The Wolf of Name Street: Hijacking Domains Through Their Nameservers

Thomas Vissers
imec-DistriNet, KU Leuven
thomas.vissers@cs.kuleuven.be

Timothy Barron
Stony Brook University
tbarron@cs.stonybrook.edu

Tom Van Goethem
imec-DistriNet, KU Leuven
tom.vangoethem@cs.kuleuven.be

Wouter Joosen
imec-DistriNet, KU Leuven
wouter.joosen@cs.kuleuven.be

Nick Nikiforakis
Stony Brook University
nick@cs.stonybrook.edu

ABSTRACT

The functionality and security of all domain names are contingent upon their nameservers. When these nameservers, or requests to them, are compromised, all domains that rely on them are affected. In this paper, we study the exploitation of configuration issues (typosquatting and outdated WHOIS records) and hardware errors (bitsquatting) to seize control over nameservers' requests to hijack domains. We perform a large-scale analysis of 10,000 popular nameserver domains, in which we map out existing abuse and vulnerable entities. We confirm the capabilities of these attacks through real-world measurements. Overall, we find that over 12,000 domains are susceptible to near-immediate compromise, while 52.8M domains are being targeted by nameserver bitsquatters that respond with rogue IP addresses. Additionally, we determine that 1.28M domains are at risk of a denial-of-service attack by relying on an outdated nameserver.

CCS CONCEPTS

• **Security and privacy → Network security**; • **Networks → Naming and addressing**;

KEYWORDS

Nameservers; DNS; typosquatting; bitsquatting

1 INTRODUCTION

The Domain Name System (DNS) is one of the most important protocols of today's Internet, seamlessly converting human-readable domain names to machine-routable IP addresses. From a branding perspective, domain names are important because they are essentially the brands which users recognize and interact with. Even though new TLDs are constantly introduced, short and generic domains in the traditional TLDs are still sold for millions of dollars [46]. From a security perspective (the focus of this paper), domain names and their properties are implicitly and explicitly trusted by users and programs alike. Users are constantly instructed to look at the domain name of websites that they visit and inspect the domain part of the sender's email address when they suspect that they have received a malicious email. Websites will send reset-password links to the mail servers listed in a domain's MX records and intrusion detection systems will treat links as less likely to be malicious if they point to long-existing domain names, rather than newly created ones.

In recent years, researchers have identified that attackers will often register old domains that were allowed to expire in order to capitalize on the residual trust of these domains. This trust has been abused to host malware on the domains of old financial institutions [26], masquerade the communication of C&C malware as traffic to and from long-established domains [23], and even hijack entire autonomous systems [32, 33]. In some cases, attackers do not even have to wait for domains to expire. In addition to guessing/stealing a domain owner's registrar password and moving that domain to a new registry [17], researchers have shown that, under the right conditions, attackers could hijack control of live domains by abusing the dangling links to stale IP addresses that arise because of environment idiosyncrasies of public clouds and managed DNS services [6, 7, 25].

In this paper, instead of focusing on individual domain names, we perform the first, large-scale investigation into the "hijackability" of nameservers and, consequently, of all the domain names that trust these nameservers for resolution purposes. More specifically, we focus on exploiting configuration issues and hardware errors to gain control over DNS requests to nameservers.

Targeting the nameserver substantially increases the attacker's potential. As the actual requested domain name remains unaltered in the DNS resolution, extreme stealthy offenses are possible. For instance, invasive MITM attacks enable miscreants to take full control over the victimized domain and its incoming traffic. Furthermore, compromising a nameserver is very efficient, as a single attack targets all domains relying on that nameserver simultaneously.

The main contributions of this study are:

- Through extensive analysis and measurements, we describe and confirm the presence of typosquatting and bitsquatting vulnerabilities, specifically applied to nameservers.
- We identify instances of targeted exploitation of both nameserver squatting attacks. Meanwhile, we find a large corpus of domains that remain vulnerable for immediate exploitation by making just a few registrations.

CCS '17, October 30-November 3, 2017, Dallas, TX, USA

ACM ISBN 978-1-4503-4946-8/17/10…$15.00
https://doi.org/10.1145/3133956.3133988

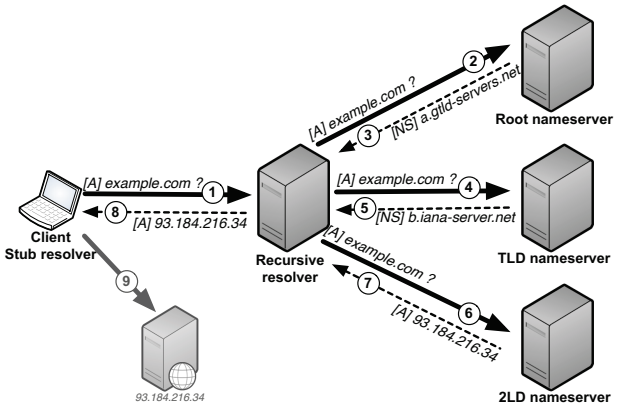

Figure 1: DNS resolution process of a client that connects to host example.com for the first time. Steps 2-7 depict the recursive resolution process. Step 9 illustrates the connection that is made to the IP address after the DNS resolution has taken place.

- We evaluate outdated WHOIS email records of nameserver domains and find several thousand domains at risk of being compromised due to negligence of their nameserver provider.
- We analyze the security practices of widely used nameservers and find that over a million domains are dependent on 8-year-old vulnerable BIND versions.

2 PROBLEM STATEMENT

In this section, we introduce the general concept behind the nameserver hijacking attacks and define the scope of this study. Furthermore, we discuss nameserver dependencies which greatly influences the impact of the presented attacks.

2.1 Hijacking requests to nameservers

When clients want to connect to a certain domain name, this domain name first needs to be resolved into an IP address. This resolution process, shown in Figure 1, will typically be executed by a recursive resolver who will first contact the root and TLD nameservers, and in a last step obtain the IP address from the second-level domain nameserver (2LD nameserver). In our evaluation, we investigate various techniques that allow an adversary to take control over such a 2LD nameserver. As virtually any type of online application or service makes use of DNS, most without realizing it, the potential consequences are widespread. In this section, we provide a brief overview of scenarios that are made possible by exploiting any one of the attacks described in this paper. It is important to note that in this overview, we only consider attacks against the most common software, and therefore the list of described attack scenarios is by no means an exhaustive one.

Man-in-the-Middle (MITM). As soon as an attacker has taken control over a domain's authoritative nameserver, clients wanting to connect to the victimized domain will send requests for the A record to the attacker's nameserver. By replying with an IP address under his control, the adversary can relay and possibly alter the traffic between the client and the domain it intended to contact. The specific consequences of such an attack will largely depend on the type of service that is being accessed. For instance, in the case of a webserver, the attacker can secretly intercept sensitive information, such as credentials and session tokens. In contrast to a MITM attack on a local network, which may only be able to target a limited number of clients, a MITM attack based on hijacking the nameserver affects *all* clients.

Rank	Nameserver domain	Domains
1	domaincontrol.com	39,674,597
2	hichina.com	4,975,760
3	dnspod.net	2,832,233
	⋮	⋮
10,000	ptserve.info	399

Table 1: First and last part of the top NSDOMs

Domain-ownership verification. In the case of a MITM attack, the time period during which the adversary can cause harm to the domain or its clients is limited to the time he has control over the nameserver. In addition to such attacks, an adversary can also perform actions that may have a more long-lasting effect. A number of such actions are related to the proof of ownership for a domain. More precisely, a number of services require (website) administrators to verify they are in fact in control of a domain, e.g. by serving a randomly-generated file at a predefined location. For many Certificate Authorities, including Let's Encrypt [22], such a verification is the only requirement in order to obtain a certificate for a domain. This means that even with only temporary control over a domain's nameserver, an adversary can obtain a certificate, which may be valid for multiple years. Moreover, as the issuance was invoked by the attacker, the domain owner does not have access to the associated private key, and thus cannot revoke the certificate. In addition to issuing SSL certificates, there are many other services that provide domain owners with more permanent access to restricted features. For instance, Google Webmaster Tools gives domain owners exclusive access to a number of features, such as the removal of pages from search results.

E-mail. In addition to the aforementioned attacks, miscreants may also leverage other types of DNS records. For instance, by returning rogue MX records, an adversary can intercept emails destined to the targeted domain. With carefully chosen TXT records, he can spoof e-mail messages from the domain, even in the most secured setups where SPF, DKIM and DMARC records are verified.

2.2 Scope of study

To evaluate the risk of hijacking domains through their nameservers, we focus on the most prominent 2LD nameservers. More specifically, we consider the top 10,000 nameserver domains that are authoritative for the largest number of domain names. To determine this set of nameservers, on December 15, 2016, we obtained the zone files of the top five gTLDs (`com`, `net`, `org`, `xyz` and `info`) with respect to the number of second-level domains present in their zones [15]. For each domain name in each zone file, we extract the `NS` records. Overall, we collect the nameserver information of over 164 million domains.

Next, we derive the *nameserver domain* (NSDOM) for each observed nameserver, e.g, we extract `dnspod.net` from the NS record listing `ns2.dnspod.net`. Then we determine the largest NSDOMs in terms of the number of domains that have NS records pointing to them. Finally, we select the top 10,000 NSDOMs as the starting point of our analyses. An excerpt of this list is shown in Table 1.

2.3 Nameserver dependencies

An important aspect of this study is the dependencies that exist between nameservers. We define a NSDOM as independent when its own NS records are in-bailiwick (e.g. the NS record for `hichina.com` is `ns1.hichina.com`) and thus the TLD nameserver will directly return the IP address of the nameserver with a glue record. In contrast, NSDOMs can be dependent on out-of-bailiwick 2LD nameservers. For instance, when querying `ns1.hostgator.sg`, we find that the NS records of the nameserver point to hosts under `dynect.net`. Since no glue record can be provided for those nameservers, an additional lookup must be made to resolve the nameserver under `dynect.net`. Only thereafter, a resolver can query `ns1.hostgator.sg` to retrieve the DNS records of a certain domain name[1]. In other words, `ns1.hostgator.sg` is completely dependent on another 2LD nameserver, and by extension, all domains relying on `ns1.hostgator.sg` are as well.

This kind of dependency is quite common. In fact, 36.4% of the top 10,000 NSDOMs are dependent on at least one out-of-bailiwick nameserver. To further illustrate these dependencies, Figure 2 maps out the NSDOMs that are dependent on `dynect.net`, the managed nameserver provider that suffered from a massive DDoS attack in October 2016, rendering offline multiple of its high-profile customers [47]. We find that `dynect.net` is the "direct" 2LD nameserver for 191,068 distinct domains. But if we take into account the other NSDOMs that are, at least partially, dependent on `dynect.net`, we come to a total of 9,242,256 domains "indirectly" relying on `dynect.net` (a 48-fold increase). Moreover, `dynect.com` is in turn dependent on a higher level nameserver. Many of these relationships we observe are *full dependencies*, i.e. when a NSDOM is completely and solely dependent on a single external NSDOM. In contrast, other NSDOMs, are only partially relying on others. These nameservers usually employ multiple managed DNS providers to prevent a single point-of-failure.

We find certain instances where long chains of nameserver dependencies emerge. In other words, there are domains that rely on out-of-bailiwick nameservers, who in turn are dependent on other out-of-bailiwick nameservers, and so forth. We call these *nameserver dependency chains*. As an example, some of the chains depicted in Figure 2 go down to 5 levels (the figure only shows up to 4 levels for visibility reasons). Moreover, we find that one NSDOM in our dataset had 8 levels of nameserver dependencies. If any of the nameservers (or the requests) involved in such a dependency chain would be compromised, the requests to all of the dependent 2LD nameservers down the chain would be affected. By extent, the attacker then has the potential of compromising all domains relying on those nameservers as well. A similar observation was made by Ramasubramanian et al. in 2008 [30]. They measured that the resolution of a domain name is, on average, dependent on 46 different nameservers, while only 2.2 of those are directly appointed by the domain owner.

[1]This assumes no caching has taken place. Furthermore, this scenario may be different in terms of glue records when the domains are in the same TLD zone. Additionally, some TLD nameservers reply with non-glue records in the additional section for performance improvement [44].

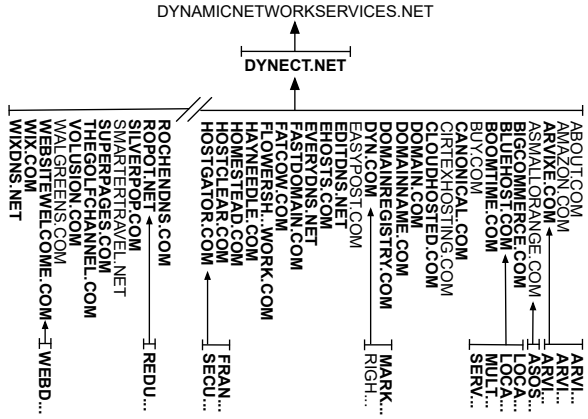

Figure 2: Fragment of the nameserver dependencies related to dynect.net. An arrow symbolizes a dependency on another nameserver. Fully dependent nameservers are marked in bold.

Listing 1: The NS records of polishop.com according to any .com TLD nameserver. All TLD nameservers return this answer.

```
$ dig NS polishop.com @a.gtld-servers.net
...
;; AUTHORITY SECTION:
polishop.com. 172800 IN NS ns-310.awsdns-38.com.
polishop.com. 172800 IN NS ns-1156.awsdns-16.org.
polishop.com. 172800 IN NS ns-1974.awsdns-54.co.uk.
polishop.com. 172800 IN NS ns-566.awsdns-06.ne.
...
```

3 NAMESERVER TYPOSQUATTING

In this section we describe the main idea of hijacking domains via typos in nameservers and present our measurements on the potential and actual abuse of this phenomenon in the wild.

3.1 Attack vector

Typosquatting is the act of registering domain names that are typographical errors of authoritative domains. The malicious actors registering these domains, called typosquatters, attempt to attract accidental visitors that mistype a domain name in their browser's URL bar. As an example, a typosquatter has registered `twittre.com` in the hopes of getting a share of `twitter.com`'s massive amount of traffic.

Typosquatting is a well-studied problem [5, 21, 27, 38, 42], however it has been limited to the scenario where a visitor of a website is making the typographical error in his browser's URL bar. In this paper, we analyze the yet uncharted phenomenon of *nameserver typosquatting*. In this scenario, the administrator of the domain mistypes the NS records while setting up the DNS configuration of the domain which usually happens through a web control panel or API offered by the registrar. To illustrate this, we take the case of `polishop.com`, a popular Brazilian web shop, which has a misconfigured (last verified on May 15, 2017) NS record (Listing 1).

Listing 2: The NS records of polishop.com according to any of the domain's authoritative nameservers. All 2LD nameservers return this answer.

```
$ dig NS polishop.com @ns-310.awsdns-38.com.
...
;; ANSWER SECTION:
polishop.com. 172800 IN NS ns-1156.awsdns-16.org.
polishop.com. 172800 IN NS ns-1974.awsdns-54.co.uk.
polishop.com. 172800 IN NS ns-310.awsdns-38.com.
polishop.com. 172800 IN NS ns-566.awsdns-06.net.
...
```

Listing 3: The A record of polishop.com according to one of the domain's authoritative nameservers

```
$ dig A polishop.com @ns-310.awsdns-38.com.
...
;; ANSWER SECTION:
polishop.com. 300 IN A 54.207.32.165
...
```

The DNS administrator of `polishop.com` mistyped the NS record for `ns-566.awsdns-06.net` while configuring his entries in the registry's zone file through his registrar. More specifically, he missed the last character of `.net` and typed `.ne` instead. Although this record is wrong, the result is still a valid domain name that can be registered and resolved (`.ne` is the ccTLD of Niger). We can verify that this domain is in fact an accidental error by querying the other authoritative nameservers of `polishop.com`. Listing 2 confirms this, as `ns-310.awsdns-38.com` returns the NS record that correctly ends in `.net`. Because of the presence of redundant nameservers, an administrator will likely not notice when a single NS record is broken.

3.2 Amount of traffic affected

In the classic typosquatting scenario, only those visitors that actually make a typographical mistake in their browser are affected. Furthermore, that single mistake impacts that visitor only once. In contrast, the impact of nameserver typosquatting is persistent for as long the misconfigured NS record is present. It is, however, far from trivial to determine the exact amount of traffic an attacker is able to control once he exploits a single misconfigured NS record. We could simplistically assume that the ratio of DNS requests going to the attacker's nameserver is equal to the ratio of nameservers the attacker now controls. In the example of `polishop.com`, this would imply that the attacker sees one-fourth of the DNS requests. This case holds when one of the nameservers is chosen randomly for every uncached request. This happens when either the TLD nameserver randomizes the returned NS records, or when the client's local resolver randomly chooses which nameserver to query. There exist, however, other possibilities [35] including one where local resolvers use the best performing nameserver or query all nameservers in parallel accepting the fastest response. In these scenarios, an attacker can increase his impact by achieving faster response times than the authoritative nameservers. Attackers could attempt to launch a DoS attack on the authoritative nameservers in order to force the clients to use the attacker's nameserver, however, we assume this approach is of limited value since it trades the ability to conduct long-term stealthy attacks for a temporary increase in traffic.

To increase the amount of traffic they can manipulate, attackers can also set a higher TTL value on the rogue DNS records that they return thereby extending their cached lifetime, e.g., as shown in Listing 4 the malicious nameserver sets the TTL of its rogue records to more than ten times higher than the authoritative records (Listing 3). Most administrators favor a short TTL to allow for more rapid adjustments to their infrastructure, however the default maximum cache time accepted by BIND, the most popular DNS software, is 7 days [48]. This allows potential attackers to drastically increase their impact since their rogue records can be cached thousands of times longer than authoritative ones.

Listing 4: The A records of polishop.com according to the attacker's nameserver

```
$ dig A polishop.com @ns-566.awsdns-06.ne
...
;; ANSWER SECTION:
polishop.com. 3600 IN A 185.53.177.31
...
```

It is clear that nameserver typosquatting poses an entirely different, complex, and more invasive threat than the traditional typosquatting attacks. An example that demonstrates this difference is that `polishop.com` nameserver typosquatters are willing to pay over $400 for the price of a single valuable `.ne` domain [2], a price that is 40 times higher than the common gTLDs.

3.3 Potential and current abuse

3.3.1 Dataset. We generated 926,742 typo variations of the top 10,000 NSDOMs and their dependencies using the typo models described by Wang et al. [42]. These models include character omission, permutation, substitution and insertion. The substitutions and insertions are based on the set of characters adjacent to the given character on a QWERTY keyboard. Additionally, there is the missing-dot typo model, where we collected the subdomains present in NS records (e.g. ns1, ns2) and directly concatenated it with the NSDOM. Overall, we find that 95% of these generated typo NSDOMs were available for registration.

3.3.2 Available typos. Of the 882,653 available typosquatting NSDOMs, 2,276 were actively used as nameservers by 6,213 misconfigured domains. Essentially, they are *unexploited* typosquatting NSDOMs, i.e. an attacker can simply register those NSDOMs and instantly compromise affected domains. As shown in Figure 3, registering just 6 typosquatting NSDOMs allows for the immediate compromise of over 2,000 domains, demonstrating the high impact of these attacks. 23 out of 6,213 domains are present within the Alexa top 1 million. Regardless of their Alexa ranking, all of them remain attractive targets for abuse of residual trust [23, 26, 32, 33].

One of the misconfigured domains is `protect-ns.com`. However, this domain serves as a nameserver for other domains as well. Thus, when we take into account nameserver dependencies as described in Section 2.3, an attacker could compromise 682 additional domains that rely on a misconfigured nameserver. Unlike the 6,213 vulnerable domains, these domains have not misconfigured their own NS records but are nevertheless vulnerable due to a mistake by a third party. The indirect nature of this error makes it particularly hard for these domain owners to, not only realize their domains can be hijacked, but also to fix the issue since the error happens at the nameserver which they trust but do not control.

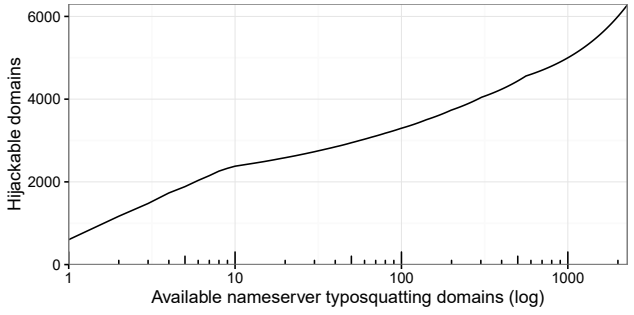

Figure 3: The amount of domains an attacker can hijack by registering a number of available typosquatting NSDOMs.

3.3.3 Registered typos. We separate the 44,089 registered typosquatting NSDOMs into two categories based on whether they appear in NS records. 3,233 (7%) of the registered typosquatting NSDOMs are actively used by domains as a nameserver. These may be *exploited* misconfigured domains or false positives where the registered typo is coincidentally similar to a domain in the top 10K NSDOMs, but is in fact the intended authoritative domain. In Section 3.3.4 we will further investigate to determine how many of these registrations are malicious. The other 40,856 (93%) registered typosquatting NSDOMs were not currently used as a nameserver by a domain in our dataset. These may also be false positives where the similarity to top NSDOMs is a coincidence or a defensive registration, however, they could potentially be *proactive* nameserver typosquatting attacks, i.e. a typosquatting NSDOM is predatorily registered, waiting for a domain to be misconfigured in the future. Given the number of new customers served by some of the largest nameservers, a well chosen proactive registration could pay off in the long term.

3.3.4 Assessing current abuse. To determine whether the registered typosquatting domains mentioned above are truly malicious or false positives we send specific DNS queries to each one and analyze the responses. More specifically, we request the A record for a target domain from both the typosquatting nameserver and the target's authoritative nameserver and compare the responses. The typosquatting nameserver can either reply with a *Rogue* IP (i.e. one that differs from the one given by an authoritative nameserver), a *Matching* IP (the same one given by the authoritative nameserver), or *No Response*. Cases where the authoritative nameserver does not respond, but the typosquatting one does are ignored since we are left without a point of comparison. We argue that a rogue response suggests active abuse.

We further analyze these responses from the *Rogue* category by making an HTTP request to the rogue IP addresses with the Host header set to the target domain, effectively mimicking a user accidentally ending up at the page due to a nameserver misconfiguration. This allows us to categorize the types of abuse used by the malicious nameservers.

Exploitive registrations. We choose a target domain for each of the 3,233 potentially exploitive typosquatting NSDOMs by selecting the highest ranked domain (according to Alexa) among all those configured to use that NSDOM.

To reduce false positives, we conservatively consider only those typosquatting NSDOMs where the target domain has NS records for both the authoritative, as well as the typosquatting NSDOM. Hence, we exclude the cases where the target domain is only configured to use typosquatting NS records. The reasoning here is that a domain would not correctly resolve if all its NS records are erroneous and domain owners would notice the mistake immediately. A possible exception to this would be if an attacker had set up a stealthy proactive typosquatting NSDOM as a recursive resolver to keep newly misconfigured domains fully operational. Nevertheless, we decide to consider these cases as likely false positives. Additionally, this filtering step also ensures that we can compare the responses of a typo and authoritative nameserver during our DNS tests.

There are 86 typosquatting nameservers serving rogue replies as shown in Table 2. These 86 malicious nameservers are capable of hijacking traffic from 423 domains including dependencies. After close inspection we find that 26 of those nameserver typosquatting registrations are all related to the same actor that performs the targeted nameserver hijacking attack on `polishop.com`. These nameservers allowed zone transfers and by probing with selective AXFR queries we find that they solely contained zone files for misconfigured domains, with every domain's A record pointing to the same IP address. This demonstrates that these malicious setups are specifically targeting those domains. When making an HTTP request to this rogue IP, our instrumented browser was shown parking pages (Table 3). Although parking pages are already known to be potentially harmful to end users [41], these can also be a front for dormant malicious activity [24].

The 391 typo domains that did not respond may not be acting maliciously at the time of our resolutions, but there is a clear security risk to the misconfigured domains since they are pointing to a third party that is not their intended authoritative domain. For the 35 nameservers with matching responses, while they appear benign, there is always the potential for attackers to lay dormant, purposefully returning the appropriate IP address, thereby avoiding detection of the hijacked nameserver until a time of their choosing.

Proactive registrations. To test the 40,856 unused typosquatting NSDOMs, we choose the target domain by selecting the highest ranked domain using the squatted authoritative NSDOM from which the typo was derived. While there was no response from most of these domains, among the 3.6% with nameservers that replied, 86% of them served rogue responses for the target domain (Table 2). HTTP requests to the rogue IPs, resulted in a wide variety of observations (Table 3). The most frequent cases were parking, empty, error and scam pages. By looking at WHOIS data, we also encounter one defensive registration though it is unclear whether it was registered to protect the website of the NSDOM, the nameserver itself, or both.

Since the typosquatting NSDOMs in this category are not found in any NS records in our dataset we assume they are not authoritative for any domain, however, 204 actually returned the same IP address as the authoritative domain. Since there is little incentive for a typical nameserver to answer queries for domains outside its zone, opening that server up to DoS attacks, this is suspicious behavior which may indicate the type of stealthy proactive attackers who wait for typos to be made and avoid detection until they

	Rogue	Matching	No Response	Other
Typo (Exploited)	86	35	366	25
Typo (Proactive)	1,295	204	39,218	139
Bitsquatting	522	85	19,141	108

Table 2: Categories of registered typo/bitsquatting NSDOMs based on their responses to target DNS queries.

	Empty page	Defensive	Security Co.	For Sale	Other	Parking page	Scam page	Error page	Redirection
Type (Exploited)	1	-	-	-	-	77	8	-	-
Type (Proactive)	210	1	15	29	7	914	48	64	7
Bitsquatting	72	1	115	21	5	265	5	36	4

Table 3: Web pages returned from the rogue IP addresses.

choose to initiate an attack. We do not expect these to be defensive registrations because it is more likely that a defensive domain would either not respond or delegate to the correct nameserver rather than answering with the correct IP address itself. Finally, while it is possible that some of these typosquatting NSDOMs are used by domains outside of the 5 TLDs in our dataset, we consider it suspicious that they answer correctly for our target domains.

3.4 Measuring vulnerable cases

In order to assess the potential impact of nameserver typosquatting from an attacker's perspective, we registered six nameserver typosquatting domains, as listed in Table 4. We partly anonymize the presented domain names in order to prevent exposure of vulnerable entities. Four of these were known to be unexploited. More specifically, we were aware of 47 domains that were currently misconfigured to use these four NSDOMs. Therefore, we expected to nearly instantly receive DNS requests to these nameservers. We also made two proactive registrations. For these NSDOMs, we had no record of them being used as nameservers in the zone files.

3.4.1 Experimental setup. Our experiment mainly aims to gauge the prevalence of hijack-able DNS resolutions. First, we intend to measure the number of DNS requests that are made to typosquatting NSDOMs. Second, we aim to determine which misconfigured domains names are effectively resolved by contacting our nameserver in error. Meanwhile, we want to minimize the impact of our measurements for the clients resolving those domains.

In order to obtain the necessary data, we adopt a specific setup, as illustrated in Figure 4. To explain this setup, assume we have registered a typosquatting NSDOM, `typo-ns.com`, and there exists a domain name, `misconfigured.com` that has listed `ns.typo-ns.com` in its NS records. Therefore, when a recursive resolver tries to resolve `misconfigured.com`, the `com` TLD nameserver will point the resolver to `ns.typo-ns.com` (1). Instead of simply setting up `ns1.typo-ns.com` with a glue record, we introduce an additional nameserver under our control on a different TLD, namely `ns.m1.xyz`, which we name NS M1. As a consequence, the resolver has to launch a request to NS M1 (3). This effectively creates the nameserver dependency scenario described in Section 2.3. As NS M1 is authoritative for the `typo-ns.com` zone, the resolver is forced to query it to get the IP address of `ns.typo-ns.com` (5), allowing us to

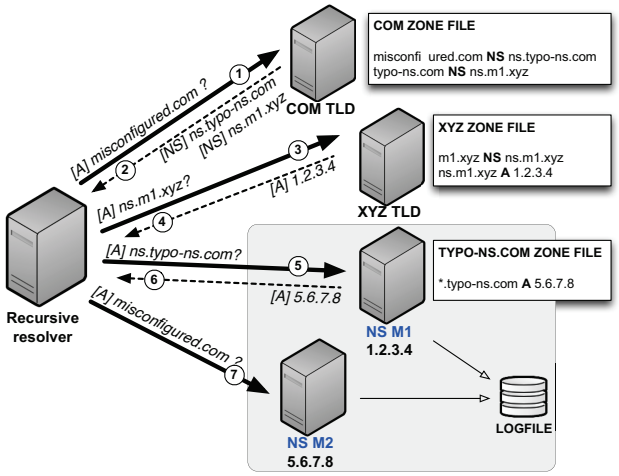

Figure 4: Experimental setup for monitoring the resolutions to typosquatting nameservers. The servers in the gray area are under our control.

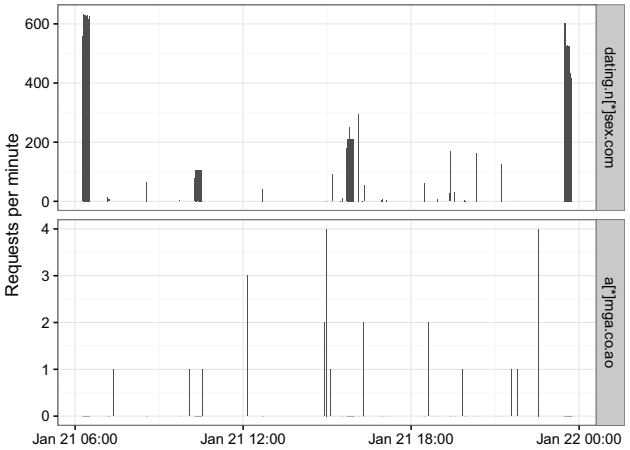

Figure 5: Requests per minute to typosquatting nameserver for two different misconfigured domains.

log that a request for `ns.typo-ns.com` has been made. Afterwards, the resolver finally obtains the IP address of `misconfigured.com`'s nameserver (NS M2) and will subsequently make a request to it (7). At NS M2, we are able to log that a request for `misconfigured.com` is made, completing the log for that resolution.

In order to gather information concerning the clients behind recursive resolvers, we enable ECS (EDNS Client Subnet) [10] on both NS M1 and M2.

Ethical Considerations. To minimize the negative impact of our experiments we set the TTL of records for the domain names we registered to only 5 seconds. We also chose not to respond to requests for domains names we did not control. As a result, the final request to the M2 nameserver for `misconfigured.com`'s IP address will timeout, just as it would have when the typo was unexploited. We used ECS in our experiments to obtain IP information of incoming requests, but this only allowed us to observe the /24 subnet for a small number of queries, maintaining clients' anonymity.

3.4.2 Findings. Over a one month period (Dec 22, 2016 - Jan 22, 2017), we received 734,300 DNS requests on NS M1 for all six registered typosquatting nameservers domains (step 5 in Figure 4). For the "missing-dot" typos (e.g. `ns[*]luehost.com`), there is generally only one nameserver queried, as that typo is specific to a particular subdomain. For the other cases, as shown in Table 4, we find that multiple nameservers on different subdomains are queried for a single typosquatting domain.

We previously determined that there were 47 domains in our dataset that were erroneously using one of our registered typosquatting NSDOMs. On NS M2, we logged resolutions for all of these expected victim domains, confirming that a typosquatting nameserver can effectively compromise all misconfigured domains. More specifically, we logged 3,013,420 "follow-up" DNS requests (step 7 in Figure 4) for those 47 domains, averaging to over 2,000 DNS requests per domain per day. The difference in the number of requests logged at NS M1 and M2 is influenced by the TTL and other factors previously discussed in Section 3.2. Interestingly, one of the two proactive registrations (`domaincon[*].com`) did receive requests, either for domains from different TLDs or for domains that were misconfigured afterwards. Other typo NSDOMs also observed requests for additional domains that suffered from temporal misconfiguration. For example, we recorded 342 queries for `p[*]hex.com` over the course of four days (Jan 18-21) while one of its NS records was mistakenly configured to `ns[*]luehost.com`.

We further record requests for a plethora of services and subdomains. For instance, we received 46 requests for DKIM public keys and 79 requests for DMARC records.

We want to note that the six experimental nameserver typosquatting registrations in this experiment were not chosen to simulate the maximum impact of an attacker, but rather to obtain diverse and representative measurements. An attacker could target more profitable cases, as described in Section 3.3.1.

The most frequently resolved FQDNs for each registered typosquatting NSDOM are shown in Table 5. Based on WHOIS data, at least the five most resolved domains using `ns2.[*]tal.co.uk` are all owned by the same entity. We further analyzed the requests of one of these domains, `dating.n[*]sex.com`, on January 21, 2017, the day we recorded the most queries. Several abnormal characteristics come to light. As displayed in Figure 5, we witnessed several intense bursts of requests lasting for exactly 15 minutes each time. The request rate stays nearly constant during such a burst, but varies from 100 to over 600 requests per minute overall. Moreover, if we look at ECS information supplied by some requests (only 1%), we find that 83% of queries were made from IP address ranges belonging to 9 different hosting and cloud infrastructure companies. In other words, these requests are not coming from human website visitors, but from hosted servers. This kind of automated, coordinated and distributed suggests a misconfigured botnet infrastructure. In contrast, the bottom part of Figure 5 shows the requests pattern of a regular domain that was misconfigured.

Interestingly, the most requested name for `domaincon[*].com` is an inverse address. The typo is present in the zone file for an IP address space managed by AT&T of which the reverse DNS lookups are partially delegated in error to `ns74.domaincon[*].com`. This peculiar case involves different possibilities than a regular DNS query. It would allow an attacker to return false hostnames for IP address owned by another organization, allowing for instance denial-of-service attacks by associating the IP address with blacklisted domain names connected to malware or spam.

Authoritative NSDOM	Typosquatting registration	N° of expected victim domains	Queried subdomains
uniregistrymarket.link	ns[*]niregistrymarket.link	19	-
krystal.co.uk	[*]tal.co.uk	11	ns1, ns2
hostgator.com	ns[*]ostgator.com	16	-
bluehost.com	ns[*]luehost.com	1	-
domaincontrol.com	domaincon[*].com	0	ns50, ns74, ns78
dnspod.net	f1[*]nspod.net	0	-

Table 4: Registered nameserver typosquatting domains and the subdomains that were queried.

Requested name	Typo NS record	Requests
www.o[*]mes.net.	ns2.[*]tal.co.uk	738,581
[*].40.12.in-addr.arpa	ns74.domaincon[*].com	81,964
g[*]ong.com	ns[*]niregistrymarket.link	36,285
a[*]mga.co.ao	ns[*]luehost.com	1,177
p[*]tor.xyz	ns[*]ostgator.com	92
-	f1[*]nspod.net	-

Table 5: The most queried name for each typosquatting nameserver registration during 31 days.

3.5 Summary

In this section we explored the potential exploitation of nameserver typosquatting. We found 6,213 unexploited misconfigured domains available in the wild and showed that a large number of them could be compromised with less than ten typosquatting registrations. 682 additional domains were found to be exploitable not through any fault of their own, but because the nameservers they rely on made typos. 86 currently registered typosquatting NSDOMs actively reply with rogue IP addresses, impacting 423 misconfigured domains. Moreover, we discovered that there exist many more proactive typosquatting registrations with 1,295 of them also responding with rogue IP addresses.

By registering 6 of our own typosquatting NSDOMs we successfully hijacked traffic from 100% of the 47 misconfigured domains pointing to our nameservers, recording more than 3 million DNS requests for those domains over a one-month period. We also found evidence of new temporary misconfigurations during this period, proving that there is value to proactive typo registrations.

4 NAMESERVER BITSQUATTING

The second attack described in this paper, nameserver bitsquatting, is related to the typosquatting attack. However, the main premise of this attack is not human error, but hardware malfunction. As in Section 3, we first describe the attack vector and its impact, followed by an analysis of registered bitsquatting NSDOMs and an experiment to measure bit-flipped DNS resolutions to nameservers.

4.1 Attack vector

Bitsquatting is the act of registering domain names to receive unintentional traffic caused by random bit-flip errors in the memory of devices and computers. These bit-flips occur due to faulty hardware, extreme temperatures or radiation, and thus are by nature rare and unpredictable. However, bitsquatting is a documented

phenomenon and multiple studies have been published reporting on its impact [13, 29], as well as conditions and causes [34, 40]. In DRAM, bit errors are typically mitigated with Error Correcting Codes (ECCs). Although the adoption of these techniques is common, they are still often missing in consumer devices and even in DRAM-containing components of enterprise class systems such as NICs and hard drives [13].

If these bit-flips alter the in-memory representation of a domain name, it can effectively lead to a request to another domain name. For instance, a bit-flip can cause a computer to accidentally connect to `twitte2.com` instead of `twitter.com` (the binary ASCII code for "2" is **0011 0010**, which is a single bit-flip away from **0111 0010**, the ASCII code for "r"). A study by VeriSign [43], reported that about one in every $10^7 - 10^8$ DNS resolutions suffers from a bit-level error.

In previous studies, researchers observed requests to bitsquatting domain names that occurred before, as well as during DNS resolution. However, these studies focussed on bitsquatting connections to a web server's domain name. In this paper, we analyze the possibility of bitsquatted DNS requests to nameservers. NSDOMs are involved in more DNS requests than "regular" domain names (making them statistically more exposed to bit-flips). Furthermore, the impact of nameserver bitsquatting is potentially larger due to cache poisoning. We identify three specific requirements for a bitsquatting nameserver attack to enfold:

(1) The bit-flip must corrupt the domain in a NS record that is or will be accepted by the recursive resolver.
(2) The attack can only occur during a DNS resolution of a domain name whose nameserver is in another TLD zone. When they are in the same TLD zone, the nameserver's IP address is returned immediately via glue records and no actual lookup for the NS records is made.
(3) The bit-flip cannot occur during transmission, since a mismatch between the DNS request and response in the question section will be rejected by the resolver [3].

4.2 Amount of traffic affected

Previously studied bitsquatting attacks, as first described by Dinarburg [13], affect only a single domain name at a time. When a rogue IP address for a domain name is cached, it can affect multiple clients for a prolonged period. Although gauging the probability of bitsquatting vectors is extremely hard, we argue that nameserver bitsquatting could be more prevalent and more impactful than its previously studied counterpart.

First, as NSDOMs are often shared by many domains, NS records are, on a global scale, involved in a lot more DNS requests than a single domain name. Thus, a bit error is in general more likely to corrupt the in-memory representation of a widely-used nameserver than that of a website's domain name.

Second, instead of just poisoning the cache entry of a domain name, the entry of a nameserver can be poisoned. In that case, the attack will affect all domains of that victimized nameserver (for all the clients of the poisoned recursive resolver). However, this is only possible in the dependent nameserver scenario, as presented in Section 2.3. More specifically, as shown in Figure 6, when a second nameserver has to be queried (step 5) to retrieve the IP address of

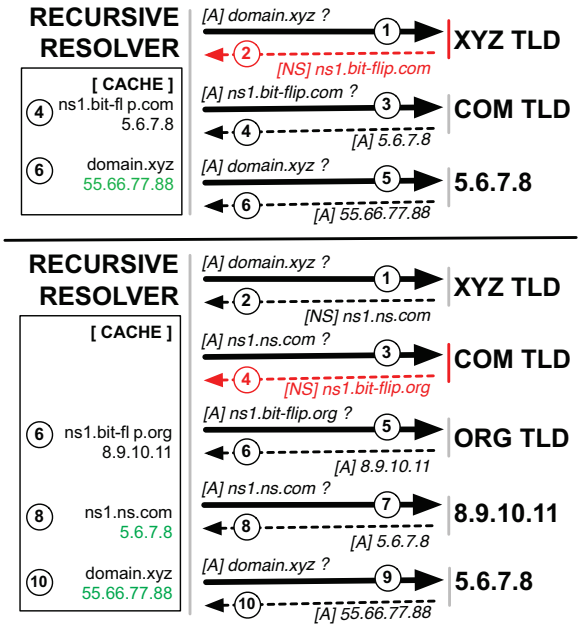

Figure 6: Bit-flip during recursive resolution involving an independent (top) and a dependent nameserver (bottom). Red indicates where bit-flips occurr and green signifies poisoned cache entries.

the dependent nameserver (7), an opportunity arises to poison the cache entry for the dependent nameserver (8).

4.3 Assessing current abuse

4.3.1 Dataset. We generated 605,965 domain bit-flips from the top 10,000 NSDOMs and their dependencies as in the work by Dinaburg [13]. As in Section 3.3, we included the subdomains of the NSDOMS since the first dot (**0010 1110**) may bit-flip to an 'n' (**0110 1110**) creating a new second level domain. 586,109 (97%) of bit-flipped domains were available for registration.

4.3.2 Finding malicious cases. For the 19,856 registered bitsquatting domains we investigate how many of them are malicious bitsquatting domains and how many are false positives. The bitsquatting scenario is similar to the proactive typosquatting in that the NSDOM is not necessarily actively used by any domains, but the attacker is betting that there will be bit-flips which will lead to their NSDOM. Therefore, we use the same methodology as in Section 3.3.4 to test the bitsquatting domains. The results of the DNS queries for the target domains are shown in Table 2. We found the categories are proportionally similar between bitsquatting and proactive typosquatting with 3.1% of domains set up as nameservers and 86% of those nameservers serving rogue IP addresses. There is some overlap of NSDOMs which were both bitsquatting and proactive typosquatting domains, but 433 of the 522 *Rogue* NSDOMs were uniquely bitsquatting names. This indicates that attackers value bitsquatting in addition to typosquatting despite its less predictable

nature. These 522 malicious NSDOMs are capable of capitalizing on potential bit-flips from at least 52,888,224 distinct domains (not taking into account dependencies).

Table 3 shows the results of HTTP requests (with the host header set to the target domain) to the rogue IP addresses served by the malicious bitsquatting NSDOMs. Compared with the same categories for proactive typos, the number of domains associated with a security company stands out. All 115 of these NSDOMs were registered by the same person which is a significant investment in bitsquatting.

As we discussed for proactive typos, it is suspicious behavior for a nameserver to respond with the correct IP if it not listed in any NS records. We find that 48 of the 85 *Matching* bitsquatting NSDOMs do not have any NS records pointing to them and therefore fall into this suspicious category.

4.4 Measuring bit-flip occurrences

We registered ten distinct bitsquatting variations of popular NSDOMs, as listed in Table 6. Nine of these have other nameservers dependent on them, creating an opportunity for cache poisoning the nameserver entry, as described in above.

In order to monitor which bitsquatting variations of nameservers are contacted and log the domains that are being resolved using them, we deploy the same experimental setup that was used for the nameserver typosquatting measurements (Section 3.4), involving two measurement nameservers NS M1 and M2. At NS M1 we receive requests for the bitsquatting nameserver, while at NS M2 we record requests for domains using that nameserver. We evaluate the data for a one-month period (Dec 22, 2016 - Jan 22, 2017).

Ethical Considerations. The same measures that were applied in the experiments of Section 3.3 were used again here to minimize the impact of our experiments. We set the TTL of our responses to only 5 seconds to prevent long term cache poisoning, and we did not respond to requests for domain names we did not own, instead allowing them to timeout as they would in the case of an unexploited bit-flip.

4.4.1 Findings. We witness resolutions for each bitsquatting NSDOM on NS M1, though the vast majority are queries for the second-level domain or common subdomains, such as `mail` or `www`, presumably made by crawlers and DNS scanners. For 3 out of 10 bitsquatting registrations however, we receive requests to very specific subdomains on which nameservers reside on the authoritative NSDOM. For instance, we observed resolvers requesting the A record of `dns9.hi[*].com` and `ns4.p18.dy[*].net`. The authoritative counterpart of those NS records are used by 3,210,418 and 9,658 domains respectively. In total we received 33 requests to specific nameserver subdomains on the bitsquatting NSDOMs over the one-month experiment, averaging to about one per day. For most requests we did not receive a follow-up request on NS M2. We assume that either a correct nameserver was queried in parallel and delivered a faster response than us, or that our response was rejected due to a question section mismatch at the resolver's side.

For three requests, however, we did receive a follow-up DNS request on NS M2 i.e., an attempt to resolve a certain domain name using the bitsquatting nameserver. These observations are shown in Table 7. The first case occurred on December 22, 2016. An IP address of a Pakistani ISP requested two nameserver subdomains of `domain[*].com` . The first is `pdns03`, where its authoritative counterpart is configured as a nameserver by 194,594 domains. We subsequently receive a follow-up request for `odin.g[*]oo.mx`, on NS M2. The domain name `g[*]oo.mx` does indeed have NS records pointing to `pdns03.domaincontrol.com` and `pdns04.domaincontrol.com`, confirming that the resolution was caused by a bit-flip. Concerning the second subdomain that was queried, `pd.304`, we deduce that this is a query for the second nameserver (`pdns04`), but containing two additional bit-flips ("n" to "." and "s" to "3").

The next two cases are very similar to each other and occurred on January 17 and 21, 2017. In both observations, we received a query for a nameserver subdomain of `domain[*].com` made by an IP address of Google's public DNS service. Afterwards, we observed three consecutive queries for a domain name on M2. As we do not respond to these queries, presumably, these are two retries of the same query. Although the source IP address differs for each of these requests, they all belong to the same Google DNS infrastructure located in Singapore [18]. Moreover, the ECS information provided in the initial, as well as the follow-up requests all match up, further confirming that all requests are part of a single DNS resolution. In both cases, the final requested domain names (`u[*]ock.global` and `s[*]ppy.global`) are using the authoritative counterpart of the bitsquatting nameserver.

For all three observations, the requested domain name is on a different TLD than its nameserver, satisfying the criteria for a successful nameserver bitsquatting hijack (Section 4.1). Since we are minimizing the impact of our measurements by not replying to the final requests and setting the TTL of the nameserver to just 5 seconds, we are unable to observe the true impact of cache poisoning.

Authoritative NSDOM	Bitsquatting registration	Dependants
domaincontrol.com	domain[*].com	✓
dynect.net	dy[*].net	✓
hichina.com	hi[*].com	✓
1and1-dns.org	[*]-dns.org	-
ui-dns.org	[*]ns.org	✓
dnsv2.com	d[*].com	✓
dynamicnetworkservices.net	dynamicnetwor[*]s.net	✓
ultradns.org	ult[*].org	✓
verisigndns.com	veri[*]s.com	✓
worldnic.com	[*]nic.com	✓

Table 6: Registered nameserver bitsquatting domains.

Time	From	ECS (Hash)	NS	Requested name	
19:02:11.4	202.[*].[*].33	-	M1	A	pdns03.domain[*].com.
19:02:11.7	202.[*].[*].33	-	M1	A	pd.304.domain[*].com.
19:02:11.9	202.[*].[*].33	-	M2	A	odin.g[*]oo.mx.
06:58:37.1	74.125.190.132	0baf1a2 /24	M1	A	ns34.domain[*].com.
06:58:37.3	74.125.190.147	0baf1a2 /24	M2	MX	u[*]ock.global.
06:58:39.0	74.125.190.145	0baf1a2 /24	M2	MX	u[*]ock.global.
06:58:40.7	74.125.190.12	0baf1a2 /24	M2	MX	u[*]ock.global.
04:03:40.5	74.125.190.141	e814a06 /24	M1	A	ns11.domain[*].com.
04:03:40.7	74.125.190.8	e814a06 /24	M2	A	s[*]ppy.global.
04:03:42.4	74.125.190.16	e814a06 /24	M2	A	s[*]ppy.global.
04:03:44.1	74.125.190.143	e814a06 /24	M2	A	s[*]ppy.global.

Table 7: Observed nameserver bitsquatting occurrences.

4.5 Summary

In this section we investigated the potential of nameserver bit-squatting. We found 522 currently registered bitsquatting NSDOMs responding with rogue IPs with the potential to abuse bit-flips that occur from 52,888,224 domains.

By registering 10 bitsquatting NSDOMs we were able to verify that bit-flipped requests, while rare, do occur. Within one month we observed 3 legitimate bit-flipped requests which would allow for hijacking and cache poisoning of the requested domain name.

5 WHOIS EMAIL HIJACKING

In this section, we introduce the techniques allowing for take-overs of entire NSDOMs by targeting email addresses listed in the WHOIS records, and evaluate their applicability.

5.1 Attack vector

Nameserver domains can be hijacked by abusing out-of-date and inaccurate information in the WHOIS records. The idea is that either access can be gained to the registrar's web control panel, or an ownership transfer of the victim domain name can be issued. Both cases allow an attacker to set up a malicious nameserver using the victim's domain. Consequently, the attacker will be able to hijack all domains dependent on that nameserver. The WHOIS field that is the most ripe for abuse is that of email contacts. Typically, the registrant contact is the person who created the account with the registrar and their email is trusted for retrieving forgotten usernames and resetting forgotten passwords.

An attacker can hijack the email accounts listed in a WHOIS record in two ways. First, some webmail providers will expire an account and make the address available again when a user does not log in for a long period of time. If the email listed in the WHOIS records is an expired webmail account, then the attacker can merely register that address again with the webmail provider. There are known cases of this type of attack. For instance, in 2009, an attacker was able to steal internal documents of Twitter by re-registering an expired Hotmail account as a way of gaining access to a Twitter employee's primary GMail account [11].

Second, if the email account listed in the WHOIS resides on a domain which has been allowed to expire, then an attacker can register that domain name and set up a mail server to receive emails destined for that domain. As soon as attackers control the email address they can initiate a password reset with the registrar and set a new password through the link sent to the stolen email address. If two-factor authentication is not set up, the attacker will gain access and have full control over the nameserver domain.

An attacker can make it more difficult for the original owner to regain control of their domain by transferring it to a different registrar. Once a domain has been transferred away, the original owner is left with little recourse [17]. In order to transfer a domain, an attacker needs to provide an authorization code (also called an EPP code) which is obtained from the original registrar either via a web-accessible control panel or through email from the admin email contact. ICANN requires registrars to respond to such email requests within five days, but the registrar may still force the owner to log in to obtain the auth code. Once the attacker has the auth code, they can provide it to the new registrar to initiate the transfer process. The new registrar will send an email to the admin contact in the WHOIS and expect a response to verify consent to the transfer. Auth codes are required for any TLDs managed by ICANN [19]. ccTLDs (managed by registries in each country and not by ICANN) may have more or less restrictive policies regarding transfers, but `.fr` and `.ca`, the two ccTLDs in our list of vulnerable domains, do require auth codes [4][8].

High Risk		Medium Risk		Low Risk	
scs[*]ver.info	394	fsi[*]ebs.net	461	pul[*]ion.fr	3,642
log[*]rks.net	565	bla[*]sun.ca	5,542	max[*]ech.com	1,912
nic[*]rup.com	1,934	[*].amsterdam	2,594	ube[*]tor.com	2,205
idc[*]com.net	689			web[*]ost.net	546
iqn[*]ion.com	1,019				
par[*]ost.net	1,425				
Affected	6,021		8,596		8,302
Dependents	29		16		112
Total	6,050		8,612		8,414

Table 8: NSDOMs with outdated WHOIS records and the number of domains dependent on them, categorized by their risk of being hijacked.

5.2 Finding vulnerable nameservers

To find nameservers vulnerable to email-based hijacking, we began by obtaining the WHOIS records for the top 10,000 NSDOMs and their dependencies using the Whoxy API [45]. From these records, we extracted the email addresses for the registrant, administrator, technical, and billing contacts. Using the Domainr API [14], we found that 11 of the domains used in these email addresses were available for registration. To find expired webmail accounts we used the Email-Hippo [16] validation API to filter active email addresses. For each email account that Email-Hippo flagged as "undeliverable", we checked whether it was available for re-registration. To that end, we developed a Selenium-based crawler that attempts to create a new email account using, as our address of choice, each of the flagged emails. If a webmail service did not present us with an availability error, that meant that that email address was available for registration. Note that in our experiments we took advantage of the UI present in the registration pages of all modern webmail providers which, through the use of appropriate AJAX calls, provides immediate feedback to the user as to whether the selected email address is available and not taken. As such, we do not need to actually register an email account in order to verify whether it is available. This allows us to ethically quantify the abuse potential of this attack vector without exploiting it and without creating any accounts on webmail providers. We found two such cases of previously existing addresses, both on hotmail.com, which had expired and were available to re-register.

5.3 Potential impact

In total, we found 13 NSDOMs with vulnerable WHOIS emails. We split them into 3 categories based on severity. Table 8 shows the nameserver domains by category. For each nameserver, the number of domains which use it in an NS record is given.

Over 6,000 domains could be impacted by hijacking the six domains in the *High Risk* category. The High Risk category includes all domains where the vulnerable email address was the registrant contact. If an attacker uses the registrant email to gain access to

the registrar's control panel then they have full control over the domain including the ability to change all other email contacts in the WHOIS record.

The *Medium Risk* category includes domains with a vulnerable admin email, but not a vulnerable registrant email. Even if it does not directly grant access to the account, control of the admin email could be used in an attempt to request an auth code from the registrar. Depending on how strict the registrar is about obtaining auth codes, this may require some amount of social engineering. Control of the admin email provides the appearance of authority which would aid such an attempt. Since the admin email is the first point of contact for domain transfers, an attacker could transfer the domain if they are able to obtain an auth code or if they are dealing with registries which do not require auth codes for transfers of particular TLDs.

The *Low Risk* category includes domains with vulnerable emails which are not admin or registrant contacts. It is unlikely that these emails could be used to gain access to the account or transfer the domain. However, there is still some amount of trust that comes along with being listed in a domain's WHOIS. For example, when obtaining an SSL certificate for a domain, certificate authorities, such as StartSSL [37], allow one to prove ownership of the domain using email addresses found in WHOIS. This assumption that the owner of an email in the WHOIS must be the owner of the domain makes any of these emails useful for social engineering. Therefore, even if attackers are not able to altogether hijack these Low-Risk domains, they could certainly request SSL certificates for them and abuse them in MITM scenarios.

Ethical Considerations While we identify vulnerable NSDOMS, we do not register their emails or attempt to compromise any of them. We have reported the WHOIS inaccuracies for the expired emails to ICANN [20] who will forward them to the appropriate registrars.

6 SECURITY PRACTICES OF NAMESERVERS

Following the idea that a domain name's security is entirely jeopardized when (the connection to) the nameserver is compromised, we set out to explore the security risks of the most widely used nameservers. To this end, we evaluated the patching practices of 312,304 nameservers (i.e., all hosts behind the fully-qualified domain names of the top 10K NSDOMs and the parent servers on which they depend), using patching as a proxy variable for a server's overall security. This decision is based on the assumption that a security-conscious administrator will be determined to update the DNS software to a version for which there are no known vulnerabilities.

6.1 Analysis

To determine whether the deployed DNS software is up-to-date, we obtained version information that is being exposed through the banners on port 53, both for TCP as well as UDP. By analyzing these banners, we found that, by far, BIND is the most popular software for DNS servers – out of the 165,012 nameservers for which we received a non-empty banner, 78.33% were using BIND. Because of this uneven distribution of DNS software in the domain name ecosystem, we focus our analysis on the patching practices in BIND.

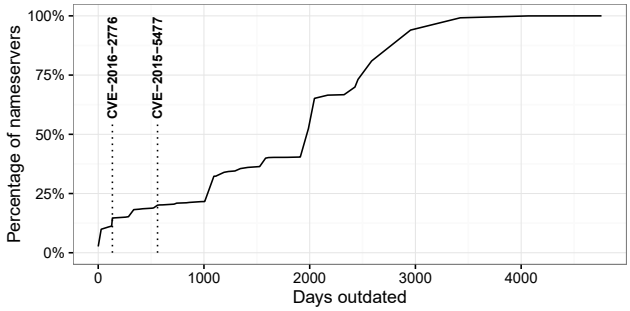

Figure 7: The cumulative distribution of nameservers by the amount of days their BIND version is outdated.

Leveraging the information extracted from the banner, we tried to determine the exact version of BIND that was used. Surprisingly, only 9,032 nameservers (6.99% of all BIND servers) reported version information. Most likely, this is because it is considered a best practice to hide this data from attackers, making it harder for them to determine which exploit they could use. For the servers where we could extract the version information, we determined the release date of the employed installation, along with the number of days it had been outdated. As a point of reference, we used the release date of the latest vulnerability-free versions that were available at the time of our scan (versions 9.9.9-P6, 9.10.4-P6, and 9.11.0-P3). Using this information, we mapped out the distribution of nameservers by the number of days they were outdated, as shown in Figure 7. This graph clearly shows that the vast majority of the nameservers for which we could determine the version are running an outdated version of BIND. More precisely, 7,703 evaluated nameservers are vulnerable to a denial-of-service attack (CVE-2016-2776), for which an exploit is publicly known [39]. Even when being more conservative with regards to considering a version out of date, we still find 7,214 nameservers (79.87% of the BIND servers that returned version information) that are vulnerable to a second denial-of-service attack (CVE-2015-5477), for which an exploit is readily available in the Metasploit framework [31].

Lastly, we want to point out that because nameservers are a common building block typically shared among thousands or even millions of domain names, all these domains are directly affected by the security of their nameservers. The 7,214 nameservers we found to be vulnerable to the DoS exploit in Metasploit, are directly jeopardizing the availability of at least 1.28M unique domain names, out of which 514 operate as nameserver themselves. As a case in point, the nameservers `yns1.yahoo.com` and `yns2.yahoo.com` report to use BIND version 9.4.3-P3, which was released in July 2009, making the software almost 8 years old. Unless the reported version is incorrect – we have no reason to believe so, as this would make the server more likely to attract unwarranted attacks – more than 646,290 domain names are put at risk by having these nameservers as their sole authoritative nameservers.

Ethical Considerations. The choice to obtain nameserver versions by reading their banners provided a non-invasive method to explore their security. This has a minimal impact on the nameservers and avoids the risk of more in depth security tests on live third-party systems.

7 DISCUSSION

Summary of findings. Hijacking domains through their nameservers is an extremely stealthy and powerful attack vector, capable of compromising domains en masse through, among others, MITM, domain-ownership verification and email attacks. In this study, we presented, for the first time, three nameserver attacks based on configuration errors and hardware issues that were evaluated on the top 10,000 nameserver domains.

We found that 6,213 domains can be hijacked, where 2,000 can be compromised with just six targeted registrations. Moreover, we raise the issue of nameserver dependencies and identify that 682 additional domains could be exploited due to a typographical error made by a third party, preventing the victims to directly locate and resolve the issue themselves. Furthermore, by evaluating the possibility of re-registering email addresses present in outdated WHOIS records of nameserver domains, we discovered that at least 6,050 additional domains are at high risk of compromise. In total, we *conservatively* estimate that 12,945 domains are directly or indirectly exposed to being hijacked through a configuration error related to their nameserver. In terms of current exploitation in the wild, we discover that attackers are already aware of these issues and register domains to exploit typos and bit-flip errors in NS records.

Lastly, our study of security practices of nameservers revealed that 7,214 nameservers are susceptible to an 8-year-old exploitable nameserver DoS vulnerability. Thereby, they are exposing 1.28M domains, enabling a large-scale denial-of-service similar to the October 2016 Dyn attack [47] without even requiring a botnet.

DNSSEC. DNSSEC is an extension to DNS which provides integrity to DNS by allowing nameservers to add digital signatures for their resource records and establishing chains of trust from the root zone to the authoritative nameserver. DNSSEC, when deployed properly, is capable of defending against the attacks described in this paper.

We refer the reader to a more complete overview of DNSSEC [9], but for the purposes of this paper the most important component is the DS record which is added to the domain's parent zone. This record tells the DNS resolver to expect signed responses from the next nameserver in the chain and contains a hash of the public key signing key for the next zone which is used to verify the source of the signed responses. When an administrator creates the DS record, they are adding a secondary reference to the correct nameserver beyond the standard NS record. If a victim domain points to a malicious nameserver, regardless of whether it was due to a mistyped NS record, a bit-flip, or stolen control of the nameserver domain, the attacker will be unable to correctly sign its responses. Without a proper signature generated by the key pairs that match the hashed public key in the DS record, a DNSSEC validating resolver will reject any response from the malicious nameserver.

However, in order for a full DNSSEC deployment to work properly there are several requirements involving responsibility and/or cooperation between domain owners, nameserver owners, registries, and ISPs. The complexity of deployment has led to slow adoption despite the age of DNSSEC [12]. For instance, in the `com` zone, only 0.56% of domains are signed at the time of writing [1].

Other defenses. Next to DNSSEC, we suggest the need for additional defenses requiring less cooperation between parties that can be adopted faster than DNSSEC.

To reduce the number of misconfigured domains, registrars can check for typos by comparing all NS records that administrators are entering into the registrar's control panel. A warning could be shown when two records fit one of the typo models proposed by Wang et al. [42], extended with our specific adjustments for NS records (Section 3.3.1). Alternatively, registrars could require administrators to enter new NS records twice, similar to creating a new password. Known typosquatting and bitsquatting defenses, such as large-scale defensive registrations, the use of ECC-enabled DRAM, and filing abuse complaints, are also applicable in the nameserver realm. These kinds of countermeasures are especially interesting for large managed nameserver providers as they are most often victimized and have the means to execute them.

Regarding outdated WHOIS information, we suggest that registrars periodically verify the email addresses listed in the WHOIS records. To prevent validation of stolen email accounts, the verification process should involve the registrant authenticating with the registrar after clicking a link received on the email account. Additionally, we encourage the adoption of two-factor authentication for access to a registrar's control panel.

Finally, we argue that many of the problems discussed in this paper are due to the inconspicuous nature of nameservers. While they are not directly visible to end users and often not even administrators, they do play an extremely crucial and security sensitive role for all Internet services.

8 RELATED WORK

To the best of our knowledge, this work is the first one that investigates the threat of hijacking domain names through nameservers by taking advantage of configuration errors and hardware issues. At the same time, in recent years, the research community has exhibited a rekindled interest in the Domain Name System because of DNS' central involvement in carrying out attacks.

8.1 Hijacking domain names

In 2015, Bryant showed that one could hijack domain names by iteratively requesting public IP addresses from AWS and identifying the domain names that were still pointing to these IP addresses because their owners had once utilized AWS for hosting purposes but had forgotten to update their DNS records after shutting down their virtual machines [6]. Liu et al. showed that these techniques could be abused to attack more public clouds and presented additional cases where websites could be hijacked by dangling DNS records [25]. Even though the authors position their work as capable of identifying all types of dangling DNS records, including dangling nameserver records (the subject of this paper), they were only able to find four confirmed cases of dangling NS records in the Alexa top 1 million list. Contrastingly, in this study, we follow a top-down methodology where we start with popular nameservers (as defined by the number of domains utilizing them for resolutions) and identify not only the domains with dangling records, but also the current name squatting abuse of misconfigured domains. Furthermore, we consider the important role that nameserver dependencies play regarding these issues and highlight the ability to hijack nameserver domains via expired WHOIS email accounts.

In recent work, Bryant identified another type of dangling DNS vulnerability related to managed DNS providers [7] showing that he could hijack control of more than 120K domain names using the managed DNS services of public cloud providers while their owners had stopped using the hosting services of the aforementioned companies. While Bryant's techniques could be straightforwardly incorporated to identify more hijack-able nameservers, we chose to focus on techniques that were hoster-agnostic i.e., techniques that do not rely on the use of specific cloud providers.

8.2 Abusing expired domains

In 2012, Nikiforakis et al. discovered that popular websites contained stale, remote script inclusions that were referring to domains that had expired [28] allowing attackers to register them and deliver malicious JavaScript code. Starov et al. investigated the ecosystem of malicious web shells discovering that some webshells were requesting remote resources from expired domains which allowed researchers (or competing hacking groups) to learn about each new shell deployment and hijack their deployed shells [36].

In 2014, Moore and Clayton investigated the use of old domain names that belonged to US banks and financial institutions and were left to expire after merges or after the companies went out of business [26]. The authors discovered that these domains were often re-registered by attackers who abused the residual trust associated with these domains for SEO activities and malware spreading. Lever et al. analyzed six years of domain data and, among others, discovered that 8.7% of the domains that appear in public blacklists are re-registered after their former owners allow them to expire [23]. Schlamp et al. took the abuse of expired domains even further by showing that attackers can (and already have [32]) hijack entire autonomous systems by re-registering the appropriate expired domains present in the databases of Regional Internet Registrars, such as RIPE and ARIN [33].

9 CONCLUSION

In this paper, we investigated the applicability of issues that are commonly thought of as end-host issues, to nameservers. We found that typosquatting, bitsquatting, and the expiration of email addresses can all be abused to hijack thousands of domain names through their nameserver records. By registering our own typosquatting and bitsquatting domains, we showed how attackers can receive millions of DNS requests by merely registering the appropriate domains. We quantified the thousands of BIND DNS servers that are running outdated software with known vulnerabilities and publicly-available exploits. Lastly we explained why poorly-adopted DNSSEC can defend against most of our described attacks, and suggested pragmatic approaches that registrars could adopt to reduce the likelihood of misconfigurations in the short-term.

ACKNOWLEDGMENTS

We would like to thank the reviewers for their valuable feedback. This research is partially funded by the Research Fund KU Leuven, the National Science Foundation (NSF) under grants, CNS-1617902, CNS-1617593, and CNS-1735396, and the Office of Naval Research (ONR) under grant N00014-16-1-2264. Some of our experiments were conducted with equipment purchased through NSF CISE Research Infrastructure Grant No. 1405641. We thank Domainr.com and Whoxy.com for their support.

REFERENCES

[1] 2017. DNSSEC Deployment Report. https://rick.eng.br/dnssecstat/. (2017).
[2] 101domain GRS Limited. 2017. .ne Domain Registration. (2017). https://www.101domain.com/ne.htm
[3] A Hubert, R van Mook. 2009. Measures for Making DNS More Resilient against Forged Answers. (2009). https://tools.ietf.org/html/rfc5452
[4] AFNIC. 2017. Changing Registrars. (2017). https://www.afnic.fr/en/your-domain-name/manage-your-domain-name/changing-registrars-3.html
[5] Pieter Agten, Wouter Joosen, Frank Piessens, and Nick Nikiforakis. 2015. Seven months' worth of mistakes: A longitudinal study of typosquatting abuse. In *Proceedings of the 22nd Network and Distributed System Security Symposium (NDSS 2015)*. Internet Society.
[6] Matt Bryant. 2015. Fishing the AWS IP Pool for Dangling Domains. http://www.bishopfox.com/blog/2015/10/fishing-the-aws-ip-pool-for-dangling-domains/. (2015).
[7] Matt Bryant. 2016. The Orphaned Internet: Taking Over 120K Domains via a DNS Vulnerability in AWS, Google Cloud, Rackspace and Digital Ocean. https://thehackerblog.com/the-orphaned-internet-taking-over-120k-domains-via-a-dns-vulnerability-in-aws-google-cloud-rackspace-and-digital-ocean/. (2016).
[8] CIRA. 2017. Register your .CA. (2017). https://cira.ca/ca-domains/register-your-ca
[9] Cloudflare. 2017. How DNSSEC Works. (2017). https://www.cloudflare.com/dns/dnssec/how-dnssec-works/
[10] Carlo Contavalli, Warren Kumari, and Wilmer van der Gaast. 2016. RFC7871: Client Subnet in DNS Queries. (2016). https://tools.ietf.org/html/rfc7871
[11] Nik Cubrilovic. 2009. The Anatomy Of The Twitter Attack. https://techcrunch.com/2009/07/19/the-anatomy-of-the-twitter-attack/. (2009).
[12] Dan York. 2011. DNSSEC Statistics. (2011). http://www.internetsociety.org/deploy360/dnssec/statistics/
[13] Artem Dinaburg. 2011. Bitsquatting: DNS Hijacking without exploitation. (2011).
[14] Domainr. 2017. Domainr Developer API. (2017). https://domainr.build/
[15] DomainTools. 2016. Domain Count Statistics for TLDs. (2016). http://research.domaintools.com/statistics/tld-counts/
[16] Email-Hippo. 2017. Email Validation Online Service. (2017). https://www.emailhippo.com/en-US
[17] Gerry Smith. 2014. When Hackers Steal A Web Address, Few Owners Ever Get It Back. (2014). http://www.huffingtonpost.com/2014/09/29/domain-theft_n_5877510.html
[18] Google Public DNS. 2017. Where are your servers currently located? (2017). https://developers.google.com/speed/public-dns/faq#locations
[19] ICANN. 2016. Transfer Policy. (2016). https://www.icann.org/resources/pages/transfer-policy-2016-06-01-en
[20] ICANN. 2017. Whois Inaccuracy Complaint Form. (2017). https://forms.icann.org/en/resources/compliance/complaints/whois/inaccuracy-form
[21] Mohammad Taha Khan, Xiang Huo, Zhou Li, and Chris Kanich. 2015. Every Second Counts: Quantifying the Negative Externalities of Cybercrime via Typosquatting. In *Proceedings of the 36th IEEE Symposium on Security and Privacy*.
[22] Letś Encrypt. 2017. How It Works. (2017). https://letsencrypt.org/how-it-works/
[23] Chaz Lever, Robert Walls, Yacin Nadji, David Dagon, Patrick McDaniel, and Manos Antonakakis. 2016. Domain-Z: 28 Registrations Later. In *Proceedings of the 37th IEEE Symposium on Security and Privacy*.
[24] Zhou Li, Sumayah Alrwais, Yinglian Xie, Fang Yu, and XiaoFeng Wang. 2013. Finding the linchpins of the dark web: a study on topologically dedicated hosts on malicious web infrastructures. In *Security and Privacy (SP), 2013 IEEE Symposium on*. IEEE, 112–126.
[25] Daiping Liu, Shuai Hao, and Haining Wang. 2016. All Your DNS Records Point to Us: Understanding the Security Threats of Dangling DNS Records. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1414–1425.
[26] Tyler Moore and Richard Clayton. 2014. The Ghosts of Banking Past: Empirical Analysis of Closed Bank Websites. In *Financial Cryptography and Data Security*. Springer, 33–48.
[27] Tyler Moore and Benjamin Edelman. 2010. Measuring the perpetrators and funders of typosquatting. In *International Conference on Financial Cryptography and Data Security*. Springer, 175–191.
[28] Nick Nikiforakis, Luca Invernizzi, Alexandros Kapravelos, Steven Van Acker, Wouter Joosen, Christopher Kruegel, Frank Piessens, and Giovanni Vigna. 2012. You Are What You Include: Large-scale Evaluation of Remote JavaScript Inclusions. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*. 736–747.

[29] Nick Nikiforakis, Steven Van Acker, Wannes Meert, Lieven Desmet, Frank Piessens, and Wouter Joosen. 2013. Bitsquatting: Exploiting bit-flips for fun, or profit?. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 989–998.

[30] Venugopalan Ramasubramanian and Emin Gün Sirer. 2005. Perils of transitive trust in the domain name system. In *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*. USENIX Association, 35–35.

[31] RAPID7. 2015. Vulnerability and Exploit Database: BIND TKEY Query Denial of Service. https://www.rapid7.com/db/modules/auxiliary/dos/dns/bind_tkey. (2015).

[32] Johann Schlamp, Georg Carle, and Ernst W Biersack. 2013. A forensic case study on as hijacking: The attacker's perspective. *ACM SIGCOMM Computer Communication Review* 43, 2 (2013), 5–12.

[33] Johann Schlamp, Josef Gustafsson, Matthias Wählisch, Thomas C Schmidt, and Georg Carle. 2015. The abandoned side of the Internet: Hijacking Internet resources when domain names expire. In *International Workshop on Traffic Monitoring and Analysis*. Springer, 188–201.

[34] Bianca Schroeder, Eduardo Pinheiro, and Wolf-Dietrich Weber. 2009. DRAM errors in the wild: a large-scale field study. In *ACM SIGMETRICS Performance Evaluation Review*, Vol. 37. ACM, 193–204.

[35] Serverfault. 2012. How is DNS lookup order determined? (2012). http://serverfault.com/questions/355414/how-is-dns-lookup-order-determined

[36] Oleksii Starov, Johannes Dahse, Syed Sharique Ahmad, Thorsten Holz, and Nick Nikiforakis. 2016. No Honor Among Thieves: A Large-Scale Analysis of Malicious Web Shells. In *Proceedings of the 25th International World Wide Web Conference (WWW)*.

[37] StartCom. 2017. StartCom Certificate Policy And Practice Statements. (2017). https://www.startcomca.com/policy.pdf

[38] Janos Szurdi, Balazs Kocso, Gabor Cseh, Jonathan Spring, Mark Felegyhazi, and Chris Kanich. 2014. The Long" Taile" of Typosquatting Domain Names.. In *USENIX Security*. 191–206.

[39] Martin Tartarelli. 2016. A Tale of a DNS Packet (CVE-2016-2776). http://blog.infobytesec.com/2016/10/a-tale-of-dns-packet-cve-2016-2776.html. (Oct 2016).

[40] Tezzaron Semiconductor. 2004. Soft Errors in Electronic Memory âĂŞ A White Paper. https://tezzaron.com/media/soft_errors_1_1_secure.pdf. (2004).

[41] Thomas Vissers, Wouter Joosen, and Nick Nikiforakis. 2015. Parking Sensors: Analyzing and Detecting Parked Domains.

[42] Yi-Min Wang, Doug Beck, Jeffrey Wang, Chad Verbowski, and Brad Daniels. 2006. Strider Typo-Patrol: Discovery and Analysis of Systematic Typo-Squatting. 6 (2006), 31–36.

[43] Duane Wessels. 2012. Evidence of Bitsquatting in COM/NET Queries. https://www.nanog.org/meetings/nanog54/presentations/Tuesday/Wessels.pdf. (2012).

[44] D Wessels. 2016. (2016). http://serverfault.com/a/819858

[45] Whoxy. 2017. Whois Lookup API. (2017). https://www.whoxy.com/#api

[46] Ben Woods. 2013. 15 of the most expensive domains of all time. https://thenextweb.com/shareables/2013/08/13/15-of-the-most-expensive-domains-of-all-time/. (2013).

[47] Nicky Woolf. 2016. DDoS attack that disrupted internet was largest of its kind in history, experts say. (2016). https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet

[48] ZyTrax, Inc. 2015. DNS BIND Operations Statements: max-cache-ttl. (2015). http://www.zytrax.com/books/dns/ch7/hkpng.html#max-cache-ttl