Risky BIZness: Risks Derived from Registrar Name Management

Gautam Akiwate
UC San Diego
gakiwate@cs.ucsd.edu

Stefan Savage
UC San Diego
savage@cs.ucsd.edu

Geoffrey M. Voelker
UC San Diego
voelker@cs.ucsd.edu

KC Claffy
CAIDA/UC San Diego
kc@caida.org

ABSTRACT

In this paper, we explore a domain hijacking risk that is an accidental byproduct of undocumented operational practices between domain registrars and registries. We show how over the last nine years over $512K$ domains have been implicitly exposed to the risk of hijacking, affecting names in most popular TLDs (including `.com` and `.net`) as well as legacy TLDs with tight registration control (such as `.edu` and `.gov`). Moreover, we show that this weakness has been actively exploited by multiple parties who, over the years, have assumed control over $163K$ domains without having *any* ownership interest in those names. In addition to characterizing the nature and size of this problem, we also report on the efficacy of the remediation in response to our outreach with registrars.

CCS CONCEPTS

• **Networks** → *Naming and addressing*; *Public Internet*.

ACM Reference Format:
Gautam Akiwate, Stefan Savage, Geoffrey M. Voelker, and KC Claffy. 2021. Risky BIZness: Risks Derived from Registrar Name Management. In *ACM Internet Measurement Conference (IMC '21), November 2–4, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3487552.3487816

1 INTRODUCTION

The security of the domain name system (DNS) is predicated on the integrity of name resolutions. When a user enters `www.amazon.com` into their browser, they assume that the Web page ultimately reached is the correct one (as intended by Amazon). Even strong security measures such as TLS implicitly assume the integrity of name resolution, since key certificate authorities, such as Let's Encrypt, predicate their due diligence on controlling a domain [10]. However, if an attacker is able to substitute their own answers in response to queries for a domain (*i.e.*, *domain hijacking*), then these security assumptions, both implicit and explicit, are violated.

To date, most domain hijacking has been the result of active attacks: either via the compromise of accounts with the authority to manipulate a domain's zone records [7] or, in the case of cache poisoning, an attack on the resolution protocol itself [20]. In this paper we explore an alternate avenue for domain hijacking that is not due to any act of attacker compromise or domain owner misconfiguration, but is instead an unintended byproduct of long-standing undocumented registrar practices.

In particular, we explore risks that emerge from the use of third-party nameservers wherein the nameserver domain is slated for removal by *its own* registrar. For such actions, registrars rely on the Extensible Provisioning Protocol (EPP), which provides a standard interface for registrars to provision and manage domain names and nameservers within each domain registry. However, in particular situations wherein the domain has subordinate host objects (typically representing nameservers) referenced by other domains, the constraints dictated by EPP do not allow the domain to be removed — *even by the registrar of the domain.* Over the years, registrars have developed an operational workaround for this limitation, in which the registrars rename host objects subordinate to the domain within the EPP system to enable removal of the domain. The host objects thus renamed are given an entirely new domain name that typically falls under the authority of a *different* top-level domain (TLD) operated by a *different* registry.[1] We call these resulting nameserver names *sacrificial nameservers.*

For example, the nameserver `ns2.example.com`, on expiry of the domain `example.com`, might be renamed *within the registry* to `{randomstring}.biz`. As a result, *any* domain name in the `.com` TLD that had delegated its nameservice to `ns2.example.com` would find that nameserver silently replaced with `{randomstring}.biz`.[2] While, as we will show, different registrars use different renaming idioms, the end result is similar. Moreover, in most cases this renaming is entirely mechanical and no attempt is made to register the new domain name (or, for that matter, to validate that the new name is not *already* registered). As a result, any party assuming control of `{randomstring}.biz` is subsequently able to control name resolution for all of the domains that had previously used `ns2.example.com` for name service. Perhaps more importantly, as a result of the renaming, a simple re-registration of `example.com` will not fix the issue.

This process that we have described is byzantine and unintuitive, which perhaps explains why it has not been identified as an issue in spite of almost two decades of practice. However, it is not an uncommon occurrence. Our analyses of zone data collected over the last nine years shows that this operational pattern has put at least a *half million domains* at risk of hijacking. Further, we will show that this is not merely a potential risk, but that it has been actively exploited by multiple parties. Together, such actors have registered the domains for at least $9,173$ sacrificial nameservers

IMC '21, November 2–4, 2021, Virtual Event, USA

ACM ISBN 978-1-4503-9129-0/21/11.
https://doi.org/10.1145/3487552.3487816

[1] The `.biz` TLD appears to have been most widely used for this purpose, inspiring our title.

[2] Indeed, for reasons we will explain, this change is not limited to the original nameserver's TLD but, depending on which TLD is used, can impact domains in a wide range of distinct TLDs, including some, such as `.edu` and `.gov`, whose registration is restricted.

and, in so doing, have obtained implicit control over more than 163, 000 domains for which they have no clear ownership interest. Moreover, of the domains that are currently exposed in this manner, our analysis shows that more than 6% maintain alternative nameservers (*i.e.*, indicating that these domains may continue to operate as going concerns without any knowledge that they are at risk). While most such domains are associated with small sites that may not be widely visited, they also include domains operated by groups in positions of authority, including law enforcement, courthouses, lawyers, health care organizations, government public health officials and religious groups.

In exploring this issue, we make four key contributions:

- Identifying sacrificial nameserver renaming practices and the hijacking risk they create. We develop a systematic methodology for identifying sacrificial nameserver renaming and characterizing the idioms used by registrars.
- Quantifying its scope and scale. Using almost a decade of archival zone file data we identify the number of domains exposed to hijacking and the dynamics of this exposure over time.
- Characterizing abuse. We empirically establish the feasibility of domain name hijacking via registering sacrificial nameserver domains, both by doing so ourselves (in controlled experiments) and by documenting a range of parties who have used this approach to acquire the traffic of many tens of thousands of domains they do not own.
- Remediation. We have been working with registrars and registries to address this issue. As a result, some registrars have changed operational practices to prevent new hijackable domains, while helping remediate existing ones.

In addition to our measurement results, we discuss the challenges in fixing this problem going forward.

2 BACKGROUND

The domain name system (DNS) is a deceptively complex artifact that relies on a broad range of technical components, organizations and procedures. In this section we provide a brief review of DNS concepts, the role of registrars and registries and existing mechanisms that have been implicated in domain name hijacking. We also provide background on the role of the Extensible Provisioning Protocol (EPP) and explain how some of its constraints impact registrars and how a popular workaround creates a hijacking risk.

2.1 DNS Namespace and Protocol

DNS is built around a namespace hierarchy (documented in RFC 1034 [17]) in which there is explicit delegation of administrative authority to individual non-overlapping *zones* following a tree-based structure. Thus, the root of the DNS name tree explicitly delegates authority for individual top-level domains (*e.g.*, `.com` or `.gov`) to nameservers who are responsible for that portion of the namespace (*i.e.*, *zone*). These nameservers in turn can further delegate their portion of the namespace to yet other servers (*e.g.*, `.com` provides nameserver records for `example.com` which thereby delegates control over all domains under `example.com` to those servers) and each zone is free to sub-delegate more specific portions of the namespace below it in the same manner.

The DNS query protocol, standardized in RFC 1035 [18], describes how DNS network queries should be issued, interpreted and appropriately routed, to ultimately find the nameservers able to provide authoritative answers for the portion of the namespace being queried. Moreover, it is designed to do so in a way that maximally exploits locality and thus reduces latency and load.

2.2 Name Registration and Provisioning

The DNS standard does not go into detail about how domain names are procured or how namespace delegation is populated and managed across nameservers. That said, those details are critical to the correct functioning of the DNS.

With a few exceptions, all top-level domains are associated with administrative entities called *registries* that primarily operate either under contract with the Internet Corporation for Assigned Names and Numbers (ICANN) (*e.g.*, for gTLDs and most legacy TLDs) or represent sovereign naming interests (ccTLDs, such as `.us` or `.ru`). Registries are responsible for the database of registered names directly underneath the TLD in the namespace hierarchy (*e.g.*, Verisign is the registry for `.com` and thus would contain `example.com` in its database) and for the nameservers that delegate authority under that namespace. Note that registries may have responsibility for multiple TLDs and some registries will outsource the technical operation of their databases to third parties who specialize in registry operations (*e.g.*, Afilias is one such specialist). Thus, for example, Verisign is the registry for the `.com` and `.net` TLDs (among others) and *also* implements the registry backend for `.edu` and `.gov` (on behalf of EDUCAUSE and the US Cybersecurity and Infrastructure Security Agency (CISA), respectively).

Via their nameservers, registries provide delegation for all the registered domain names used in the DNS. However, provisioning new domain names or changing the details of their delegation is a responsibility typically shared with third parties called *registrars*. Registrars act as an interface between customers who wish to obtain or manage a domain name and the registries that maintain authoritative delegation information for those domains. Thus, a customer seeking to obtain `riskybiziness.com` (available as of this writing) would contract with a registrar (*e.g.*, GoDaddy) who would, in turn, engage with the registry (Verisign) to claim the name and install the customer's choice of nameserver (NS) records in the `.com` zone.[3] Importantly, registrars can contract with many registries *and* there can be many registrars who contract with each individual registry.

Finally, although not formally part of either the DNS or the name registration and provisioning systems, nameserver hosting plays an important role in practical DNS operations. While some name registrants host their own nameservers, others outsource this function to a third party. Thus, consider the situation in which `example.com` is registered via GoDaddy. The owner of this domain could choose to manage their own nameservers, in which case they might request that `example.com`'s NS records point to `ns1.example.com` and

[3] Note that some TLDs are restricted to particular classes of registrants and do not use registrars. For example, `.edu` domains are only made available to educational institutions (via EDUCAUSE), and `.gov` domains are only available to US Government entities (via CISA).

ns2.example.com.[4] However, they might instead choose to just use GoDaddy to provide nameservice. Alternatively, they might choose a third-party nameservice provider that offers DDoS protection and, in many cases, they might do some combination of all of these, possibly for reasons of diversity and redundancy. Thus, example.com might have NS records that point to multiple different domains that are owned and operated by third parties.

2.3 Domain Hijacking

Any time the name resolution for a domain name is controlled by an outside party, without the consent of the domain owner, it is commonly referred to as *domain hijacking*. If an outside party can control the resolution, then their lack of ownership interest in the domain is irrelevant because their control over resolution is the capability that matters. Hijacking can be employed for a range of purposes including site defacement, phishing, man-in-the-middle attacks and/or further compromise. Those visiting the hijacked domain will have no way of knowing that they are not visiting the site that they expect.[5]

There are a number of ways domain hijacking can occur. Perhaps the best known are direct attacks on the DNS protocol itself, particularly a family of attacks called *DNS cache poisoning* that inject carefully forged and timed DNS responses to convince recursive resolvers to accept and cache false authoritative information [6, 20]. Such attacks involve repeated and active network-layer attacks and, typically, are only able to directly impact one DNS cache (*i.e.*, resolver) at a time. Another class of attacks results from the theft of credentials: either the domain owner's credentials (*i.e.*, their account with their registrar and/or their nameserver hosting provider) or the credentials of a registrar or registry administrator with authority to update records on behalf of the domain owner [7, 15]. In these cases, the adversary simply replaces the nameserver records (either at the registry level or, if dealing with subdomains, for the domain's zone).

The other opportunity for hijacking occurs as a byproduct of errors or inconsistency in how nameserver delegation is specified. If a nameserver to whom responsibility for a domain is delegated is unable to provide authoritative information, it is referred to as a *lame delegation*. *Dangling delegations* are a special case of this phenomenon in which some resource (*e.g.*, the domain name or the IP address) is unclaimed and thus might be acquired by an attacker for domain hijacking. Liu *et al.* first documented the presence of domains whose nameserver domains have expired and thus an adversary could simply purchase them [16]. Expired nameserver domains are conceptually similar to the situation we study, but for the fact that in our study the vulnerable nameserver domains are completely new, created by registrars. Bryant documented a large-scale version of this problem in which stale NS records at the .io registry provided a mechanism to hijack *all* subordinate domains [4]. Vissers *et al.* extended these ideas to cover nameserver domains whose own nameservice is dependent on dangling names, as well as dangling that occurs via accident (typos) and bit errors (so-called bit squatting) [21]. Recently, Alowaisheq *et al.* [3] showed that stale records in the domain's zone (as opposed to the zone of the parent TLD) provided sufficient purchase for hijacking (and, in demonstrating this risk, provided an improved mechanism for exploiting dangling delegation hijacking in general).

Finally, Akiwate *et al.* provided a recent measurement survey of lame delegations across the Internet (not motivated by hijacking in particular) and in the course of that measurement first identified the issue of registrar-based renaming [2]. Our work specifically builds on this paper and seeks to fully explore how widespread this practice is among registrars, characterize the scale and scope of the risk, identify the extent to which it is being actively used to hijack domains, and work with registrars to remediate the problem.

2.4 EPP and the Host Object Renaming Trick

As we have discussed, a multiplicity of registrars contract to register and manage domain names under the authority of each registry. To manage the attendant complexity, the provisioning and management of domain names and nameserver delegation records are standardized via the Extensible Provisioning Protocol (EPP). Each registry operator provides an EPP interface to its object repository, which allows its contracted registrars to make provisioning requests (*e.g.*, creating domains, deleting domains, updating their nameserver records, etc.). Chief among the properties that EPP guarantees is isolation: a domain registered by one registrar cannot be modified by another without permission.

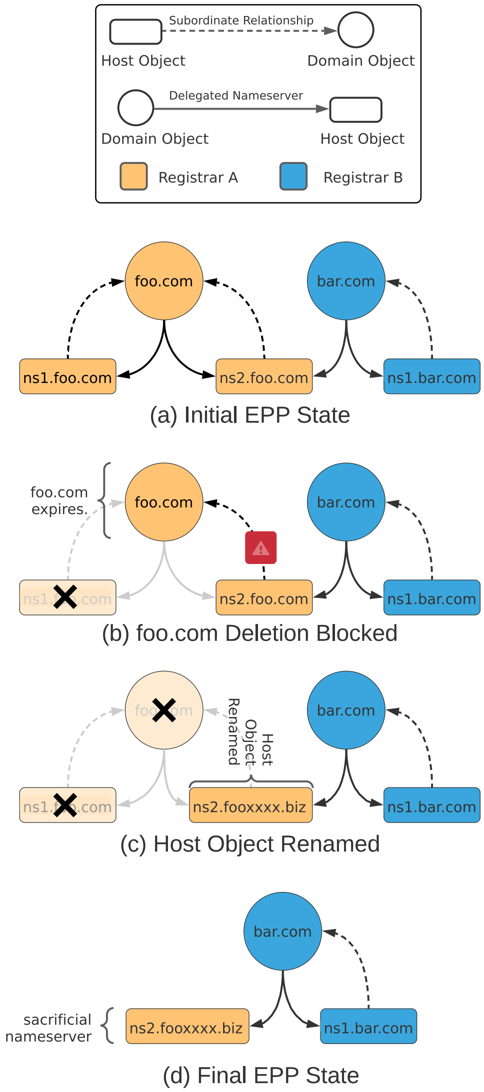
EPP is standardized in RFC 5730 [11] and the domain and host mapping (critical for this paper) is documented in RFCs 5731 [12] and 5732 [13]. An EPP object repository contains two kinds of objects: domain objects, which represent the information about registered domain names; and host objects, which hold information about nameservers including their host name. However, the two are inexorably linked through their use of domain names. In EPP terminology, a domain object (foo.com) is *superordinate* to individual *subordinate* host objects that make use of that domain (*e.g.*, ns1.foo.com or ns2.foo.com). The EPP object mapping standards include rules to ensure that references between objects are sound, *i.e.*, you cannot delete an object that is referred to by another. Two EPP rules are critically important to this paper:

> A domain object SHOULD NOT be deleted if subordinate host objects are associated with the domain object. For example, if domain "example.com" exists and host object "ns1.example.com" also exists, then domain "example.com" SHOULD NOT be deleted until host "ns1.example.com" has either been deleted or renamed to exist in a different superordinate domain. [RFC 5731]
>
> A host name object SHOULD NOT be deleted if the host object is associated with any other object. For example, if the host object is associated with a domain object, the host object SHOULD NOT be deleted until the existing association has been broken. [RFC 5732]

These consistency rules, combined with the isolation property protecting registrars from one another, leads to the problem demonstrated in Figure 1. Registrar A is responsible for the domain foo.com and wishes to delete it (in this case because its registration has expired). However, before the domain object can be deleted, registrar

[4] Note that in these situations it is key that additional "glue" Address (A) records also be provisioned to allow example.com's nameserver names to be resolved. However, these details are not critical for this paper.

[5] In principle, while TLS is designed to protect against such attacks, it has been repeatedly demonstrated that attackers can use control over a domain's name resolution to acquire new valid certificates from certificate authorities.



(a) Initial EPP State

(b) foo.com Deletion Blocked

(c) Host Object Renamed

(d) Final EPP State

Figure 1: Nameserver renaming in EPP as a mechanism to bypass domain deletion constraints

A must first delete any subordinate host objects (`ns1.foo.com` and `ns2.foo.com`). This step is straightforward for `ns1.foo.com`, but `ns2.foo.com` is referred to by the domain object `bar.com` which has delegated nameservice to that host object. Unfortunately, since `bar.com` is under the control of registrar B, EPP's protections prevent registrar A from changing that delegation.

However, there is a workaround. As per RFC 5731, registrar A can *rename* the host object (`ns2.foo.com`), which it controls, to something in another domain that it also controls. As a result, the host object is no longer subordinate to `foo.com`.

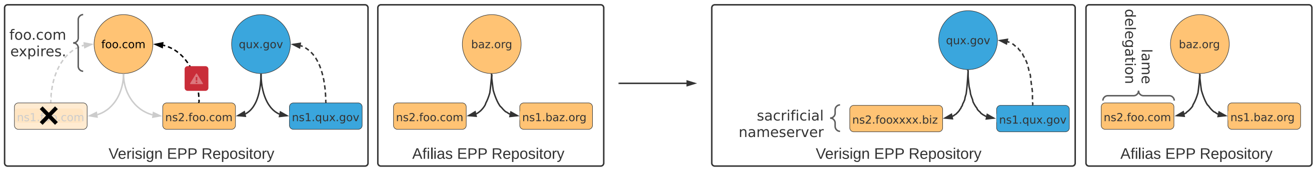
For example, for a time one registrar renamed their unwanted nameservers to `{randomstring}.dummyns.com`, where `dummyns.com` was a "sink" domain that they operated expressly for this purpose. This approach prevents hijacking, but has the disadvantage that the registrar must manage this domain carefully to ensure it is not itself hijacked.[6]

Another approach is to rename each unwanted host object to an *entirely new* domain that does not exist. This approach minimizes load and responsibility to the registrar, but does create a potential risk of future hijacking. However, it also introduces a new complication: it is not possible to create a dangling domain reference inside an EPP repository. In particular, EPP will not allow a host object to be renamed subordinate to a non-existent domain object within the namespace of its repository (*i.e.*, you cannot create an `ns2.foobar.com` host object in Verisign's EPP repository unless the `foobar.com` domain object already exists). However, some registrars discovered a loophole. EPP relaxes its rules if the namespace is *external* to the EPP repository. Specifically, if the new superordinate domain is in `.biz` (or any other TLD not managed by Verisign), then the Verisign EPP repository declares no authority over it and lets the rename take place.

Returning to our example in Figure 1 we see just such a transformation take place. The `ns2.foo.com` host object is renamed to `ns2.fooxxxx.biz`, which EPP allows. Thus, all references to `ns2.foo.com` in the EPP repository now point to this host object. Since the `.com` TLD nameservers are populated from this repository it means that a DNS request for any domain (*e.g.*, such as `bar.com`) that had previously pointed at `ns2.foo.com` will now return NS records for the sacrificial nameserver `ns2.fooxxxx.biz` (which refers to an unregistered domain in a different TLD). This outcome is unintuitive to the operator of `bar.com` since neither they, nor their registrar, took any action and yet their NS records have changed. It is similarly unintuitive to the operator of the `.biz` registry who does not participate in this transaction. In particular, the resultant sacrificial nameserver is not directly visible to the `.biz` registry since no objects are created in its registry database, except insofar as the `.biz` TLD servers will be forced to handle additional name service requests for the non-existent domain. Finally, having completed this transformation, the registrar who initiated the action now lacks the authority to "undo" it, both because host objects referring to an external TLD cannot be modified, and changing nameserver records for domains (*e.g.*, such as `bar.com`) managed by another registrar is outside their direct control.

Finally, it is important to note that the scope of a host object renaming operation is *not* a TLD, but is the scope of the collective namespaces managed by the particular EPP repository (*i.e.*, *all* TLDs whose registries are operated by that provider). Thus, in the context of Figure 2, because Verisign also operates `.gov` (and `.net` and `.edu`), the domain `qux.gov` that pointed to `ns2.foo.com` would also be silently updated to use the new sacrificial nameserver, while the domain `baz.org` (operated by Afilias) would be unchanged since it belongs to a separate EPP repository. As a result, even though both `qux.gov` and `baz.org` initially delegated to the same nameserver `ns2.foo.com`, the final nameserver delegation after `foo.com` expires is dependent on the EPP repository. Note, it is this scoping property that allows domains under *restricted* TLDs (*e.g.*, `.gov` and `.edu`

[6] Ironically, it appears that `dummyns.com` was abandoned for this purpose and is now being operated to hijack nameserver traffic for all domains that pointed to it.



(a) Handling of foo.com expiration in different EPP repositories

(b) Final EPP State

Figure 2: Handling of domain expiration in different EPP repositories. The renaming operation affects all TLDs supported by a registry's EPP repository, but other EPP repositories are unaffected by it.

operated by Verisign) to also be affected by this issue in spite of the fact that they do not use registrars.

In the remainder of this paper we provide a comprehensive assessment of the prevalence of this practice, the scope of the exposure, exploitation of the exposure, and efforts to remediate this practice.

3 IDENTIFYING SACRIFICIAL NAMESERVERS

In this section we describe our methodology for identifying sacrificial nameservers. Using nine years of TLD zone files, we first generate a candidate set of nameservers that match the properties expected of newly created sacrificial nameservers. From this candidate set, we then identify renaming idioms used by various registrars over time and the nameservers in the zone files that match these idioms.

3.1 Properties of Sacrificial Nameservers

Based on the EPP constraints that lead to the creation of sacrificial nameservers, we expect them to have the following three properties when originally created:

(1) **Visibility**: Sacrificial nameservers are a result of renaming host objects by registrars via EPP at the registry level (typically with domain owners unaware of these changes). As such, we only expect to see sacrificial nameservers as authoritative nameservers for domains at the level of the registry TLD servers (parent zone) and not in the authoritative nameservers configured by the domain owner (child zone).

(2) **Unresolvability**: When created, sacrificial nameservers are simply names in a registry database, and as such are not intended to refer to operational nameservers that actively resolve delegated domains. As a result, we expect sacrificial nameservers to be "unresolvable" when created (*i.e.*, we expect the domains delegated to sacrificial nameservers to be lame delegated). Even if a sacrificial nameserver uses a sink domain, we expect it to be lame delegated assuming the registrar does not want their nameservers to handle queries for domains that they are not authoritative for and hence cannot resolve.

(3) **Single Repository**: Since different registries operate different EPP repositories, the renaming of a host object should only affect domains hosted in the same EPP repository. As a result, the domains that delegate to sacrificial nameservers cannot span multiple EPP repositories (maintained by different registries) since renaming only affects domains in the same EPP repository. For example, a sacrificial nameserver cannot affect domains in `.com` and `.info` since it would span two different registry repositories, namely Verisign and Afilias.

We use these properties as the basis for discovering sacrificial nameservers.

3.2 Finding Sacrificial Nameservers

The visibility property of sacrificial nameservers means that the TLD zone files should capture their creation via renaming. As a result, the primary data set we use is the zone file data in CAIDA-DZDB [5].[7] The data set covers nine years of daily snapshots of zone files from April 2011 through September 2020. As of September 2020, CAIDA-DZDB contained zone files for over 1250 zones. These 1250 zones include 530.4M domains and 20.8M nameservers spanning the legacy gTLDs, the new generic TLDs (ngTLDs), and the `.us`, `.nu`, and `.se` country-code TLDs (ccTLDs). While the zone data was obtained through a combination of signed access agreements for early years of data, the ICANN Centralized Zone Data Service (CZDS) [14], and publicly available zone data, CAIDA now provides uniform *research access* to the DZDB data set used in this paper both interactively and via a programmatic API.

To find sacrificial nameservers, we first narrow the full set of roughly 20M initial nameservers in CAIDA-DZDB to a set of around 300K unresolvable nameservers. We then look for patterns in the names of the unresolvable nameservers that reflect renaming idioms registrars have used to create sacrificial nameservers, resulting in a refined candidate set. As such, our ability to identify sacrificial nameservers with confidence is contingent on their use of either a unique identifier in the renaming scheme (*e.g.*, `dropthishost`) or the use of the original nameserver in the sacrificial nameserver (*e.g.*, `ns2.foo.com` renamed to `ns2.fooxxxx.biz`).[8] As a consequence, we are conservative in our estimate of sacrificial nameservers.

We then manually confirmed the registrar renaming idioms we discover, and then went back and systematically matched them to the entire longitudinal zone file data set to create our final set of sacrificial nameservers. Of the roughly 300K unresolvable nameservers, we find more than 200K nameservers are sacrificial. The following subsections describe each of these steps in more detail.

[7] CAIDA-DZDB data set is a clone of the DNS Coffee data set used in Akiwate *et al.* [2].

[8] A sacrificial nameserver with a completely random string is hard to disambiguate from typos with absolute certainty.

3.2.1 Unresolvable Nameservers. Our first step collects nameservers that are unresolvable when they are first referenced by domains into an initial candidate set. Recall that registrars create sacrificial nameservers to remove dependencies on host objects in a registry database. For this purpose, the sacrificial nameserver is just a name in the database, and is not intended to refer to a domain that resolves to a host with an operational nameserver. Sacrificial nameservers typically either refer to a sink domain controlled by the registrar, or to a randomly generated name in another registry. In either case, we expect the sacrificial nameserver to be unresolvable at the time it is created,[9] and thus the domains that delegate to it become at least partly lame delegated at that moment.

Based on that observation, our approach is to identify all nameservers that are referenced by some domain in the zone files before the nameserver itself first became resolvable (if ever). To determine the resolvability of a nameserver we use a simplified version of the static resolution methodology from Akiwate *et al.* [2] for identifying lame delegations. In essence, we use the daily snapshots of the zone files to derive the date ranges for when each nameserver has a valid static resolution path (*e.g.*, via glue records in the zone files). When a nameserver is referenced by any domain for the first time, and the nameserver is unresolvable at that time, then we add the nameserver to the candidate set. Using this method reduces the initial 20 M nameservers in the zone files to a candidate set of 312,328 nameservers.

3.2.2 Identifying Patterns. Our next step identifies unique patterns among the candidate nameservers that reveal renaming idioms used by registrars. These idioms reflect patterns in the use of sink domains for sacrificial nameservers, such as `LAMEDELEGATION.ORG`, or patterns in the generation of random names, such as using the prefix `DROPTHISHOST`.

To discover patterns in nameserver names we built a tool that, given a list of domain names as input, looks for common substrings across them. We applied it to the set of roughly 300K candidate nameservers, revealing the most common substrings among nameservers in the candidate set. We then manually examined the output from the tool and identified nine such patterns. For each, we manually confirmed that the nine patterns consistently reflect sacrificial nameserver renaming idioms.

During this analysis we discovered two naming patterns used for testing purposes. Nameservers such as `EMT-NS1.EMT-T-407979799-1575645880157-2-U.COM` and other nameservers with the `EMT-` prefix are one such pattern. Similar to our reaching out to registrars to confirm their renaming practices, reaching out to a registry confirmed the nature of these nameservers. We removed 28, 614 such test nameservers from the candidate set.

3.2.3 Original Nameserver Matching. Next we use a host name matching tool on the remaining candidate nameservers. The intuition is that some renaming idioms generate names for sacrificial nameservers partly off the nameserver being renamed. To take advantage of this pattern, we first need to identify the nameservers whose renaming led to the creation of the sacrificial nameservers.

To that end, we look at the nameserver history for domains delegated to each of the candidate nameservers. Specifically, we look at the day just before the candidate nameserver was created: the nameserver that was renamed would last show up in the zone file the day before we first see it as a sacrificial nameserver. If the two nameservers (original and renamed) match our criteria, we then classify the renamed server as a sacrificial nameserver.

For example consider `ns2.internetemc1aj2kdy.biz`, a candidate nameserver and the domain `whitecounty.net` that delegates to it. The history for the domain[10] shows that the candidate nameserver first appears on July 1st, 2019. We then look at the nameserver history for the domain (`whitecounty.net`) to find nameservers last seen on June 30th, 2019. There is one nameserver `ns2.internetemc.com` that matches our criteria. Next, we check if the registered domain of the original nameserver is a substring of the sacrificial nameserver registered domain. In this example, `internetemc` is a substring of `internetemc1aj2kdy`, and we conclude that the original nameserver `ns2.internetemc.com` was renamed to `ns2.internetemc1aj2kdy.biz`.

For all the candidate nameservers that pass this match test, we identify the registrar for the nameserver domain at the time of renaming (Enom for `internetemc.com` in the example above) using data from DomainTools [8], and then group the nameservers by registrar. Next, we manually inspect the registrar clusters to identify the renaming scheme. Based on this technique we identified four registrars that used renaming idioms with the previous nameserver as the basis for creating the sacrificial nameserver domain.

Note that before performing the history match we can eliminate some candidate nameservers because they violate the single repository property: the renamed nameserver is in the same TLD as the domains, or the domains delegated to the nameserver span known different registry EPP repositories. We eliminate 11, 403 such nameservers because they violate the single repository property.

3.3 Limitations

Our methodology has limitations that likely prevent us from identifying all sacrificial nameservers. First, our methodology does not detect renaming idioms that do not have a consistent pattern. Moreover, if a registrar creates sacrificial nameservers using a function that does not preserve the original nameserver in a recognizable form, then our last matching step (Section 3.2.3) will not identify them. Second, we assume that sink domains used by registrars are unresolvable. However, it is possible that some registrars could monetize the traffic sent to domains delegated to sacrificial nameservers. Our methodology will not detect these as sacrificial nameservers since they are resolvable. Finally, our data set includes only three ccTLDs, so we have limited insight into sacrificial nameservers among the full set of ccTLDs.

Given these limitations, our results are therefore a lower bound on the overall prevalence of sacrificial nameservers. However, since our methodology was able to uncover the sacrificial renaming practices used (and confirmed) by many major registrars, we believe that our results reflect common practice (at least among non-ccTLDs).

[9] If a hijacker later registers the sacrificial nameserver domain, then it does become resolvable later in its lifetime.

[10] https://dzdb.caida.org/domains/WHITECOUNTY.NET

Renaming Idiom	**Registrar**	**# of Sacrificial**	**# of Affected**
Sink Domain		**Nameservers**	**Domains**
DUMMYNS.COM	Internet.bs	10,147	38,936
LAMEDELEGATION.ORG	Network Solutions	5,902	113,496
NSHOLDFIX.COM	TLD Registrar Solutions	3,527	3,248
DELETE-HOST.COM	GMO Internet	1,224	41,408
DELETEDNS.COM	Xin Net Technology Corp.	535	29,620
LAMEDELEGATIONSERVERS.{COM, NET}	SRSPlus	447	2,009
Total		21,782	228,698

Table 1: Non-hijackable renaming idioms using registered sink domains. Note that a given domain may be affected by more than one sacrificial nameserver over time, so the sum of all rows can be greater than the overall total. The non-hijackable nature depends on registrars maintaining control over the sink domain.

Renaming Idiom	**Registrar**	**# of Sacrificial**	**# of Affected**	**Example Renaming**
Sink Domain		**Nameservers**	**Domains**	**ns1.foo.com**
PLEASEDROPTHISHOST	GoDaddy	75,030	217,952	pleasedropthishostxxxxx.foo.biz
DROPTHISHOST	GoDaddy	40,374	109,478	dropthishost-xxxxx.biz
DELETED-DROP	Internet.bs	3,511	9,289	deleted-xxxxx.drop-xxxxxx.biz
123.BIZ	Enom	5,799	7,157	ns1.foo123.biz
xxxxx.{BIZ, COM}	Enom	54,752	164,264	ns1.fooxxxxx.biz
xxxxx.BIZ	DomainPeople	654	3,304	ns1.fooxxxxx.biz
xxxxx.BIZ	Fabulous.com	334	1,223	ns1.fooxxxxx.biz
xxxxx.BIZ	Register.com	388	1,570	ns1.fooxxxxx.biz
Total		180,842	512,715	

Table 2: Hijackable renaming idioms using random sacrificial names. The xxxxx is a place holder for random strings of various lengths depending on the registrar and the time. Note that a given domain may be affected by more than one sacrificial nameserver over time, so the sum of all rows can be greater than the overall total.

4 REGISTRAR RENAMING IDIOMS

This section presents the results of our methodology for identifying sacrificial nameservers and the renaming idioms that registrars use to create them. Overall we identified more than a dozen registrar renaming idioms that were used to create 202,624 sacrificial nameservers, and ultimately impacted 741,413 domains.

We divide the renaming idioms into two classes, non-hijackable and hijackable. The non-hijackable renaming idioms use a registered sink domain and thus cannot be hijacked. Table 1 lists the registrars that have used non-hijackable idioms and the sink domains they used for renaming. This renaming approach ensures that affected domains are not at risk, but requires that the registrar ensures that the sink domain does not expire (otherwise all affected domains could be hijacked by a single sacrificial nameserver registration). Indeed, in our analysis, we see evidence of a registrar switching renaming idioms and simply abandoning the sink domain. This instance highlights the long term risks of using sink domains and the potential benefits of a more permanent solution.

In contrast, the hijackable renaming idioms rename the nameserver to a random (likely unregistered) sacrificial name. We classify them as hijackable since an attacker can register the random sacrificial nameserver domain and take over resolution of all domains that were delegated to it. Table 2 shows the renaming idioms adopted by different registrars, the number of hijackable sacrificial nameservers created, and the number of domains affected. Note that some registrars have adopted different renaming idioms over time, which we list separately. The last column shows an example of the resulting sacrificial nameserver created by each renaming idiom.

In the rest of this section, we discuss the renaming idioms of the three most prominent registrars that create hijackable domains as well as a significant accidental renaming event in more detail. Sections 5 and 6 then discuss the extent to which hijackable domains are exploited and who is exploiting them, respectively.

GoDaddy: GoDaddy has adopted different renaming idioms over time. The earliest is the PLEASEDROPTHISHOST idiom, which simply replaced the subdomain with PLEASEDROPTHISHOST and a random string. The domain second-level name was kept unchanged while the TLD was typically changed to .biz, unless the nameserver being renamed was itself in .biz. In that case, the sacrificial nameserver used .com. However, this simple renaming idiom meant that at times the sacrificial nameserver inadvertently pointed to an existing domain. In fact, 3,704 sacrificial nameservers created by the PLEASEDROPTHISHOST renaming idiom accidentally used domains that were already registered.

In 2015, GoDaddy adopted the DROPTHISHOST renaming idiom. In this case, the renamed nameserver is DROPTHISHOST followed by a unique random identifier. The sacrificial nameserver is always in the .biz TLD. While this idiom avoided using names in use by existing domains, it still left domains delegated to the sacrificial nameserver at risk of hijack.

Enom: Enom also changed renaming idioms over time. The earliest renaming idiom simply replaced the TLD of the nameserver with 123.biz. By 2012 Enom switched to a new renaming idiom which replaced the TLD by a random string followed by .biz; if the nameserver being renamed was itself in .biz, the sacrificial nameserver instead used .com.

Internet.bs: The registrar Internet.bs is an interesting case. Internet.bs originally used a non-hijackable renaming idiom with DUMMYNS.COM as the sink domain. However, in 2015 after it was acquired by CentralNIC, Internet.bs switched to using a hijackable renaming idiom. In doing so, though, it abandoned its registration of DUMMYNS.COM, leaving it available for registration by other parties who have hijacked nameserver traffic for all domains that point to it. This case highlights the benefits of a more permanent solution codified in the EPP standard (Section 7).

Namecheap's accidental deletion. Our analysis also revealed one large-scale example of an *accidental* renaming event that exposed domains to hijacking in a similar manner. In particular, we identified 46 nameservers renamed under registrar-servers.com, the default nameserver domain for Namecheap, in July of 2016. In communicating with Namecheap, we learned that this event resulted from an employee accidentally sending a deletion request to Enom (at the time this event happened Namecheap registered domains via Enom) for the registrar-servers.com domain. Since this deletion request could not be satisfied while a subordinate host object (*e.g.*, ns1.registrar-servers.com) still existed, the deletion machinery for Enom (since they registered the domains) renamed each of the 46 host objects (default nameservers used by Namecheap) to the .biz TLD (*e.g.*, ns1.registrar-serversxxxx.biz) to eventually delete the registrar-servers.com domain.

As a result, for a brief period of time, 1.6 million domains (including tiktok.com) had dangling delegations that would have permitted hijacking. Luckily, the vast majority of affected domains quickly fixed their delegations: only 51, 699 of the original $1.6M$ domains still delegated to a sacrificial nameserver after three days, and four years later only 51 of them had not fixed their delegation. However, this example further illustrates how the registrar "rename to delete" practice can have risky side effects. Due to the accidental nature of this event, we do not include these nameservers, nor the domains affected as a result, in our subsequent analyses.

5 EXPLOITATION OF SACRIFICIAL NAMESERVERS

The results in Section 4 showed that more than half a million domains were placed at risk because they delegated to a hijackable sacrificial nameserver. However, as we show in this section, this risk is not merely hypothetical. In fact, nearly a third of these domains have been hijacked when their sacrificial nameserver domains were registered. We classify these as hijacks since the sacrificial nameserver domains (*e.g.*, dropthishost-xxxx.biz) have no apparent value other than the domains that delegate to them. As such, the registration of these "random" nameserver domains is unlikely to be accidental in nature. In this section, we characterize this hijacking activity, its dynamics over time, and the nature of the vulnerable domain population.

Overall (2011–2020)	Hijackable	Hijacked	(%)
Sacrificial NS	180,842	9,173	5.07%
Affected Domains	512,715	163,827	31.95%

Table 3: Number of hijackable and hijacked sacrificial nameservers and their delegated domains.

5.1 Hijacking Summary

Table 3 shows the number of sacrificial nameservers that were hijackable and hijacked over the lifetime of our data set. It also shows the number of domains delegated to these nameservers: if a domain delegates to a hijacked sacrificial nameserver, then it is considered hijacked.

Only a small fraction (5%) of hijackable nameservers have been registered over time. Yet, more than 30% of hijackable domains have been hijacked as a result. This disparity is not an accident, and reflects the fact that hijackers are selective in the sacrificial nameservers they register, preferring those used by many domains.

5.2 Hijacking Over Time

As we have discussed, these registrar renaming practices have been in use for many years, and it is evident in our data going back to April of 2011.

Figure 3 longitudinally shows the number of newly hijackable domains that appear each month due to the creation of sacrificial nameservers. Encouragingly, the trend has been downward over the years (perhaps due to the increasing use of third-party nameservers, *e.g.*, domaincontrol.com). However, it is still the case that each month thousands of domains are newly placed at risk of hijacking.

Figure 4 covers the same time period, but shows the number of such domains that are newly hijacked each month. It is clear that hijacking has been a long-standing behavior as well: as long as domains in our data set have been at risk, hijackers have taken advantage of them by registering sacrificial nameservers. Unfortunately, unlike the clear downward trend in newly hijackable domains, the trend in newly hijacked domains is bursty: the hijacking activity occurs throughout our data set, with some months — even recently — seeing thousands of newly hijacked domains.

5.3 Desirability

If we assume that the domains themselves are equally valuable (a clearly simplified assumption, but essentially valid for some business models such as search engine optimization (SEO) for attracting traffic), then the value of hijacking a sacrificial nameserver depends upon how many and how long domains delegate to it. We see indications that hijackers select for registering sacrificial nameservers that enable the hijacking of many domains and potentially for long durations. To provide a visualization of this behavior, for each sacrificial nameserver we define a "hijack value" for it as the sum of all

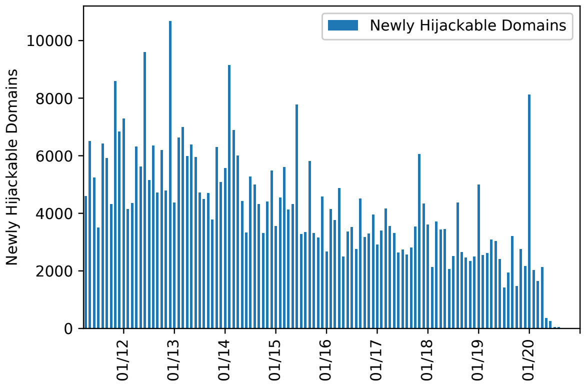


Figure 3: New hijackable domains per month from April 2011 to September 2020.

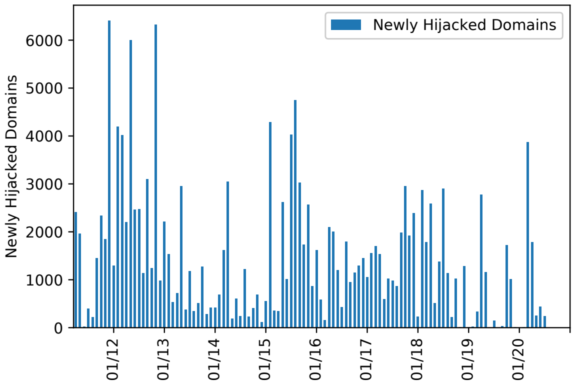


Figure 4: New hijacked domains per month from April 2011 to September 2020.

the days that domains delegated to it were hijackable. For example, if a sacrificial nameserver has one domain that was delegated to it for 30 days and another for 50 days, then the hijack value of the nameserver is 80 days.

Figure 5 shows the relationship between the hijack value of each sacrificial nameserver and the number of domains that delegate to it. Note that the x-axis is log scale, and we cap the y-axis at 1000 domain delegations to maintain clarity. While hijackers do register sacrificial nameservers across the spectrum, the scatter-plot shows that hijackers have registered most of the sacrificial nameservers with the highest value and largest number of delegated domains in our data set.

5.4 Time to Exploit

Next, we characterize how quickly hijackers exploit sacrificial nameservers. For every sacrificial nameserver that was hijacked, we count the number of days from when the sacrificial nameserver was created until it was registered. Figure 6 shows the distributions of these counts as two CDFs. The bottom CDF shows the time to

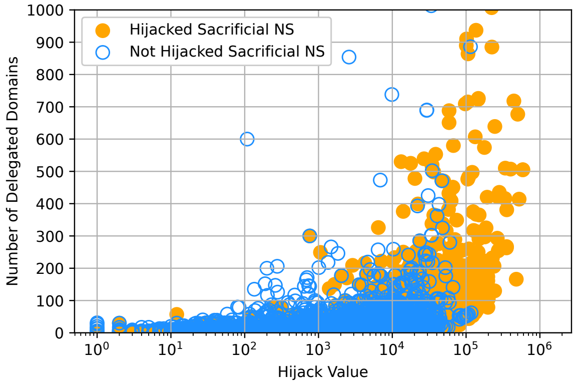


Figure 5: Scatter plot showing the number of domains delegated (capped at 1,000) and the hijack value of both hijackable and hijacked sacrificial nameservers.

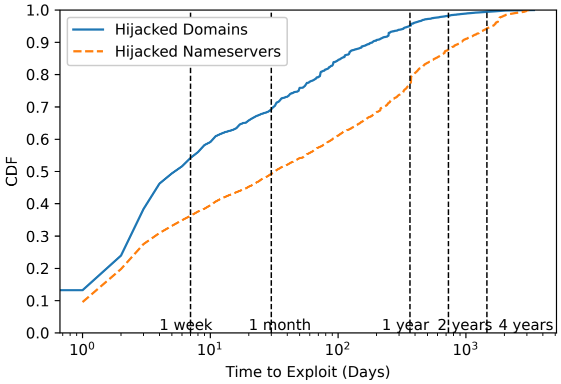


Figure 6: Time to exploit hijackable sacrificial nameservers and vulnerable domains eventually hijacked.

exploit for sacrificial nameservers, and the top CDF for their associated domains. The results show that hijackers move quickly: 50% of vulnerable domains are hijacked within 5 days of when a sacrificial nameserver is created, and more than 70% of vulnerable domains within a month. The quick turnaround time between creation and exploitation suggests actors who routinely monitor for these opportunities and exploit them when they become available.

Moreover, comparing the two CDFs reinforces the notion that hijackers are selective when registering sacrificial nameserver domains. The sacrificial nameservers with the most value are the ones associated with many domains, and the CDFs reflect this difference: the CDF for sacrificial nameservers shows a longer time to exploit consistently relative to the CDF for their associated domains. For instance, whereas 50% of vulnerable domains are registered within a week, only 35% of sacrificial nameservers are registered in the same time span.

5.5 Duration

Finally, we examine the durations for which domains are hijacked further revealing interesting hijacking behaviors. Figure 7 compares the durations for which domains are hijacked with the durations

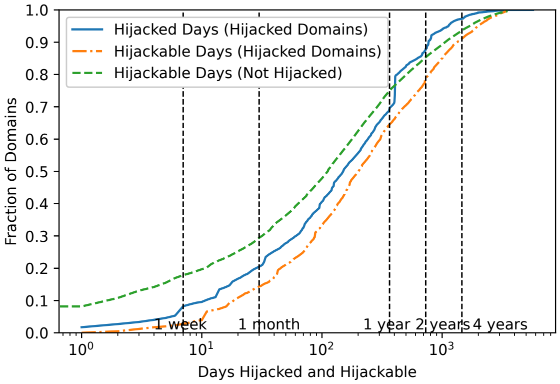


Figure 7: Fraction of domains hijacked or hijackable for at most X days.

for which they are hijackable (at risk of being hijacked). The green and red curves show the CDFs of the number of days for which domains were at risk of being hijacked: the green CDF for domains that were never hijacked, and the red CDF for domains that were hijacked at least once. For domains that were hijacked, the blue CDF further shows the number of days for which they were hijacked.

Comparing the green and red CDFs indicates that hijackers select for domains that are hijackable for longer durations. For domains that are not hijacked, 15% of them are hijackable for less than a week. In contrast, 15% of hijacked domains are hijackable for a month. The two steps in the curve for hijacked domains correspond to domain registrations expiring after one and two years: 10% of hijacked domains are hijacked for one year, and 5% are hijacked for two years, after which they are not renewed even though at times they are hijackable. Often registrars offer lower prices for initial registrations, and then higher prices for renewals. Presumably the domains hijacked via the registered sacrificial nameserver domains were not providing sufficient value to the hijackers, and so they stopped renewing the sacrificial nameserver domains.

We believe altogether these results indicate that hijackers are sensitive to the return on investment — the cost to register the sacrificial domain name — for the domains that they hijack.

5.6 The Nature of Hijacked Domains

If we examine the nature of the domains being hijacked we can sometimes infer aspects of the hijacker's intent. In our analysis, the vast majority of hijacked domains are completely delegated to a hijacked nameserver. While this provides the hijacker complete control over the domain's resolution, it also means that the domain likely lost all nameservice when the renaming transition occurred. This group of "fully hijacked" domains appears to select for unpopular or moribund domains that are not in regular and active use. We believe that the most prolific hijackers are insensitive to the underlying nature of the affected domains and treat them primarily as a source of cheap traffic or reputation. Indeed, of the domains on the Alexa Top 1M list as of September 11, 2020, only ∼500 domains were hijackable at some point of time before September 2020 as a result of the renaming.

However, we note that even for unpopular domains, hijacking carries risk in situations where the hijacked name carries reputation even if it does not receive much traffic. For example, as we describe later in Section 6.1, in a controlled experiment we were able to obtain complete control over a `.edu` domain for an operating educational institution and over an operating `.gov` domain. Controlling such names, further embellished with working certificates and legitimate-looking web sites, would allow an attacker to implicitly invoke the authority of the organization even if the organization rarely used the domain prior to the hijack. For example, approximately 200 of the affected domains were registered by MarkMonitor which specializes in protecting "the online presence of the world's leading brands". These names typically include brand names as part of the domain name (*e.g.*, supporting particular contests or advertising campaigns). These domains, though not in current active use, would be attractive for phishing campaigns since they explicitly invoke the brand in their name *and* are registered by the same registrar used by the brand holder. Fortunately, we have not identified any such attacks using these domains.

Finally, 3,520 of the currently hijackable domains use multiple nameservers where *only a subset* are sacrificial. This situation is particularly worrisome because, when one of their nameservers becomes a sacrificial nameserver, these domains still have fully functional name service as a result of the redundancy provided by their other functional nameservers. Thus, it is entirely likely that the domain owners may not realize that their domains have become hijackable or even hijacked. Indeed, of the 3,520 hijackable domains with alternate resolvable nameservers, 1,105 of them use a sacrificial nameserver that has been hijacked.

Such "partially hijacked" domains include both those of sufficient popularity to appear on the Alexa Top 1M List, but also those used by parties whose communications are particularly sensitive, including public health departments, law offices, law enforcement organizations and courthouses. As an example of this sensitivity, we note that the law enforcement portals of most large Web services — used for serving legal process such as warrants and subpoenas — perform their initial user authentication in large part based on the ability of users to receive e-mail at existing well-known law enforcement domains. Similarly, many courts now routinely issue orders via e-mail — with an implicit authenticity accorded to messages arising from the court's well-known domain names. We have identified a number of partially-hijackable domains that fit this criteria. While we have not identified attackers making sophisticated use of partially hijacked domains, it is unsurprising because we also know of no clear methodology for testing for such attacks.

We have disclosed these domains, as well as the others affected by sacrificial nameservers, to the appropriate registrars and registries for remediation with domain owners. Five months after notification, fewer than 500 of these partially hijackable domains have fixed their delegation. However, as we will describe further in Section 7, the registrar community has deemed the issue of sufficient concern to change their operational procedures and, as of this writing, there are very few sacrificial nameservers still being created.

Hijacker NS Domain	NS	Domains
mpower.nl	3,261	63,759
protectdelegation.{ca,eu,com}	2,551	48,871
yandex.net	2,468	36,001
phonesear.ch	433	14,324
dnspanel.com	549	14,293

Table 4: Top five hijackers overall by number of domains hijacked (April 2011 – September 2020).

	Nameservers		Affected Domains	
	Vuln.	Hijacked	Vuln.	Hijacked
Sep 2020	36,553	1,186 (3.2%)	53,970	16,888 (31.3%)
Feb 2021	26,796	1,210 (4.5%)	40,578	14,606 (36.0%)
Delta	-9,757	+24	-13,392	-2,282

Table 5: Change in number of hijackable (vulnerable) and hijacked sacrificial nameservers and affected domains after notifications starting in September 2020.

6 CHARACTERIZING HIJACKERS

Sacrificial nameservers are clearly being registered that hijack the domains that delegate to them. As a final analysis, we explore what the hijacked domains are being used for — first using a controlled experiment to confirm the capabilities of hijackers, and then more broadly examining how bulk hijackers have been using hijacked domains.

6.1 Controlled Experiment

When a registrar performs a renaming operation that creates a sacrificial nameserver name, it is just a name in the registry database. A hijacker can then register the domain that corresponds to a sacrificial nameserver name, and operate a nameserver that answers queries for delegated domains.

To confirm the capabilities that registering sacrificial nameserver domains affords a hijacker, we registered five such domains without issue. We then used our own infrastructure and confirmed that we observed incoming queries for the domains, while being careful to never respond. Surprisingly, we also saw queries for .edu and .gov domains[11] at our server. These queries were unexpected since the host object renaming should not affect other TLDs, particularly restricted TLDs that do not have traditional registrars.

This phenomenon revealed the situation described in Section 2.4: in practice the renaming operation affects all of the TLDs managed on the same shared EPP repository of a registry. In short, since Verisign manages .edu and .gov, renaming a host object in .com can affect domains in .edu and .gov (among others). Since the .edu and .gov registries manage each registrant themselves, they may not realize or anticipate that NS records in their zone can be changed without their express involvement. Finally, to confirm that we could truly hijack resolution for a domain in a restricted TLD, we updated our infrastructure to respond to queries for the hijackable .edu domain but only for queries coming from a /24 we controlled. Note that we were extremely careful about both legal issues (working closely with our general counsel in designing the experiment) and ethics with this experiment. We discuss the ethical considerations in more detail in Section 8.

6.2 Bulk Hijackers

While it is straightforward to identify that a sacrificial nameserver domain has been registered (and hence that it is likely being used to hijack the domains for which it receives DNS requests), it is far harder to identify who is behind such actions. The combination of long-standing domain registration proxy services, and the impact of the GDPR on information in public registration records, means that we rarely know much about a domain registrant. Moreover, given the tremendous flexibility available to attackers with such control, it is difficult to know precisely the intent of any particular hijack without witnessing an attack in progress.

However, one category of use — bulk traffic exploitation — *is* amenable to cursory automated analysis. In particular, we can distinguish hijacker groups based on the choice of NS records used to support sacrificial name server domains (*i.e.*, what nameservers are used when a sacrificial nameserver domain is looked up?). Table 4 shows the most popular controlling nameserver domains over the course of our study.[12]

Manually visiting the hijacked domains associated with these controlling nameservers in September 2020 is consistent with our hypothesis about their underlying motivation. The most prevalent use of hijacked domains is to host a traditional parking site, with topic links related to the original domain content designed to drive low-quality advertising clicks. For example, sacrificial nameservers controlled by mpower.nl direct their domains in this manner (*e.g.*, alicornarts.com as of this writing). A mass monetization strategy is offered by phonesear.ch, which is not only the controlling nameserver but also the destination site for its hijacked domains (via redirect). phonesear.ch serves a Web site containing links to all North American telephone numbers and appears to use its thousands of hijacked domains to support a search engine optimization (SEO) strategy for attracting traffic. We believe visitors are then monetized via an affiliate relationship with Spokeo (each page at phonesear.ch advertises Spokeo's service to obtain more information about a phone number).

Retrospectively, we also analyzed screenshots of 100 random hijacked domains using the Internet Archive Wayback Machine and confirmed that the use of hijacked domains has not changed significantly over time, with parking sites dominating the sample. We also specifically examined domains hijacked by phonesear.ch in the past, but the screenshots were blank presumably due to how the Wayback Machine handles redirections.

7 NOTIFICATION AND REMEDIATION

Beginning in September 2020, we initiated a broad outreach effort to communicate our findings to the registrar community. The outreach had two main goals: to remediate currently affected domains,

[11] The .gov domain delegation has since been fixed based on our outreach. The .edu domain is no longer hijackable due to our defensive registrations pending outreach.

[12] yandex.net includes default nameservers for domains registered by Yandex.

and to prevent new domains from being exposed. There was considerable surprise in the community about the nature of the issue and sufficient concern to drive a range of efforts to address it. We assess the impact of such actions here, first characterizing the remediation of *existing* hijackable domains and then describing the effects of new renaming practices on the creation of *new* hijackable domains. Finally, we propose potential options for modifications to the EPP standard and registrar operational practices that could form a more robust permanent solution.

7.1 Remediation of Existing Affected Domains

As we have explained, once renamed outside an EPP repository, a host object cannot be subsequently modified (Section 2.4). Consequently, existing sacrificial nameservers cannot simply be renamed by registrars to fix vulnerable domains in a centralized fashion. Instead, any fix requires individual actions for each hijackable domain (either by their registrars or registrants). To facilitate such remediation, we notified the top ten registrars with the most affected domains. Additionally, given the long tail of registrars with affected domains, we collated per-registrar lists of the 54K hijackable domains and made them available, in November 2020, to the registrar community via the DNS Abuse Working Group. At least 12 additional registrars availed themselves of the collated lists from the working group. Since remediation of any form is a cost, we were uncertain how such remediation would play out.

Since we were unable to get concrete communication from any of the 22 registrars on their plan for tackling the affected domains, we had to rely on indirect measures to ascertain impact. As one measure of impact, Table 5 shows the change in number of affected nameservers (down 9K from 36K) and domains (down 13K from 54K) roughly five months (Sep 2020 to Feb 2021) after we started notifications to registrars.[13] We cannot attribute all of these changes to registrar actions since domains will expire naturally and some domain holders may change their delegations organically. To account for this confound, we calculated the baseline rate of "organic" expiration over the equivalent time period a year prior (Sept 2019 to Feb 2020). During that time, we saw the disappearance of 4K sacrificial nameservers and 11K affected domains.

The significant relative improvement in remediation of hijackable sacrificial nameservers (*i.e.*, 9K compared to 4K) is primarily a result of action from GoDaddy. GoDaddy appears to have updated delegations for hijackable domains that they controlled — domains for which they are the current registrar — from their old hijackable renaming to their new renaming idiom. Nearly 60% of the domains remediated (7,877 out of 13,392) and 70% of hijackable nameservers remediated (6,932 out of 9,757) were a result of such actions from GoDaddy. Another notable, albeit smaller, remediation effort was from MarkMonitor who successfully remediated roughly 200 domains (domains with significant brand names).

Interestingly, the smaller relative change in the number of affected domains (13K compared to 11K) suggests that there is a long tail of sacrificial nameservers affecting a few domains whose remediation does not have much overall impact on the situation.

Registrar	New Renaming Idiom	NS	Domains
GoDaddy	`EMPTY.AS112.ARPA`	13,988	28,750
Internet.bs	`NOTAPLACETO.BE`	563	1,330
Enom	`DELETE-REGISTRATION.COM`	459	1,121
Total		15,010	31,201

Table 6: Domains protected due to renaming idiom changes as of September 2021.

7.2 Preventing New Exposure

Of the six registrars that used a hijackable renaming idiom, we were able to successfully notify the three with the largest impact: GoDaddy, Enom, and Internet.bs. In response to our notifications, all three registrars committed to adopting a non-hijackable domain for their future renaming actions. Internet.bs chose a dedicated sink domain `notaplaceto.be` for creating new sacrificial nameservers going forward, as did Enom (using `delete-registration.com` for this purpose). Finally, rather than designating a dedicated sink domain, GoDaddy chose to create sacrificial nameservers under `empty.as112.arpa`, originally envisioned as an anycast sink for queries [1].[14]

Table 6 shows the breakdown of sacrificial nameservers created under these new renaming idioms and the domains protected as a result. As of September 2021, these modifications have prevented the creation of roughly 15K hijackable sacrificial nameservers, thus protecting over 31K domains.

7.3 Robust Long-term Fixes

The ubiquitous use of sink domains is a good short term fix. However, it is also inherently fragile as it relies on existing registrars to maintain these special domains in perpetuity (as well as depending on new registrars to adopt similar measures). Given the dynamism in the registrar market it seems difficult to count on perfection and, indeed, we have past evidence of registrars abandoning sink domains in the past (Section 4). Moreover, because sink domains concentrate dangling delegations, if one such domain is not renewed it could allow an attacker to control tens of thousands of domains with a single registration.

As such, a more permanent solution to this problem likely requires a change to the EPP standard. One potential change to the EPP standard would be to require the use of a reserved TLD for renaming. The IETF-reserved `.invalid` TLD, first reserved in 1999 [9] with additional guidance on its use published in 2013 [19], fits this scenario perfectly. The use of `.invalid` is a promising solution as it eliminates the non-renewal problem. In fact, the idea of creating sacrificial nameservers under a reserved label motivated the use of `empty.as112.arpa` by GoDaddy. However, because the `as112.arpa` domain is anycast, it introduces some new risks. In particular, an attacker controlling an AS112 anycast server could hijack all requests in its vicinity and resolve all such delegations.[15]

[13] Note that a sacrificial nameserver "disappears" when it loses all of its delegated domains.

[14] While this solution avoids the use of a sink domain it may introduce other risks (Section 7.3).

[15] To partially mitigate this risk one could use DNSSEC to sign the `empty.as112.arpa` zone, or use a new signed sibling zone in `as112.arpa`.

A more ambitious approach would combine protocol and operational changes to remove the underlying "garbage collection" problem for deleted nameserver domains. In particular, by changing the deletion rules in EPP — so that deletion of a domain also removes all references (*i.e.*, nameserver delegations) to any subordinate host objects — would prevent the creation of new dangling delegations inside an EPP repository. However, fully addressing inter-registry links across EPP repositories (*e.g.*, a nameserver domain in `.com` that is used by domains in `.org`) would require a new mechanism to report such domain deletions among registries so that they too could automate the removal of links to deleted nameservers.

Based on our findings, the ICANN Security and Stability Advisory Committee (SSAC) is considering the launch of a multi-stakeholder effort to consider tradeoffs among proposed solutions and, ultimately, to publish an advisory of recommended practice.

8 ETHICAL CONSIDERATIONS

We carefully designed our study to identify and address potential ethical risks up front, evaluating potential harms through a consequentialist lens. We believe that our work introduces no new harm and, in fact, reduces the potential harm that would have existed without our research.

First, this work primarily relied on publicly available datasets and data that is implicitly public by virtue of how the DNS works (*i.e.*, the current resolution of a DNS name). Where we identified concrete risks or harms (*i.e.*, of domain hijacking) we reached out to affected registrars and registries. Moreover, we worked with these communities not only to aid in mitigating currently exposed domains but also to prevent future exposures via changes in operational practice. Finally, we chose not to highlight currently vulnerable names in this paper to avoid facilitating their exploitation.

Second, we designed our controlled experiment (Section 6.1) to have zero impact on the `.edu` domain name in question. We selected this particular domain because it did not have any operational authoritative nameservers. Thus, the domain neither resolved nor was used by the institution.[16] To further reduce potential for impact, we configured the sacrificial nameserver (under our control) to return an A record *if and only if* the request originated from our client IP address during a short testing window. All other queries received no response (as they always had before). Thus, only in our restricted environment did the sacrificial nameserver in our control return a response. Given that we did not respond, the only information that could have been revealed was the identity of the recursive resolver trying to look up one of the associated domains. While we believe such a risk is low, we further mitigated that concern by deleting all log data (and hence any record of who looked up the domain). We balanced this minimal residual risk against the value in conducting this experiment, which we conducted to validate our understanding of the problem, and that there were no mechanisms that would prevent hijacking from succeeding.

Finally, because our Institutional Review Board (IRB) is focused squarely on overseeing human subjects research (which this work is not), they were in no position to give us independent oversight. For this reason, we conferred with campus general counsel — whose remit is broader than simply human subjects research — and received their approval for our experimental design and its controls, *before* any active measurements were conducted.

9 CONCLUSION

Our primary technical discovery in this work is how an unforeseen interaction between registrar operational practices and the constraints of registry provisioning systems have made at least a *half million domains* vulnerable to hijacking. This risk arises from a long-standing undocumented registrar operational practice that bypasses restrictions on domain deletion by first renaming nameservers slated for removal. Moreover, these nameservers are commonly renamed to point to domains in different TLDs in which the registrar does not have interest or control. As a result of the process, a simple re-registration of the deleted domain does not address the vulnerability. This subtlety, combined with the fact that affected domain owner's nameserver records are modified without their knowledge, make this vulnerability particularly insidious. While most of the domains placed at risk in this manner are either unpopular or moribund, some include sites where the names carry reputation even if they do not receive much traffic (*e.g.*, law enforcement, law offices, public health departments, and even parked domains for popular brands in alternate TLDs).[17] Our work provides a comprehensive picture of this long-standing vulnerability and also describes how our outreach has led to changes in operational practices at registrars that should significantly minimize these risks going forward.

10 ACKNOWLEDGMENTS

We thank our anonymous shepherd and reviewers for their insightful and constructive suggestions and feedback. This work would not have been possible without help from many individuals and organizations. We are indebted to Ian Foster for his help with CAIDA-DZDB/DNS Coffee, and also thank Sean McNee, Jackie Abrams, Ariella Robinson, and Susan Prosser from DomainTools for help with additional data. At UCSD we thank Cindy Moore, Alistair King, Bradley Huffaker, Daniel Andersen, and Paul Biglete for supporting the software and hardware infrastructure for this project, and we thank Daniel Park for his legal guidance.

We are also very appreciative of the thoughtful help and feedback we have received from many sectors of the registrar and registry community. We are particularly grateful to Brian Dickson for his wisdom and insight with DNS arcana, Reg Levy for facilitating remediation with registrars, and Duane Wessels and Joe Abley for their insights from the registry perspective. We also are thankful to Steven DeJong, Dave Knight, Volker Greimann, Elliot Noss, Owen Smigelski, Tim April, Patrik Fältström, James Galvin, Mattijs Jonker, Warren Kumari, Raffaele Sommese, and the ICANN Security and Stability Advisory Committee for their help and feedback.

This work was supported in part by National Science Foundation grants CNS-1629973, CNS-1705050, OAC-1724853, and OIA-1937165, the Irwin Mark and Joan Klein Jacobs Chair in Information and Computer Science, and generous support from Google.

However, this approach would require consensus in the AS112 and DNSOP community, including IANA, and a revision of RFC 7534.

[16] The institution uses a related `.com` domain to host their content.

[17] Moreover, the accidental deletion of Namecheap nameservers affecting $1.6M$ domains highlights that the risk is not limited to inexperienced domain owners.

REFERENCES

[1] J. Abley, B. Dickson, W. Kumari, and G. Michaelson. 2015. AS112 Redirection Using DNAME. RFC 7535. https://rfc-editor.org/rfc/rfc7535.txt.

[2] Gautam Akiwate, Mattijs Jonker, Raffaele Sommese, Ian Foster, Geoffrey M. Voelker, Stefan Savage, and KC Claffy. 2020. Unresolved Issues: Prevalence, Persistence, and Perils of Lame Delegations. In *Proceedings of the ACM Internet Measurement Conference (IMC)*. Virtual Event.

[3] Eihal Alowaisheq, Siyuan Tang, Zhihao Wang, Fatemah Alharbi, Xiaojing Liao, and XiaoFeng Wang. 2020. Zombie Awakening: Stealthy Hijacking of Active Domains through DNS Hosting Referral. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*. Virtual Event.

[4] Matthew Bryant. 2017. The .io Error – Taking Control of All .io Domains With a Targeted Registration – The Hacker Blog. https://thehackerblog.com/the-io-error-taking-control-of-all-io- domains-with-a-targeted-registration/.

[5] CAIDA and Ian Foster. 2020. CAIDA-DNS Zone Database (DZDB). https://dzdb.caida.org.

[6] David Dagon. 2008. DNS Poisoning: Developments, Attacks and Research Directions. USENIX Security 2008, DNS Panel Talk. https://www.usenix.org/legacy/events/sec08/tech/slides/dagon_slides.pdf.

[7] Department of Homeland Security. 2019. Emergency Directive 19-01: Mitigate DNS Infrastructure Tampering. https://cyber.dhs.gov/ed/19-01/.

[8] DomainTools. 2020. Whois History. https://research.domaintools.com/research/whois-history/.

[9] D. Eastlake and A. Panitz. 1999. Reserved Top Level DNS Names. RFC 2606. https://rfc-editor.org/rfc/rfc2606.txt.

[10] Let's Encrypt. 2020. Challenge Types – DNS-01 Challenge. https://letsencrypt.org/docs/challenge-types/.

[11] S. Hollenbeck. 2009. Extensible Provisioning Protocol (EPP). RFC 5730. https://rfc-editor.org/rfc/rfc5730.txt.

[12] S. Hollenbeck. 2009. Extensible Provisioning Protocol (EPP) Domain Name Mapping. RFC 5731. https://rfc-editor.org/rfc/rfc5731.txt.

[13] S. Hollenbeck. 2009. Extensible Provisioning Protocol (EPP) Host Mapping. RFC 5732. https://rfc-editor.org/rfc/rfc5732.txt.

[14] ICANN. 2020. Centralized Zone Data Service. https://czds.icann.org.

[15] Krebs on Security. 2019. A Deep Dive on the Recent Widespread DNS Hijacking Attacks. https://krebsonsecurity.com/2019/02/a-deep-dive-on-the-recent-widespread-dns- hijacking-attacks/.

[16] 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 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, Vienna, Austria, 1414–1425. https://doi.org/10.1145/2976749.2978387

[17] P. Mockapetris. 1987. Domain Names – Concepts and Facilities. RFC 1034. https://rfc-editor.org/rfc/rfc1034.txt.

[18] P. Mockapetris. 1987. Domain Names – Implementation and Specification. RFC 1035. https://rfc-editor.org/rfc/rfc1035.txt.

[19] S. Cheshire and M. Krochmal. 2013. Special-Use Domain Names. RFC 6761. https://rfc-editor.org/rfc/rfc6761.txt.

[20] Sooel Son and Vitaly Shmatikov. 2010. The Hitchhiker's Guide to DNS Cache Poisoning. In *Proceedings of the 6th International ICST Conference (SecureComm)*. Singapore, 466–483.

[21] Thomas Vissers, Timothy Barron, Tom Van Goethem, Wouter Joosen, and Nick Nikiforakis. 2017. The Wolf of Name Street: Hijacking Domains Through Their Nameservers. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, Dallas, TX, 957–970. https://doi.org/10.1145/3133956.3133988