiltration from Internal Hosts," *IEEE TNSM*, vol. 17, no. 1, pp. 265–279, 2019.

[18] J. Ahmed *et al.*, "A Tool to Detect and Visualize Malicious DNS Queries for Enterprise Networks," in *Proc. IFIP/IEEE IM*, 2019, pp. 729–730.

[19] Y. Kazato *et al.*, "Towards Classification of DNS Erroneous Queries," in *Proc. Asian Internet Engineering Conference*. Chiang Mai, Thailand: Association for Computing Machinery, 2013, p. 25–32.

[20] Y. Takeuchi *et al.*, "Detection of the DNS water torture attack by analyzing features of the subdomain name," *Journal of Information Processing*, vol. 24, no. 5, pp. 793–801, 2016.

[21] R. Alonso *et al.*, "Mining IP to domain name interactions to detect DNS flood attacks on recursive DNS servers," *Sensors*, vol. 16, no. 8, p. 1311, 2016.

[22] F. T. Liu *et al.*, "Isolation forest," in *Proc. IEEE Data Mining*, Pisa, Italy, Dec 2008, pp. 413–422.

[23] X. Luo *et al.*, "A large scale analysis of dns water torture attack," in *Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence*, 2018, pp. 168–173.

[24] T. M. Million. (2018) Top 1 million website in the world. [Online]. Available: http://downloads.majesticseo.com/majestic_million.csv