A Centralized Monitoring Infrastructure for Improving DNS Security

Manos Antonakakis, David Dagon, Xiapu Luo, Roberto Perdisci, Wenke Lee, and Justin Bellmor

{manos, dagon, csxpluo, perdisci, wenke}@cc.gatech.edu, justin@gtisc.gatech.edu
Georgia Institute of Technology, College of Computing,
Atlanta, GA 30332, USA

Abstract. Researchers have recently noted [14, 27] the potential of *fast poisoning* attacks against DNS servers, which allows attackers to easily manipulate records in open recursive DNS resolvers. A vendor-wide upgrade mitigated but did not eliminate this attack. Further, existing DNS protection systems, including bailiwick-checking [12] and IDS-style filtration, do not stop this type of DNS poisoning. We therefore propose Anax, a DNS protection system that detects poisoned records *in cache*.
Our system can observe changes in cached DNS records, and applies machine learning to classify these updates as malicious or benign. We describe our classification features and machine learning model selection process while noting that the proposed approach is easily integrated into existing local network protection systems. To evaluate Anax, we studied cache changes in a geographically diverse set of 300,000 open recursive DNS servers (ORDNSs) over an eight month period. Using hand-verified data as ground truth, evaluation of Anax showed a very low false positive rate (0.6% of all new resource records) and a high detection rate (91.9%).

Keywords: DNS Poisoning, Attack Detection, Local Network Protection

1 Introduction

The Domain Name System, or DNS, maps domain names to IP addresses and other records essential for email, web, and nearly every significant network protocol. DNS security problems in turn affect numerous other services and critical resources. Recently, the security community has identified *fast poisoning* techniques that allow the trivial corruption of DNS records [23, 14]. A poisoning attack allows an adversary to manipulate resolution caches, usually through a "blind" off-path guessing of the transaction components used for DNS message integrity.

Several secure DNS protocols have been proposed, including DNSSEC [6, 7] and DNSCurve [9]. DNSCurve provide link-level security while DNSSEC provide object-based security of DNS messages using cryptographic means. However, the deployment of DNSSEC has proven slow [26], and many hosts have on-path hardware that interferes with DNSSEC's larger packet sizes [8].

The delay in deploying secure DNS motivates the need for local networks to protect their recursive DNS resolution infrastructure. Traditional solutions such as IDS and packet-inspection tools provide limited protections against some classes of attacks, but do not detect DNS poisonings. Indeed, poisoning attacks generally use valid, "RFC-compliant" DNS messages that contain *misleading* answers (e.g., associating a domain with the wrong IP address or nameserver—one under the control of an attacker).

For this reason, DNS security systems are generally concerned with records *in cache* (or in the resolver), as opposed to *in flight* (or on the wire). In this work, we focus on *in cache* detection of DNS poisoning for similar reasons:

1. The in-line inspection of DNS traffic can introduce latency. Some protocols are tolerant of this delay, but for DNS, even adding a few tens of milliseconds delay can have detrimental impact on other services (e.g., VoIP, DNSBL validation, etc.). In extreme cases, adding such delays can result in SERVFAIL responses.
2. Several tools already detect classes of DNS attacks, such as packet format violations (e.g., name pointer loops [4]). These attacks are orthogonal to DNS poisoning, and must be done *on the wire data*, as opposed to the cached data.
3. Some DNS attacks, such as out-of-bailiwick record injection [35], are already rejected by the DNS resolvers themselves. Such attacks are technically DNS poisoning, but have been addressed by RFC 2181 [19] (and related policies) and are routinely dropped by recursive servers. (This is known as "answer validation" in most DNS resolvers [12].) The DNS poisoning attacks we consider are in the newer family of *fast poisoning* or "Kaminsky-class" attacks, which evade these forms of basic RFC 2181 trustworthiness checks. Note that the answer validation phase is usually opaque in a DNS resolver, and server logging of rejected records is often infeasible, mainly due to system performance and volume of the logs.

For these reasons, we focus on the detection of DNS poisoning that is found *in cache*, in order to identify attacks that have evaded all existing layers of protection. To detect DNS poisoning that has evaded all other layers of filtration, we need access to large, busy recursive servers. In practice, such access is difficult to obtain, because of the operational risk it poses to a critical network component, and because of potential privacy concerns in witnessing stub traffic. We therefore decided to use data obtained from the inspection of open recursive caches run by third parties on the Internet. Open recursive resolvers [16] generally permit the inspection of their caches. Since we can successfully detect poisonous Resource Records (RR) in Internet scale measurements, we will be able to do the same when we inspect a less diverse set of recursive DNS servers, e.g., those in a single organization.

We select 300,000 open recursive servers, in order to obtain a diversity of DNS resolvers based on geography, network size, and organizational type (e.g., corporate vs university networks). The network properties of these hosts are discussed in Section 3. Using this data source, we designed and evaluated a large-scale, centralized poisoning detection system called *Anax*. Our implementation of Anax provides a scalable, centralized view of DNS poisoning. Further, it works in an automated manner with minimal human intervention. Anax is able to perform these measurements without being on the same network path as the attacker and victim. During our experiments, Anax was able to successfully detect 319 unique poisoned resource records (RRs) that were subsequently manually verified as DNS poisoning attacks. In addition, because Anax works on arbitrary DNS caches, it can also protect local networks against poisoning even when the local resolver is not open recursive.

Anax relies on a fundamental observation about DNS. Despite being dynamic, DNS records generally direct users to a known, usually stable set of `NS` records. Poisonings on the other hand, generally redirect victims to new, different IP addresses often set up for furtive, short-lived harvesting of information (such as banking credentials, credit card numbers and email passwords). We therefore created detection heuristics that note the statistical DNS properties of answers. Our analysis shows that our features are stable even against significant changes in legitimate DNS hosting.

We operated the Anax poisoning detection system for several months, resulting in a database of tens of millions of DNS answer records. Using extensive classification filters and heuristics we can reliably label the majority of the IPs in recorded RRs. Using manual effort we verified by hand and labeled the remaining 1,264 unique IPs address record as "legitimate" and "poisonous". This labeled data set was then used to train and test our detection module, as described in Section 3. The evaluation of Anax based upon real world data proved so promising that it makes our system an efficient real-time poisoning detection system.

The remainder of this paper is organized as follows. Section 2 provides in-depth technical details of poisoning attacks and related work. Section 3 presents the detection methodology that Anax utilizes. Section 4 details our experiments with Anax, including validation and labeling steps of Anax's dataset. In Section 5 we elaborate on the details of the detection heuristics that Anax uses and present the detection results based on our real-world data analysis. Finally, we conclude in Section 6.

2 Background and Related Work

This section offers a brief overview of the Domain Name System (DNS), addressing aspects relevant to poisoning and detection. Readers familiar with DNS may skip over this section. Further background on DNS can be found in [37].

2.1 Background on DNS Poisoning

DNS provides a distributed database of domain names organized as a tree structure. A domain name is a node in the tree and is labeled with the minimum path used to reach the node from the root. When expressed as a fully qualified domain name, each node is a label separated by period. A *zone* is a collection of nodes under a common parent. Such collections form a subtree, the top of which is called the *start of authority*. Authority DNS servers answer queries about nodes in their zones, and generally provide answers about mappings of *leaf nodes* (or terminus nodes), or a referral to another sibling authority when sub-zones have been delegated to another authority server. The answers from such authority servers are recorded by recursive DNS servers for caching on local networks.

Although DNS poisoning could occur between the stub and forwarder (step one), or the forwarder and resolver (step three), we are primarily concerned with attacks on the path between the resolver and authority (step four in Figure 1). This path is by necessity

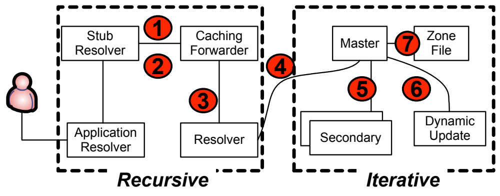

Fig. 1: **An overview of DNS resolution, and risks posed at each phase of the resolution path. DNS poisoning is most commonly concerned only with risks experienced on step four, the communication between resolvers and authorities.**

exposed to the Internet. Since DNS responses are (with noted exceptions [36]) usually a single UDP packet, attackers can send large numbers of spoofed, malicious answers that are "off-path". By "off-path" we mean that an attacker can spoof a UDP packet, claiming to be the authority for a zone from any point on the Internet. Witnessing such poisoning attacks requires the observer to be "on-path" (e.g., as a transit provider or below/above the resolver). If one is not "on-path", it is often difficult to observe such DNS attacks [15].

The basic properties of traditional and Kaminsky-class DNS poisoning attacks have been extensively studied [15, 16, 27]. The Kaminsky-class of DNS attack greatly speeds up traditional DNS poisoning attacks that have historically been done by changing stub DNS settings [16], shown in step two of Figure 1. This increase in the attack speed due to Kaminsky class of poisoning can be achieved by repeatedly attempting to poison "new" nonce names in a zone of interest. Even a bandwidth limited attacker will eventually win the packet race for one of the nonce child names [14], allowing for replacement of the `NS-type` of record in cache. Recent industry studies have noted that DNS manipulations are not only used for phishing, but commonly used for "click-fraud" and by spammers to drive traffic to malicious sites [32], as well.

2.2 Related work

Our work combines ideas from two areas of literature: DNS cache poisoning detection and Internet-wide DNS-based measurement. While Anax is the first system to detect Kaminsky-style DNS cache poisoning, it owes much to previous related research.

DNS cache poisoning is not a new phenomenon. Cache poisoning has been a known vulnerability in DNS since at least 1993 [33], and has seen a resurgence issue in 1997 [35], 2002 [10], 2007 [25], and 2008 [23]. Despite many years of research in eliminating cache poisoning, the latest attack was judged serious enough to warrant multi-vendor coordinated patching [3].

Several vulnerability assessment tools and technologies allow the discovery of DNS vulnerabilities often caused by misconfiguration. Nessus [1] and specific DNS related tools such as DNSStuff [17] and PorkBind [11], detect DNS servers vulnerable to specific cache poisoning attacks. In contrast, Anax detects actual cache poisoning instead of vulnerabilities.

No available tool exists to detect actual *in-cache* poisoning. DoX [41] would use a peer-to-peer network to detect cache poisoning, but it has never been tested in practice nor deployed on the Internet, and this system would require a significant infrastructure and the cooperation of other DoX nodes to be effective. In contrast to DoX, Anax is a centralized system, does not require any external cooperation, and has been tested on real world network scenarios.

Several solutions, such as DNSSEC [6, 7], DNSCurve [9], 0x20 encoding [15] and WSEC-DNS [27], have been proposed to eliminate cache poisoning vulnerabilities entirely. While these solutions would reduce or eliminate cache poisoning, they require explicit or implicit changes to the DNS protocol, are not widely deployed, or are not likely to find wide-spread adoption in the short term (maybe except DNSSEC).

Internet-wide measurement via DNS has been previously used to estimate delay between two arbitrary hosts in King [21]. Anax's goal is not to measure distances between arbitrary hosts, as King does, but to collect IP information about a set of "domain names of interest" (detailed in Section 3.2) that King does not. Internet-wide DNS poisoning scans have been performed by The Measurement Factory [20], but these scans only investigate parent zone poisoning, to which very few name servers are vulnerable, while Anax can detect Kaminsky-class attacks, to which many currently deployed servers are vulnerable. Anax is also able to detect cache poisoning targeted at a specific resolver or set of resolvers. Wendlandt et ll. [38], proposed "Perspectives", a system that uses multiple hosts to verify a server's public key. Our system has a similar scanning methodology but the scope of the two systems is orthogonal; Anax deals with DNS RR validation within cache, while "Perspectives" reactively validates public keys.

Finally, we note that our work has a superficial similarity to the Notos domain reputation system [5]. Notos, created by many of the same authors of this work, uses machine learning to assign a reputation score to unknown domains according to given trained categories (e.g., spam-related domains, botnet domains). In contrast, the present study uses a very limited set of features to identify poisonous DNS records. While Notos allows one to identify groups of similar domains, Anax lets one judge the integrity of selected *in-cache* records.

3 Methodology

In this section we describe the methods that Anax uses to detect cache poisoning. We start with a discussion of the features inherent to cache poisoning attacks, in particular how poisoning attacks may be detected by observing changes in records cached by open-recursive DNS server (ORDNS).

Figure 2 shows the overview of the Anax poisoning detection system. In step one the scanning engine sends to the scanning host a list of domain names and ORDNS servers. The raw DNS answers from scanning (step two) are stored in the raw DNS data collector. A one-time training step labels and verifies a portion of these records (step three). After manually labeling the dataset, we send it to the detection engine for modeling (step four). The resulting models around the benign and poisonous classes of

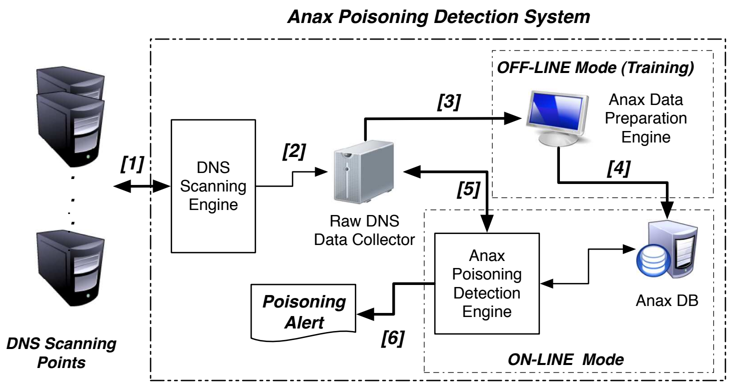

Fig. 2: **Anax Poisoning Detection System.**

RRs will be stored in the Anax DB. At this point the system can be directly utilized (step five) to classify new unknown RRs in DNS answers as they arrive from the scanning points to the raw DNS data collector. Then, Anax can be switched to an on-line mode, and detect new poisonous records using step six.

3.1 Abnormality in DNS Answers Due to Cache Poisoning

Kaminsky-class attacks have made cache poisoning even easier, especially against unpatched servers or servers that cannot take advantage of full source port randomization due to network configurations like NAT. As noted in section 2, poisoning attacks create inherently *local* impacts, making it hard to observe once you are “off-path” of the resolver.

The consensus of answers observed in the wild can be used to validate the resource records (RRs) presented as valid answers. In practice, there are several nuances to this simple approach. DNS can be used for load balancing, localizing content, and to monetize typographical errors, so query results often vary, even without malicious manipulation. To avoid effects of load balancing and content localization, it is necessary to obtain consensus results based on *network* and *geographic diversity*.

An ORDNS that has been the victim of a cache poisoning attack, will answer “on-path” queries using somehow different IP(s) in these RRs (Table 1, lower sub-table) or NS(s) that cannot be correlated with the domain name we try to resolve (Table 1, upper sub-table). Usually these IP(s) point to a different, attacker controlled server. The answer for the poisoned record should inevitably contain at least a single different IP than the IPs found in legitimate RRs for the same domain name. As noted in Section 2, the only possible way to observe this variation in answers is to be “on-path” with the ORDNS. In other words, one needs to be able to directly query the resolver for the poisoned RR. Since we did not have access to customer transit data for this study, we generated such data by utilizing two DNS scanning points: one located in California

Domain Name	NS	CC	Date	ORDNS
amazon.com	hu-bud02a-dhcp09-main.chello.hu	HU	2009-07-26	Cisco CNR
americanexpress.com	c.exam-ple.com	PA	2009-03-20	BIND 9.2.3
americanexpress.com	d.exam-ple.com	PA	2009-05-05	Win DNS NT4
bankofamerica.com	209.59.194.246	US	2009-06-18	Win DNS 2003
bankofamerica.com	209.59.195.246	US	2009-06-18	Win DNS 2003

Domain Name	IPs	CC	Owner	ORDNS
americanexpress.com	189.38.88.129	BR	CYBERWEB	BIND 9.2.3
google.com	85.10.198.253	DE	HETZNER-AS	Win DNS 2000
visa.com	61.207.9.4	JP	OCN NTT	BIND 9.2.0
update.microsoft.com	205.178.145.65	US	Net. Sol.	No Match
google.com	65.98.8.192	US	FORTRESSITX	QuickDNS

Table 1: **Poisoning cases observed by Anax. In the upper part of the table we can see NS replacements observed in `NS-type` RRs. In the lower table we can see IPs in `A-type` RRs that were manually labeled as poisoning cases. With the ORDNS column we provide the type of ORDNS software from the poisoned resolver using the fpdns tool.**

and one located in Ottawa. Using these two scanning points we probed a large, geographically and network diverse set of open recursive DNS servers, as discussed below.

To identify Kaminsky-class attacks (`NS-type` record replacements) and simple DNS poisonings (`A-type` record manipulations), Anax relies on an inherent feature of DNS poisoning: namely, that the poisoned ORDNS will report cached RRs that are "abnormal" with respect to zone and the IP address space. We define as an abnormal the RR with an IP that **should not** reside nor can be linked in any way with the poisoned zone's **"network provisioning"** — a network that can be associated with the zone's operator or a major Content Delivery Network (CDN). For example, a poisonous NS record for amazon.com will point hosts to an authoritative name server (ANS) outside of Amazon's typical DNS provisioning address space. In other words, the IP address of the attacker controlled ANS along with the IP address in the poisoned `A-type` records, cannot be linked with Amazon's IP address space or even worse it might be in dynamic address space. This variation in the RRs can be measured externally as long as we can be "on-path" with the ORDNS.

3.2 Probes and Measurements

Anax's poisoning detection works in three discrete phases: *preparation, measurement*, and *analysis*. The preparation phase consists of collecting IP addresses of open-recursive DNS servers located throughout the world, determining which domains could be likely targets of poisoning attacks, and probing open-recursive servers for poisoning detection (DNS Scanning Engine, Figure 2).

During the *measurement* phase, Anax's scanning engine performs a series of queries while recording matching answers. All the resulting raw DNS traffic is placed in a fully indexed database (Raw DNS Data Collector, Figure 2). Finally, in the analysis phase

(Data Labeling and Detection Engine, Figure 2), Anax performs a series of checks on the recorded RRs from all scanned open-recursive servers. Anax will be able to assign a label for each unique RR of a given zone, and decide its legitimacy. The preparation and measurement phases are described below; the analysis phase is described in Section 5.

Preparation The preparation phase of Anax is composed of three parts: the *gathering* of ORDNS servers, the *identification* of domains likely to be poisoned, and the *probing* of each ORDNS server for poisoning detection. ORDNS servers are gathered using the method proposed by Dagon, et al. in [16]. Using this method, we were able to obtain 8,274,341 open-recursive DNS servers distributed throughout the world. Anax also periodically re-checks DNS resolvers to ensure they continue to behave as open-recursive servers. It is very expensive to regularly probe all discovered ORDNS, therefore we sampled a smaller but geographically diverse set of 300,000 ORDNSs. We made hundreds of thousands of DNS queries to a large, geographically and network diverse set of these 300,000 ORDNSs for 131 zones of interest. A small glimpse of the overall ORDNS diversity from our scanning list with regard to the country code (CC), the autonomous systems (AS) and CIDR block can be found in Figure 4.

Since traditional cache poisoning attacks only affect DNS cache entries for a specific domain, poisoning may only be checked on a per-domain basis. To create a list of domains that are likely to be attacked, we combined the top 100 worldwide websites as ranked by Alexa with the world's top 100 e-business websites, yielding 131 unique domains. These 131 domains are globally distributed, focus on a variety of industries, and all have very high visitor counts. To the best of our knowledge, none of these domains are used for malicious operation, and theoretically the domain names and IPs from these sites should not be part of any black list. The amount of financial transactions conducted through these sites also makes them very tempting targets for phishing attacks (as noted by several on-line phishing analysis resources [34]), that potentially could be staged via DNS poisoning. We refer to this list of 131 domains as the "domains of interest".

Measurement Anax uses repeated queries to discover IP address records for the domains of interest. Using the following scanning protocol, Anax maintains `A-type` record information and `NS-type` record information for the domains of interest.

Anax's scanning points issue a series of typical DNS queries like the one presented in Figure 3. These scan points use such queries in order to capture the *on-path* behavior of the ORDNS. A scan point always makes four types of queries to an ORDNS for each of the domains of interest. The type of queries are `A`, `NS`, `MX` and `AAAA`. The main reason for selecting these different types of queries is to discover as many different RRs as possible for each zone without requiring access to the zone itself.

Let us assume that d is a domain of interest. The complete probing protocol we use is the following querying sequence: the **first** query is an `A-type` and for the domain $d_{control}$, which we own and for which we operate the only "legitimate" authority name server (ANS). The `A-type` of record for this domain contains a single IP that we never change. Using this simple technique we can check if the ORDNS server we probe is acting as a real open-recursive, is misconfigured or provides a DNS-tunneling service. The **second** query is an `A-type` and for the domain d. These queries provide Anax

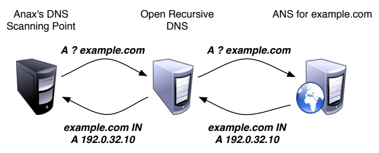
Anax's DNS Scanning Point
Open Recursive DNS
ANS for example.com
A ? example.com
A ? example.com
example.com IN A 192.0.32.10
example.com IN A 192.0.32.10

Fig. 3: **A typical `A-type` query for example.com to an open-recursive server (ORDNS). In this case the ORDNS's cache is empty, and the ORDNS needs to ask the authoritative name server (ANS) of example.com in order to find the IP that is currently "mapped" to the domain name.**

CC	#ORDNS	#ASs	#CIDRs
US	116213	3785	14340
CN	34778	90	2574
JP	20147	329	1760
NL	17651	172	483
FR	16261	164	482
KR	14822	326	1316
IT	12824	204	569
GB	9587	414	952
DE	9441	408	818
SE	9119	113	355

Fig. 4: **A summary of the diverse ORDNS scanning targets.**

with the current IP address of the probed domain d. The **third** is an `A-type` query for $random\_value.d$. Since the random nonce ($random\_value$) does not exist, and the zone does not use wild-card entries, this query ensures that the remote ORDNS always reaches the authority name server (ANS) of d. This results in an answer of NXDOMAIN. We are certain about this since we can trivially verify that none of the zones of interest are wild-carded in the 2^{nd} level domain (2LD). The **fourth** query is an `NS-type` query for d, from which Anax discovers the current IP and domain name information about the authority name servers for the domain name d. Some CDN enabled zones (e.g., bestbuy.com) tend to have their authority name servers operated by the CDN network. We want to capture this diversity in our dataset.The **fifth** query is an `MX-type` query for d, which provides us with the current email servers for the domain name d. Typically this IP location is managed by the same owner of the domain name d. Some zones, however, simply outsource their e-mail services (e.g., AT&T and MessageLabs). The **sixth** is an `AAAA-type` query for the domain name d, based on which we can record again the start of authority record (`SOA-type`) for the domain d.

As noted above, we probe using this sequence of queries so we can obtain key characteristics about the open-recursive server in a single scan event: IP and nameserver information for each domain of interest. We discover as many IP to domain name mappings for each zone as possible due to query type variation (`A, NS, MX, AAAA`). In Figures 5 and 6 we can see a sample of the two observed IP address discovery growth trends that we observe with Anax's scanning engine. Figure 5 shows that domain names that exhibit significant network diversity for IP address present in legitimate `A-type` answers (e.g., blogger.com, amazon.com) or utilize CDN networks (e.g., bestbuy.com uses akamai.net), Anax needs more time to identify all possible IPs addresses. In this case Anax will identify 95% of address records for these network diverse zones in 8-12 days while at the same time continue discovering new IP addresses months after the start of scanning. In Figure 6 domain names with a more stable network profile utilize significantly fewer IP addresses over time. In this case Anax takes less time (95% of address records will be discovered in 2-3 days) to discover almost all of them.

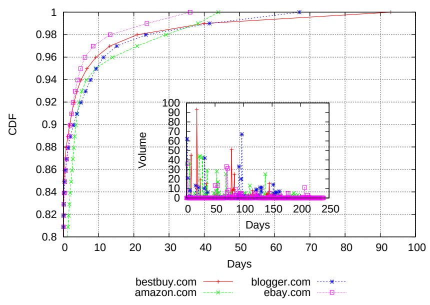

Fig. 5: **IP discovery trend in Anax for zones that use CDN networks or are network diverse.**

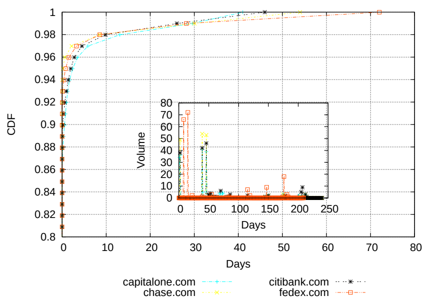

Fig. 6: **IP discovery trend in Anax for network stable zones.**

4 Dataset Evaluation

Using the Anax infrastructure described in Section 3, we periodically made a large number of DNS queries to a set of 300,000 ORDNSs for the 131 zones of interest. The raw DNS collector holds the DNS data generated by Anax's scanning engines. The scanning points periodically synchronize their data to this server for analysis. When the system is in on-line mode (Figure 2; step five) the data can be instantly classified as it arrives in the raw DNS collector. Potentially, the detection engine could be placed directly at the scanning points and classify new RRs on-the-fly.

We evaluated the system in its off-line mode since it was necessary for us to carefully obtain ground truth for our classification process. We used part of the captured traffic to evaluate our detection algorithm. For training our detection system, we used 23 million DNS answers recorded between January 2009 ant the end of February 2009. To create the testing dataset, we used 57 million DNS answers recorded between March 2009 and August 2009.

The raw DNS data gathered by Anax holds all possible observations made about the resource records (RRs) in the received answers for all zones of interest. Our dataset provides "evidence"—that is the RRs ("Domain Name to IP" and "Domain Name to NS server") returned by the ORDNS. A portion of the unique RRs present in these datasets were manually classified to provide the ground truth for our study. At this point we should note that Anax is able to classify `A-type` resource records or "Domain Name to IP" mappings. The collection of the `NS-type` records (or "Domain Name to Name Server" mappings) helped us in the manual classification process and forensic analysis of the hand verified poisoning cases.

4.1 Dataset Labeling

We constructed our limited whitelist by selecting 23 "major" recursive DNS servers across the US. Using a one-time probe against these open-recursive servers, we obtained all address records for the 131 zones of interest. We hand verified that each address found in the returned answers was indeed part of the legitimate domain resolution. After mapping the returned IPs to the corresponding Classless Inter-Domain Routing (CIDR) block, we used this newly created set of CIDRs as our **only** CIDR based whitelist.

During our eight month scanning period, while most of the answers were deemed legitimate, not all illegitimate answers were necessarily poisonous. DNS misconfiguration is a common phenomenon, and sometimes resembles malicious behavior [39, 16]. To account for this, we created several labels for a range of responses: Legitimate Response, CDN, Misconfiguration, NXDOMAIN rewriting, DNS Proxy, and Poisoning, as described below:

Legitimate Response: Legitimate responses indicate a properly functioning ORDNS server returning correct results. The resolutions and authoritative name servers for the list of domain names of interest all point to machines in the same autonomous system among open-recursive servers in the same geographic region, and match prior, verified answers. Our initial whitelist is a small subset of this category.

Content Delivery Networks (CDN): Many zones use DNS to load balance and localize web traffic for popular destinations. This appears where the domain's addresses are assigned to a known content delivery network such as Akamai or Limelight. Network blocks operated by content delivery networks are highly diverse and it is difficult to whitelist all their members. We consulted passive DNS databases (e.g., [22]), to assist on labeling IPs on this category.

Misconfiguration: Some answers showed clear signs of misconfiguration. This is often seen when hosts answer as an authoritative for the root servers or common TLDs such as `.com`, `.net`, or `.org`, or instead return an RFC 1918 or RFC 3330 address. These errant authoritative answers are described in [39] as misconfiguration.

NXDOMAIN rewriting Services: When an IP address was returned for a query that should elicit an NXDOMAIN response, the result was labeled as NXDOMAIN rewriting. These results are not cases of malicious poisoning, and can be detected when an open-recursive DNS server returns an IP address instead of NXDOMAIN for a domain known not to exist. Generally, the resulting IP address points to an advertising portal or a search engine.

DNS proxy: When the ORDNS always provides the same IP address for multiple zone and query types, and at the same time we can identify it as DNS tunnel [2] or a ToTD [31] server, we classify it as DNS proxy. Most tellingly, such resolvers exhibit no IP variations, since they never consult authorities and maintain no cache. Strictly speaking, we do not treat this as DNS poisoning, even though local networks may likely wish to ban the use of DNS proxies.

Poisoning: We **hand-verify** and label as "poisonous" any address returned by an ORDNS that was not owned by the domain name owner, and pointed to a machine under the control of a malicious party. To assist with this labeling, we consulted numerous IP blacklists [30, 24, 13], do-not-route-lists [28], dynamic IP space [29] and passive DNS databases [22]. IPs in RRs that pointed to such hosts indicated malicious poisoning of an ORDNS.

5 Detection Model and Results

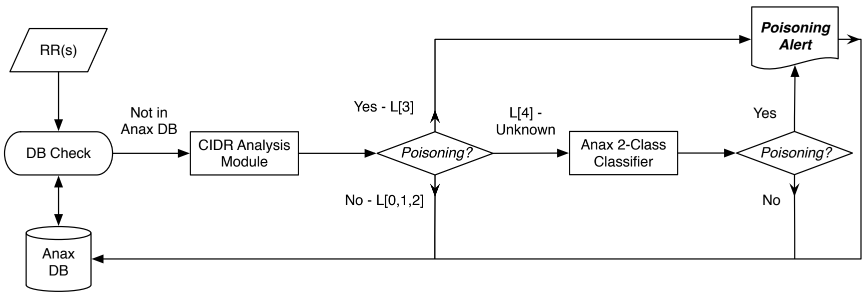

Fig. 7: **Poisoning detection flow in Anax.**

The poison detection flow in Anax consists of detection modules placed in series to reduce false positives and produce as few false negatives as possible. (As noted below, we arranged these detection modules to place the highest false positive rate first, to maximize the final true detection rate) Figure 7 shows the various steps of the detection flow. RRs not in the Anax DB are forwarded to the CIDR analysis module defined in Section 5.2. The L[0]-L[4] categories (defined in Section 5.1) represent the various ways the CIDR analysis module may classify a new RR. The 2-Class classifier (defined in Section 5.3) handles the unknown RRs from the CIDR analysis module (category L[4]) and flags them as benign or poisonous, based on trained models of benign and poisonous RRs. In the case of poisonings, Anax produces a detailed poisoning report on the RR that caused this alert. The detection flow ends by updating the Anax DB on the analyzed RRs. In the following sections we examine in detail the key modules of Anax's detection flow, namely the CIDR analysis module and Anax's 2-class classifier.

5.1 Categories of Resource Records

In Section 4.1 we identified the type of DNS responses we anticipate to receive. Before we elaborate on the details of the CIDR analysis module and 2-Class classifier we introduce the categories of resource records that the modules can handle. Anax groups RRs in the following five categories in order to clearly define the detection actions that each detection module will enforce. These five categories are:

L[0] - Whitelisted RRs: This category is comprised of address records known to be benign, based on our small CIDR-based whitelist.

L[1] - Misconfiguration & "non-routable" IPs: This category is comprised of RRs with IPs that should be considered as misconfigurations since they point to "non-routable" address space. Although interesting, they are not useful for detecting DNS poisoning and are counted as benign when calculating the final detection rates.

L[2] - NXDomain rewriting & Proxies: This category contains two special cases of `A-type` records. The first category contains address records that are meant to produce NXDOMAIN answers according to our probing protocol but did not. Addresses from such NXDOMAIN rewriting services are of no interest as poison, and are deemed benign. The second category is composed of address records from ORDNS acting as DNS tunneling servers. There is no interest in further analyzing these RRs as poison.

L[3] - Poisonous RRs: This category includes address records that reside in any of the following public lists: do-not-route or peer list [28], dynamic IP address space (PBL) [29], hosts reported to drop malware (XBL) [30] or engage in other malicious activity [24]. These RRs will cause the detection algorithm to exit while creating a poisoning alert. The reason why these records will be considered as cases of poisoning attacks is that none of the domains present in our list of "domains of interest" would ever internationally serve malware. Therefore, none of the IPs present in their resource records should ever be in any of these lists.

L[4] - Unknown RRs: This category contains address records from which the CIDR analysis module (defined in Section 5.2) can make no immediate detection decision based on the four previous categories. Finer grained analysis is needed for these RRs. As we will describe in Section 5.3, this can be achieved by computing a six-dimension statistical feature vector.

5.2 CIDR Analysis Module

Address records in RRs that fall into these five categories will initially be used by the CIDR analysis to either create a poisoning report (category L[3]), claim that the RR is not the subject of a poisoning attack (categories L[0]-L[2]) or forward any RR that requires more expensive, fine-grained analysis to the Anax 2-Class classifier (category L[4]). The CIDR analysis module receives RRs, such us `google.com IN 74.125.67.104`, with IP addresses from the monitored zones. Its goal is to make an immediate detection decision about the address record. Based on the categories mentioned in Section 5.1, any IP address within the RR will reside in one of the five categories (L[0]-L[4]).

The primary motivation behind the use of this module is to reduce the overall false positives and to eliminate unnecessary analysis of IPs that fall into the L[0]-L[3] categories. Address records marked as L[0], L[1] and L[2] will cause the detection algorithm to exit without producing a poisoning alert. Simultaneously, the detection algorithm will update the Anax database. For address records that will be placed in the L[3] category by the CIDR analysis module, a poisoning alert will be generated for the corresponding RR. RRs that falls into the L[4] category will be forwarded to the 2-Class classifier, which will make the final detection decision based on statistical models from known benign and poisonous RRs profiles.

5.3 Anax 2-Class Classifier

RRs in category L[4] that cannot be directly checked with our limited white and black listing. Therefore, we use a 2-class K-nearest neighbors (IBK) classifier to make the final detection decision on them. This statistical classifier differentiates between benign and poisonous RRs based on benign and malicious RR profiles built using passive DNS information. Passive DNS data collection is a very common technique that gathers historic DNS resolutions. We use such passive DNS data traces (pDNS) to produce six statistical features for Anax's 2-class classifier.

In order to compute these features, Anax requires a resource record (RR) as an input. An RR of `A-type`, as we already mentioned in previous sections, is composed of a domain name d and an IP d_{ip}. We define $BGP(d_{ip})$ as the set of all IPs in the same BGP prefix of d_{ip}. Each domain name present in our list of "domains of interest" is composed of two parts: the top level domain or TLD (e.g., .com, .org) and the second level domain or 2LD (e.g., ebay, google). We represent every domain name d in our list as $d_{2ld}.d_{tld}$. Using the same logic, when we query the pDNS against an IP, the pDNS will report back to us a list of domain names that are historically linked with this particular IP. We refer to each returned domain name from the pDNS DB as AD. Each returned domain name can also be represented as $AD = ad_{nld}.....ad_{2ld}.ad_{tld}$, assuming that it is a n^{th} level domain.

The set of all **unique** domain names returned from a passive DNS query on d_{ip} is $APDNS_{d_{ip}} = \bigcup_{k=1..m} AD_k$, where m is the number of unique domain names (AD) that historically can be linked with the d_{ip} in the passive DNS database [22]. Also, we define $APDNS_{BGP(d_{ip})} = \bigcup_{k=1..m} AD_k$, where m is the number of unique domain names (AD) that historically can be linked with any IP in the BGP prefix of d_{ip} in the passive DNS DB. Next we define $AD^{3ld.2ld.tld} = ad_{3ld}.ad_{2ld}.ad_{tld}$, $AD^{2ld.tld} = ad_{2ld}.ad_{tld}$ and $AD^{2ld} = ad_{2ld}$.

Now we can define the set $APDNS^{3ld.2ld.tld}_{d_{ip}} = \bigcup_{k=1..m} AD^{3ld.2ld.tld}(k)$ which include all $AD^{3ld.2ld.tld}$ domains (e.g., www.example.com) from all domain names in the set $APDNS_{d_{ip}}$. We also can define the set $APDNS^{2ld.tld}_{d_{ip}} = \bigcup_{k=1..m} AD^{2ld.tld}(k)$ which include all $AD^{2ld.tld}$ domains (e.g., example.com) from all domain names in the set $APDNS_{d_{ip}}$. We define as $APDNS^{2ld}_{d_{ip}} = \bigcup_{k=1..m} AD^{2ld}(k)$ the set of strings which include all AD^{2ld} (e.g., example) from all domain names in the set $APDNS_{d_{ip}}$.

Similarly, we define the two sets $APDNS^{2ld.tld}_{BGP(d_{ip})} = \bigcup_{k=1..m} AD^{2ld.tld}(k)$ and $APDNS^{2ld}_{BGP(d_{ip})} = \bigcup_{k=1..m} AD^{2ld}(k)$ that include all second level domain names ($AD^{2ld.tld}$) and all strings (AD^{2ld}) from all domain names in the set $APDNS_{BGP(d_{ip})}$ respectively.

Finally, we define a list of popular second level domains (2LD) that belong to content delivery networks (CDN) like Akamai, CoralCDN, Limelight and Redcondor. We refer to this list as $ACDN = \bigcup_{k=1..n} cdn_k$, where cdn_k is a distinct **fully qualified** second level domain name (e.g., akamai.net, akamaiedge.net, coralcdn.net). We now elaborate on how we compute the six statistical features based on each newly received resource record:

[Φ_1] - Domain Name Diversity: The number of **unique** domains in the set $APDNS_{d_{ip}}$ that historically have been mapped with the d_{ip} in the RR.

[Φ_2] - 2LD Diversity: The number of **unique** $AD^{2ld.tld}$ present in the set $APDNS^{2ld.tld}_{d_{ip}}$ and have been historically mapped with the d_{ip} in the RR.

[Φ_3] - 3LD Diversity: The number of **unique** $AD^{3ld.2ld.tld}$ present in the set in the set $APDNS_{d_{ip}}^{3ld.2ld.tld}$ and have been historically mapped with the IPs in d_{ip}.

[Φ_4] - Relative BGP CDN Occurrence: The frequency of the $AD^{2ld.tld}$ that historically are present in the set $APDNS_{BGP(d_{ip})}$ and at the same time the $AD^{2ld.tld} \in ACDN$.

[Φ_5] - Relative BGP $\mathbf{d_{2ld}.d_{tld}}$ Occurrence: The frequency of the $d_{2ld}.d_{tld}$ in the set $APDNS_{BGP(d_{ip})}^{2ld.tld}$ that historically have been mapped with any IP present in the set $BGP(d_{ip})$.

[Φ_6] - Relative BGP $\mathbf{d_{2ld}}$ String Occurrence: The frequency of the string d_{2ld} in the set $APDNS_{BGP(d_{ip})}^{2ld}$ that historically have been mapped with any IP present in the $BGP(d_{ip})$.

The statistical features Φ_1, Φ_2 and Φ_3 will provide us with historic DNS information based only on the d_{ip} in the RR. The statistical feature Φ_4 will capture the participation of commonly used CDN second level domains that historically have been mapped with any IP in the same BGP prefix as the d_{ip}. Finally, the statistical features Φ_5 and Φ_6 will capture the participation of all other domain names that point into the same BGP prefix with the d_{ip} and at the same time match with the $2ld.tld$ and the $2ld$ of the domain d.

If the 2-Class classifier labels the IP as poisonous, a poisoning alert will be created for the corresponding RR. Otherwise, it will be marked as benign and it will be added into Anax DB.

5.4 Model Selection and Detection Results

We evaluate the 2-Class classifier in two modes: standalone mode and "in-line" with the CIDR analysis module. In the standalone mode we seed the classifier with any new RRs directly, while in the in-line mode we feed the RRs to the CIDR analysis module and the classifier receives only RRs that belong solely to the L[4] (unknown) category. We evaluated our modules with this process to better justify our decision of assembling the detection flow the way we did. It is straightforward, from an efficiency-minded point of view, that placing the CIDR module in-front of the classifier should lessen the workload on the classifier (since IPs labeled L[0] - L[3] need no further processing). The question we try to answer in this section is the following: will the classifier perform better in in-line or in standalone mode?

We start by carrying out the model selection, a very common technique from the machine learning community. Model selection is used in order to select the optimal machine learning method for solving a given classification problem [18]. We select one classifier for each major family of commonly used classifiers:

I. **Simple Logistic Regression - SLR;** a classifier for building linear logistic regression models.
II. **K-nearest neighbors classifier - IBK;** a "lazy" K-nearest neighbors classifier.
III. **LAD Decision Tree;** a classifier for generating a multi-class alternating decision tree using the LogitBoost strategy.
IV. **Support Vector Machine - SVM;** a SVM based classifier with radial basis function kernel.

	CIDR and Classifier	**Classifier only**
Families	$TP\%$ / $FP\%$ / $Preci.$	$TP\%$ / $FP\%$ / $Preci.$
NBayes (Poi)	94.1% / 63.4% / 15.1%	95.0% / 28.9% / 55.4%
IBK (Poi)	**91.9% / 0.6% / 94.6%**	96.4% / 2.7% / 93.1%
SVM (Poi)	57.0% / 0.9% / 88.6%	81.9% / 5.9% / 83.9%
MLP (Poi)	34.4% / 0.8% / 83.8%	54.2% / 3.7% / 84.8%
LAD (Poi)	73.9% / 3.6% / 70.8%	81.5% / 7.4% / 80.7%

Table 2: **Model Selection for Anax 2-Class Classifier in two modes; standalone and "in-line" with the CIDR analysis module.**

V. **Neural Network - MLP;** a classifier that uses back-propagation to classify instances.

We used several different classifiers, and found that with a 2-Class K-nearest neighbors IBK classifier we obtain the best detection results with $FP_{rate} = 0.6\%$ and $TP_{rate} = 91.9\%$. This is not an unusual phenomenon in machine learning, that a simple classifier like the IBK performs significantly better than more sophisticated and complex classification methods like neural networks [18, 40]. The Receiver Operating Characteristic (ROC) curves for the poison class while using the IBK classifier can be seen in Figure 8.

The reader should note that the $FP_{rate} = 0.6\%$ and $TP_{rate} = 91.9\%$ are **not** packet rates. ROC analysis usually works on rates of detection over network traces, but doing so would unfairly bias the classification results in Anax's favor because the vast majority of the packets are benign. By the definition of the false positive rate (incorrectly classified negatives over total negatives), the number of negatives (or benign packets) is significantly higher than the very sporadic cases of poisoning. Therefore, we decided to instead conservatively calculate the FP_{rate} and the ROC curve based on the unique RRs. In this case, the 0.6% of false positive rate means that for every 1000 unique benign RRs Anax observes for a zone, the poisoning detection system will misclassify six of them as poisonous. To further place the FP_{rate} results into real world context we can look into the domain name "ebay.com", where Anax classified 137 unique RRs over the period of eight months, which means that over an eight month period of time it would misclassify less than a single RR. This indicates that Anax is able to produce low false positive rates due not to the relative volume of the negatives, but due to the accuracy of the 2-Class classifier.

The goal of the 2-Class classifier is to lower the FP_{rate} inherent to the CIDR analysis module due to the limitations of white and black lists. At the same time, we need to keep TP_{rate} as high as possible. We observe that when the modules are "in-line", both the FP_{rate} and TP_{rate} are typically better. The only exception is the case of the Naive Bayes (NBayes) classifier where the TP_{rate} decreases in the "in-line" mode. Unfortunately, NBayes cannot be considered as a candidate for our modeling due to the very high FP_{rate} that exhibits in both modes. The "in-line" mode is typically better since

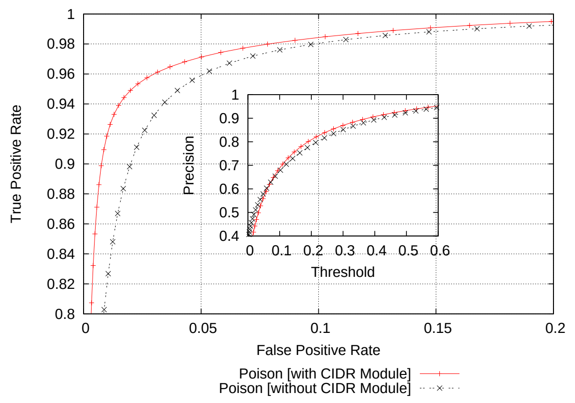

Fig. 8: **The ROC curve for poisoning detection in Anax.**

the majority of RRs escaping the CIDR analysis module will have the following two characteristics.

First, they are not commonly seen in RRs for the monitored zones. Our whitelist will have very small visibility of the whitelisted address space because we do not risk re-probing and re-verifying correct answers from a small set of trusted recursive servers. In general, maintaining a whitelist has proven to be a very inefficient task. Instead, we use the classifier to leverage the task of identifying other whitelisted RRs. This is possible because the classification features we used to compute the statistical vectors from the passive DNS database will place these uncommon legitimate RRs closer to legitimate trained vectors due to the history of the given IP (present in the newly observed RR) within the passive DNS database.

The second category of RRs that will escape the CIDR analysis module will inevitably contain IPs that belong to CDNs and mainly serve news sites. CDNs tend to fluctuate the network addresses that they use to ensure better quality of service to the end-user of the domain. Static whitelisting cannot keep up with these frequently changing addresses so the CIDR analysis module will not be able to whitelist all CDN addresses. Anax successfully addresses this issue in the 2-Class classification module. IPs from CDN networks produce vectors that are very distinct. Such IPs tend to be mapped to a large number of distinct domain names historically. This list of domain names also shows very small diversity in the number of unique 2LDs and large participation of typical domain names (2LDs) directly correlated with CDNs (e.g., akamai.net, cloudfront.net, llnwd.net). A portion of some CDN related vectors will always be present in the training dataset and the classifier will have no problem correctly classifying similar statistical patterns in the testing dataset.

Anax utilizes passive DNS data for computing its statistical features, therefore it is sensitive to the relative passive DNS window (how long are retained passive DNS data) and how the passive DNS data are aggregated. Operators should collect passive DNS data below the resolver in order to protect their database against out-of-Bailiwick

RRs. Furthermore, the utilization of past-CDN IP address space for poisoning could be a significant evasion threat for Anax if the passive DNS window is more than a few weeks. If the window is on the order of several months, then any past-CDN IP address space will still contain past-CDN signal, (considered benign by Anax). This increases the difficulty in identifying poisoning attempts with IPs originating from such addresses.

6 Conclusion

Recently discovered flaws in the DNS protocol require new, innovative techniques to detect poisoning. We have suggested and explored a new area for such research: the detection of DNS poisoning using network observations. We built a system, Anax, that aims to examine the nature of cache poisoning attacks. Anax is able to detect cache poisoning locally and in a fully automated manner.

Leveraging the fact that DNS poisoning is an inherently localized attack, Anax provides useful insights into attacks, based largely on limited whitelisting and statistical IP and domain name metrics. Anax's detection engine shows that these heuristics can be refined, and placed in order to yield a low (RR-based) FP_{rate} (0.6%), high (RR-based) TP_{rate} (91.9%). Our work has focused on "zones of interest" that are historically targets of phishing attacks.

Anax relies on a fundamental observation about DNS: benign DNS records from major zones generally direct users to a known, usually stable set of `NS-type` and `A-type` records. Poisonings on the other hand generally point victims to new IP addresses. Anax utilizes detection heuristics based on historic passive DNS observations and is able to accurately model benign and malicious RRs. The eight month, real world evaluation shows that Anax is an effective and efficient real-time poisoning detection system.

7 Acknowledgments

We thank Robert Edmonds and Paul Royal for their valuable comments. This material is based upon work supported in part by the National Science Foundation under grant no. 0831300, the Department of Homeland Security under contract no. FA8750-08-2-0141, the Office of Naval Research under grants no. N000140710907 and no. N000140911042. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation, the Department of Homeland Security, or the Office of Naval Research.

Bibliography

[1] Nessus: The network vulnerability scanner. `http://www.nessus.org/nessus/`.

[2] OzymanDNS: Kaminsky DNS tunnel. `http://www.doxpara.com`, 2005.

[3] DNS multi vendor patch: CVE-2008-1447. `http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2008-1447`, March 2008.

[4] CERT Advisory. Vulnerability Note VU-23495 - DNS implementations vulnerable to denial-of-service attacks via malformed DNS queries. August 2001.

[5] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster. Building a Dynamic Reputation System for DNS. In *Proceedings of the 19th USENIX Security Symposium*, August 2010.

[6] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. RFC 4033 - DNS Security Introduction and Requirements.

[7] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. RFC 4034 - Resource Records for the DNS Security Extensions. `http://www.ietf.org/rfc/rfc4034.txt`, 2005.

[8] R. Bellis and L. Phifer. Test report: DNSSEC impact on broadband routers and firewalls. `http://download.nominet.org.uk/dnssec-cpe/DNSSEC-CPE-Report.pdf`, 2008.

[9] D. J. Bernstein. Introduction to DNSCurve. `http://dnscurve.org/`, 2008.

[10] Ccais/RNP (Brazilian Research Network CSIRT) and Vagner Sacramento. Vulnerability in the sending requests control of Bind versions 4 and 8 allows DNS spoofing, November 2002.

[11] D. Callaway. PorkBind - Recursive multi-threaded nameserver security scanner. `http://innu.org/~super/#tools`, 2008.

[12] Computer Academic Underground. bailiwicked_domain.rb. `http://www.caughq.org/exploits/CAU-EX-2008-0003.txt`, 2008.

[13] Team Cymru. The Darknet Project. `http://www.team-cymru.org/Services/darknets.html`, 2004.

[14] D. Dagon, M. Antonakakis, K. Day, X. Luo, C. Lee, and W. Lee. Recursive DNS Architectures and Vulnerability Implications. In *Proceedings of the 16th NDSS*, San Diego, CA, 2009.

[15] D. Dagon, M. Antonakakis, P. Vixie, T. Jinmei, and W. Lee. Increased DNS Forgery Resistance Through 0x20-Bit Encoding. In *Proceedings of the 15th ACM CCS*, Alexandria, VA, 2008.

[16] D. Dagon, N. Provos, C. Lee, and W. Lee. Corrupted DNS Resolution Paths: The Rise of a Malicious Resolution Authority. In *Proceedings of 15th NDSS*, San Diego, CA, 2008.

[17] DNSstufff. DNS Network Tools: Network Monitoring and DNS Monitoring. `http://www.dnsstuff.com/`, 2008.

[18] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley-Interscience, 2nd edition, 2000.

[19] R. Elz and R. Bush. http://www.faqs.org/rfcs/rfc2181.html, July 1997.

[20] The Measurement Factory. DNS Survey: Cache Poisoners. `http://dns.measurement-factory.com/surveys/poisoners.html`, 2008.

[21] K. Gummadi, S. Saroiu, and S. Gribble. King: Estimating latency between arbitrary internet end hosts. In *Procceding of the 2nd ACM SIGCOMM IMW*, 2002.

[22] ISC. SIE@ISC. `http://sie.isc.org`.

[23] D. Kaminsky. Black ops 2008: It's the end of the cache as we know it or: "64k should be good enough for anyone". `http://www.doxpara.com/DMK_BO2K8.ppt`, 2008.

[24] Karmasphere. The open reputation network. `https://dnsparse.insec.auckland.ac.nz/dns`, 2006.

[25] A. Klein. BIND 9 DNS Cache Poisoning. `http://www.trusteer.com/files/BIND_9_DNS_Cache_Poisoning.pdf`, 2008.

[26] E. Osterweil, D. Massey, and L. Zhang. Observations from DNSSEC deployment. In *Proceedings of the 3rd NPSec*, 2007.

[27] R. Perdisci, M. Antonakakis, X. Luo, and W. Lee. WSEC DNS: Protecting Recursive DNS Resolvers from Poisoning Attacks. In *Proceedings of DSN-DCCS*, Estoril, Lispon, July 2nd 2009.

[28] The Spamhaus Project. Lasso: The Spamhaus Don't Route Or Peer List. `http://www.spamhaus.org/drop/drop.lasso`, 2008.

[29] The Spamhaus Project. PBL: The Policy Block List. `http://www.spamhaus.org/pbl`, 2008.

[30] The Spamhaus Project. XBL: Exploits block list. `http://www.spamhaus.org/xbl`, 2008.

[31] WIDE Project. The TOTD ('trick or treat daemon') dns proxy. `http://www.vermicelli.pasta.cs.uit.no`, January 2006.

[32] D. Samosseiko. The PARTNERKA - What is it, and why should you care? In *Proceedings of USENIX, Workshop on Hot Topics in Cloud Computing*, 2009.

[33] C. Schuba. Addressing weaknesses in the domain name system protocol. Master's thesis, Purdue University, 1993.

[34] D. Ulevitch. Phishtank: Out of the Net into the Tank. `http://www.phishtank.com/`, 2009.

[35] USDJ. Eugene E. Kashpureff pleaded guilty to unleashing malicious software on the internet. July 1997.

[36] P. Vixie. RFC 2671 - Extension Mechanisms for DNS (EDNS0). `http://www.faqs.org/rfcs/rfc2671.html`, 1999.

[37] P. Vixie. DNS complexity. *ACM Queue*, 5(3), April 2007.

[38] D. Wendlandt, D. Andersen, and A. Perrig. Perspectives: Improving ssh-style host authentication with multi-path probing. In *Proceedings of the Usenix ATC*, June 2008.

[39] D. Wessels. DNS Cache Poisoners Lazy, Stupid, or Evil? `http://www.nanog.org/mtg-0602/pdf/wessels.pdf`, 2002.

[40] I. Witten and E. Frank. *Data mining : practical machine learning tools and techniques*. Morgan Kaufmann series in data management systems. Morgan Kaufman, June 2005.

[41] L. Yuan, K. Kant, P. Mohapatra, and C. Chuah. DoX: A Peer-to-Peer Antidote for DNS Cache Poisoning Attacks. In *ICC '06*, 2006.