UNIVERSITY OF CALIFORNIA SAN DIEGO

Identifying DNS Infrastructure Hijacks using Large Scale Measurements

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Computer Science

by

Gautam Akiwate

Committee in charge:

Professor Geoffrey Voelker, Co-Chair
Professor Stefan Savage, Co-Chair
Professor Kimberly Claffy, Co-Chair
Professor George Papen
Professor Aaron Schulman

2022

The Dissertation of Gautam Akiwate is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2022

DEDICATION

To my family, and friends. Both old and new.

EPIGRAPH

The time that my journey takes is long and the way of it long.

I came out on the chariot of the first gleam of light,
and pursued my voyage through the wildernesses of worlds
leaving my track on many a star and planet.

It is the most distant course that comes nearest to thyself,
and that training is the most intricate
which leads to the utter simplicity of a tune.

The traveller has to knock at every alien door to come to his own,
and one has to wander through all the outer worlds
to reach the innermost shrine at the end.

My eyes strayed far and wide
before I shut them and said "Here art thou!"

The question and the cry "Oh, where?"
melt into tears of a thousand streams
and deluge the world with the flood of the assurance "I am!"

- Rabindranath Tagore

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

A journey, especially a long one like a PhD, can never be successful without the support of countless people both known and unknown. While enumeration comes with the risk of omissions, I will be remiss if I did not at least attempt to thank the people I know had an important role in my success.

First and foremost, I would like to thank my advisors. I had the fortune of working with not one, not two, but three amazing advisors and mentors. I would like to thank Geoff Voelker for taking a chance on a wide eyed bushy tailed student who wanted to do "research". It has been Geoff's support, guidance and attention that has been instrumental in making this work possible. I am grateful to Stefan Savage for his ability to find exciting problems and to motivate students into tackling seemingly unsolvable problems. I am also grateful to KC Claffy whose focus on the practical realities of the Internet translated this work from just academic research to having real-world impact. Finally, I would also like to thank Aaron Schulman and George Papen for being on my thesis committee and being available whenever I needed any help.

I have had the privilege of working with some amazing collaborators both from UC San Diego and other universities. And while it is a long list I insist on including them despite the unavoidable omissions. Alex Gamero-Garrido, Ariana Mirian, Audrey Randall, Ben Du, Bradley Huffaker, Cindy Moore, Duane Wessels, Enze Liu, Louis DeKoven, Mattijs Jonker, Raffaele Sommese, and Ramakrishna Padmanabhan have all been instrumental in making much of the work during my time in PhD happen.

The CSE Community at large has been an integral part of this journey. I was lucky enough to contribute to many aspects of the department like Chez Bob that make CSE special. I am thankful to the students, faculty and staff who make this department tick. Specifically, I would like to thank Nishant Bhaskar, Sorin Lerner, Alex Snoeren, John Renner, Dhiman Sengupta, Benjamin Pullman, Joe DeBlasio, Sunjay Cauligi, Jennifer Folkstead, and Matthew Scott. Additionally, my office mates over the years, who inspired me, and cheered me on: Ariana Mirian, Audrey Randall, Alisha Ukani, Enze Liu, Rob McGuinness, Alex Forencich, Stewart

Grant, Anil Yelam, and Keegan Ryan.

I cannot even begin to properly thank my family who has been unfailingly supportive. I owe much to my mother who has anchored me, and cheered on for me at my lowest points. Finally, I am greatly indebted to Purvi Desai for her patience and encouragement while I finished my degree.

Chapter 1, 2, and 3, in part, is a reprint of the material as it appears in *Proceedings of the International Measurement Conference 2020*. Gautam Akiwate, Mattijs Jonker, Raffaele Sommese, Ian Foster, Stefan Savage, Geoffrey M. Voelker, and KC Claffy. The dissertation author was the primary investigator and author of this paper.

Chapter 1, 2, and 4 in part, is a reprint of the material as it appears in *Proceedings of the International Measurement Conference 2021*. Gautam Akiwate, Stefan Savage, Geoffrey M. Voelker, and KC Claffy. The dissertation author was the primary investigator and author of this paper.

Chapter 1, 2, and 5 in part, has been submitted for publication of the material as it may appear in *Proceedings of the International Measurement Conference 2022*. Gautam Akiwate, Raffaele Sommese, Mattijs Jonker, Zakir Durumeric, KC Claffy, Geoffrey M. Voelker, and Stefan Savage. The dissertation author was the primary investigator and author of this paper.

VITA

2008 – 2012 Bachelor of Technology, College of Engineering, Pune

2013 – 2015 Master of Science, UC San Diego

2017 – 2022 Doctor of Philosophy, UC San Diego

PUBLICATIONS

"IRR Hygiene in the RPKI Era". Ben Du, Gautam Akiwate, Thomas Krenc, Cecilia Testart, Alexander Marder, Bradley Huffaker, Alex C. Snoeren, and KC Claffy. In *International Conference on Passive and Active Network Measurement, Springer, 2022.*

"Risky BIZness: Risks Derived from Registrar Name Management". Gautam Akiwate, Stefan Savage, Geoffrey M. Voelker, and KC Claffy. In *Proceedings of the Internet Measurement Conference, 2021.*

"Who's Got Your Mail? Characterizing Mail Service Provider Usage". Enze Liu, Gautam Akiwate, Mattijs Jonker, Ariana Mirian, Stefan Savage, and Geoffrey M. Voelker. In *Proceedings of the Internet Measurement Conference, 2021.*

"Home is Where the Hijacking is: Understanding DNS Interception by Residential Routers". Audrey Randall, Enze Liu, Ramakrishna Padmanabhan, Gautam Akiwate, Geoffrey M. Voelker, Stefan Savage, and Aaron Schulman. In *Proceedings of the Internet Measurement Conference, 2021.*

"Clairvoyance: Inferring Blocklist Use on the Internet". Vector Guo Li, Gautam Akiwate, Kirill Levchenko, Geoffrey M. Voelker, and Stefan Savage. In *International Conference on Passive and Active Network Measurement, Springer, 2021.*

"Unresolved Issues: Prevalence, Persistence, and Perils of Lame Delegations". Gautam Akiwate, Mattijs Jonker, Raffaele Sommese, Ian Foster, Geoffrey M. Voelker, Stefan Savage, and K. C. Claffy. In *Proceedings of the Internet Measurement Conference, 2020.*

"MAnycast2 – Using Anycast to Measure Anycast". Raffaele Sommese, Leandro Bertholdo, Gautam Akiwate, Mattijs Jonker, Roland van Rijswijk-Deij, Alberto Dainotti, KC Claffy, and Anna Sperotto. In *Proceedings of the Internet Measurement Conference, 2020.*

"Trufflehunter: Cache Snooping Rare Domains at Large Public DNS Resolvers". Audrey Randall, Enze Liu, Gautam Akiwate, Ramakrishna Padmanabhan, Geoffrey M. Voelker, Stefan Savage, and Aaron Schulman. In *Proceedings of the Internet Measurement Conference, 2020.*

"Measuring Security Practices and How They Impact Security". Louis F. DeKoven, Audrey Randall, Ariana Mirian, Gautam Akiwate, Ansel Blume, Lawrence K. Saul, Aaron Schulman, Geoffrey M. Voelker, and Stefan Savage. In *Proceedings of the Internet Measurement Conference, 2019.*

"Inferring Persistent Interdomain Congestion". Amogh Dhamdhere, David D. Clark, Alexander Gamero-Garrido, Matthew Luckie, Ricky KP Mok, Gautam Akiwate, Kabir Gogia, Vaibhav Bajpai, Alex C. Snoeren, and KC Claffy. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication, 2018.*

ABSTRACT OF THE DISSERTATION

Identifying DNS Infrastructure Hijacks using Large Scale Measurements

by

Gautam Akiwate

Doctor of Philosophy in Computer Science

University of California San Diego, 2022

Professor Geoffrey Voelker, Co-Chair
Professor Stefan Savage, Co-Chair
Professor Kimberly Claffy, Co-Chair

DNS infrastructure hijacks are a class of attacks wherein the attack is the result of an attacker controlling part or all of the DNS infrastructure for a domain. These hijacks are typically a byproduct of attackers exploiting errors and inconsistencies in how nameserver delegations are specified, or attackers gaining authority to update delegation records on behalf of the domain owner. Significantly, attacks on DNS infrastructure can impact nearly all users of a domain. Thus, understanding DNS infrastructure hijacks is of critical importance given that it undermines trust in services hosted at the hijacked domain.

In this dissertation, I directly address the challenges inherent in identifying DNS infrastructure hijacks. In particular, I demonstrate it is feasible to infer hijacks as a third-party by leveraging large-scale measurements of the DNS ecosystem supplemented by a wide array of complementary data sources which help provide a broader context for interpreting the DNS measurements. In doing so, I show how large-scale measurements can help not only identify instances of high-value domains being hijacked, but also uncover long-standing operational practices exposing large numbers of domains unbeknownst to the domain owner. I first describe a large-scale measurement study across the Internet to comprehensively identify the extent of errors and inconsistencies in nameserver delegations and how it affects the security and efficiency of the resolution process. In the course of this first study, I discovered long-standing operational practices that exposed nearly half a million domains over nine years to the risk of hijack. In a second study, I then explored in depth the domain hijacking risk caused by these undocumented operational practices in the DNS ecosystem. While the two studies highlighted *opportunistic hijacks* wherein the security of the DNS infrastructure is undermined due to actions of the domain owner or registrar, in a final project, I explored *targeted hijacks* wherein an attacker actively takes control of DNS configuration for the domain. Overall in this dissertation, I present a qualitative *and* quantitative exploration of DNS infrastructure hijacks.

Chapter 1

Introduction

The primary role of the Domain Name System (DNS) is to resolve human-readable domain names to IP addresses. In its most familiar usage, a DNS client (such as a web browser) seeking to reach a webpage will request the IP address corresponding to the fully-qualified domain name (FQDN). This seemingly simple role of DNS hides the considerable complexity in not only how this resolution works, but also how the entire system is configured and operated. DNS depends on a range of complex interactions between many technical components and organizations to function. As such, the ability of DNS to resolve a domain *securely* is dependent on these various interactions working correctly.

At the same time, the Internet has become central to essential services such as email, communication, multimedia, banking, and shopping. To facilitate quick access, organizations host these services under easy to remember domain names. For instance, an organization like UC San Diego (`ucsd.edu`) will configure their DNS nameserver delegations to map `mail.ucsd.edu` to the IP address of the mail server. Consequently, the correct and secure functioning of these services is predicated on the correct and secure resolution of the corresponding domain. As a result, the ability to *influence* the resolution of a domain is of particular significance. If a third-party can control the resolution of a domain, they can redirect users of that domain to a malicious IP address providing an avenue to compromise users. The "ownership" of the domain is immaterial in this case, since the ability to control the domain resolution makes the

third-party the *de facto* owners of the domain. This scenario, wherein the resolution of a domain is controlled by a third-party without the consent of the domain owner, is typically referred to as *DNS hijacking*. While the most widely studied DNS hijacking attacks are direct attacks on the DNS query protocol [28, 61, 72, 85, 94] itself, I study a different class of DNS hijacks where the attack results from of an attacker controlling part or all of the DNS infrastructure for the domain. I refer to this class of domain hijacking attacks as *DNS infrastructure hijacks*.

DNS infrastructure hijacks are typically a byproduct of attackers exploiting errors and inconsistencies in how nameserver delegations are specified, or attackers gaining authority to update delegation records on behalf of the domain owner [30, 56]. Significantly, attacks on DNS infrastructure can impact nearly all users of a domain. Thus, understanding DNS infrastructure hijacks is of critical importance given that it undermines trust in services hosted at the hijacked domain. In this dissertation, I directly address the challenges inherent in identifying DNS infrastructure hijacks. In particular, I demonstrate it is feasible to infer hijacks as a third-party by leveraging large-scale measurements of the DNS ecosystem supplemented by a wide array of complementary data sources which help provide a broader context for interpreting the DNS measurements. In doing so, I show how large-scale measurements can help not only identify instances of high-value domains being hijacked, but also uncover long-standing operational practices exposing large numbers of domains unbeknownst to the domain owner.

To that end, I first conducted a large-scale measurement study across the Internet to identify the extent of errors and inconsistencies in nameserver delegations, and how these issues affect the security and efficiency of the resolution process. In particular, this first study focuses on *lame delegations*, which occur when a nameserver responsible for a domain is unable to provide authoritative information about it. Using an Internet-wide measurement targeting authoritative nameservers to identify lame delegations, I showed that lame delegations of various kinds are common (affecting roughly 14% of domains queried) that they can significantly degrade lookup latency (when they do not lead to outright failure), and that they expose hundreds of thousands of domains to adversarial takeover.

Based upon my analysis of the common causes of lame delegations, I identified operational practices that exposed nearly half a million domains over nine years to the risk of hijack. In a second study, I then explored the domain hijacking risk caused by undocumented operational practices of registrars — organizations who obtain and manage domains. I found domains implicitly exposed to the risk of hijacking included domains in most popular top-level domains (TLDs) (including `.com` and `.net`), as well as legacy TLDs with tight registration control (such as `.edu` and `.gov`). Moreover, I found that this vulnerability has been actively exploited by multiple parties who, over the years, have assumed control of $163K$ domains without having any ownership interest in those names. In addition to characterizing the nature and size of this problem, I also worked extensively with the registrars on remediating the issue.

While the two studies highlighted *opportunistic hijacks* wherein the security of the DNS infrastructure is undermined due to actions of the domain owner or registrar, in my third project, I explored *targeted hijacks* where an attacker actively takes control of the DNS configuration for the domain. I used four years of longitudinal Internet-wide IP scans to build a model of deployed Internet infrastructure for every domain over time. I then analyzed this model — a deployment map — to identify suspicious deployments that strongly correlate with hijacks. I identified 41 domains, primarily belonging to government agencies over the world, that were hijacked. This work to retroactively identify targeted hijacks sets the stage for future work aiming to identify these hijacks in near real time.

This dissertation is structured as follows. Chapter 2 lays out the details of the Domain Name System (DNS) relevant to this dissertation. In Chapter 3, I describe the measurement study undertaken to comprehensively document the prevalence of lame delegations across the Internet. In Chapter 4, I document the domain hijacking risk caused by undocumented registrar renaming practices. In Chapter 5, I describe an empirical methodology to identify targeted DNS infrastructure hijacking campaigns retroactively. Finally, Chapter 6 summarizes this dissertation and charts directions for future work.

Chapter 1, in part, is a reprint of the material as it appears in *Proceedings of the International Measurement Conference 2020*. Gautam Akiwate, Mattijs Jonker, Raffaele Sommese, Ian Foster, Stefan Savage, Geoffrey M. Voelker, and KC Claffy. The dissertation author was the primary investigator and author of this paper.

Chapter 1, in part, is a reprint of the material as it appears in *Proceedings of the International Measurement Conference 2021*. Gautam Akiwate, Stefan Savage, Geoffrey M. Voelker, and KC Claffy. The dissertation author was the primary investigator and author of this paper.

Chapter 1, in part, has been submitted for publication of the material as it may appear in *Proceedings of the International Measurement Conference 2022*. Gautam Akiwate, Raffaele Sommese, Mattijs Jonker, Zakir Durumeric, KC Claffy, Geoffrey M. Voelker, and Stefan Savage. The dissertation author was the primary investigator and author of this paper.

Chapter 2

Background

The Domain Name System (DNS) provides a distributed lookup service that maps hierarchical namespace to a variety of associated resource records (RRs). In its most familiar usage, a DNS client (such as a web browser) will request the address resource records (either `A` for IPv4 or `AAAA` for IPv6) corresponding to the fully-qualified domain name (FQDN) found in a URL. To operate successfully, the DNS depends on a range of complex interactions between technical components and organizations. In this section we provide an overview of these technical components, organizations, and interactions between the different organizations.

2.1 DNS Namespace

The DNS namespace is based on a hierarchical inverted tree structure (RFC 1034 [66]). The top of this inverted tree is the DNS Root. The individual non-overlapping branches (*i.e.*, *zones*) directly below the root are commonly referred to as *top-level domains (TLDs)*. There are two types of TLDs: generic TLDs (gTLDs), and country-code TLDs (ccTLDs). While gTLDs (*e.g.*, `.com`, `.org`) are designated for general use, the use of ccTLDs (*e.g.*, `.us`, `.uk`) is determined by a relevant authority from the country. The level below the TLDs is referred to as a *registered domain* or a *base domain*. These are the "domain names" (*e.g.*, `ucsd.edu`) that belong to an organization or a person. The domain owner (an organization or a person) may use the levels under the domain to map to different services. While there are no formal conventions

as to how this namespace is used, there are many common naming practices. For instance, the web server IP address is typically mapped to the `www` subdomain. Together the label, the domain, and the TLD (*e.g.*, `www.ucsd.edu`) is referred to as the fully-qualified domain name (FQDN).

In this hierarchical namespace, there is explicit delegation of individual non-overlapping *zones* starting at the root of the DNS namespace. The DNS Root explicitly delegates authority of each TLD (*e.g.*, `.com` or `.edu`) to nameservers which are then responsible for that *zone*. These authoritative nameservers can further delegate branches of their namespace to other nameservers in turn. For instance, `.com` provides nameserver delegations for `example.com` which consequently delegates, to those nameservers, authority over all zones under `example.com` (*e.g.*, `www.example.com`). Moreover, each of these zones can further sub-delegate specific branches of the namespace under it similarly.

2.2 DNS Protocol

DNS is fundamentally a lookup service. Clients make requests, following the protocol first specified in RFC 1035 [67], to resolve individual RR's (such as `A` records) for a given fully-qualified domain name. Thus, a client seeking to reach `www.sysnet.ucsd.edu` might request its `A` record and obtain the IP address `137.110.222.10` in return. In typical use, a client's request is directed to a configured recursive resolver, either a local DNS server usually provisioned to the operating system via DHCP, or a public resolver such as Google's `8.8.8.8`. If they do not have an appropriate and fresh answer in their cache, recursive resolvers take responsibility for performing the series of distributed requests needed to complete the resolution, or to identify that the resolution cannot be satisfied (*e.g.*, resulting in an `NXDOMAIN` response).

2.2.1 Recursive resolution

Recursive resolvers use the same protocol as clients, but parse the domain from left to right, dropping a domain name's prefixes until they encounter a portion of the namespace for which they know of an authoritative server to query.

Absent any previously cached information, all recursive resolvers at least include the hard-coded IP addresses of the global DNS root servers. These servers will not be able to provide authoritative information about the FQDN being queried, but will return authoritative information about the nameserver (`NS`) records for the associated top-level domain (TLD).[1] We say that these `NS` records represent a *delegation* of the namespace. For example, nameservers for `.edu` are delegated responsibility for the namespace below `.edu`. Then, using an appropriate TLD nameserver, the recursive resolver can issue its query again, each time obtaining answers about nameservers responsible for a more narrowly delegated portion of the namespace until a nameserver is reached that can provide an authoritative `A` record, identifying the IP address for the original query received from the client.

As a concrete example, a query for `www.sysnet.ucsd.edu` to a newly started recursive resolver might produce a request to a root server which, in turn, would reply with `NS` records for the `.edu` nameservers (*i.e.*, `[a-m].edu-servers.net`). Sending the same request to these servers would produce a reply pointing to the `ucsd.edu` nameservers (`ns-auth1.ucsd.edu`, among others) which, upon being queried themselves, would point to the `sysnet.ucsd.edu` nameservers (*i.e.*, `ctrl.ucsd.edu`, among others).[2] Finally, the authoritative nameservers for `sysnet.ucsd.edu` would provide the resulting `A` record for `www.sysnet.ucsd.edu`.[3]

2.2.2 Glue records

It is important to note that `NS` records are names themselves (*e.g.*, `ns-auth1.ucsd.edu`) and a recursive resolver must obtain `A` records for those names to properly contact them. This resolution can be problematic, however. For instance, if the domain `example.com` is dele-

[1] These records include legacy gTLDs such as `.com` and `.edu`, country-code TLDs (ccTLDs) such as `.uk` and `.ru`, and 1000+ new generic TLDs such as `.xyz`.

[2] Note that there is no requirement that each "." in the domain name represent a delegated portion of the namespace. Indeed, while it so happens that `sysnet.ucsd.edu` operates in a separately delegated "zone" from `ucsd.edu`, that delegation is an administrative choice. In an alternate implementation, `ns-auth1.ucsd.edu` could have provided an authoritative `A` record for `www.sysnet.ucsd.edu` directly.

[3] Note that this complete set of queries is rarely performed in practice because answers, at each level of the namespace, are cached for the period of time designated in the time-to-live (TTL) field in each nameserver answer.

gated to `ns1.example.com` (a common idiom), there is no way to query `ns1.example.com` to obtain its IP address. For this reason, the DNS protocol allows nameservers to provide additional records, called *glue records*, which are `A` or `AAAA` records for the identified nameservers (`ns1.example.com` in this example). To improve latency, nameservers may also provide *sibling glue records*, which are glue records for sibling domains in the zone file. Thus, it is common for nameservers to provide corresponding `A` or corresponding `AAAA` records (*i.e.*, IP addresses) for any `NS` records they return authoritative answers for. Critically, a requester will only accept additional records that are *in-bailiwick*, *i.e.*, portions of the namespace for which the server provides authoritative answers. `NS` records that are *out of bailiwick* for a domain will typically not include glue, since resolvers will not accept them. For example, delegating `example.com` to nameserver `ns1.example.org` would be glue-less; the `.com` TLD nameservers would not provide glue for `ns1.example.org`.

It is important to note that in the resolution process, each zone implicitly trusts its *parent* zone (*e.g.*, `.com` is the parent zone of `example.com`) to provide the correct nameserver delegations. A delegation to an incorrect nameserver compromises the security of the entire underlying zone.

2.3 DNS Namespace Management

Together, the DNS standard (RFC 1034 and RFC 1035 [66, 67]) specifies the implementation of the namespace, domain names, and the protocol to query the domain names. However, it does not detail how these domain names can be procured, or how the namespace delegation is managed. Those aspects of DNS are important to its correct functioning. Moreover, those aspects of DNS also present an attack surface.

For most TLDs, the Internet Corporation for Assigned Names and Numbers (ICANN) delegates the administration of those zones under contract to organizations called *registries*.[4]

[4]While there is considerable latitude in how TLDs representing sovereign naming interests (ccTLDs) are administered, ICANN requires an organization to administer the ccTLD as the ccTLD registry in the public

Registries manage the database of domains under the TLD along with the nameserver delegations. Note, the registries are also responsible for operating the authoritative nameservers that delegate authority for the domains under the TLD. At times a single organization is responsible for multiple TLDs; at times some registries will outsource the technical operation of the TLD to organizations who specialize in running registry operations. For example, Verisign is the registry for the `.com` and `.net` TLDs (among others) and *also* implements the registry backend for `.edu` on behalf of EDUCAUSE and `.gov` on behalf of the US Cybersecurity and Infrastructure Security Agency (CISA).

While registries provide nameserver delegation for the domains under the TLD, 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` 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.[5] Importantly, registrars can contract with many registries *and* there can be many registrars who contract with each individual registry.

2.3.1 Name Registration and Provisioning

The technical mechanism for interfacing between registrars and registries is the Extensible Provisioning Protocol (EPP) principally documented in RFC 5731 and RFC 5732 [48, 49]. Registries use EPP to provide a degree of administrative access to the registry database and to allow registrars the ability to install newly registered domains into the database and manage the records for those domains. EPP provides a degree of isolation between registrars and ensures the

interest [40].

[5]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).

consistency of the overall database.

EPP's consistency constraints can have unintuitive consequences. For example, one registrar can create a host object entry in EPP to delegate a nameserver (`ns1.example.com`) for a domain that they have registered. If a different registrar registers a domain that uses `ns1.example.com` as *its* nameserver, then the first registrar will no longer be able to delete the host object `ns1.example.com` nor its associated domain object `example.com`, so long as the other domain registered by the second registrar continues to use `ns1.example.com`. In addition to the baseline constraints of EPP, registries and registrars can impose their own restrictions on names registered through them.

Finally, many names in the DNS rely on multiple registries. For instance, `example.com` might have two nameservers spanning two TLDs: `ns1.example.com` and `ns1.example.org`. While the registry for `.com` (Verisign) is in a position to validate and enforce policies about `ns1.example.com`, it is unable to do the same for the `NS` records (`ns1.example.org` in this case) outside its authority.

2.3.2 Updating DNS Delegations

Following the successful purchase of the domain until the domain expires or transferred, the registrar is also responsible for updating the registry database with the DNS configuration for the domain (including nameserver delegation) as requested by the registrant. To resolve the domain, the registrant has to ensure that delegations are correctly reflected in the registry database — the *parent zone*. Moreover, the registrant needs to ensure that any updates to the nameservers or the nameserver IP addresses are communicated to the parent zone. These updates are communicated to the registry by the registrant via the registrar. As shown in Figure 2.1, the registrant uses their account at the registrar to update their nameserver or other DNS delegations. In turn, the registrar, uses the Extensible Provisioning Protocol (EPP) [47, 48, 49] to interface with the registry to propagate any changes to the nameserver delegation and the corresponding IP addresses — glue records — to the registry.

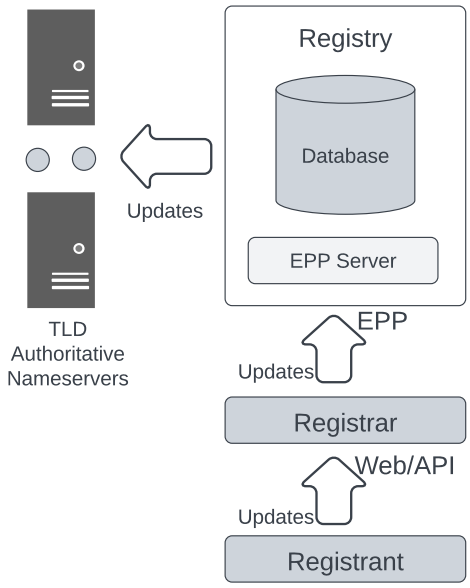

Figure 2.1. Flow of DNS Delegations Updates.

2.3.3 Nameserver Hosting

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. 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 `ns2.example.com`.[6] 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.

[6]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 dissertation.

2.4 DNS Hijacks

In normal circumstances, resolution of a domain name is under the control of the registrant or someone with whom the registrant has a contractual agreement. When an attacker controls the domain resolution, they are in essence the *de facto owners* of the domain since their control over resolution is the capability that matters with regards to a domain. When resolution for a domain is thus controlled by a third party, without the consent of the actual domain owner, it is typically referred to as a *hijack.* Crucially, if an attacker controls resolution, they can redirect users of that domain to a malicious IP address providing an avenue to compromise users. While in principle Domain Name System Security Extensions (DNSSEC) [78] and Transport Layer Security (TLS) Certificates [75] are designed to protect against these attacks, these mechanisms can be subverted [56].

The focus of this dissertation is a class of DNS hijacks referred to as *DNS infrastructure hijacks* where the hijack is the result of an attacker controlling part or all of the DNS infrastructure for the domain. DNS infrastructure hijacks are typically a byproduct of attackers *opportunistically* exploiting errors and inconsistencies in how nameserver delegations are specified, or attackers gaining authority to update delegation records by *targeting* high-value domains [30, 56].

Opportunistic DNS Infrastructure Hijacks. In these hijacks, the attack on the DNS infrastructure is not due to actions of an attacker but that of the domain owners or the registrars. Typically, these attacks exploit *lame delegations* which occur when a domain has one or more delegated nameservers that are unable to provide authoritative responses for the domain [69]. Sometimes the underlying nameserver exists in a non-existent second domain, which an attacker can acquire and (partially) control resolution for the first domain [62, 8]. This *dangling delegation* subcase can occur due to expired domains [18][7] or intentional operational practices by registrars to deal with constraints in the DNS provisioning protocol. We explore lame delegations in

[7] In 2017, a dangling delegation at the `.io` TLD left every domain in the TLD vulnerable to hijack [18].

Chapter 3, and the intentional registrar operational practices in Chapter 4.

Targeted DNS Infrastructure Hijacks. In these hijacks, the attack results from an attacker targeting and taking control of mechanisms to update DNS configuration for a domain. At a high level, these hijacks are supply chain attacks, wherein an attacker targets a service provider (in this case DNS nameservice) to eventually compromise the target. In these attacks, the attacker replaces the `NS` records for the domain with nameservers controlled by the attacker, and updates specific `A` records pertaining to targeted subdomains (*e.g.*, `mail`, `vpn`) to point to the attacker's infrastructure. Initial reports of this type of attack identified use for activism or mischief [45, 87]. More recently, sophisticated nation state actors used this attack to compromise government agencies as well as large infrastructure providers [4, 29, 46, 65, 86]

Chapter 2, in part, is a reprint of the material as it appears in *Proceedings of the International Measurement Conference 2020*. Gautam Akiwate, Mattijs Jonker, Raffaele Sommese, Ian Foster, Stefan Savage, Geoffrey M. Voelker, and KC Claffy. The dissertation author was the primary investigator and author of this paper.

Chapter 2, in part, is a reprint of the material as it appears in *Proceedings of the International Measurement Conference 2021*. Gautam Akiwate, Stefan Savage, Geoffrey M. Voelker, and KC Claffy. The dissertation author was the primary investigator and author of this paper.

Chapter 2, in part, has been submitted for publication of the material as it may appear in *Proceedings of the International Measurement Conference 2022*. Gautam Akiwate, Raffaele Sommese, Mattijs Jonker, Zakir Durumeric, KC Claffy, Geoffrey M. Voelker, and Stefan Savage. The dissertation author was the primary investigator and author of this paper.

Chapter 3

Unresolved Issues: Prevalence, Persistence, and Perils of Lame Delegations

In this chapter, we detail our large-scale measurement study across the Internet to identify the extent of errors and inconsistencies in nameserver delegations and how it affects the efficiency and security of the resolution process. Since these errors and inconsistencies in nameserver delegations typically manifest themselves as lame delegations, which occur when delegated nameservers are unable to provide authoritative answers for a domain, we chose to measure the prevalence, persistence, and perils of lame delegations in the DNS ecosystem longitudinally.

3.1 Overview

The Domain Name System (DNS) plays a critical role in the functioning of the Internet by resolving human-readable domain names into routable IP addresses (among other tasks). Because this function is distributed, its operation implicitly depends on the nature of the delegations configured across the DNS namespace. In particular, the ability of a domain to be efficiently resolved is predicated on all of its nameservers being resolvable and that those nameservers, in turn, are able to provide authoritative answers. In the common case, all of these requirements are satisfied, but there are a significant minority where they are not.

When a nameserver is delegated authority over a domain, but is unable to provide authoritative answers about that domain, a *lame delegation* is created. In the best case, lame

delegations can result in increased resolution latency, as queries must timeout and be redirected to other hopefully correctly configured nameservers. However, in other situations, lame delegations can provide sufficient purchase for attackers to monitor or hijack DNS resolution.

In this chapter, we explore the prevalence and causes of such lame delegations in the DNS name hierarchy. We explore this issue both longitudinally, using nine years of zone snapshot data comprising over 499 million domains in both legacy and new generic TLD (gTLD) namespaces (respectively, *e.g.*, `.com` and `.xyz`) as well as in the current DNS namespace using active measurements covering over 49 million domains. We find that lame delegations are relatively common, roughly 14% of registered domains actively queried have at least one lame delegation and the clear majority of those have no working authoritative nameservers. We identify reasons why lame delegations persist, including: cross-zone delegations, which current protocols are unable to validate; and non-working IP addresses in glue records, which similarly cannot be validated statically using registry zone data. Moreover, we identify an unforeseen interaction between existing registrar practice and the constraints of registry provisioning systems that has inadvertently created hundreds of thousands of lame delegations.

Our measurements show that lame delegations can have significant impacts even when there are alternative working authoritative nameservers for a domain. Lame delegations can result in a significant increase in average resolution latency (3.7×), unnecessary load on existing nameservers (roughly 12% of requests to GoDaddy's nameservers are for domains for which they are not authoritative [73]) and, most importantly, the potential for malicious parties to monitor or hijack DNS lookups. We have identified many tens of thousands of domains vulnerable to such hijacking and, in several instances, we have identified single domains that, if registered by an attacker, would have allowed the hijacking of thousands of domain names. Finally, we describe our efforts working with the registrar and registry communities to understand the source of these problems and establish efforts to address them going forward.

Table 3.1. Example lame delegation due to typos.

Records	Type
`foo.com` NS `ns1.example.com`	Well Configured
`foo.com` NS `ns2.exmple.com`	Misconfigured
`ns1.bar.com` A 132.239.1.1	Well Configured
`ns2.bar.com` A 13.239.1.1	Misconfigured

3.2 Lame Delegations

Absent issues like network or server outages, every fully-qualified domain name should be resolvable by any nameserver delegated to provide authoritative answers for that portion of the namespace. However, as this chapter documents, there are a significant number of cases where this is not so. In particular, a range of configuration errors produce *lame delegations* — a situation where an `NS` record for a given domain does not lead to authoritative answers for that domain. Lame delegations result in wasted DNS queries, sometimes to hosts that do not even exist [14, 77].

In some cases all of a registered domain's delegations are lame. It is also possible for a domain to be *partly lame*, *i.e.*, at least one nameserver is deficient, but not all of them. The former case is likely to be fixed quickly because the namespace is unusable. Partly lame domains are more insidious because name resolution continues to operate, but with increased latency and potential security risks. The increased latency arises because if a recursive resolver uses the lame nameserver first, it will need to timeout before it will try a correctly configured nameserver.

The potential for security risk is more nuanced. Consider the case in which the misconfiguration is a result of a typo such as shown in Table 3.1. Whoever controls `exmple.com` can control the resolution for the fraction of requests for `foo.com` that are resolved through the `ns2.exmple.com` nameserver. Similarly, whoever has control of the mistyped IP address can control the resolution of the domain names that use `ns2.bar.com`. Lame delegations create an attack surface for would-be hijackers of the delegating domains.

3.3 Related Work

The complexity of DNS configuration, and associated prevalence of misconfigurations, was recognized decades ago [14, 77]. In 2004, Pappas *et al.* used active measurements to study ∼52 k domain names and found that on average about 15% of registered domains under several TLDs (*i.e.*, `.com`, `.net`, `.org`, `.edu` and various ccTLDs) had lame delegations [69].

A TLD may contain glue records for a nameserver, even when the registered domain name of the nameserver has expired. Such a nameserver is considered *orphaned*. Kalafut *et al.* [55] passively analyzed six TLDs over a 31-day period in April 2009, and identified 16 k orphan nameservers per day on average. The TLDs under consideration accounted for about 60% of all domains on the Internet at the time. Kalafut *et al.* also found that certain TLDs accounted for a disproportionate number of orphan records, and that some orphans were evidently used for malicious purposes. In 2019, Sommese *et al.* [83] revisited this behavior. They found that some TLDs had fewer orphan nameservers than 10 years earlier, but other TLD operators had more orphan records than before, and they were prevalent among new gTLDs. Notably `.com` and `.net` no longer had any, implying those TLD operators are now automatically preventing them.

Liu *et al.* investigated the presence of *pointers* to invalid resources in the DNS, a type of *dangling DNS record* [62]. They used active measurement to highlight dangling records created by use of ephemeral IP addresses on cloud services and via expiring domains.

Lame delegations can also occur with reverse delegations. Some Regional Internet Registries (RIRs) automatically detect lame reverse delegations, such as APNIC [10] and LACNIC [59]. ARIN previously had a similar policy, but retired it in 2014 [11]. In 2016, Phokeer *et al.* showed that reverse delegations are frequently lame in AFRINIC's `41.in-addr.arpa` zone [70]. At the time AFRINIC did not have automated detection, but later instituted it [5] and substantially reduced the prevalence of lame reverse delegations [6]. Our study focuses on forward delegations, which determine control and availability of mappings.

In 2020, Sommese *et al.* [84] found that ∼8% of registered domains under the largest

Table 3.2. TLD zone files per year in DNS Coffee data set.

Year	TLD Zone Files	Year	TLD Zone Files
2011	12	2016	1221
2012	12	2017	1237
2013	49	2018	1241
2014	462	2019	1235
2015	828	2020	1206

Table 3.3. Records in the DNS Coffee data set.

Domains	**Nameservers** (`NS`)	**IPv4** (`A`)	**IPv6** (`AAAA`)
499.3 M	19.9 M	5.1 M	91.9 k

gTLDs (*i.e.*, `.com`, `.net` and `.org`) have inconsistent parent (delegation) and child zones. They investigated the risk of such inconsistencies to the availability of misconfigured domain names.

These previous studies used only active measurements to study delegation-related security risks in the DNS namespace. Ours is the first to use comprehensive collections of both active and passive DNS measurements to explore and quantify these risks, allowing us to not only identify long-term trends in lame delegations, but also analyze root causes of their surprising prevalence in some cases.

3.4 Data Sets

We use two data sets for analysis: a passive collection of TLD zone files, and a data set of active DNS resolutions.

3.4.1 DNS Coffee: TLD Zone Data

Our primary data set is a large collection of zone files from the `dns.coffee`[1] service [26]. This data set contains daily snapshots of zone files from April 2011 through January 2020, covering nearly nine years. Over time, as zone files for new TLDs became available, `dns.coffee`

[1] CAIDA now offers the same collection through `CAIDA-DZDB` at https://dzdb.caida.org

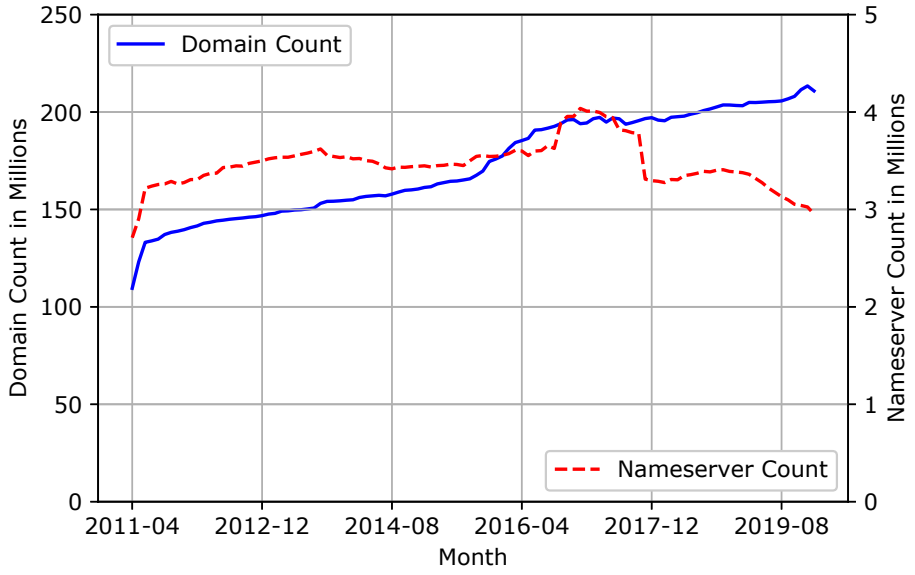

Figure 3.1. Number of distinct domains and nameservers in DNS Coffee zones over the years.

added them to its collection. Table 3.2 shows the number of unique TLDs collected over time, and Figure 3.1 shows the number of distinct domains and nameservers across the zone files every year. As of September 2020, the service collects zone files for over 1250 different zones on an ongoing basis. The snapshots include the zone files of legacy generic TLDs (gTLDs), the `.us`, `.se` and `.nu` country-code TLDs (ccTLDs), and new generic TLDs (ngTLDs) made available through the ICANN Centralized Zone Data Service [53].

One issue when analyzing zone files is that records can refer to TLDs outside of that zone. As we aggregate records from the zone files together, cross references across zones are automatically consolidated in the data set. However, for records that refer to TLDs for which we do not collect zone files, we have to make assumptions, *e.g.*, that the resolution is valid.

3.4.2 Active DNS Measurement

Certain characteristics of real-world DNS behavior cannot be learned from zone files. Zone files may list `NS` records for nameservers that do not have authoritative data, are not reachable, or do not even exist. Active measurement data can reveal these additional insights into

lame delegations, although capturing comprehensive data would require exhaustively querying all nameservers listed in the zone files for a given domain, and all IP addresses for each nameserver. Open data sets like the OpenINTEL [89] project do not exhaustively query all nameservers in the zone file; instead they perform resolutions as a typical nameserver would, and stop when they receive an authoritative response for a domain. This approach will not capture comprehensive data on availability or authoritativeness of nameservers listed in the zone file, a particular problem for partly lame delegations. Thus, to gain a more comprehensive picture, we perform our own active DNS measurements. We describe our methodology for doing so later, in Section 3.6.1. Given the intrusive nature and overhead of exhaustive probing, we limit the number of domains we actively probe. Further, we supplement these measurements with OpenINTEL data to ascertain the potential "real-world" impact of lame delegations.

3.5 Lame Delegations Inferred from Zone Files

Our first analysis uses the nine years of zone file data to identify unresolvable nameservers that cause lame delegations. We delineate three periods of a nameserver's lifetime during which lame delegations occur, each period associated with different causes and implications. In this context, we characterize the prevalence of unresolvable nameservers and affected domains overall, how long domains are lame delegated, and how an unusual concentration in the `.biz` TLD reveals an undocumented registrar operational practice. We then examine unresolvable nameservers and lame delegations longitudinally over the nine years, identifying trends, prominent events that indicate causes of large-scale lame delegations, and their associated risks.

3.5.1 Methodology for Static Analysis

Our analysis of longitudinal zone file data performs "static resolution" of domains and nameservers to identify unresolvable nameservers that lead to lame delegations. Specifically, we infer lame delegations by following chains of records in zone files to establish that a nameserver has a valid resolution path. We use the zone file snapshots over time to derive the date ranges for

Table 3.4. Mock `NS` and `A` records to illustrate static resolution.

TLD	Records	Start Date	End Date
.com	foo.com NS ns1.bar.in	2011-04-11	2013-10-31
.com	foo.com NS ns1.baz.org	2011-06-18	2013-10-31
.com	foo.com NS ns1.qux.org	2011-04-11	2013-10-31
.com	foo.com NS ns.1qux.org	2011-06-11	2013-10-31
.com	foo.com NS ns1.thud.org	2011-06-11	2013-10-31
.org	thud.org NS ns1.baz.org	2011-06-06	2013-10-31
.org	ns1.baz.org A 93.14.2.34	2011-06-06	2013-10-31
.org	ns1.qux.org A 93.14.2.36	2011-06-06	2013-09-01

Table 3.5. Resolvability at the end of static resolution.

Nameserver	Start Date	End Date	Reason
ns1.bar.in	2011-04-11	2013-10-31	Other TLD
ns1.baz.org	2011-06-06	2013-10-31	Glue Record
ns1.qux.org	2011-04-11	2011-06-05	Late Access
ns1.qux.org	2011-06-06	2013-09-01	Glue Record
ns1.thud.org	2011-06-06	2013-10-31	Parent Resolution
ns.1qux.org	-	-	No Records

when each nameserver has a valid resolution path. We then identify the registered domains that depend upon the nameservers during their valid time periods. Any $(domain, nameserver)$ pair where the domain relies on a nameserver outside of that nameserver's periods of valid resolution is a lame delegation. We refer to registered domains in the zone files simply as domains, and specifically mention in context if a domain is a fully qualified domain name.

To explain this static resolution process, we use mock `NS` and `A` records (Table 3.4) to show how we use four criteria to derive the "valid resolution" date ranges for the nameservers (Table 3.5). Each record has a start and an end date. For each TLD we record the first time we imported the zone file for that TLD. The earliest information we have for `foo.com`, and generally any domain in `.com`, is `2011-04-11`. We derive resolution validity if any of these four criteria hold:

(1) **Other TLDs:** Domains in our set of zone files can have `NS` records with nameservers

in TLDs for which we do not have a zone file. In Table 3.4, `foo.com` has a nameserver `ns1.bar.in`. Since we do not have the zone file for the `.in` TLD we conservatively assume that `ns1.bar.in` can be resolved from `2011-04-11` to `2013-10-31` (Table 3.5). Of the ∼20 M nameservers in our zone file data set, 1.4 M (7%) of them belong to such TLDs and we assume that they are resolvable.

(2) **Late Access TLDs:** We do not always have the earliest zone file for a given TLD, *e.g.*, our earliest copy of the `.org` TLD zone file is from `2011-06-06`. If we see earlier references to nameservers in the `.org` TLD in other zone files, we conservatively mark them as resolvable for the duration before we have visibility into the TLD.

(3) **Glue Records:** If a nameserver has a glue record in the zone files, then we assume that the nameserver is resolvable for the duration of the glue record. In Table 3.4, `ns1.baz.org` and `ns1.qux.org` have glue records that make them resolvable for the durations shown in Table 3.5.

(4) **Parent Resolution:** Domains using a nameserver that does not have a glue record can still resolve via the resolution on the nameserver's parent domain. In Table 3.4 consider `ns1.thud.org`. While `ns1.thud.org` does not have a glue record, the nameserver parent `thud.org` can be resolved by `ns1.baz.org` since it has a valid resolution path via its glue records. Thus, in Table 3.5 we consider `ns1.thud.org` resolvable from `2011-06-06` to `2013-10-31` as a result of parent resolution. Determining parent resolution may involve multiple layers of redirection before reaching a nameserver with a valid resolution path. Otherwise, a nameserver without a glue record is unresolvable.

We illustrate this static analysis process by working through the mock examples in Tables 3.4 and 3.5. Table 3.5 presents the durations for which a nameserver is conservatively resolvable. Nameservers `ns1.bar.in`, `ns1.baz.org`, and `ns1.thud.org` have a valid resolution path for the entire period during which they are the nameservers of `foo.com`. However, consider `ns1.qux.org` whose glue record is valid only until `2013-09-01`. Thus, we infer `ns1.qux.org` was unresolvable for the period `2013-09-02` to `2013-10-31`. Additionally `ns.1qux.org`, an

Table 3.6. Summary of unresolvable nameservers in our zone file data set, broken down by nameserver TLDs. Includes unresolvable nameservers as percentage of all nameservers in same TLD. We categorize by TLD of nameserver and not the domain. Recall that nameservers and domains can be lame in more than one time period, so the sum of the time period columns is generally larger than the overall total.

	Unresolvable Nameservers by TLD				All TLDs
TLD	**Unr. NS**	**Pre Unr. NS**	**In Life Unr. NS**	**Post Unr. NS**	**Lame Dom.**
.com	367,054 (4.25%)	17,660 (0.20%)	277,379 (3.22%)	85,899 (1.00%)	2,122,825
.net	85,039 (4.91%)	2,531 (0.14%)	61,372 (3.55%)	24,997 (1.45%)	899,082
.org	34,669 (3.51%)	828 (0.08%)	17,540 (1.78%)	17,438 (1.77%)	246,486
.info	39,184 (3.28%)	831 (0.07%)	29,092 (2.44%)	10,207 (0.86%)	67,796
ccTLDs	9,480 (1.41%)	333 (0.05%)	4,947 (0.74%)	4,920 (0.73%)	28,193
ngTLDs	28,472 (0.57%)	2,830 (0.06%)	12,351 (0.25%)	14,474 (0.29%)	446,906
.biz	191,211 (50.8%)	7,968 (2.12%)	8,454 (2.25%)	181,211 (48.1%)	551,201
Total	755,109 (4.07%)	32,981 (0.18%)	411,117 (2.22%)	339,146 (1.83%)	4,114,750

example of a typo of the actual nameserver, never has any records associated with it. This typo results in a security risk since someone can register `1qux.org`, set the glue record for `ns.1qux.org` to a private nameserver, and control the resolution of `foo.com` for the fraction of requests that come its way.

Applying the static analysis across all nameservers for the full time period of our data set, we delineate a nameserver's "unresolvability", *i.e.*, when it is unresolvable, across three periods:

1. **Pre-Life**: The nameserver is referenced by a domain before the nameserver is first resolvable, typically due to delayed glue or delayed registration of the nameserver domain.

2. **In-Life**: The nameserver is temporarily unresolvable after previously being resolvable. The most common type of lame delegation, it is frequently the result of a nameserver domain expiring and then being renewed, or its glue records being misconfigured.

3. **Post-Life**: The nameserver is no longer resolvable or was never resolvable. Typically, it is a result of an expired nameserver domain not being renewed, or a typo when entering the nameserver domain.

We found these categories useful for identifying causes and implications of lame delegations.

Our static resolution assumes that a nameserver with a glue record is routable, reachable, and operates an authoritative DNS server for the domain. Consequently, the static analysis results are lower bounds on unresolvable nameservers and lame delegations. Even so, static analysis uncovers a wide variety of DNS behavior that leads to lame delegations. Complementing this analysis, Section 3.6 describes our active measurements that derive a snapshot of lame delegations via operational execution of the DNS protocol.

3.5.2 Prevalence of Lame Delegations

We start by characterizing the overall prevalence of unresolvable nameservers across the zone files in our data set. Table 3.6 shows the total number of unresolvable nameservers and the total number of domains affected. The table also includes two breakdowns of the overall numbers: by time period (columns), and by TLD (rows).

Unresolvable nameservers may be a small percentage of nameservers (4%), but they result in more than 4.11 M lame delegated domains. Most unresolvable nameservers are unresolvable in-life, which is not surprising since they correspond to issues at any point during a nameserver's lifetime. The smallest category of unresolvable nameservers are those that are unresolvable pre-life; these cases are typically delayed registration of the nameserver domain. EPP constraints do not allow unregistered nameserver domains in the same TLD, so this situation arises only when the nameserver domain is in a different TLD from the domain itself.

Characterizing the prevalence of unresolvable nameservers by TLD, we found that unresolvable nameservers appear more often and with roughly similar prevalence in the old generic TLDs in the first four rows: between 3–4%. Country-code TLDs have comparatively fewer unresolvable nameservers, and the many new gTLDs grouped under ngTLDs have the fewest unresolvable nameservers.[2] While domain management practices could be better in the

[2]We examined unresolvable nameservers among the individual gTLDs in the ngTLDs group and no particular gTLD stood out.

newer TLDs, a common practice in the new gTLDs is to have the NS records for domains point to nameservers in another TLD, often .com. Our method attributes any resulting unresolvable nameserver to .com and not the newer gTLD.

3.5.3 DROPTHISHOST Anomaly

We placed .biz at the end of the Table 3.6 since it stands out in sharp contrast to other TLDs. The .biz TLD has had 381,475 nameservers across nine years of zone files. Of these, nearly half had no valid resolution path ever, yet domains still pointed to them. These results uncovered a long-standing undocumented practice among some registrars when dealing with expired nameserver domains.

Nearly 66% of these unresolvable .biz nameservers (118,905) have the substring "DROPTHISHOST" followed by a random unique string (indicating a generated GUID) in their FQDN. Very few of these nameserver domains have ever been registered, placing the domains served by the nameservers at risk of hijacking. Examining the history of such domains revealed a pattern: the change in their NS records to a unique "DROPTHISHOST" nameserver happens after the previous nameservers in the NS records expire.[3]

The naming, scale, and longevity of this pattern suggested systematic behavior. We reached out to the .biz registry and a large registrar to understand our findings. The registry was unaware of the extent of the issue because they had no visibility into it—these nameservers are not actually registered in .biz, and hence .biz does not have any records for them in its registry database. They just appear as names in NS records in the databases of other TLDs.

The registrar solved the mystery. For decades registrars have used an undocumented practice to clean up expired nameserver domains, a practice developed in response to a situation created by requirements of the EPP specification. A registrar cannot delete the record for a nameserver domain that expires if there are other records (*e.g.*, domains) in the same TLD that

[3]As an example, see the current and past nameservers of a test domain at https://dns.coffee/domains/ORPHAN-FINDER.COM.

refer to a host object for that domain (Section 2.3.1). However, by crafting a nameserver hostname in another TLD, and updating the host object record to use this "sacrificial" nameserver hostname instead — in effect updating the `NS` record of all domains referring to the original nameserver to use the new sacrificial nameserver host in a different superordinate domain — the registrar can then garbage collect the original expired nameserver object (RFC 5731 Section 3.2.2 [48]). Domains pointing to the sacrificial nameserver become lame delegated, but domain owners can always change the `NS` records to use a valid nameserver again if they choose. Anecdotally, it appears registrars chose `.biz` because it was a new gTLD at the time.

There are a few potential options to solve the problem going forward. The first option is to create sacrificial nameserver domains under a "sink" domain that the registrar controls. Some registrars already use this option. However, this option leaves the registrar responsible for answering queries for lame delegated domains, and for operating the "sink" domain. Another option would rely on the AS112 project `empty.as112.arpa` [3], which established a distributed anycast service that DNS operators could use to sink DNS traffic relating to parts of the global namespace under their control. Doing so would not require coordination among zones, and would ensure that such nameserver domains would never be registered by another party. To minimize query latency, responses could return `NXDOMAIN` with a long TTL. But this project relies on volunteers willing to donate resources to operate an AS112 anycast server. More concerning, a malicious actor could set up their own AS112 server and hijack queries intended for the AS112 server. More recently, however, ICANN's Security and Stability Advisory Committee (SSAC) has recommended that ICANN reserve a private-use TLD that might offer a useful path forward [54] to resolving this issue.

The registrar we talked with was also surprised at the extent of the current situation; indeed, our findings motivated a change in their operational practice. However, cleaning up the existing DROPTHISHOST and similar sacrificial nameservers is more challenging. Given the restrictions in EPP that prevent external records from being modified (Section 3.2.5 of RFC 5732 [49]), purging these records will require coordination among registrars whose domains

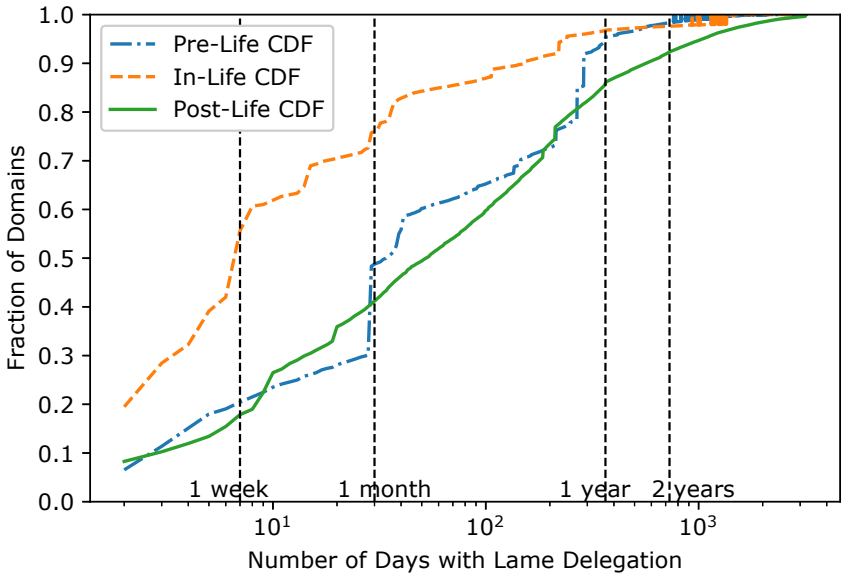

Figure 3.2. Fraction of domains with lame delegations for at most X days.

point to such nameservers and the registrars who created them. We plan to continue working with the registrar and registry communities to find a viable alternative approach to renaming expired nameservers as well as cleaning up the existing records.

3.5.4 Duration of Lame Delegations

How long do lame delegations persist? Figure 3.2 shows the fraction of domains with lame delegations to pre/in/post-life unresolvable nameservers for at most X days. Domains that are lame as a result of in-life lame nameservers are lame for the shortest time: nearly 50% of the affected domains are lame delegated for less than a week. These lame delegations suggest intermittent causes such as misconfigurations that are discovered relatively quickly. The mode at five days reflects an event in November 2011 where `cwgsh.com` and all of the nameservers under it became unresolvable after the domain `cwgsh.com` expired, causing nearly 60 thousand domains to have lame delegations.

Both pre-life and post-life unresolvable periods of nameservers have durations substantially longer than in-life unresolvable periods. For pre-life periods, the inflection at 29 days is due

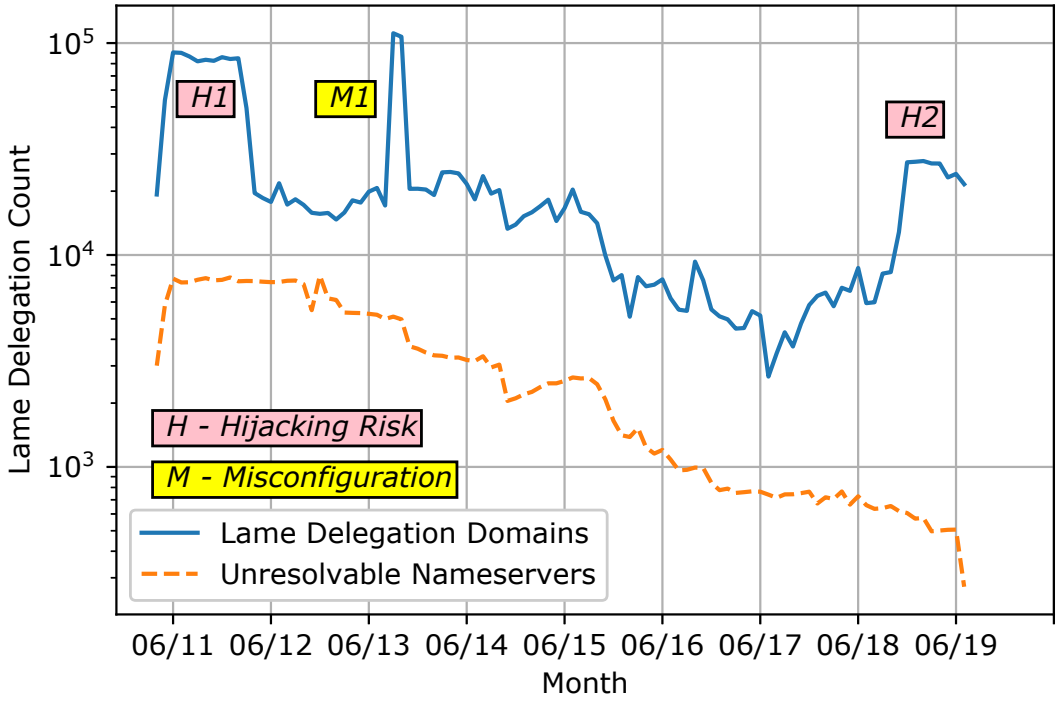

Figure 3.3. Pre-life lame delegations (blue) due to dependency on nameservers that are not yet unresolvable (red), because the nameserver domain or associated glue is not yet active.

to a misconfiguration of `nic.tel`, and the last inflection corresponds to an issue with `cwgsh.org`, which had domains pointing to it for 289 days before it was registered (Section 3.5.5).

The distribution of post-life unresolvable periods has the longest tail, reflecting intentional use of lame delegations to park domains. Some domains are lame for up to 3,000 days, nearly the timeframe of our data set. Parking domains for long durations is a risk since the nameserver domain can mistakenly be allowed to expire, exposing them to hijacks (Section 3.5.5).

3.5.5 Lame Delegations over Time

The duration of our zone data set allowed us to analyze long-term trends in lame delegations caused by unresolvable nameservers. We observed trends, discovered prominent events, and considered associated risks. For pre-life, in-life, and post-life, Figures 3.3—3.5 show the number of unresolvable nameservers causing lame delegations, and the number of domains

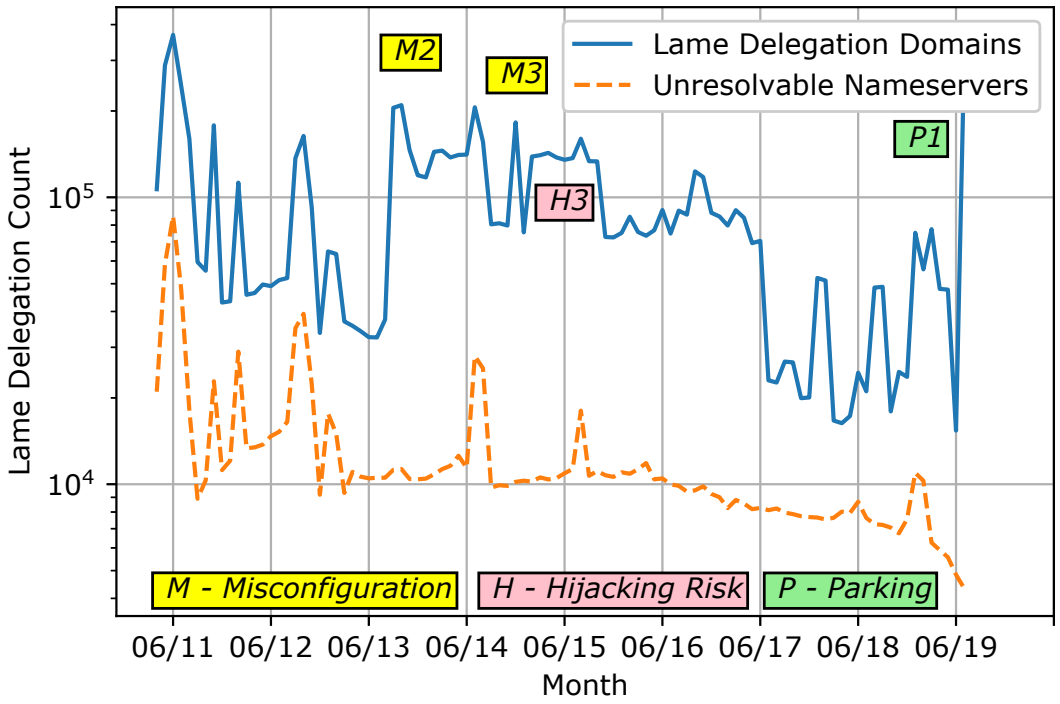

Figure 3.4. In-life lame delegations due to nameservers that *become* unresolvable (red), often due to temporary expiration of nameserver domain or misconfiguration of glue.

affected by them, over time.

The pre-life timeseries (Figure 3.3) shows a downward trend in this kind of unresolved nameserver. Over the last few years, significantly fewer nameservers are named in `NS` records before those nameservers are resolvable. Yet the number of domains affected has increased substantially. The contrast indicates that the practice of adding nameservers in `NS` records before they are resolvable is on the rise, but concentrated on fewer nameservers. The sudden increase in concentration (`H2`) is a result of a single typo causing roughly 20,000 domains to be lame delegated.

The in-life timeseries (Figure 3.4) shows a generally stable baseline number of unresolvable in-life nameservers through 2014, and a slight decreasing trend since then. The most common cause of in-life periods is mismanagement, *e.g.*, failure to renew, deleting required glue records.

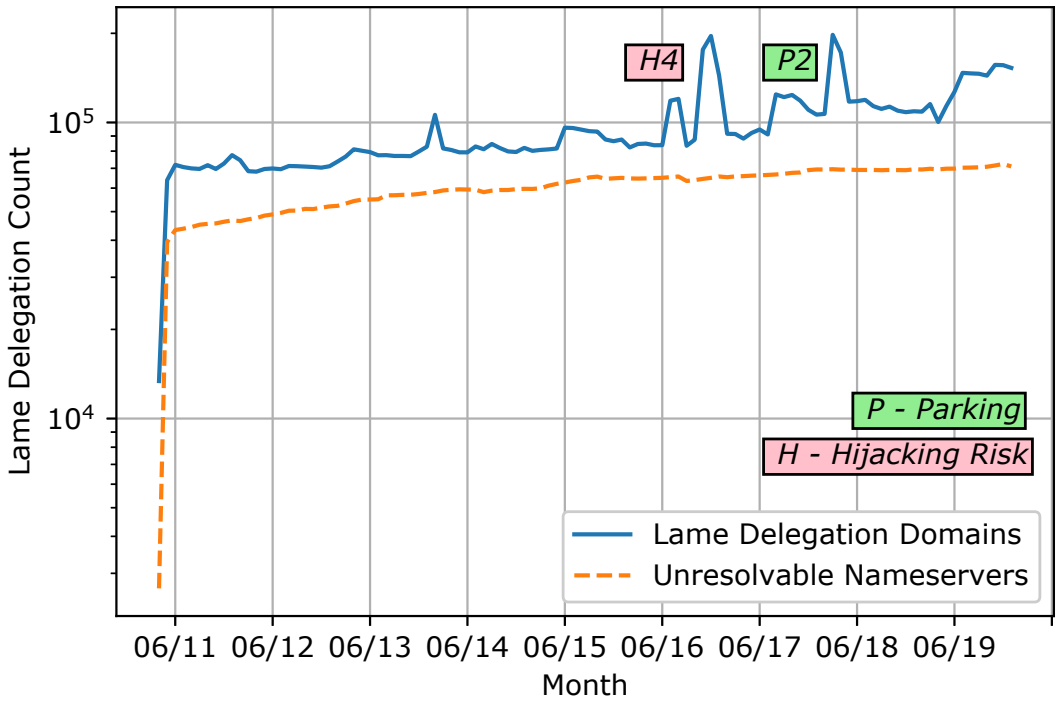

Figure 3.5. Post-life lame delegations (blue) due to nameservers that are no longer or were never resolvable (red), typically due to permanent expiration of a nameserver domain or typo of a nameserver.

The post-life timeseries (Figure 3.5) shows increasing trends in the number of post-life unresolvable nameservers and the number of domains affected by them. The steady increase could reflect the increasing use of unresolvable nameservers for parked domains, or for domains that have expired but have yet to be released.

These timeseries also show spikes in the number of unresolvable nameservers and and their associated lame delegated domains. These spikes correspond to significant events that caused many domains to become lame delegated. In the rest of this section, we study these events to highlight major causes of lame delegation and associated risks.

Hijacking Risk

Lame delegations can pose a risk to domain owners since attackers can take advantage of expired nameserver domains or typos to hijack domain resolutions. Consider the events

labeled "Hijacking Risk" in Figures 3.3—3.5. In May 2011 (H1 in Figure 3.3) roughly 29,000 domains pointed to three unresolvable nameservers. These lame delegations were a result of three nameservers created by the Conficker Working Group (CWG) to use for sinkholed and preemptively registered domains used by Conficker [71]. However, these nameserver domains expired and someone else acquired them, thus controlling resolution of the domains using those nameservers [9]. Further, in May 2015 (H3 in Figure 3.4) the `cwgsh` nameserver registrations expired again.

In December 2016 (H4 in Figure 3.5) nearly 100,000 domains suddenly become lame when their nameserver's domain expired. Specifically, the domains using the nameservers `ns[1,2].oigjaeiug.xyz` become unresolvable when the registered domain `oigjaeiug.xyz` expired. Surprisingly, domains continued to point to these unresolvable nameservers for five more months, until May 2017. Further, the domain `oigjaeiug.xyz` was available for registration at the end of this period, posing a hijacking risk: an attacker registering that domain name could immediately have become authoritative for domains that pointed to it in this period.

Finally, in December 2018 (H2 in Figure 3.3) the appearance of roughly 20,000 lame delegated domains was due to the use of the unregistered nameserver `ns5.dsndun.net`, which is a typo on the intended `ns5.dnsdun.com`. The domain `dsndun.net` was registered six months later, but the historical zone files reveal that `ns5.dsndun.net` did not resolve to the same addresses as `ns5.dnsdun.net`. In this case, whoever registered `dsndun.net` hijacked resolutions for nearly 20,000 domains for six months before the original domain owner removed the typoed nameserver from its list of authoritative nameservers.

Quantifying the Hijacking Risk. To make this risk concrete, we quantified the hijacking opportunity, *i.e.*, the potential to gain some degree of DNS resolution control over currently lame delegated domains. Our zone file data showed that as of January 2020, there were 70,605 nameservers under 48,185 unique registered domains used by 151,422 lame delegated domains. Of these nameserver domains, 42,579 (88%) were available for purchase, placing nearly 75,000 domains at risk. For instance, by purchasing just 10 of these domains (each under $10 per

domain), anyone could have potentially become the authoritative nameserver for around 4,000 domains.

While these domains may not have much intrinsic value, they could be a source of cheap domains. For the cost of registering a nameserver domain, an actor effectively gains use of all domains that name it in their NS record. Even though a purchaser does not own the delegated domains, they have control over how they are resolved and can even get SSL certificates signed for them.

This risk is not hypothetical. We see evidence of actors purchasing nameserver domains to take advantage of lame delegations. For instance, the owner of phonesear.ch has been registering nameserver domains that are authoritative for many lame delegated domains,[4] apparently for search engine optimization. Section 3.5.6 describes a set of lame delegations that left a county government in the U.S. at risk of hijacking for over a year.

Misconfiguration

A common cause of lame delegation is misconfiguration. We describe the three examples (M1-M3) annotated in Figures 3.3 and 3.4.

In September 2013, new nameservers were added to the nic.tel zone without glue records (M1), followed by existing nameserver glue records being dropped (M2). These configuration issues are consistent with reports of ongoing troubles the registry operator had with their delegations [51]. In May 2017 .tel transferred ownership [52], after which issues with the nic.tel nameservers disappeared.

The nameservers conficker-sinkhole.{com,net} were registered as a fix for letting the cwgsh domains expire, and efforts were made to move some domains over to these new nameservers from the cwgsh nameservers (which were no longer under the Conficker Working Group Control). Unfortunately, in December 2014 (M3), these domains expired and for five days were unresolvable while the registrar held them for the grace period. Fortunately, based

[4] https://dns.coffee/nameservers/A.NS.PHONESEAR.CH

on `whois` information, the domains were renewed in the grace period avoiding a repeat of the hijacking seen with the `cwgsh` domains (Section 3.5.5).

Parking

Registrars often try to monetize traffic to parked or expired domains. Typically, this monetization takes the form of many domains serviced by a single nameserver that directs visitors to advertisements. When such nameservers become unresolvable, the number of lame delegations jumps. We highlight two examples.

In July 2019 (`P1` in Figure 3.4) roughly 285 k domains became lame, caused by the domain `domainparkingserver.net`, along with the glue records for its nameservers in the zone, disappearing for seven days from the zone files.

Similarly, the spike (`P2`) in March 2017 was due to a nameserver used for parked domains expiring. Since domains still pointed to the expired nameserver, the registrar could not delete the nameserver. The registrar followed industry practice and changed the `NS` record to `ns1.pendingrenewaldeletion.com.lamedelegation.org`,
making the original nameserver domain available for registration again. In this case, the registrar used a domain it owns to act as a "sacrificial nameserver", and therefore created no hijacking risk.

3.5.6 Discussion

The lame delegation issues highlighted by our longitudinal passive analysis may involve only a small fraction of nameservers and domains in the DNS, and relatively unpopular ones at that. However, we argue that these issues are still important for a variety of reasons.

First, misconfigurations due to expired nameservers, nameserver records with typos, etc., represent a gap between expected and actual operation. When all nameservers for a domain are lame (fully lame), the domain is entirely unresolvable. When a subset of nameservers for a domain are lame (partly lame), the domain may still resolve but persistent unresolvable

nameservers reduce the resiliency of DNS resolution for those domains. Section 3.6 discusses the operational impact of these issues.

Second, lame domains have sufficient value in practice to motivate some actors to capture their traffic by strategically registering dangling nameservers, as illustrated by the `phonesear.ch` example in Section 3.5.5.

Finally, even "unpopular" domains may identify critical infrastructure. As a concrete example, consider `whitecounty.net`, the official domain for White County, Georgia. This domain had the same two authoritative nameservers `ns2.internetemc.com` and `ns1.hemc.net` from our first import of the `.net` zone file until June 30, 2019 when the domain `internetemc.com` expired. To work around the EPP constraint of freeing a domain (`internetemc.com` in this case) when host objects associated with the domain have live references, the registrar renamed the host object associated with the domain `ns2.internetemc.com` to a sacrificial nameserver `ns2.internetemc1aj2tkdy.biz` in a different TLD.[5] This renaming followed a similar practice to that described in Section 3.5.3, just using a different pattern for the sacrificial nameserver.

As a result, starting on July 1, 2019, one of its nameservers was unresolvable and `whitecounty.net` was partly lame delegated. By registering `internetemc1aj2tkdy.biz`, an attacker could have received a fraction of the resolution requests for an official county government domain. Note that redundancy in DNS worked as intended since the other nameserver still worked and resolved everything correctly, albeit with a delay at times if the resolver chose to query the lame nameserver first. Ironically, though, because redundancy masked the long-term unresolvable nameserver, this issue went undiscovered by the domain owner. Given the sensitive nature of White County's domains, we reached out to the registry who notified the domain registrant. The domain configurations were fixed soon after.

[5] See the timeline illustrated at https://dns.coffee/domains/WHITECOUNTY.NET

Table 3.7. Active DNS Resolution Lame Delegation Results: Breakdown by TLD.

	.com	ngTLDs	.net	.org	Total
Domains	13,000,000	13,000,000	13,174,611	10,015,702	49,190,313
Fully Lame	8.7%	9.6%	10.5%	9.2%	9.5%
Partly Lame	11.8%	19.8%	13.5%	11.7%	14.3%
Nameservers	620,561	278,657	724,518	552,665	1,325,856
IPs	299,319	143,095	347,413	273,906	534,214
Fully Lame	14.5%	17.1%	16.2%	16.4%	15.7%
Partly Lame	41.9%	44.0%	43.3%	44.1%	45.3%
˜AA	0.3%	0.2%	0.5%	0.4%	0.5%

3.6 Lame Delegations Measured with Active Queries

Static analysis revealed many aspects of lame delegations, particularly over time, but it is a lower bound. Active measurement shows that the prevalence of lame delegations is significantly higher in operational practice. We can detect lame delegations operationally by performing active domain resolutions, much as clients do when resolving domains. We characterize the prevalence of lame delegations across the major gTLDs, explore nameserver consistency issues, and quantify the impact of lame delegations on domain resolution time.

3.6.1 Methodology

We targeted `NS` queries at all nameservers listed in the zone file for a domain, from a single, well-provisioned vantage point connected to the Netherlands NREN. We supplemented our measurements with active resolution data provided by OpenINTEL for additional context about lame delegated domains within the recent past.

We started with a snapshot of the ngTLD zone files and `.com`, `.net` and `.org` to learn all the registered domain names under these zones. Next, we extracted the nameservers specified in their `NS` records. Finally, we extracted IP addresses in any existing glue records for nameservers.

We performed the following measurement steps:

1. Actively resolve all NS names and record the IPv4 addresses[6] learned per name.[7]

2. For each registered domain name, and for every NS name of each particular domain, we targeted up to *five* actively resolved IP addresses for the NS name in question with an explicit NS query for the registered domain name. We instantiate a local DNS resolver to contact the nameserver, so caching mechanisms will not affect our measurements.

    - We recorded the set of NS records returned by the NS query, including response flags set by the nameserver.
    - In case of an error (*e.g.*, a connection timeout or a DNS-specific error), we record the error type.

Between March and May 2020 we queried over 49 million domains: 13 million randomly sampled domains from .com, 13 million randomly sampled domains from the combined set of all ngTLDs, and all domains from .net and .org. This selection balances coverage against the overhead of an exhaustive crawl of the entire DNS with the exponential fan-out from multiple nameservers per domain, and then multiple IP addresses per nameserver.

When resolvers cannot use a provided NS record (*i.e.*, delegation) to obtain authoritative answers for a registered domain, we infer the delegation is *lame* (Section 3.2). Our measurement reveals cases in which NS hosts do not exist, do not run a nameserver, or are not able to provide authoritative responses. It is not always possible to distinguish non-operational servers from network outages. Nameservers that we cannot reach after repeated attempts, we infer to be lame.

3.6.2 Domain Perspective

Table 3.7 summarizes the results of our active measurements, including the number of domains resolved, the total number of nameservers used by those domains, and the total number

[6] We contacted nameservers over IPv4 only. Our rationale is that a nameserver that is unresponsive over IPv4 and reachable only over IPv6 is still lame to resolvers (*e.g.*, clients) with no IPv6 connectivity.

[7] Successful resolution requires any part of the delegation chain for the NS name to work. We do not exhaustively check every step of the chain as our perspective does not require it and doing so would exponentially increase measurement overhead.

Table 3.8. Partly lame domains by number of delegated NS.

#NS	#Lame Domains (%)	#NS	#Lame Domains (%)
1	11,926 (54.5%)	9	675 (40.7%)
2	5,732,799 (14.7%)	10	304 (12.6%)
3	499,652 (14.1%)	11	80 (61.1%)
4	551,592 (10.7%)	12	295 (3.4%)
5	132,428 (13.1%)	13	71 (28.9%)
6	97,472 (30.6%)	14	2 (100%)
7	17,817 (26.3%)	15	2 (100%)
8	9,176 (5.7%)	16	1 (100%)

of IP addresses associated with the nameservers. The table classifies domains into two categories, fully and partly lame. A fully lame domain means that we did not obtain an authoritative answer from *any* nameserver or IP enumerations for that domain. A partly lame domain means that we did not obtain an authoritative answer from *at least one* nameserver and IP enumeration for that domain. Note that the partly lame metric also includes the fully lame cases.

At the time of our measurements roughly 10% of domains were fully lame (not resolvable) consistently across the TLDs.[8] This number increased to 14% of domains when considering partly lame domains (has at least one lame delegation, but not all). There are various reasons why actively resolving a domain can fail, from typos in names to placing recursive (non-authoritative) resolvers in NS records. For the 10% fully lame domains, the most prevalent issues that we encountered are nameservers that do not (or cannot) provide an authoritative answer, or nameservers that cannot be reached (*i.e.*, query timeouts). Only a small percentage of cases resulted from typos in NS records.

Partly lame delegations were only 3–5% more common than fully lame. Since Table 3.7 counts the fully lame cases as also partly lame, it shows that more often than not, if a domain has any lame nameserver path, all of its paths do not resolve. The exceptions are the new gTLDs grouped under ngTLDs. Nearly 20% of domains we queried in ngTLDs had at least

[8]Note that this percentage is similar to the results from Pappas *et al.* [69] in 2004. As the timeseries from Section 3.5 highlights, lame delegations have long been a persistent issue in the DNS.

Table 3.9. Fully lame nameservers relative to all nameservers in the same TLD.

NS TLD	Total NS	Fully Lame NS(%)
.com	176,897	57,137 (32.3%)
.net	97,160	30,896 (31.8%)
.org	38,825	14,792 (38.1%)
.info	2,690	731 (27.2%)
ccTLDs	65,041	16,585 (25.5%)
ngTLDs	40,792	19,213 (47.1%)
.biz	14,311	10,533 (73.6%)
Total	435,716	149,887 (34.4%)

one nameserver path that did not resolve. This behavior could derive from ngTLDs domains being concentrated on many fewer nameservers than other TLDs. ngTLDs have roughly half the number of nameservers and corresponding IPs when compared to legacy gTLDs with similar number of domains.

Table 3.7 also breaks down the nameserver IP addresses into fully and partly lame. A fully lame IP address means that, when querying that IP to resolve a domain, that IP does not return an authoritative answer for *all* domains for which we queried it. A partly lame IP address means that the IP does not return an authoritative answer for *at least one* domain for which we queried it.

The fact that partly lame domains still resolve underscores the benefits of redundancy in the DNS. Table 3.8 classifies partly lame domains by the number of delegated nameservers. The first row corresponds to domains with just one nameserver, which by definition are misconfigured since RFC 1034 requires a domain have two nameservers at least [66]. With one lame nameserver, these domains are all unresolvable. As the number of nameservers increases, the percentage of partly lame domains naturally increases. The more delegated nameservers, the higher the probability that at least one of them at any time is lame. These results highlight the fact that DNS redundancy may obscure configuration issues, since the domain is often still resolvable even when misconfigured.

3.6.3 Nameserver Perspective

We next look at the nameservers responsible for lame delegations. Table 3.9 shows the number and percentage of actively discovered fully lame nameservers across TLDs. Consistent with the results of our static analysis, `.biz` stands out with an unusually high percentage of fully lame nameservers (Section 3.5.3). There were 14,311 nameservers in `.biz` in our active measurement set, and 73.6% of them were fully lame. Domains using the unresolvable `.biz` nameservers predominantly come from the legacy TLDs `.com` and `.net`, again consistent with the long-standing practice of handling expired nameservers within those TLDs.

Looking at nameserver domains and their IPs more closely, lame delegation is concentrated and the most prevalent nameservers and IPs suggest that they, at least, are lame delegated by design and not due to misconfiguration. Table 3.10 reports the top fully lame delegated `NS` records and IPs. The top nameserver domain is associated with suspicious bulk domain registrations.[9] Manual inspection shows that the others are primarily sinkholes for security, abuse, and expired domains where delegated domains have been made lame intentionally. The top IP serves parked and for-sale domains, the next two IPs are used by `0088dns.com`, and the last two IPs are used in glue records for nameservers in `maff.com` with a large number of apparently abusive domains.

3.6.4 Consistency

The DNS ultimately depends upon multiple independent sources of information to operate correctly. However, as a hierarchical, delegation-based distributed system, the DNS does not contain inherent internal mechanisms to ensure consistency across these independent records. Inconsistencies arise for a variety of reasons, two of which we describe: inconsistencies in authoritative responses and glue records.

[9]For example, see current and past delegated domains at https://dns.coffee/nameservers/NS1.0088DNS.COM.

Table 3.10. Top fully lame delegated NS IPs and domains.

NS IP	Country	#Lame Domains
52.20.26.87	US	144,327
60.12.122.226	CN	117,462
103.26.77.114	CN	117,462
218.98.111.162	CN	80,142
183.2.194.161	CN	80,142

NS Domain	#Lame Domains
0088dns.com	117,462
sinkhole.shadowserver.org	45,401
verification-hold.suspended-domain.com	41,804
sav.com	35,431
icmregistry.net	32,377
expirenotification.com	32,369

Authoritative Consistency

By definition, authoritative nameservers should reply with authoritative responses (setting the `AA` flag). In a small percentage of cases (0.1–0.3%) authoritative nameservers do not reply as authoritative, creating lame delegations as a result. Table 3.11 shows the top 10 nameserver IPs that do not set the `AA` flag, ranked by the number of nameserver domains associated with those IPs. These turn out to be wildcard nameservers, which set the `AA` flag to false for `NS` queries and true for any `A` queries. The top nameservers group in pairs based on country code. In fact, the nameserver pairs are used as the two nameservers for parking domains, which obviates the need to update nameserver zone file records.

Glue Consistency

One can obtain IP addresses of nameservers by examining glue records in zone files, or by actively querying for their `A` records. These two methods should ideally yield the same set of IP addresses, but we find a surprising degree of inconsistency between the glue records in zone files and those returned by active queries. We examined the consistency between the set of IP

Table 3.11. AA false lame delegated IPs.

NS IPs	Country	#Domains	Wildcard
91.195.241.7	DE	59,628	Y
91.195.240.7	DE	56,744	Y
185.230.61.173	IL	22,897	Y
185.230.60.173	IL	22,894	Y
31.31.205.59	RU	14,710	Y
31.31.205.62	RU	14,710	Y
209.235.147.130	US	5,751	Y
209.235.147.131	US	5,746	Y
151.236.17.126	DE	5,616	N
149.154.159.77	GB	4,048	N

Table 3.12. Parent-Child Glue Record Consistency.

	.com		ngTLDs		.net		.org	
Unresponsive	80554	16.4%	32410	42.0%	58189	21.9%	28961	29.5%
$P = C$	355055	72.2%	40528	52.5%	184407	69.3%	60739	61.8%
$P \neq C$	56038	11.4%	4223	5.5%	23534	8.8%	8524	8.7%
$P \cap C = \emptyset$	47716	85.1%	3832	90.7%	20541	87.3%	7754	91.0%
$P \cap C \neq \emptyset$	8322	14.9%	391	9.3%	2993	12.7%	770	9.0%
$P \subset C$	5742	69.0%	290	74.2%	2061	68.9%	488	63.4%
$P \supset C$	2317	27.8%	95	24.3%	805	26.9%	269	34.9%
Rest	263	3.2%	6	1.5%	127	4.2%	13	1.7%

addresses in the zone glue records ("parent" zone glue P) and the glue records retrieved via DNS queries (the "child" zone glue C).

Table 3.12 shows two interesting results between the two perspectives. First, similar to the results for domains in Table 3.7, a significant number of glue records cannot be resolved by querying, particularly for glue record IPs used by nameservers in the ngTLDs.

Second, for the glue records that can be queried, most are consistent ($P = C$). But from 5.5–11.4%, depending on the TLD, have inconsistent glue records. Table 3.12 further breaks down the relationships between the two sets P and C into four categories: the sets of glue records are completely disjoint ($P \cap C = \emptyset$); the parent zone glue records are a subset of the

child zone glue records ($P \subset C$); and the parent zone glue records are a superset of the child zone ($P \supset C$); and sets that otherwise overlap in at least one address ("Rest"). The breakdown shows that the dominant inconsistencies are entirely disjoint: the child zone glue records are completely different from the parent zone glue records. As a result, for nearly 10% of the cases these inconsistencies create two entirely separate resolution paths for the same nameservers: in-bailiwick domains will use the parent zone glue records, whereas out-of-bailiwick domains will use the child zone glue records.

3.6.5 Impact of Lame Delegation

In addition to their security risks (Section 3.5.6), lame delegations also degrade DNS resolution performance. In this section we quantify this performance impact and show that it affects even popular domains.

Lame delegations cause useless DNS queries. When resolving a domain that has at least one lame delegated nameserver, a resolver may have to contact multiple nameservers to successfully resolve the domain. As a result, the average resolution time for lame delegated domains will increase. To quantify this impact experimentally, we used data from OpenINTEL [89] to calculate the average resolution time for resolving the roughly 49 million domains in our active measurement set over the month of March 2020. OpenINTEL performs active measurement using a *normal resolver* to resolve domains. The normal resolution method approximates the user experience, and averaging resolution performance measurements over a month minimizes short-term variance.

The average resolution time for domains that are fully resolvable (without any lame delegated nameserver) was 172 ms, whereas domains with lame delegated nameservers had a significantly higher resolution time. For partly lame delegated domains (where a subset of the nameservers are lame), the average resolution time was 720 ms. For fully lame delegated domains, the resolution time was 1743 ms, an order of magnitude higher than fully resolvable domains. Note that these resolution times were bounded by timeout errors and caching since

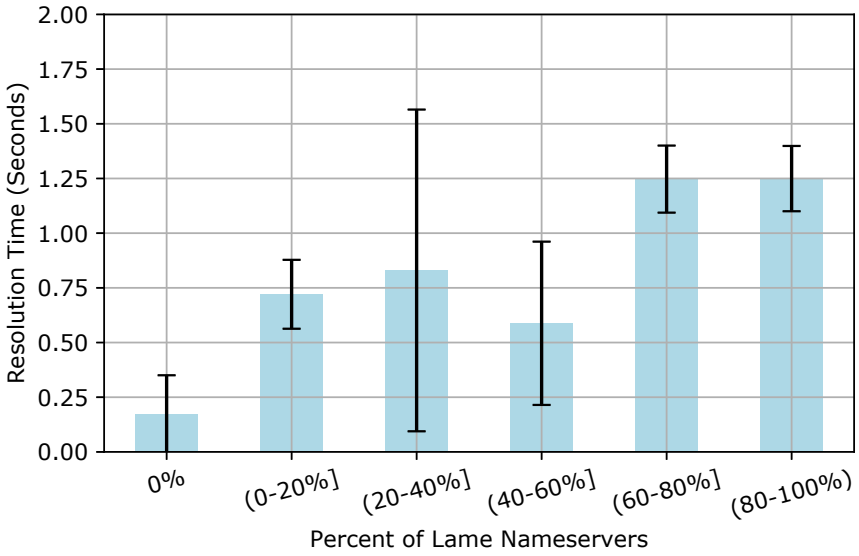

Figure 3.6. Average times to resolve domains over a month of daily resolutions. Domains are aggregated by the percentage of lame delegated authoritative nameservers they have, *e.g.*, domains with five nameservers where three are lame delegated fall into the "(40,60%]" bucket. The whiskers show standard deviations.

this data came from using a *normal resolver* process. Even entirely lame delegated domains ultimately have a maximum finite resolution time.

Figure 3.6 breaks down resolution times for the domains in our data set by the percentage of their lame delegated nameservers. For example, for domains with (40,60%] lame delegated nameservers (*e.g.*, domains with two nameservers where one of them is lame, or domains with five nameservers where three are lame), the average resolution time was 0.59 seconds, 3.4× higher than domains with no lame delegated nameservers (the "0%" bucket). Overall the figure shows that a higher percentage of lame delegated nameservers per domain resulted in higher average resolution time.

We also observed that lame delegations occurred even on popular domains. Table 3.13 shows the number of domains in our active measurement set that are on Alexa Top lists [7], and the number of those that were fully and partly lame. We used the Alexa list for April 13, 2020, which corresponds to the midpoint of our active measurement campaign.

Table 3.13. Popular domains with lame delegations: the number of domains in our active measurement set that are on Alexa Top lists, and the number of those that are fully and partly lame.

	Measured	Fully Lame	Partly Lame
Alexa Top 100k	14,483	146	439
Alexa Top 1M	82,420	943	2,867

Table 3.13 shows that lame delegations, while not as ubiquitous, were present even for popular domains. Consider `archive.org`, an Alexa Top 200 site, which has one lame delegation of five possible delegations. As of September 12, 2020, `archive.org` was still partly lame delegated.[10] Surprisingly, we also encountered fully lame delegations in popular domains. We found that most domains switched their nameservers soon after, remediating the lame delegation. These observations reinforce our hypothesis that fully lame delegations are likely to be fixed more quickly than partly lame delegations because the domains are unusable when fully lame delegated.

Finally, as yet another perspective indicating that lame delegations are a notable operational issue, GoDaddy estimates that roughly 12% of requests to their nameservers are for domains for which they are not authoritative [73].

3.7 Ethical Considerations

We had to consider ethical aspects of characterization and responsible disclosure of lame delegations. Domains with lame delegations may be at risk of being hijacked. Given the many thousands of at-risk nameserver domains, we cannot defensively register all of them, which would raise its own ethical issues if we could. Without the ability to protect these lame domains, disclosing them increases the risk of harm to their owners and users. We are working on a responsible way to disclose our snapshot of lame delegations.

[10] Note that `archive.org` while misconfigured is not at risk of being hijacked.

3.8 Summary

The Internet, as it is commonly taught, is constructed from simple abstractions implemented via a number of key network protocols. Invariably, however, there is significant daylight between this clean abstract model of how the Internet functions and the frequently messy reality of its concrete operation. Measurement studies such as this one are the mechanisms we use to characterize this gap in understanding. Our work characterizing the presence and risks of lame delegation in the DNS exemplifies the value of this kind of empirical study.

Using comprehensive collections of both active and passive DNS measurements (covering 49 M and 499 M domains respectively), we found that lame delegations are surprisingly common: roughly 14% of registered domains that we actively measured had at least one lame delegation, and most of those had no working authoritative nameservers. However, even for domains with working alternative nameservers, our measurements show that these lame delegations impair DNS performance (average resolution latency increasing by 3.7×) in addition to producing substantial unnecessary load on existing nameservers.

Finally, we found that unregistered or expired domains in lame delegations can create significant security risk. Indeed, over the last nine years, we identified at least three instances in which an attacker could have hijacked thousands of domains by registering a single nameserver domain. Analysis of this phenomenon led us to discover an unforeseen interaction between registrar practice and the constraints of registry provisioning systems that has inadvertently made hundreds of thousands of domains vulnerable to hijacking due to accidental lame delegations. This practice has persisted for over twenty years, but we are now working with registrars to remediate it and its effects.

Going forward, we are exploring ways to combine daily zone data and periodic active measurements to automatically identify and report lame delegations as they are created. An open question remains about the most effective mechanisms for communicating these findings to appropriate stakeholders to incent corrective action. As well, the security issues that arise as

unintended byproducts of registrar/registry practices deserve further attention as this aspect of the domain name ecosystem is largely opaque to the research community.

Many domain operators configure redundancy in resolution infrastructure, which can hide underlying systemic issues for long periods of time. Ironically, this engineered robustness poses a security threat, as domain operators rarely take notice of DNS configurations unless their domain stops resolving completely. Thus they are likely to fail to notice partly lame domains that attackers can exploit.

While some systematic issues such as the "DROPTHISHOST anomaly" require registrar-level intervention to fix, domain owners can proactively monitor their own domain configurations. In pursuit of improved monitoring and remediation, we are developing a monitoring tool to allow domain owners to check static zone files for potential delegation-related security risks, and will integrate it into our zone analysis platform. Finally, we have begun an effort to work with the registrar and registry communities to responsibly disclose such risks, establish their underlying causes, and develop improved operational practices to minimize lame delegations going forward.

Chapter 3, in full, is a reprint of the material as it appears in *Proceedings of the International Measurement Conference 2020*. Gautam Akiwate, Mattijs Jonker, Raffaele Sommese, Ian Foster, Stefan Savage, Geoffrey M. Voelker, and KC Claffy. The dissertation author was the primary investigator and author of this paper.

Chapter 4

Risky BIZness: Risks Derived from Registrar Name Management

In Chapter 3, we first identified long-standing undocumented operational practices between registrars and registries that exposed domains to the risk of hijack unbeknownst to the domain owner. In this chapter, we study the domain hijacking risk caused by these undocumented operational practices in the DNS ecosystem. We find that over nine years more than $512k$ domains have been implicitly exposed to the risk of hijacking, affecting names even in TLDs with tight registration control (such as `.edu` and `.gov`). Moreover, we find that this vulnerability 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 our outreach with the registrars wherein the registrars not only acknowledged the issue, but also remediated the hijacking risk.

4.1 Overview

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 [37].

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 [30] or, in the case of cache poisoning, an attack on the resolution protocol itself [85]. In this chapter 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

[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.

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 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.

4.2 Background

In this section we provide a brief 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.

4.2.1 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 [47] and the domain and host mapping (critical for this chapter) is documented in RFCs 5731 [48] and 5732 [49]. 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 chapter:

> 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 4.1. 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 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 using a scheme

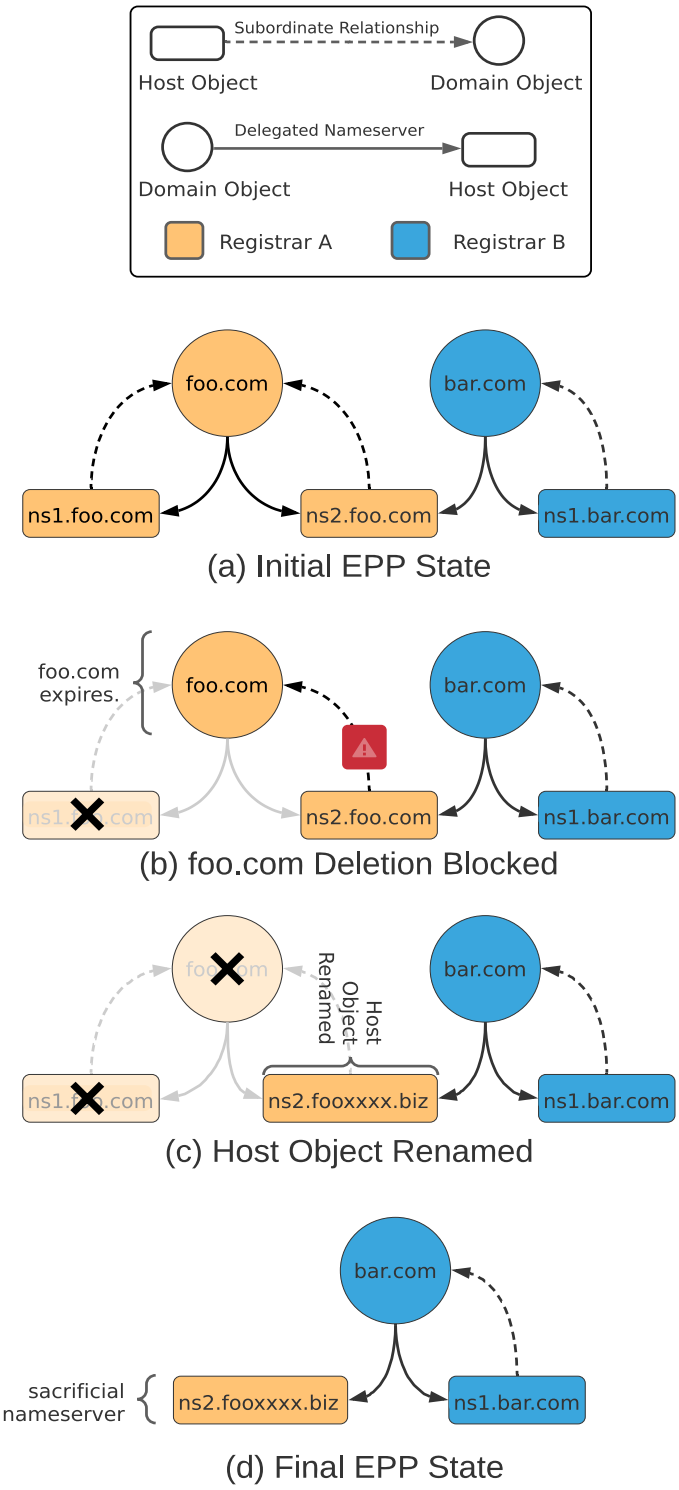

Figure 4.1. Nameserver renaming in EPP as a mechanism to bypass domain deletion constraints

like {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.[3]

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 4.1 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

[3]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.

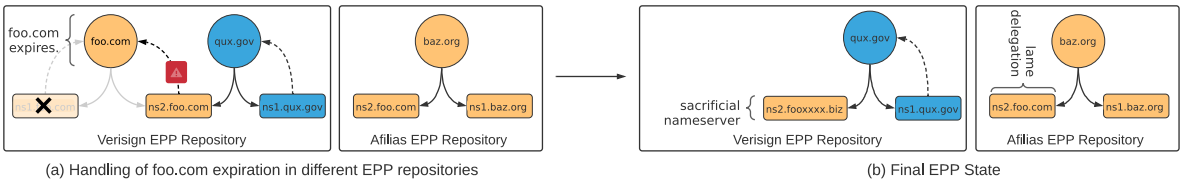

Figure 4.2. Handling of domain expiration in different EPP repositories. The renaming operation affects all TLDs supported by a registry's EPP repository, but other EPP repositories are unaffected by it.

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 4.2, 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` 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 chapter 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.

4.3 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.

4.3.1 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.

4.3.2 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 [21].[4] The data set covers nine years of daily snapshots of zone files from April 2011

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) [53], and publicly available zone data, CAIDA now provides uniform *research access* to the DZDB data set used in this work 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`).[5] As a consequence, we are conservative in our estimate of sacrificial nameservers.

[4] CAIDA-DZDB data set is a clone of the DNS Coffee data set used in Chapter 3.

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

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.

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,[6] 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 Chapter 3 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 $20M$ nameservers in the zone files to a candidate set of 312,328 nameservers.

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

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.

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.

Table 4.1. 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 Sink Domain	Registrar	# of Sacrificial Nameservers	# of Affected 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 4.2. 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.

Renaming Idiom Sink Domain	Registrar	# of Sacrificial Nameservers	# of Affected Domains	Example Renaming 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	

For example consider `ns2.internetemc1aj2kdy.biz`, a candidate nameserver and the domain `whitecounty.net` that delegates to it. The history for the domain[7] 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 [33], 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.

4.3.3 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 4.3.2) will not identify them. Second, we assume that sink domains used by registrars

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

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).

4.4 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 4.1 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 4.2 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 4.5 and 4.6 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 by `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 4.7).

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.

4.5 Exploitation of Sacrificial Nameservers

The results in Section 4.4 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

Table 4.3. Number of hijackable and hijacked sacrificial nameservers and their delegated domains.

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

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.

4.5.1 Hijacking Summary

Table 4.3 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.

4.5.2 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 4.3 longitudinally shows the number of newly hijackable domains that appear each month due to the creation of sacrificial nameservers. Encouragingly, the trend has been

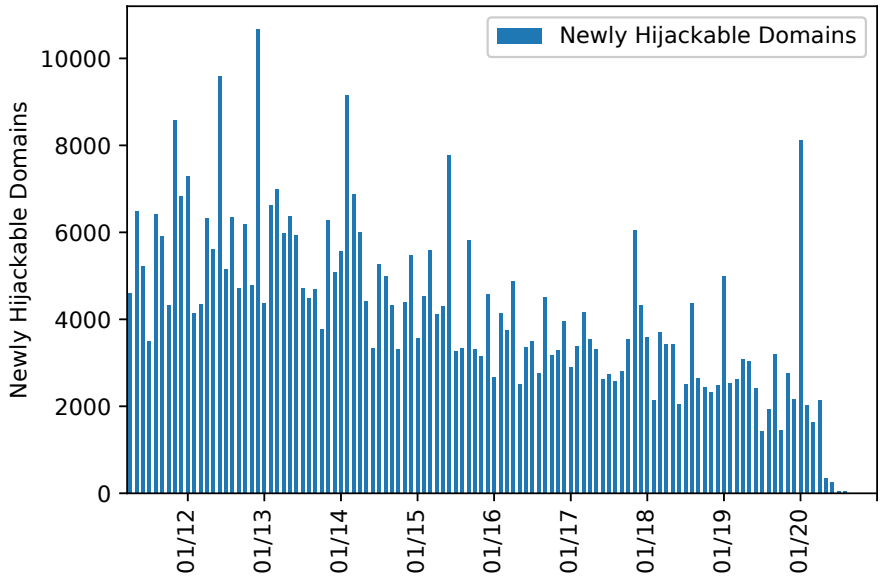

Figure 4.3. New hijackable domains per month from April 2011 to September 2020.

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 4.4 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.

4.5.3 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

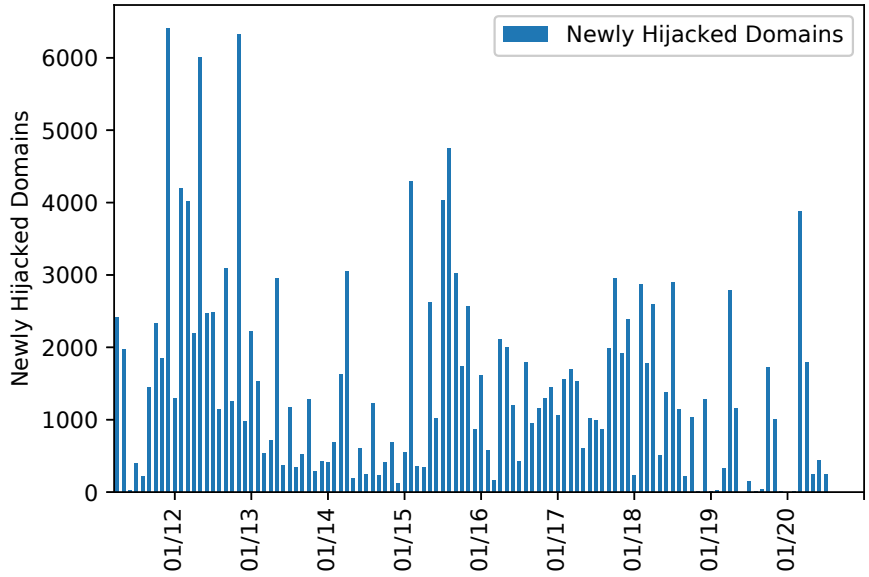

Figure 4.4. New hijacked domains per month from April 2011 to September 2020.

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 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 4.5 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.

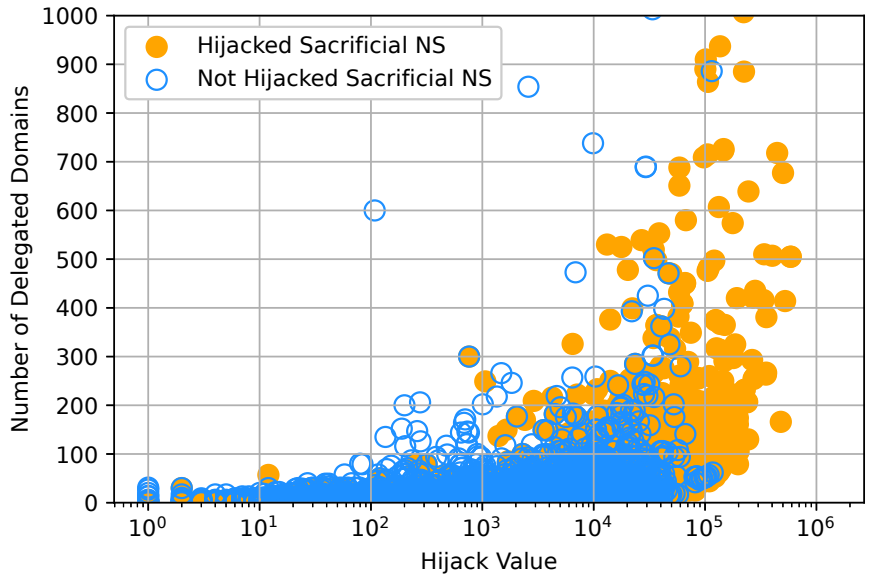

Figure 4.5. Scatter plot showing the number of domains delegated (capped at 1,000) and the hijack value of both hijackable and hijacked sacrificial nameservers.

4.5.4 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 4.6 shows the distributions of these counts as two CDFs. The bottom CDF shows the time to 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

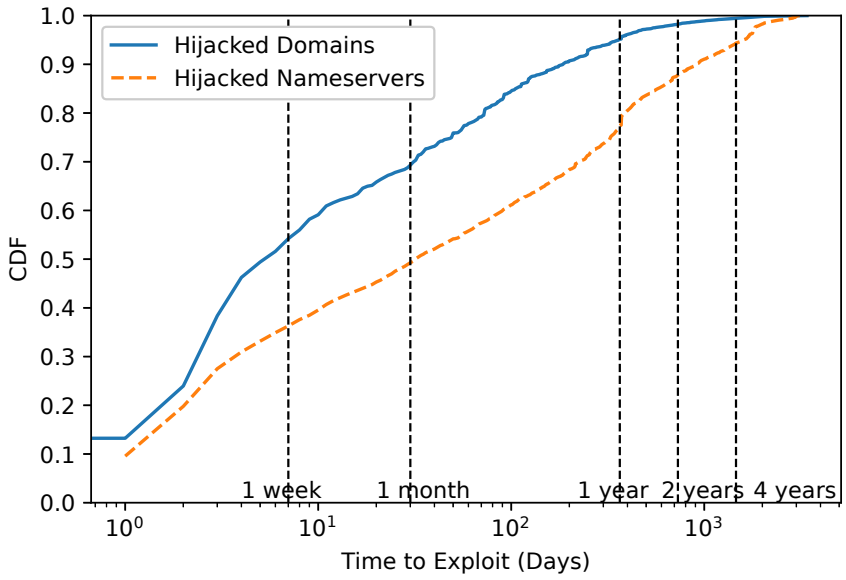

Figure 4.6. Time to exploit hijackable sacrificial nameservers and vulnerable domains eventually hijacked.

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.

4.5.5 Duration

Finally, we examine the durations for which domains are hijacked further revealing interesting hijacking behaviors. Figure 4.7 compares the durations for which domains are hijacked with the durations 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

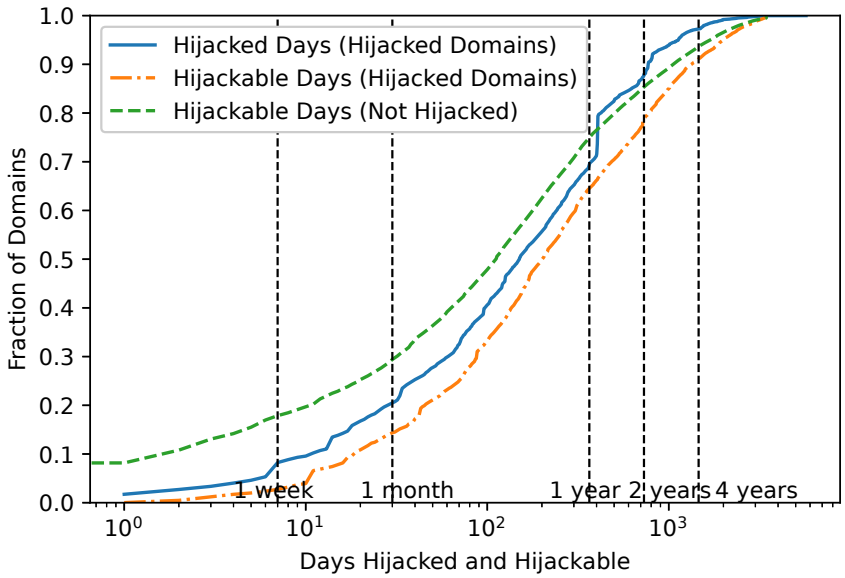

Figure 4.7. Fraction of domains hijacked or hijackable for at most X days.

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.

4.5.6 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 4.6.1, 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 4.7, 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.

4.6 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.

4.6.1 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

Table 4.4. Top five hijackers overall by number of domains hijacked (April 2011 – September 2020).

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

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[8] 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 4.2.1: 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 4.8.

[8]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.

4.6.2 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 4.4 shows the most popular controlling nameserver domains over the course of our study.[9]

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).

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

Table 4.5. Change in number of hijackable (vulnerable) and hijacked sacrificial nameservers and affected domains after notifications starting in 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

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.

4.7 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, 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.

4.7.1 Remediation of Existing Affected Domains

As we have explained, once renamed outside an EPP repository, a host object cannot be subsequently modified (Section 4.2.1). 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 4.5 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.[10] 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 ($13k$ com-

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

Table 4.6. Domains protected due to renaming idiom changes as of September 2021.

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

pared 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.

4.7.2 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 [3].[11]

Table 4.6 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.

4.7.3 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

[11] While this solution avoids the use of a sink domain it may introduce other risks (Section 4.7.3).

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 4.4). 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 [1] with additional guidance on its use published in 2013 [24], 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.[12]

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.

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

4.8 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 chapter to avoid facilitating their exploitation.

Second, we designed our controlled experiment (Section 4.6.1) 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.[13] 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.

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

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.

4.9 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).[14] 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.

Chapter 4, in full, is a reprint of the material as it appears in *Proceedings of the International Measurement Conference 2021*. Gautam Akiwate, Stefan Savage, Geoffrey M. Voelker, and KC Claffy. The dissertation author was the primary investigator and author of this paper.

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

Chapter 5

Retroactive Identification of Targeted DNS Infrastructure Hijacking

While previous chapters highlighted *opportunistic hijacks*, in this chapter we explore *targeted hijacks* wherein an attacker actively takes control of the DNS infrastructure for the domain. Unfortunately, this risk is not hypothetical. In 2019, the US Department of Homeland Security issued an emergency warning about *DNS infrastructure tampering*. This alert, in response to a series of attacks against foreign government websites, highlighted how a sophisticated attacker could leverage access to key DNS infrastructure to then hijack traffic and harvest valid login credentials for target organizations. However, even armed with this knowledge, identifying the existence of such incidents has been almost entirely via post hoc forensic reports (*i.e.*, after a breach was found via some other method). Indeed, such attacks are particularly challenging to detect because they can be very short lived, bypass the protections of TLS and DNSSEC, and are imperceptible to users. Identifying them retroactively is even more complicated by the lack of fine-grained Internet-scale forensic data. This chapter is a first attempt to make progress at this latter goal. Combining a range of longitudinal data from Internet-wide scans, passive DNS records, and Certificate Transparency logs, we have constructed a methodology for identifying potential victims of sophisticated DNS infrastructure hijacking and have used it to identify a range of victims (primarily government agencies), both those named in prior reporting, and others previously unknown.

5.1 Overview

Sophisticated attackers use a variety of methods to gain access to targeted organizations. These methods can include spear phishing (*e.g.*, the Democratic National Committee [93]), abusing software vulnerabilities (*e.g.*, Equifax [15]), or exploiting weak passwords (*e.g.*, SolarWinds [64]). However, a less widely-appreciated vector involves the careful manipulation of DNS infrastructure *in order to acquire valid login credentials or session tokens* to a targeted organization.

This chapter focuses on such a class of attack in which an adversary has obtained the capability to manipulate a target domain's DNS configuration. This capability is typically obtained by compromising the domain holder's account with its registrar or compromising the registrar itself, although we are also aware of versions of the attack that involve compromising accounts at DNS nameserver hosting providers. Using this capability, an attacker can temporarily divert a domain's traffic in order to pass the domain validation check of Certificate Authorities (CA) such as Let's Encrypt or Comodo. Having obtained a CA-signed TLS certificate, attackers then — at a time of their choosing — can arrange to divert traffic for specific subdomains that host TLS-protected services requiring cleartext user credentials (*e.g.*, SMTP, VPN, IMAP, etc.). The attacker can then extract any such credentials as users interface with these services, and can repurpose them for further access inside the organization.

Versions of such attacks, in use by state-affiliated actors, date back to 2013 [17, 88] but they became much more widely known in early 2019 when Cisco Talos [65] and FireEye's Mandiant [46] documented particular attacks and victims in the Middle East. This led the US DHS to issue an emergency directive about the threat and mandate a range of mitigations on government systems [30]. However, identifying the victims of such attacks was left to each organization since it relied upon their individual diligence and site-specific knowledge (*e.g.*, in auditing DNS records and validating issued TLS certificates for their domains). The challenge of third-party auditing, along with the short time scales over which such attacks can operate,

perhaps explains why there have been limited investigations of this threat and its victims.

This chapter sets out to explore this question empirically and retroactively, identifying domains that may have been hijacked in this manner and focusing on those cases that are likely to represent real victims. Using four years of longitudinal data across multiple data sources, including certificate transparency logs, passive DNS logs and active Internet-wide scans, we construct a methodology for identifying domains whose anomalous network behavior matches the pattern of such attacks and are qualitatively valuable to an attacker. This work makes three contributions:

- *Attack-centric operational signatures.* By identifying the requirements of attack (*i.e.*, issuing a new certificate, staging a server to host that certificate, then using DNS hijacking to divert traffic for a subdomain that handles user credentials to the new server) we construct a model for how such attacks produce network-visible side effects.

- *Opportunistic filtering with existing data sets.* Using a wide array of longitudinal data sets we identify how these side-effects likely manifest in our data. Combined with limited assumptions about attacker behavior, we filter the set of potential victim domains to a modest number.

- *Manual qualitative evaluation.* We evaluate the resulting set of domains manually and qualitatively for their likelihood as potential victims. Our analysis predominantly identifies sensitive government agencies, consistent with the sophisticated mode of attack, and we show that our approach independently identifies virtually all victims documented in the 2019 industry reports.

Ultimately, this chapter provides a framework for identifying such attacks, and their precursors, as a third party. We explain the complexity in making such identifications but show that existing data, though imperfect, is sufficient to retroactively identify a range of real victims — including sensitive government sites previously undocumented.

5.2 Background

Targeted attackers seek to gain access to an organization and, from there, expand their capabilities. While a small number of such data breaches result directly from exploiting software vulnerabilities (only 3% according to the 2021 Verizon Data Breach report [91]), the vast majority involve the acquisition of remote access credentials (*i.e.*, user names and passwords) typically via phishing, credentials theft and reuse, or brute force. However, all these techniques are fundamentally opportunistic and produce side-effects that can alert system operators (*i.e.*, reported phishing e-mails, failed logins, etc.) An alternative approach is to covertly acquire these credentials in real-time as they are used by remote workers. However, to do this requires the attacker to solve two problems: first, they must be able to divert traffic from remote workers to a server under their control and second, they must bypass any cryptographic protection employed to protect against such diversion.

In this section, we will briefly summarize the relevant aspects of the Domain Name System (DNS), review DNS hijacking in general and DNS infrastructure hijacking in particular, and briefly highlight why existing protections such as DNSSEC and TLS do not protect against these attacks.

5.2.1 DNS and DNS hijacking

The primary role of the Domain Name System (DNS) is to map human-readable domain names to routable IP addresses for use in a variable of higher-layer protocols and services. DNS provides a hierarchical namespace such that the owner of a given registered domain name (*e.g.*, `nsf.gov`) is delegated authority to resolve that name and any *fully qualified domain names (FQDN)* under it (*e.g.*, `fastlane.nsf.gov`). This works via a query protocol, first specified in RFC 1035 [67], by which any party on the Internet can ask to be directed to an *authoritative nameserver* for a given domain name (its `NS` record). This nameserver then provides the IP address (the `A` record in DNS parlance) that the FQDN maps to.

Any time an attacker can control this resolution process, such that a domain ultimately resolves to an IP address of the attacker's choosing, this is referred to as *domain hijacking*. There are a variety of ways that a domain might be hijacked, largely owing to the considerable complexity of the DNS protocol and its implementation. Perhaps the best known is *cache poisoning*, which occurs when an attacker anticipates queries for a domain from a *recursive resolver* (*i.e.*, a service taking responsibility for client DNS resolutions) and injects carefully crafted (but incorrect) responses to satisfy these queries. Because DNS recursive resolvers cache their results, once a domain is poisoned in this way, all the resolver's clients will receive erroneous resolutions. [28, 63, 85]. Another class of attack, *query interception* occurs when the attacker can mediate communications from the requester to its recursive resolver (or from the recursive resolver to other nameservers) and substitute incorrect responses [61, 72, 94]. Yet other attacks, including ones discussed in Chapter 3 and Chapter 4, involve exploiting configuration errors wherein a domain's `NS` records include so-called *dangling delegations* — nameserver FQDNs whose own domains can be controlled by an attacker (*e.g.*, because they have expired) and then used to influence resolution [8, 62].

Finally, *DNS infrastructure hijacks*, which are the focus of this chapter, result from an attacker taking control of the mechanism used to update DNS configurations — typically by compromising the domain holder's account with their domain registrar or the systems of the registrar itself.[1] The domain's registrar enjoys privileged access to update the domain's records in its top-level domain (TLD) registry database that is, in turn, relied upon when the domain is resolved. In these attacks, the attacker replaces the `NS` records for the domain with nameservers controlled by the attacker, and arranges that specific `A` records pertaining to targeted

[1] Sadly, this is not idle speculation, and we are aware of more than a few instances of registrar compromises. For example, quoting from the indictment of several officers of the Chinese Ministry of State Security *United States v Zhang et al.*: "On August 28, 2013, LIU sent MA a link to a news article that explained how the Syrian Electronic Army (SEA) had hacked into the computer systems of Company L, a domain registrar, in order to facilitate intrusions. On December 3, 2013, members of the conspiracy used the same method as the SEA to hack into the computer systems of Company L and hijack domain names of Company H, which were hosted by Company L." [88]. Similarly, the attacks against `pch.net` (which provides DNS infrastructure services for the ccTLDs of over 130 countries) involved attackers obtaining privileged credentials at `pch.net`'s registrar, Key-Systems [56].

subdomains (*e.g.*, `mail`, `vpn`) will point to the attacker's infrastructure. While initial reports of such attacks identified use for activism or mischief [87, 45], more recently several large security firms have reported on their use to compromise government agencies and large infrastructure providers [4, 29, 46, 65, 86].

Thus far there is little academic literature on this issue. One notable exception is the recent paper from Houser *et al.* using a combination of past domain hijacks and Farsight's Passive DNS data set to train a machine learning classifier to detect such hijacks (although the authors do not attempt to use this classifier to detect any new hijacks) [50]. Our work focuses on the same problem domain, but has both different aims and means — we seek to retroactively identify sophisticated attacks in the wild and do so via a constructive framework based on concrete attacker objectives.

5.2.2 DNSSEC and TLS

There are multiple standard security mechanisms used to protect against domain hijacking. Specific to DNS, the Domain Name System Security Extensions (DNSSEC) [78] provides cryptographic guarantees of authenticity and integrity for each DNS record, via a trust hierarchy that mirrors the DNS delegation hierarchy. However, DNSSEC is not widely deployed in practice [76] and is of limited benefit in DNS infrastructure hijacks, because it is commonly the very authority for updating DNS records (including their signatures) that has been hijacked [56].

Another defense is provided by Transport Layer Security (TLS) [75] wherein an application provides certificates signed by a third-party Certificate Authority (CA). These certificates attest that the endpoint really represents one or more FQDNs (specified in the Common Name and Subject Alternative Name fields of a TLS certificate) and that the public key contained in the certificate is valid and should be used to bootstrap a secure session. Thus, assuming that clients (*e.g.*, a VPN) validate such certificates before allowing further communication, hijacking a target's DNS resolution will not be sufficient to read the traffic being intercepted. However, the premise underlying this arrangement is that the CA's signed attestation is backed by their

appropriate due diligence that the party obtaining the certificate is really who they say they are. However, as we will explain in Section 5.3, modern certificate provisioning practices have rendered this guarantee itself vulnerable to domain hijacking.

5.3 DNS Infrastructure Hijacks

In this section, we describe the stages of the DNS infrastructure hijacks on which we focus, and the challenges in identifying them. DNS infrastructure hijacks are complex and depend on several advanced capabilities. In this section we outline the stages of such an attack.

Develop Capability. Attackers start by developing the ability to modify the target domain's `NS` records or `A` records. This step can leverage three different paths: (a) compromising the account credentials the registrant uses with their registrar or DNS provider; (b) compromising the registrar that administers the domain; or (c) compromising the registry DNS configuration database [4, 56]. In the latter two cases, the hijack can extend to all domains under the registrar's or registry's control. In any of these cases the attacker can also typically disable protections provided by DNSSEC [56].

Attacker Infrastructure. In the previous step the attacker establishes control over a domain's DNS resolution, but the ultimate objective of the attack is to gain control over the *infrastructure* served by the domain. To this end, the adversary must redirect sensitive subdomains — used to receive authentication credentials — to counterfeit infrastructure imitating those services (such as a mail login page), with the goal of accessing credentials via adversary-in-the-middle (AitM) techniques.

Adversary-in-the-Middle Capability. In recent years, a combination of users expecting a "secure connection" and browser vendors initiatives [42, 43, 44] make harvesting credentials without a TLS certificate significantly harder.[2] To obtain a TLS certificate that will satisfy modern clients, a domain owner must request one from a browser-trusted *Certificate Authority* (CA) who,

[2] Attacks without TLS certificates can still succeed either by socially engineering users to click through security warnings or through the use of drive-by malware [17, 25, 82], but they are not the focus of this work.

in turn, is responsible for verifying domain ownership before provisioning a certificate. While CAs verify ownership using several methods, the one most relevant is *domain validation* which uses demonstrations of real-time control over the domain as a proxy for proof of ownership [31, 38] (*e.g.*, posting a given challenge token in a `TXT` record for the domain). Most certificates today are requested and issued in this manner, using a fully-automated process called the Automatic Certificate Management Environment (ACME) [13] network protocol.[3]

Thus, an attacker's ability to control DNS resolution can be sufficient to obtain a browser-trusted TLS certificate for that domain. In fact, there is clear documentation of attackers obtaining certificates for sensitive subdomains (*e.g.*, `mail`, `vpn`) in recent DNS infrastructure hijacks [56, 65]. On obtaining such a certificate, an attacker can then deploy it to its counterfeit infrastructure (*e.g.*, servers for webmail, VPN, IMAP, etc.). Traffic diverted from the target domain to its counterfeit counterpart will accept and use the presented certificate, allowing the attacker to extract the cleartext of any credentials sent. However, acquiring a counterfeit certificate is not entirely covert since the Certificate Transparency (CT) standard [60] requires CA's to publish new certificates in a public audit log before they are issued.[4]

Active Hijack. In this stage the attacker actively redirects resolution of the target domain to their own infrastructure. Typically, this happens serially for short durations over weeks to evade detection [4]. Moreover, if the imitating infrastructure is a reasonable replica of the target and uses a browser-trusted (though maliciously obtained) TLS certificate, users have no immediate way of knowing they have been redirected. Of course, users are more likely to report suspicious activity if they log into the mail service but do not see their mail, so sophisticated attackers tunnel traffic back to the legitimate infrastructure (*e.g.*, using the proxy-like Internet Content Adaption Protocol (ICAP) [23]) to further hide the attack [46]. While traffic is redirected, the attacker collects credentials which can then be used to laterally move into the target organization. While

[3]For a full description on the ACME workflow, we refer readers to RFC 8555 [13] and the original Let's Encrypt paper [2].

[4]While not strictly mandatory, major browser vendors have made CT participation a pre-requisite for CAs to be trusted by their software.

the redirections may last for short durations, the attacker infrastructure typically is functional and responsive throughout.

Post Hijack. Upon successful conclusion of the DNS hijack (*i.e.*, the attacker has acquired login credentials and established a beachhead inside the organization), the attacker can then decommission its counterfeit infrastructure. However, we have seen attackers not only reuse its infrastructure (based on IP addresses observed) for different targets, but also leave infrastructure up for days, sometimes months, after the hijack (Section 5.5.1).

Summary. This class of DNS hijack is characterized by: (a) multiple brief updates to DNS configuration minimize opportunities for detection; (b) attacker infrastructure that is responsive for the duration of the hijack and sometimes much longer; (c) attacker infrastructure that responds with a browser-trusted maliciously obtained certificate.

5.4 Methodology

We now describe our methodology for identifying historical DNS infrastructure hijacks. The main insight of our approach is that from a third-party vantage point, it is far more feasible to identify the precursor of a hijack — the infrastructure attackers establish to impersonate the domain — rather than tracking changes in DNS configuration that reflect a hijack of the domain. This infrastructure, such as a server that mimics the mail login page of a target domain, has a key requirement for a successful attack: a valid browser-trusted TLS certificate controlled by the attacker to assert authority over the target domain. Thus, once the attacker has established their infrastructure, the certificate will appear in global IPv4 scans of specific TCP ports that return TLS certificates.

Based on this insight, our approach identifies hijacks by first discovering the attacker infrastructure used to target domains. We use data from Internet-wide scans to build a model of deployed infrastructure — a *deployment map* — for every domain over time. We analyze these deployment maps to identify suspicious deployments that strongly correlate with hijacks.

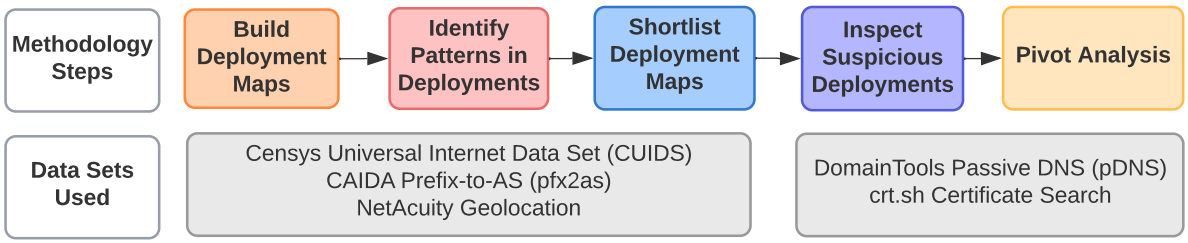

Figure 5.1. Our five step methodology to identify DNS infrastructure hijacks and the data sets used in the steps.

Moreover, this approach provides additional context in the form of the longitudinal use of IP addresses, certificates, and CAs associated with each domain. This context provides a baseline for characterizing new infrastructure that appears with authoritative claims on the domain.

Our approach consists of five steps, illustrated in Figure 5.1 and described in the following sections. We first *build deployment maps* for every domain. Next, we *use patterns* in these maps to *shortlist* domains with potentially suspicious deployments. Then, we *manually inspect* this shortlist with heuristics and supplemental data to establish confidence that the suspicious deployments reveal hijacks. Finally, using the domains inferred as hijacked, we *pivot* to examine if other domains not captured by our methodology were targeted using the same attacker infrastructure.

5.4.1 Building Deployment Maps

A deployment map models *where* and *when* infrastructure on the Internet provided service for a domain. By design, it clusters deployments to reveal suspicious infrastructure used to mimic sensitive target domains. We build deployment maps using *longitudinal* Internet-wide scans that retrieve TLS certificates from responsive hosts. In our work we use the Censys Universal Internet Data Set (CUIDS) [22] with records for more than 71M IP addresses spanning January 2017 to March 2021.

The CUIDS includes comprehensive weekly scans for TLS certificates across the entire IPv4 address space. This weekly scan data captures when a certificate was seen at a specific IP

Table 5.1. Annotated IP scan data related to `kyvernisi.gr` for the month of April, 2019. **T** indicates if certificate is browser trusted. **S** indicates if certificate secures a subdomain where credentials are typically entered.

Scan Date	IP Address	Ports (TCP)	ASN	CC	crt.sh ID	Issuing CA	T	S	Name(s) Secured
2019-04-09	84.205.248.69	[443, 993, 995]	35506	GR	1245068498	DigiCert Inc	T	T	[mail.kyvernisi.gr]
2019-04-16	84.205.248.69	[443, 993, 995]	35506	GR	1245068498	DigiCert Inc	T	T	[mail.kyvernisi.gr]
2019-04-23	95.179.131.225	[993]	20473	NL	1394170951	Let's Encrypt	T	T	[mail.kyvernisi.gr]
2019-04-23	84.205.248.69	[443, 993, 995]	35506	GR	1245068498	DigiCert Inc	T	T	[mail.kyvernisi.gr]
2019-04-30	84.205.248.69	[443, 993, 995]	35506	GR	1245068498	DigiCert Inc	T	T	[mail.kyvernisi.gr]

address and port.[5] Starting with this data set, we annotate the IP address with the origin AS (using CAIDA Prefix-to-AS mappings [19]) and its geolocation (using NetAcuity [36]). We further annotate this data with information extracted from the certificate, including the Subject Alternative Names (SANs) [74] specifying the domain names secured by the certificate [39] and the Issuer CA. Additionally, we identify if the certificate is browser-trusted,[6] As an example, Table 5.1 shows these annotations for four scans that found certificates securing `kyvernisi.gr` in April 2019. Using this annotated data, we identify the observable infrastructure associated with every domain. For the example in Table 5.1, scans at two different IP addresses (`84.205.248.69` and `95.179.131.225`) returned a certificate securing the domain `kyvernisi.gr`. We refer to those IP addresses and the certificates they return as the *observable infrastructure* for `kyvernisi.gr`.

We cluster the observable infrastructure for a domain into separate *deployment groups*. For a given domain, we define a deployment group as the observable infrastructure associated with IP addresses originated by the same ASN on a given date. Our assumption is that the infrastructure used by an attacker will be distinct from the infrastructure used by the domain owner. Thus our goal is to have the legitimate and the attacker infrastructure appear as separate deployment groups in the map.

A deployment group seen longitudinally over a period of time is referred to as a *deployment*, and *all* deployments for a domain together represent the domain's *deployment map*. For example, the two rows for the April 23, 2019, scan of `kyvernisi.gr` in Table 5.1 have IP addresses in two different ASNs (20473 and 35506) that each return certificates for `kyvernisi.gr`. Each forms its own deployment group for that date. As a result, for the period of April 2019 the domain `kyvernisi.gr` has two deployments which, together, form its deployment map as shown in Figure 5.2.

Instead of building a single deployment map for every domain, we break the period from

[5] We use data for ports that are typically associated with TLS certificates and, hence, targeted by attackers (ports [443, 465, 587, 993, 995]).

[6] We mark a certificate as trusted if it is trusted by either Apple, Microsoft, or Mozilla. We do not use the Chrome Root Store because it was rolled out after the time frame of our study.

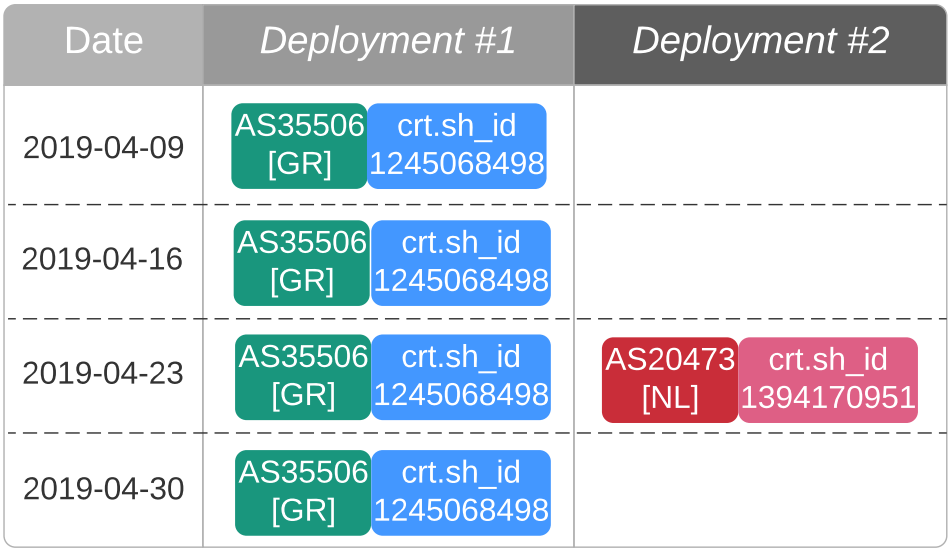

Figure 5.2. Deployment Map of `kyvernisi.gr` for the month of April 2019 capturing the two deployments. Deployment #1 is a stable deployment. Deployment #2 is a transient deployment since it only shows up in one scan indicating suspicious behavior.

January 2017 to March 2021 into nine six-month periods. For each of these periods, we build a deployment map for all domains with a publicly-visible TLS certificate in the corresponding six-month period. We consider each period independently, *i.e.*, a domain's lifetime may span multiple deployment maps, each of which we evaluate separately. Breaking up the four-year period leverages temporal locality in deployments to better account for both long-term stable transitions and brief transient changes. From the four years of CUIDS scan data, we construct deployment maps for more than $22M$ domains.

5.4.2 Identify Suspicious Patterns

Our goal is to identify attacker infrastructure that appears as new *and* temporary deployments in location and time. To that end, we categorize deployment maps into three patterns: *stable*, *transition*, and *transient* deployments. The first two patterns correspond to benign deployments, and the third matches suspicious deployments.

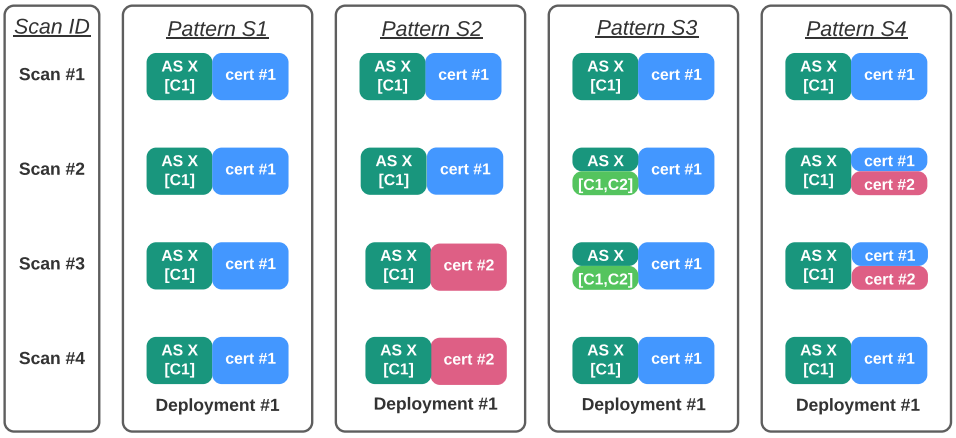

Figure 5.3. Representative stable patterns (S) in deployment maps. The consistent use of the same ASNs over time indicate stable and benign deployment patterns.

Stable Patterns

The patterns in Figure 5.3 represent benign, stable deployment maps. Most domains fall into this category: $21.25M$ (96.5%) of the $22M$ domains are stable.

Pattern S1 represents a single deployment presenting the same certificate from IP addresses in AS X and geolocated to country $C1$. In this pattern the observed infrastructure for a domain does not change over time, and the certificates associated with the domains have long validity periods. Pattern S2 is similar, but represents the rollover of certificates on expiry in a stable deployment. The only change in observed infrastructure for the domain is a change in the certificate associated with the domain.

Patterns S3 and S4 capture domains that show minor changes in an otherwise stable deployment, such as the appearance of new IP addresses geolocated to a different country but the same AS, or a new certificate securing the domain being deployed on the same observed infrastructure. Such changes could reflect the domain owner testing new endpoints or services, or expanding geographical deployment with the same provider. We consider these patterns as stable and benign given the consistent use of the same AS.

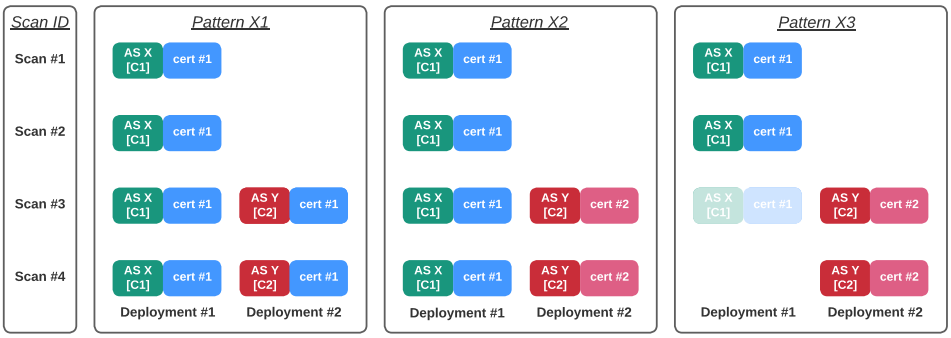

Figure 5.4. Representative transition patterns (X) in deployment maps. These deployment patterns capture long-term stable changes in deployment.

Transition Patterns

While most domains have stable deployments over short time scales, over longer time periods infrastructure associated with domains can change deployments for a variety of legitimate reasons. The patterns in Figure 5.4 represent transitions that reflect a significant change in observed infrastructure, but the change is stable going forward in time. Such observable transitions in infrastructure will appear as new deployments in the deployment maps. $650k$ (2.95%) of the domains have transition patterns.

The first two patterns reflect the expansion of domain deployments. Domain owners could be scaling up or diversifying their infrastructure into a new AS (Pattern X1), perhaps even with an additional certificate for use with a new provider (Pattern X2). From our experience examining such deployment maps, these patterns typically correspond to the adoption of cloud services in addition to on-premises infrastructure. The third pattern X3 reflects a shift to completely new infrastructure, with a new certificate for the domain being served from IP addresses in a different AS. A common cause of this pattern is a domain owner switching hosting providers, or the domain changing ownership. At times we see a small overlap between the old and new deployments (the shaded old deployment in the figure). In most cases, DNS resolution switches to the new infrastructure and the previous infrastructure is torn down.

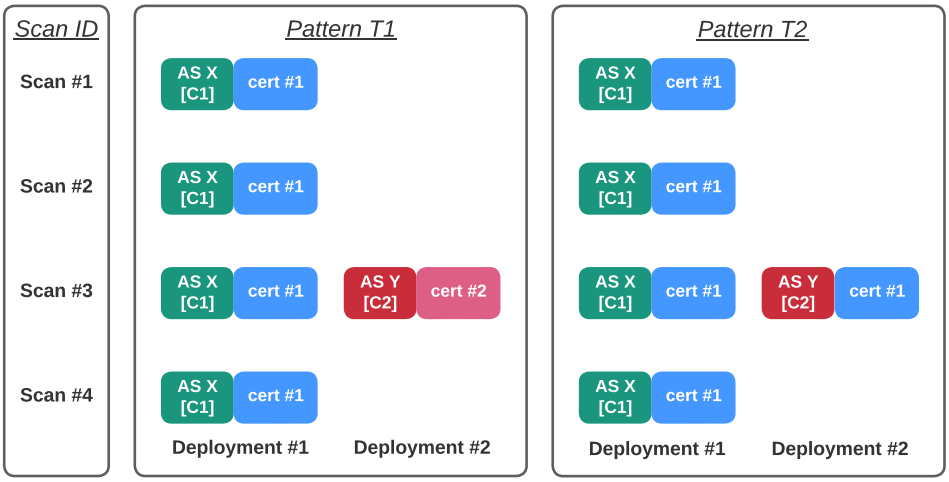

Figure 5.5. Representative transient patterns (T) in deployment maps. The transient nature of the attacker infrastructure created to the mimic the target domain indicates suspicious deployments.

Transient Patterns

The third category of patterns reflects transient changes in deployment maps, and 28k (0.13%) deployment maps fall into this category.[7] Given the transient nature of attacker infrastructure created to mimic the target domain, we expect these patterns to capture such malicious deployments. The key to this identification is the time threshold used to distinguish transient from transition changes. The threshold for attacker infrastructure lifetime needs to be long enough to reflect a malicious deployment, but short enough to avoid false positives. We find three months — the typical validity period of free certificates — to usefully balance these tradeoffs. The intuition is that the attacker infrastructure is tied to the validity of the maliciously-obtained certificate. In most cases during this three-month period, the attacker either harvests credentials to laterally move into the target domain, or the attacker has been discovered. Thus we focus on searching for attacks that use *transient deployments* that do not persist beyond three months ($\sim$12 certificate scans).

Pattern T1 (Figure 5.5) reflects a deployment map that consists of a long-term stable deployment combined with a transient deployment using a new certificate and different infras-

[7] We find 77k (0.35%) of domains too noisy or unstable to categorize.

tructure. While this suspicious pattern often indicates a hijack, sometimes the evidence is not so conclusive. For instance, a domain may use AS16509 (Amazon) for their stable deployment, but briefly also use AS14618 (also Amazon). It is difficult for a third party to conclude that this activity is malicious — the transient appearance of a different AS for the same provider is not uncommon, but using a new certificate for the domain is. As a result, we label deployment maps matching Pattern T1 as suspicious, and then evaluate them on a case-by-case basis for a final verdict (Section 5.4.4).

Pattern T2 also reflects the appearance of a transient deployment against the background of a stable deployment, but the certificate associated with the transient deployment is the same as the one used by the stable deployment. This pattern typically indicates a legitimate expansion in infrastructure by the domain owner, but can also reflect malicious activity. In particular, it can capture the prelude to hijacks: the stage of an attack in which the attacker sets up a parallel infrastructure that proxies to the actual IP. Due to the proxy setup, the certificate scan at an IP address controlled by an attacker returns the legitimate certificates by proxying to an IP address of the stable deployment. Use of the proxy implies that the domain was targeted but not necessarily hijacked, although it is possible the original certificate was exfiltrated and is being used by the attacker [86]. As with other suspicious deployments, we must manually evaluate deployment maps matching this pattern with care.

5.4.3 Shortlist Deployment Maps

We narrow our candidate set of hijacks to deployment maps to those matching Patterns T1 or T2. We then use a set of heuristics to remove common cases that result in false positive or inconclusive results.

We first check if the transient deployment ASN is organizationally related to the ASN of its stable deployment using the CAIDA AS-to-Organization Mapping [20]. Second, we also check if the transient deployment geolocates to the same countries as any stable deployment. Third, we check if a domain is missing from 20% of scans in the six-month period, or if the the

domain displays similar transient deployments in multiple (three or more) consecutive six-month periods. These checks identify domains where *our visibility* into the domain's deployment is too unstable to make a determination. We prune deployment maps that match any of these three criteria from our candidate set.

As a final step, we only keep deployment maps where the transient deployment has a browser-trusted certificate securing a *sensitive* subdomain: a subdomain with a substring matching [`secure`, `mail`, `remote`, `login`, `logon`, `portal`, `admin`, `owa`, `vpn`, `connect`, `citrix`, `signin`, `cloud`, `box`, `account`, `intranet`, `imap`, `smtp`, `pop`, `ftp`, `api`]. We manually compiled this list based on common names of subdomains targeted in early attacks. For deployment maps not matching this naming criteria, we still shortlist them if we observe an extremely stable deployment for a six-month period before and after the transient, thus indicating a truly anomalous occurrence.

Thus, our final candidate set contains transient deployments originated by a different ASN *and* geolocated to a different country (relative to the stable deployment) either affecting a sensitive domain or representing a rare anomalous occurrence. The deployment map for `kyvernisi.gr` (Figure 5.2) is in this set since the transient deployment is for the `mail` subdomain, is originated by a different ASN (AS 20473), and geolocates to a different country (Netherlands) relative to the stable deployment. After applying these heuristics, we shortlist 8143 domains as suspicious. Of these 47 domains are shortlisted for being truly anomalous.

5.4.4 Inspect Suspicious Deployments

After shortlisting, we have a candidate set of domains whose deployment maps have very suspicious transient deployments. To decide whether the domains have been involved in hijacking attacks, we inspect the 8143 domains. This step is the most time consuming since it manually examines several features that can provide corroborating evidence that a hijack has occurred.

We first cross-reference these domains with passive DNS (pDNS) and certificate trans-

parency (CT) data sets. The main advantage of pDNS data is that it requires no cooperation from zone owners (*e.g.*, it does not require access to restricted zone files for ccTLDs).[8] We use DomainTools' pDNS data set [34]. Since domains targeted for hijacking are by their nature in active use, we expect them to be actively queried. For each domain, pDNS reports the first and last time a specific resolution was seen, if at all. However, pDNS data captures only domains that are actively queried on networks observed by DomainTools' sensors [95]. We also cross-reference these domains with a certificate transparency (CT) data set using the crt.sh [79] service which allows us to search CT logs for certificates issued to a domain. Based on this cross-referencing, we find only 1256 of the 8143 shortlisted domains worth manually examining. For the remaining domains, we neither saw relevant data in the pDNS or CT data sets, nor were they truly anomalous occurrences.

For domains matching Pattern T1 we check pDNS for changes in nameserver delegation and in the resolution of subdomains listed in the suspicious certificate returned from the transient deployment. If the suspicious certificate is *issued* near the time pDNS observes changes in nameserver delegations or changes in domain resolution, then we conclude that the certificate was maliciously obtained and the pDNS changes reflect a hijack. We make this determination given that: the transient deployment is in a different ASN and different country; it returns a suspicious new certificate for a sensitive domain not used elsewhere; and corroborating data in pDNS records a short-lived change in DNS resolution. We find 22 domains that match these criteria.

For domains matching Pattern T2, confidently inferring that an attack occurred is more challenging, since the transient deployment only captures the prelude to an expected hijack (Section 5.3). We first check if pDNS captures either a change in DNS resolution for the targeted subdomain or a change in nameserver delegation. We then check the CT logs to see whether a new certificate was issued in the same time period that the transient deployment was observed. If a new certificate exists that secures a sensitive subdomain for which we also observe a change in

[8]The domain hijacks we identified spanned 15 TLDs; of those, we have access to only three of their zone files.

DNS resolution, we consider the certificate suspiciously obtained.

We conclude a domain was hijacked in the presence of corroborating evidence from both pDNS and CT. We find 6 domains that match this criteria; of those, 4 domains were shortlisted for being truly anomalous occurrences. Notably for `ais.gov.vn`, while we see evidence of redirection in pDNS, we do not find any suspiciously issued certificates. As a result, we mark the domain as *targeted* as opposed to hijacked.

We conclude that a domain has been hijacked only when there is significant corroborating evidence (Section 5.5.1). In the absence of corroboration, if a domain has a transient deployment that is truly anomalous — it is the only transient deployment in an otherwise stable deployment map — we mark the domain as *targeted but not hijacked*. This situation can arise if the attack was unsuccessful or if none of our data sources captured it. Of the domains shortlisted for being truly anomalous, we find convincing evidence that 24 domains were targeted in this fashion (Section 5.5.4).

5.4.5 Pivot Analysis

Using the attacker infrastructure of the 28 hijacked domains (22 T1 domains and 6 T2 domains) identified from manual inspection, we pivot to identify other domains referencing the same nameserver delegations or resolving to the same IP addresses using the pDNS logs. This step identifies 13 more domains. We discuss the reasons why deployment maps do not capture these domains in Section 5.5.2. For these domains, we try to also identify the maliciously obtained certificate using the process detailed above.

5.4.6 Limitations

Our methodology has several limitations. First, performing longitudinal inference as an independent third party restricts us to what is visible in publicly available data sets. Thus, in absence of groundtruth, delineating an attack from legitimate changes in the extremely dynamic DNS ecosystem relies on multiple data sources to discover corroborating evidence matching the

attack profile, and false positives are still a risk. Consequently, we find ways to aggressively prune the data to minimize false positives and inconclusive results (suspicious events with no corroborating data). As a result, our pruning potentially biases our inferences toward domains with more stable standardized deployments for which we could confidently infer were hijacked.

At the same time, we are also limited by coverage issues with the data sets: either too coarse-grained to catch the ephemeral hijack activity (*e.g.*, weekly active IPv4 scans); addresses that do not respond to scanning; or in the case of traffic data, being limited to those networks where passive DNS traffic is gathered for commercial use (pDNS).

Finally, our method requires manual inspection of candidate hijacks. While our focus on domains that rely on TLS certificates renders this requirement tractable, we hope our experiences enable development of more automated techniques in the future.

Given these limitations, our results are therefore a lower bound (perhaps a severe one). However, our methodology did uncover domains not previously identified, establishing that this class of hijack is a serious ongoing concern.

5.5 Results

Applying our methodology to historical data between January 2017 and March 2021, we identified 41 domains as hijacked and 24 domains as targeted. To illustrate this approach concretely, we first describe how we use deployment maps to identify a set of related hijacked domains in Kyrgyzstan. We then discuss the overall features of the full set of 41 hijacked domains, independent sources of validation, and the longitudinal implications of the hijacks. We then discuss the features of the 24 domains that were targeted, but for which we did not observe a hijack. Finally we discuss trends in the organizations that were targeted, the infrastructure used by the attackers, and our disclosure efforts.

5.5.1 The Kyrgyzstan Hijacks

As a concrete example of our method, we describe how we found that a set of Kyrgyzstan government domains were the targets of attacks: `mfa.gov.kg`, `invest.gov.kg`, `fiu.gov.kg`, and `infocom.kg`.

For the four-year duration of our study, the deployment map for `mfa.gov.kg` (the Ministry of Foreign Affairs, Kyrgyzstan) contains a stable infrastructure deployment in Kyrgyzstan, hosted on IP addresses originated by AS 39659 (Infocom, Kyrgyzstan). The deployment map also contains a transient deployment starting December 22, 2020. This transient deployment has a new certificate for the sensitive subdomain `mail.mfa.gov.kg` that is returned from an IP address located in Russia and originated by AS 48282 (VDSINA Hosting, Russia). As a result, the deployment map matches Pattern T1. Additionally, the domain appears in more than 80% of the scans in the six-month period under consideration, the stable and transient deployments are not related to the same AS organization, and are not geolocated in the same countries.

At this point the transient deployment is flagged as suspicious. For a final determination, we use additional data sources for corroborating evidence. From the pDNS data, the stable authoritative nameservers for `mfa.gov.kg` were `ns1.infocom.kg` and `ns2.infocom.kg`. On December 20, 2020, the authoritative nameserver delegations were updated to `ns1.kg-infocom.ru` and `ns2.kg-infocom.ru`, and those nameservers resolved `mail.mfa.gov.kg` to the IP address using the suspicious certificate. Both NS and A redirections continued until January 12, 2021. Using the certificate transparency logs from `crt.sh`, the TLS certificate returned from the transient deployment for `mail.mfa.gov.kg` was issued on December 21, 2020, by Let's Encrypt.[9]

Given all of the evidence — a transient deployment in a different AS that returns a new, suspicious certificate for `mail.mfa.gov.kg`, together with changes to the authoritative nameservers at the same time that the new certificate was issued — we conclude that: the domain was hijacked, and the attacker infrastructure used IP address 94.103.91.159 and nameservers

[9] https://crt.sh/?id=3810274168

ns{1,2}.kg-infocom.ru to target the domain.

The domain invest.gov.kg (Investment Portal, Kyrgyzstan) was attacked on December 28, 2020, a week later than mfa.gov.kg. The domain invest.gov.kg also has a transient deployment with a new certificate for mail.invest.gov.kg, and it uses the same anomalous AS and authoritative nameservers as the attack on mfa.gov.kg. These nameservers redirected mail.invest.gov.kg to the transient deployment on December 28, 2020, for a week.

After identifying that mfa.gov.kg and invest.gov.kg were attacked, as the last step we pivot to investigate whether other domains were targeted using the same attacker network and nameserver infrastructure. From the pDNS data, we see ns{1,2}.kg-infocom.ru being briefly used as authoritative nameservers for fiu.gov.kg (Financial Intelligence Service, Kyrgyzstan) in December 2020 and for infocom.kg (State Agency for Information Services) in January 2021. Those anomalous nameservers returned resolutions for mail. fiu.gov.kg and mail.infocom.kg to a server in the same AS as the attacker infrastructure for the other two domains. [10] Cross-referencing with the CT logs at crt.sh shows new TLS certificates issued for mail.fiu.gov.kg and mail.infocom.kg in the same time frame. Based upon this evidence, we also conclude that the domains fiu.gov.kg, and infocom.kg were the targets of a hijack.

The reason why the deployment maps for fiu.gov.kg and infocom.kg did not match a transient pattern is that the IP scans did not find any stable observable infrastructure for the domains. This case demonstrates the utility of the pivot step, enabling discovery of attacks for domains that do not have stable observable infrastructure. While these hijacks focus on harvesting credentials, mfa.gov.kg was redirected again in May 2021 (outside the period of our study) luring users into installing a "security update". This executable links to a malware family known as Tomiris which researchers have linked to SolarWinds [58].

```
<div id="errorMessageDiv" class="errorMessage">
  Для продолжения работы с сервисом электронной почты
    необходимо установить обновление безопасности:
  <br>
<a href="update-mfa.exe">Скачать обновление</a></div>
```

Figure 5.6. Error message added to the counterfeit `mail.mfa.gov.kg` site to trick users into installing malware. Translation courtesy Google Translate: "To continue using the email service, you must install the security update: Download Update".

Evolution of Kyrgyzstan Hijacks

Starting late 2020, Censys started collecting additional service and device context [35] including HTTP Responses. This additional context provides further visibility into the nature and evolution of these attacks. For example, during our study we observed that visitors to `mail.mfa.gov.kg` (a Zimbra-based mail login page) were redirected to a site in Russia that also provided a (valid) counterfeit certificate for the site — consistent with the well-established strategy of harvesting credentials typed into these web pages. Indeed, using Censys' HTTP response data we were then able to verify that while the page mimicked the Zimbra login page's look and feel, it differed from the standard Zimbra code. Moreover, in May 2021, we observed the domain redirected to *a new* IP: `178.20.46.22` (again originated by AS 48282 in Russia). This server also mimicked the Zimbra mail login page, but included *additional JavaScript code* to render an error message (shown in Figure 5.6) intended to socially engineer users into installing the "security update" software: *update-mfa.exe*. We identified this executable on VirusTotal, uploaded shortly after the redirection is initiated [92]. This software, written in Go, has since been identified as the downloader for the *Tomiris* implant, itself loosely associated with the SolarWinds attack [58]. We surmise that the attacker found that credentials interception alone was insufficient for their needs (*e.g.*, perhaps due to the use of multi-factor authentication for some users).

[10]The legitimate nameservers for these domains are under `infocom.kg`.

5.5.2 Hijacked Domains

Table 5.2 and Table 5.3 together report the 41 domains we infer as hijacked. These domains span government agencies, infrastructure providers, and even registrar and registry operators. Since most of the domains are associated with government agencies, we group the domains by their country (CC) and order each group by the time of hijack (Hij). For each domain, we report the reason we identify the domain (Type) and the subdomain targeted (Sub). We also report whether there is corroborating nameserver (pDNS) or certificate transparency (CT) evidence, as well as the network infrastructure and location of the victim and attacker deployments.

We identify 20 domains as hijacked directly using deployment maps, indicated by the pattern T1. As with the Kyrgyzstan domains discussed above, the deployment maps revealed: a transient deployment in a different AS and country; the transient deployment returned a suspicious newly-issued certificate targeting one specific sensitive subdomain; and resolutions in pDNS revealed short-lived changes to the authoritative nameservers that briefly redirected traffic to the infrastructure (IP address) in the transient deployment.

We identify another 6 domains as hijacked using a combination of deployment maps and CT data, indicated by the pattern T2. Typically in these cases, we first captured the prelude to the hijack. While we see a transient deployment in a different AS and country, the transient deployment returns a certificate associated with the stable deployment. However, on further inspection the pDNS logs revealed short-lived changes in resolution for a sensitive subdomain from the stable deployment to the transient deployment. Moreover, cross-referencing the subdomain with the CT logs reveals a suspicious newly-issued certificate for the sensitive subdomain in the same time frame: although the transient deployment did not return the suspicious certificate in the CUIDS scans, it did appear in the CT logs.

Then, using the attacker infrastructure identified to pivot, we identify another 13 domains as hijacked. We identify 6 domains pivoting on the attacker infrastructure IP address (indicated

Table 5.2. List of 28 domains identified as hijacked between January 2017 and March 2021 using deployment maps. Type indicates how we identified the domain. For every domain, we report the time of first hijack, the targeted subdomain, the country associated with the organization behind the domain, corroborating evidence from pDNS and CT, as well the network infrastructure and geolocation of the attacker and the target domain. The domains not highlighted are associated with Sea Turtle campaigns. The domains highlighted in gray have not previously been identified. The `.kg` domains partially match findings in a report by Kaspersky [58].

Type	Hij.	Targeted Domain Information			Cross Ref		Attacker Infra. (Transient)			Legitimate Infra. (Stable)	
		CC	Domain	Sub.	pDNS	crt	IP	ASN	CC	ASNs	CCs
T1	May'18	AE	mofa.gov.ae	webmail	✔	✔	146.185.143.158	14061	NL	[5384,202024]	[AE]
T1	Sep'18	AE	adpolice.gov.ae	advpn	✔	✔	185.20.187.8	50673	NL	[5384]	[AE]
T1*	Sep'18	AE	apc.gov.ae	mail	✖	✔	185.20.187.8	50673	NL	[5384]	[AE]
T2	Sep'18	AE	mgov.ae	mail	✔	✔	185.20.187.8	50673	NL	[202024]	[AE]
T1	Jan'18	AL	e-albania.al	owa	✔	✔	185.15.247.140	24961	DE	[5576]	[AL]
T2	Nov'18	AL	asp.gov.al	mail	✔	✔	199.247.3.191	20473	DE	[201524]	[AL]
T1	Nov'18	AL	shish.gov.al	mail	✔	✔	37.139.11.155	14061	NL	[5576]	[AL]
T1	Dec'18	CY	govcloud.gov.cy	personal	✔	✔	178.62.218.244	14061	NL	[50233]	[CY]
T1	Dec'18	CY	webmail.gov.cy	.	✔	✔	178.62.218.244	14061	NL	[50233]	[CY]
T1	Jan'19	CY	sslvpn.gov.cy	.	✔	✔	178.62.218.244	14061	NL	[50233]	[CY]
T1	Feb'19	CY	defa.com.cy	mail	✔	✔	108.61.123.149	20473	FR	[35432]	[CY]
T1	Nov'18	EG	mfa.gov.eg	mail	✔	✔	188.166.119.57	14061	NL	[37066]	[EG]
T2	Nov'18	EG	mod.gov.eg	mail	✔	✔	188.166.119.57	14061	NL	[25576]	[EG]
T2	Nov'18	EG	nmi.gov.eg	mail	✔	✔	188.166.119.57	14061	NL	[31065]	[EG]
T1	Nov'18	EG	petroleum.gov.eg	mail	✔	✔	206.221.184.133	20473	US	[24835,37191]	[EG]
T1	Apr'19	GR	kyvernisi.gr	mail	✔	✔	95.179.131.225	20473	NL	[35506]	[GR]
T1	Apr'19	GR	mfa.gr	pop3	✔	✔	95.179.131.225	20473	NL	[35506,6799]	[GR]
T2	Sep'18	IQ	mofa.gov.iq	mail	✔	✔	82.196.9.10	14061	NL	[50710]	[IQ]
T1	Dec'20	KG	invest.gov.kg	mail	✔	✔	94.103.90.182	48282	RU	[39659]	[KG]
T1	Dec'20	KG	mfa.gov.kg	mail	✔	✔	94.103.91.159	48282	RU	[39659]	[KG]
T1	Dec'17	KW	csb.gov.kw	mail	✔	✔	82.102.14.232	20860	GB	[6412]	[KW]
T1*	Apr'19	KW	moh.gov.kw	webmail	✖	✔	91.132.139.200	9009	AT	[21050]	[KW]
T2	May'19	KW	kotc.com.kw	mail2010	✔	✔	91.132.139.200	9009	US	[57719]	[KW]
T1	Nov'18	LB	medgulf.com.lb	mail	✔	✔	185.161.209.147	50673	NL	[31126]	[LB]
T1	Nov'18	LB	pcm.gov.lb	mail1	✔	✔	185.20.187.8	50673	NL	[51167]	[DE]
T1	Oct'18	LY	noc.ly	mail	✔	✔	188.166.119.57	14061	NL	[37284]	[LY]
T1	Jan'18	NL	ocom.com	connect	✔	✔	147.75.205.145	54825	US	[60781]	[NL]
T1	Mar'19	SY	syriatel.sy	mail	✔	✔	45.77.137.65	20473	NL	[29256]	[SY]

Table 5.3. List of 13 domains identified as hijacked between January 2017 and March 2021 using pivot analysis. Type indicates how we identified the domain. For every domain, we report the time of first hijack, the targeted subdomain, the country associated with the organization behind the domain, corroborating evidence from pDNS and CT, as well the network infrastructure and geolocation of the attacker and the target domain. The domains not highlighted are associated with Sea Turtle campaigns. The `.kg` domains partially match findings in a report by Kaspersky [58].

Type	Hij.	Targeted Domain Information			Cross Ref		Attacker Infra. (Transient)			Legitimate Infra. (Stable)	
		CC	Domain	Sub.	pDNS	crt	IP	ASN	CC	ASNs	CCs
P-IP	Dec'18	CY	owa.gov.cy	.	✔	✔	178.62.218.244	14061	NL	[50233]	[CY]
P-IP	Jan'19	CY	cyta.com.cy	mbox	✔	✔	178.62.218.244	14061	NL	—	—
P-IP	Nov'18	IQ	inc-vrdl.iq	.	✔	✔	199.247.3.191	20473	DE	[50710]	[IQ]
P-NS	Dec'18	JO	gid.gov.jo	.	✔	✔	139.162.144.139	63949	DE	—	—
P-NS	Dec'20	KG	fiu.gov.kg	mail	✔	✔	178.20.41.140	48282	RU	—	—
P-NS	Jan'21	KG	infocom.kg	mail	✔	✔	195.2.84.10	48282	RU	—	—
P-IP	Dec'18	KW	dgca.gov.kw	mail	✔	✔	185.15.247.140	24961	DE	—	—
P-IP	Nov'18	LB	finance.gov.lb	webmail	✔	✔	185.20.187.8	50673	NL	—	—
P-IP	Nov'18	LB	mea.com.lb	memail	✔	✔	185.20.187.8	50673	NL	—	—
P-IP	Oct'18	LY	embassy.ly	.	✔	✖	188.166.119.57	14061	NL	—	—
P-NS	Oct'18	LY	foreign.ly	.	✔	✔	188.166.119.57	14061	NL	—	—
P-NS	Jan'19	SE	netnod.se	dnsnodeapi	✔	✔	139.59.134.216	14061	DE	—	—
P-NS	Dec'18	US	pch.net	keriomail	✔	✔	159.89.101.204	14061	DE	—	—

Table 5.4. Description of 41 domains identified as hijacked including the broad sector level categorization.

CC	Domain	Description	Sector
AE	adpolice.gov.ae	Abu Dhabi Police, UAE	Law Enforcement
AE	apc.gov.ae	Police College Website, UAE	Law Enforcement
AE	mgov.ae	Telecomm. Regulatory Authority, UAE	Govt. Organization
AE	mofa.gov.ae	Ministry of Foreign Affairs, UAE	Govt. Ministry
AL	asp.gov.al	Albanian State Police, Albania	Law Enforcement
AL	e-albania.al	E-Govt. Portal, Albania	Govt. Internet Services
AL	shish.gov.al	State Intelligence Service, Albania	Intelligence Services
CY	cyta.com.cy	Telecommunications Provider, Cyprus	Infrastructure Provider
CY	defa.com.cy	Natural Gas Public Company, Cyprus	Energy Company
CY	govcloud.gov.cy	Govt. Internet Services, Cyprus	Govt. Internet Services
CY	owa.gov.cy	Govt. Internet Services, Cyprus	Govt. Internet Services
CY	sslvpn.gov.cy	Govt. Internet Services, Cyprus	Govt. Internet Services
CY	webmail.gov.cy	Govt. Internet Services, Cyprus	Govt. Internet Services
EG	mfa.gov.eg	Ministry of Foreign Affairs, Egypt	Govt. Ministry
EG	mod.gov.eg	Ministry of Defense, Egypt	Govt. Ministry
EG	nmi.gov.eg	National Institute for Governance, Egypt	Govt. Organization
EG	petroleum.gov.eg	Petroleum and Mineral Ministry, Egypt	Govt. Ministry
GR	kyvernisi.gr	Govt. Internet Services, Greece	Govt. Internet Services
GR	mfa.gr	Ministry of Foreign Affairs, Greece	Govt. Ministry
IQ	mofa.gov.iq	Ministry of Foreign Affairs, Iraq	Govt. Ministry
IQ	inc-vrdl.iq	E-Govt. Portal, Iraq	Govt. Internet Services
JO	gid.gov.jo	General Intelligence Directorate, Jordan	Intelligence Services
JO	psd.gov.jo	Public Security Directorate, Jordan	Intelligence Services
KG	fiu.gov.kg	Financial Intelligence Service, Kyrgyzstan	Govt. Ministry
KG	invest.gov.kg	Investment Portal, Kyrgyzstan	Govt. Ministry
KG	mfa.gov.kg	Ministry of Foreign Affairs, Kyrgyzstan	Govt. Ministry
KG	infocom.kg	Internet Services	Infrastructure Provider
KW	csb.gov.kw	Central Statistical Bureau, Kuwait	Govt. Ministry
KW	dgca.gov.kw	Civil Aviation, Kuwait	Civil Aviation
KW	kotc.com.kw	Kuwait Oil Tanker Company	Energy Company
KW	moh.gov.kw	Ministry of Health, Kuwait	Govt. Ministry
LB	finance.gov.lb	Ministry of Finance, Lebanon	Govt. Ministry
LB	mea.com.lb	Middle East Airlines, Lebanon	Civil Aviation
LB	medgulf.com.lb	Insurance Company, Lebanon	Insurance
LY	embassy.ly	Libyan Embassies	Govt. Organization
LY	foreign.gov.ly	Ministry of Foreign Affairs, Libya	Govt. Ministry
LY	noc.ly	National Oil Corporation, Libya	Energy Company
NL	ocom.com	Internet Services	Infrastructure Provider
SE	netnod.se	Internet Services	Infrastructure Provider
SY	syriatel.sy	Telecommunications Provider, Syria	Infrastructure Provider
US	pch.net	Internet Services	Infrastructure Provider

as P-IP), and another 7 domains pivoting on the attacker-controlled authoritative nameservers (indicated as P-NS). For all but one, we also find corroborating evidence in the CT logs in the form of suspicious newly-issued certificates for the sensitive subdomains in the same time frame.

These domains are not directly flagged using deployment maps for a couple of reasons. First, there might not be any observable infrastructure for the domain revealed using the IP address scans (*e.g.*, `dgca.gov.kw`), or a domain might not use a TLS certificate (*e.g.*, `embassy.ly`). Second, some deployment maps have many deployments, making it challenging to conclude which correspond to a suspicious transient deployment; both `netnod.se` and `owa.gov.cy`, for instance, have multiple transient deployments.

The final two domains, `apc.gov.ae` and `moh.gov.kw`, do not have corroborating pDNS evidence (indicated as T1*). However, we identify them as hijacked because we see a transient deployment with a suspicious newly-issued certificate securing a sensitive subdomain *and* the exact IP address associated with the transient deployment was also used to hijack other domains.

Validation. As discussed in Section 5.3, a key challenge with a third-party approach to identifying these attacks is the lack of ground truth. The goal is to compromise an organization, and organizations are typically reluctant to publicize such events when they happen.

However, there are multiple reasons to believe that the attacks we have identified are authentic. First, our methodology is not a machine learning approach subject to overfitting — there is no training or training data. Our methodology is to identify the critical operational requirements for this class of attacks. Thus, while we may miss attacks for lack of data, all real attacks that appear in our dataset should be identified by our approach. Second, in spite of our constructive approach, the sites our methodology identified are almost exclusively *government agencies* — precisely the sites that we would expect to be targeted in sophisticated attacks.

Finally, in addition to these circumstantial observations, a number of the sites we identified have been independently confirmed as targets of DNS hijacking attacks — either directly (*i.e.*, the site itself is named) or indirectly (*i.e.*, the attacker infrastructure IPs are named). Thus, for every hijacked domain and attacker IP address, we searched online for articles that include

either feature.

Notably, the attacker infrastructure targeting 33 domains matched the IP addresses implicated in the Sea Turtle hijacks [4, 29]. Additional articles confirm 28 of these 33 domains as being targets of DNS hijacks [56, 80]. Further, a recent report by Kaspersky partially matches our findings about the four `.kg` domains [58].

Only four domains (highlighted in gray) do not have independent confirmations. We continue to believe that DNS hijacking is the most likely explanation, even if these attacks have not been previously discovered or publicly disclosed.

Longitudinal Patterns. The hijacks span the entire four years of our data set, with a significant uptick in 2018 corresponding with the Sea Turtle hijacks. Perhaps more significant, we find recurring hijacks of domains under the same TLD spanning months and in some cases years. This pattern suggests that the attackers were sophisticated enough to evade detection for long periods. Moreover, we identify domain hijacks as late as January 2021, long after the Sea Turtle hijacks were publicized (early 2019), indicating that these types of attacks remain an ongoing problem.

5.5.3 Observability

Attackers are very careful in limiting the durations of domain hijacks to minimize observability via DNS. When attackers change the domain resolutions to the malicious infrastructure, they generally do so for less than a day at a time. For 51% of the domains hijacked, pDNS captures evidence of the attack itself — domain resolutions to the malicious infrastructure — for at most one day. This approach prevents the hijack from appearing in the daily zone file snapshots, even if the domain is hijacked more than once. Of the three domains whose TLDs we have zone file access to, the hijack is not visible in the zone file for two of the domains (`ocom.com` and `netnod.se`). For the third, `pch.net`, the hijack is visible in the zone for one day — yet resolutions to the malicious infrastructure appear in pDNS spanning a 20-day period.

Attacker infrastructure is observable for longer periods. Attackers create the malicious

certificate and deploy it relatively quickly. More than 50% of the malicious certificates for the hijacked domains are visible in the certificate scans within 8 days of the certificate being issued and appearing in the certificate transparency logs. Once deployed, though, the malicious certificate is often observable in only a small number of weekly CUIDS scans. For more than 50% of the domains, the malicious certificate only appeared in one scan, and another 20% of the certificates appeared in just two.

5.5.4 Targeted Domains

Recall that the second pattern of transient deployments (T2) capture the prelude to the hijack. The deployment maps reveal a transient deployment in an unrelated AS located in a different country relative to a long-term stable deployment. However, the transient deployment returns the same certificate as the stable deployment, and there is no evidence that the authoritative nameservers are changed to use the transient deployment. Our interpretation of these deployment maps is that they either correspond to attacks that never took place, our data sets failed to capture the relevant events, or that, while highly anomalous, they reflect legitimate activity that we cannot discern from a third-party perspective.

Table 5.5 lists the 24 domains we identify as targeted. Similar to the hijacked domains, we group the domains by their country and order them by the time of hijack. Many of these domains show attacker infrastructure being reused. For instance, attacker infrastructure targeting four domains across two ccTLDs (`.ae`, `.sa`) uses the same IP address (194.152.42.16). The same IP address (103.213.244.205) is also used to target two `.vn` domains with two distinct stable deployments. Moreover, attackers use AS 45102 (Alibaba) to target domains across eight TLDs (`.ch`, `.kz`, `.lt`, `.lv` `.ma`, `.mm`, `.gov`, `.vn`) between June 2020 and November 2020.

We also note that 21 of the 24 domains were targeted in 2020, after the Sea Turtle disclosures, so perhaps this activity reflects a completely different set of attackers. Unfortunately, we do not have a way of making that determination, nor do we find any online reports mentioning either these domains or the IP addresses of the transient deployments.

Table 5.5. List of 24 Domains identified as targeted for hijacking between January 2017 and March 2021. The deployment maps for all these domain match Pattern T2 which is typically a prelude to the actual attack. These domains represent truly anomalous occurrences — an IP from another ASN from another country returned a certificate for the domain. Similar to the hijacked domains, we report on both the inferred victim and attacker infrastructure.

Tar. Date	CC	Targeted Domain Domain	Sub	Cross Ref. pDNS	Cross Ref. crt	Attacker Infra. (Transient) IP	Attacker Infra. (Transient) ASN	Attacker Infra. (Transient) CC	Legit. Infra. (Stable) ASNs	Legit. Infra. (Stable) CCs
Apr'20	AE	milmail.ae	—	✗	✗	194.152.42.16	47220	RO	[5384]	[AE]
Apr'20	AE	mocaf.gov.ae	—	✗	✗	194.152.42.16	47220	RO	[5384]	[AE]
Apr'20	AE	moi.gov.ae	—	✗	✗	194.152.42.16	47220	RO	[5384]	[AE]
Dec'20	AE	epg.gov.ae	—	✗	✗	159.69.193.152	24940	DE	[202024]	[AE]
Jun'20	CH	parlament.ch	—	✗	✗	8.210.146.182	45102	SG	[61098,3303]	[CH]
Nov'20	GH	nita.gov.gh	—	✗	✗	78.141.218.158	20473	NL	[37313]	[GH]
Sep'17	JO	psd.gov.jo	mail	✗	✗	185.162.235.106	50673	NL	[8934]	[JO]
Jun'20	KZ	zerde.gov.kz	—	✗	✗	8.210.190.81	45102	SG	[48716,15549]	[KZ]
Nov'20	LT	stat.gov.lt	—	✗	✗	8.210.190.214	45102	SG	[6769]	[LT]
Jul'20	LV	iem.gov.lv	—	✗	✗	8.210.199.85	45102	SG	[8194, 25241]	[LV]
Nov'20	LV	zva.gov.lv	—	✗	✗	8.210.36.66	45102	SG	[8194, 199300]	[LV]
Apr'18	MA	justice.gov.ma	micj	✓	✗	188.166.160.110	14061	DE	[6713]	[MA]
Apr'20	MA	mem.gov.ma	—	✗	✗	47.75.34.153	45102	HK	[6713]	[MA]
Oct'20	MM	mofa.gov.mm	—	✗	✗	47.242.150.18	45102	US	[136465]	[MM]
Nov'20	PL	knf.gov.pl	—	✗	✗	103.195.6.231	64022	HK	[34986]	[PL]
May'20	SA	cmail.sa	—	✗	✗	194.152.42.16	47220	RO	[49474]	[SA]
Sep'20	TM	turkmenpost.gov.tm	—	✗	✗	185.229.225.228	41436	NL	[20661]	[TM]
Aug'20	US	manchesternh.gov	—	✗	✗	8.210.210.235	45102	SG	[13977]	[US]
Dec'20	US	batesvillearkansas.gov	host	✗	✗	95.179.153.176	20473	NL	[32244]	[US]
Apr'19	VN	ais.gov.vn	intranet	✓	✗	45.77.45.193	20473	SG	[131375,63748]	[VN]
Dec'20	VN	mofa.gov.vn	—	✗	✗	45.77.27.9	20473	JP	[24035]	[VN]
Mar'20	VN	cpt.gov.vn	—	✗	✗	103.213.244.205	136574	JP	[63747]	[VN]
Mar'20	VN	most.gov.vn	—	✗	✗	103.213.244.205	136574	JP	[38731,131373]	[VN]
Sep'20	VN	vass.gov.vn	—	✗	✗	47.74.3.121	45102	JP	[18403]	[VN]

Table 5.6. Description of 24 domains identified as targeted including the broad sector level categorization.

CC	Domain	Description	Sector
AE	epg.gov.ae	Emirates Post, UAE	Postal Service
AE	milmail.ae	Armed Forces Mail, UAE	Law Enforcement
AE	mocaf.gov.ae	Ministry of Cabinet Affairs, UAE	Govt. Ministry
AE	moi.gov.ae	Ministry of Interior, UAE	Govt. Ministry
CH	parlament.ch	Parliament, Switzerland	Govt. Organization
GH	nita.gov.gh	National Info. Tech. Agency, Ghana	Govt. Organization
KZ	zerde.gov.kz	National Info. Holdings, Kazakhstan	Govt. Organization
LB	pcm.gov.lb	Council of Ministers, Lebanon	Govt. Ministry
LT	stat.gov.lt	Statistics Lithuania	Govt. Ministry
LV	iem.gov.lv	Ministry of the Interior, Latvia	Govt. Ministry
LV	zva.gov.lv	State Agency of Medicines, Latvia	Govt. Organization
MA	justice.gov.ma	Ministry of Justice, Morocco	Govt. Ministry
MA	mem.gov.ma	Ministry of Sustainable Dev., Morocco	Govt. Ministry
MM	mofa.gov.mm	Ministry of Foreign Affairs, Myanmar	Govt. Ministry
PL	knf.gov.pl	Polish Financial Supervision Authority	Govt. Ministry
SA	cmail.sa	Al-Elm Information Security	IT Firm
TM	turkmenpost.gov.tm	Turkmen Post	Postal Service
US	batesvillearkansas.gov	City of Batesville, AR	Local Govt.
US	manchesternh.gov	City of Manchester, NH	Local Govt.
VN	ais.gov.vn	Authority of Info. Security, Vietnam	Govt. Organization
VN	cpt.gov.vn	Central Post Office, Vietnam	Postal Service
VN	mofa.gov.vn	Ministry of Foreign Affairs, Vietnam	Govt. Ministry
VN	most.gov.vn	Ministry of Science and Tech., Vietnam	Govt. Ministry
VN	vass.gov.vn	Vietnam Academy of Social Sciences	Govt. Organization

Table 5.7. Affected Organizations break down by sector.

Organization Sector	# of Domains Hij.	Tar.	Total
Government Ministry	12	11	23
Government Organization	4	6	10
Government Internet Services	7	0	7
Infrastructure Provider	6	0	6
Law Enforcement	3	1	4
Energy Company	3	0	3
Intelligence Services	3	0	3
Postal Service	0	3	3
Civil Aviation	2	0	2
Local Government	0	2	2
Insurance	1	0	1
IT Firm	0	1	1
Total	41	24	65

5.5.5 Affected Organizations

We manually identified the organizations associated with the domains identified as hijacked or targeted (Table 5.4 and Table 5.6). These organizations span government ministries, government organizations, infrastructure providers, and even some private firms. Table 5.7 breaks down the affected organizations by sector. For every sector, we list the number of domains identified as hijacked or targeted. Domains associated with government ministries top this list, suggesting state-affiliated motivations behind the attackers and the style of their attacks. Domains related to government Internet services (mail, cloud, VPN services) reflect their value for credential theft.

5.5.6 Attacker Infrastructure

As a final analysis we examine features of the attacker infrastructure used to hijack or target domains.

Network. Table 5.8 lists the networks used by attackers and the number of domains targeted. It shows a concentration in the use of Digital Ocean, Vultr, and Serverius. While we

Table 5.8. Networks used by Attackers. Number of domains hijacked or targeted from each network.

Network Information		# of Domains		
ASN	**AS Name**	**Hij.**	**Tar.**	**Total**
14061	Digital Ocean	15	1	16
20473	Vultr	7	4	11
45102	Alibaba	0	9	9
50673	Serverius	7	1	8
48282	VDSINA	4	0	4
47220	ANTENA3	0	4	4
9009	M247	2	0	2
24961	MYLOC	2	0	2
63949	Linode	2	0	2
136574	Zheye Network	0	2	2
20860	IOMart	1	0	1
54825	Packet Host	1	0	1
24940	Hetzner	0	1	1
41436	CloudWebManage	0	1	1
64022	Kamatera	0	1	1
	Total	41	24	65

see concentrations in networks used, we do not believe they are reliable features for detection since attacker infrastructure is largely portable. As one example, for the domain `owa.gov.cy` the attacker targeted the domain from four separate ASNs: 14061 , 20473, 33387, 44901. We also see a difference in the ASes used between hijacked and targeted domains, which likely simply reflects different attackers being observed.

Certificates. Another important aspect of the attacker infrastructure are the certificates. Of the hijacked domains, we identified a suspicious certificate for 40 domains. Table 5.9 lists these certificates along with the CAs which issued them. These certificates were issued from two CAs: 28 from Let's Encrypt, and 12 from Comodo.[11] Let's Encrypt offers free, automated certificate issuance via its ACME protocol [13], and Comodo (now Sectigo) offers free trial certificates [12]. Both use Domain Validated (DV) certificates which only requires control over DNS infrastructure [32]. Given that other CAs (*e.g.*, ZeroSSL) now also offer automated

[11] Comodo has been since rebranded to Sectigo. Two domains issued by Sectigo are counted as issued by Comodo.

Table 5.9. List of suspiciously obtained certificates for 40 hijacked domains. (`embassy.ly` did not use TLS certificates.) Let's Encrypt is the Issuer CA for 28 while Comodo is the Issuer CA for 12. Both of the CAs provided certificates for free. Only 4 certificates were revoked. Let's Encrypt does not provide a CRL for the leaf certificates and instead relies on Online Certificate Status Protocol (OCSP). As a result, we cannot determine retroactively if any of the 28 certificates issued by Let's Encrypt were revoked.

CC	Domain	Target	crt.sh ID	Issuer CA	CRL
AE	adpolice.gov.ae	advpn	835334320	Let's Encrypt	—
AE	apc.gov.ae	mail	820893483	Let's Encrypt	—
AE	mgov.ae	mail*	804429558	Let's Encrypt	—
AE	mofa.gov.ae	webmail	495595690	Comodo	✖
AL	asp.gov.al	mail*	929142682	Comodo	✔
AL	e-albania.al	owa	296537802	Let's Encrypt	—
AL	shish.gov.al	mail	912593168	Let's Encrypt	—
CY	cyta.com.cy	mbox	1150009364	Comodo*	✔
CY	defa.com.cy	mail	1225501249	Comodo*	✖
CY	govcloud.gov.cy	personal*	1021403642	Comodo	✖
CY	owa.gov.cy	.	1056463948	Comodo	✖
CY	sslvpn.gov.cy	.	1088915811	Comodo	✖
CY	webmail.gov.cy	.	1039430428	Comodo	✖
EG	mfa.gov.eg	mail	946136592	Let's Encrypt	—
EG	mod.gov.eg	mail*	970178538	Let's Encrypt	—
EG	nmi.gov.eg	mail*	961982738	Comodo	✖
EG	petroleum.gov.eg	mail	962230186	Let's Encrypt	—
GR	kyvernisi.gr	mail	1394170951	Let's Encrypt	—
GR	mfa.gr	pop3	1382284606	Let's Encrypt	—
IQ	mofa.gov.iq	mail	775703946	Let's Encrypt	—
IQ	inc-vrdl.iq	.	961752433	Let's Encrypt	—
JO	gid.gov.jo	.	1024142638	Let's Encrypt	—
KG	fiu.gov.kg	mail	3848797679	Let's Encrypt	—
KG	invest.gov.kg	mail	3842234495	Let's Encrypt	—
KG	mfa.gov.kg	mail	3810274168	Let's Encrypt	—
KG	infocom.kg	mail	3913246526	Let's Encrypt	—
KW	csb.gov.kw	mail	2288836441	Let's Encrypt	—
KW	dgca.gov.kw	mail	291715835	Let's Encrypt	—
KW	kotc.com.kw	mail2010*	1485763752	Let's Encrypt	—
KW	moh.gov.kw	webmail	1394227599	Let's Encrypt	—
LB	finance.gov.lb	webmail	922787324	Let's Encrypt	—
LB	mea.com.lb	memail	923463031	Let's Encrypt	—
LB	medgulf.com.lb	mail	983855608	Let's Encrypt	—
LB	pcm.gov.lb	mail1	983220130	Let's Encrypt	—
LY	embassy.ly	.	—	—	—
LY	foreign.gov.ly	.	893184607	Let's Encrypt	—
LY	noc.ly	mail	885156392	Let's Encrypt	—
NL	ocom.com	connect	314340862	Comodo	✖
SE	netnod.se	dnsnodeapi	1071765455	Comodo	✔
SY	syriatel.sy	mail	1349974775	Let's Encrypt	—
US	pch.net	keriomail	1075482666	Comodo	✔

certificates, it would not be surprising to see other CAs being used by attackers going forward.

Significantly, only four of these certificates were revoked based on the Certificate Revocation List (CRL) indexed by `crt.sh` as provided by the issuer CA.[12] This lack of revocation suggests that, in most of the cases, the victim is unaware of the hijack until after the certificate expiry, if at all.

As an interesting data point, we found that the legitimate infrastructure of some domains used certificates which were not browser-trusted, indicating use of an internal trusted CA by the domain owner. This use of an internal trusted CA means that the CT logs only contain the suspicious certificates associated with a transient deployment.

5.5.7 Disclosure and Ethical Considerations

In this chapter, we identified a number of historical hijacks that were previously unidentified. While the direct harms surrounding these events have long since past, we did not know if the victims were aware of these incidents (and hence able to make appropriate decisions concerning their security moving forwards; *e.g.*, such as resetting passwords, etc.) Thus, we believe that victim notification was our primarily ethical obligation. To this end, we reached out to the previously unidentified 8 hijacked domains and 24 targeted domains, directly and via their national CERTs and reported all domains and inferred attacker infrastructure to allow for full auditing. Over two months have passed since our notifications so we now believe all affected parties are well aware of these potential issues. While we recognize that our publication of this data also creates the potential for secondary reputational harms (governments, in particular, do not like to be seen as victims – perhaps explaining the lack of substantive responses to our notifications) we believe those interests are secondary to the value of full transparency for the broader research and security communities. Indeed, just as we have benefited from detailed third-party reporting to help evaluate our own research, we believe our data can and will provide purchase for other

[12] Let's Encrypt does not provide a CRL for the leaf certificates and instead relies on Online Certificate Status Protocol (OCSP) [27]. As a result, we cannot determine if the certificates issued by Let's Encrypt were revoked.

researchers to further investigate strategies used by attackers (*e.g.*, Section 5.5.1) and to provide further examples for reasoning about detection and prevention.

5.6 Discussion

The key result of this chapter is a methodology for retroactively identifying evidence of targeted DNS infrastructure hijacking. We identify a range of potential victims (predominantly government agencies) in over twenty countries — including many that have been independently confirmed via forensic reporting but also a variety that have never been reported publicly. However, perhaps the most remarkable result from this study is that it was possible at all. Here we reflect on the challenges associated with identifying such attacks and the work required to improve our visibility into such threats.

Transparency at Short Time Scales. Unlike other ecosystems such as routing (BGP), the DNS ecosystem (especially ccTLDs) is comparatively opaque at short time scales. Because the goal of these attacks was to gather credentials, they only needed to be active for extremely short periods of time — once to acquire a certificate and again, to harvest credentials. Indeed, with vanishingly few exceptions, these attacks are entirely invisible in DNS zone files because their daily granularity is orders of magnitude too coarse to capture the attack. Similarly, both active and passive DNS measurements will only record such attacks if they are lucky enough to measure the DNS state at *precisely* the time that a hijack is taking place. Brief anomalies — both benign and malicious — are largely invisible to existing measurement infrastructure. This is well-understood by sophisticated adversaries, and addressing this lack of transparency in DNS is an important challenge for our community — whether through online change detection, reactive measurement, or systems (such as CT logs) that log all potential state-impacting changes.

Implicit Trust Dependence. Another issue highlighted by our study is the ongoing challenges we face with implicit trust dependence. While TLS is designed to protect us from actors mounting adversary-in-the-middle attacks, its security depends on the due diligence of

trusted CAs. Yet the economic efficiencies of CAs using domain validation has produced an environment where a DNS infrastructure hijack is sufficient to subvert this due diligence and thus bypass TLS — an authentication ouroboros. Similarly, protocols like DNS and DNSSEC implicitly place trust in registrars and registries. But if one is compromised, the guarantees made by these protocols are easily bypassed. This is not unique to the situation described in this work, and there are a plethora of well-documented attacks around trust dependency issues ranging from BGP and DNS [16, 81] to package managers [68]. This remains an open area of research, but virtually all solutions take some page from the "trust but verify" book. In much the same way that Let's Encrypt now guards against BGP hijack [16], we will need to develop similar capabilities against DNS hijack and improved versions and deployment of Registrar and Registry Lock features [41, 57, 90].

Cleartext Credentials. Given the targeting of credentials, we may want to rethink the practice of sending cleartext user credentials. While violations of TLS' integrity and confidentiality guarantees are problematic, that a single violation should provide long-term arbitrary access inside a target organization is an asymmetric threat. The independent efforts to transition to password-less authentication, via WebAuthn, CTAP and its successors, may provide an opportunity to eliminate password theft as a potential attack vector.

Collective Action Problems. Complicating all of these issues is that they require systematic attention across disparate entities — registrars, registries, CAs, DNS service providers, ISPs, and software developers. No single party is in an ideal position to address this problem effectively, and moreover the overheads for needed improvements will frequently accrue to those other than the beneficiaries. Thus, addressing this problem effectively will, in addition to further research and development, require significant investment in leadership and coordination.

Chapter 5, in part, has been submitted for publication of the material as it may appear in *Proceedings of the International Measurement Conference 2022*. Gautam Akiwate, Raffaele Sommese, Mattijs Jonker, Zakir Durumeric, KC Claffy, Geoffrey M. Voelker, and Stefan Savage. The dissertation author was the primary investigator and author of this paper.

Chapter 6

Conclusion

This dissertation explores two classes of DNS infrastructure hijacks. The first class of hijacks — *opportunistic DNS infrastructure hijacks* — is the result of errors and inconsistencies introduced in nameserver delegations due to actions of the registrant or the registrar (Chapter 3 and Chapter 4). The second class of hijacks — *targeted DNS infrastructure hijacks* — is the result of an attacker targeting and subverting otherwise correctly configured delegations to attacker controlled delegations (Chapter 5).

While the DNS primarily relies on complex technical and contractual interactions between different organizations to successfully operate, it also relies on human interaction to configure the delegations correctly. As such, the avenues for errors (*i.e.*, lame delegations) that affect the security and the efficiency of the resolution process is high. In Chapter 3, we presented a methodology to identify lame delegations at scale for the global Internet. Using this methodology, we found nearly 14% of domains analyzed were lame delegated highlighting the widespread inconsistencies in nameserver delegations. Our work has inspired at least one large organization to check their domains regularly for lame delegations.

In Chapter 4, we showed that inconsistencies are not always a result of faulty human interaction (*e.g.*, typos when entering nameserver delegations), but also systematic choices made by registrars on technical interaction (*e.g.*, renaming on domain name expiration). Building on our work to identify lame delegations, we presented a methodology to uncover operational

practices responsible for exposing domains. In all, we identified half a million domains made vulnerable in a nine-year period. As a result of our work, at least three registrars updated their operational practices to prevent this exposure while many more have remediated previously exposed domains.

Finally in Chapter 5, we presented a methodology to identify targeted DNS infrastructure hijacks over a four-year period. This methodology collated a wide array of data sources including Internet-wide scans, routing information, Certificate Transparency Logs, and Passive DNS to infer hijacked domains by identifying attacker infrastructure used to target the hijacked domain. Using this methodology we identified 41 hijacked domains, 8 of which were not previously identified. As such, this methodology presents an avenue for third-party organizations to infer occurrences of this class of hijacks without requiring disclosure from hijacked domains.

6.1 Future Directions

The work in this dissertation indicates a few different directions for future research. While our work in Chapter 4 largely addresses the problem, it does not *permanently* fix it. A potential permanent solution is to modify the underlying protocols so as to eliminate the constraints that required the operational workaround in the first place. While permanent solutions are deliberated over by the community, another direction to consider is to equip domain owners with tools to monitor their own domains. A "trust tree" tool that exhaustively maps the delegations implicitly and explicitly trusted by the resolution process for a domain can enable domain owners to not only monitor their domain to avoid erroneous delegations, but also determine their exposure in terms of the domains trusted.

For targeted hijacks, a future direction is to identify potential interventions that can impede the attacker workflow in targeted hijacks. However, any such intervention hinges on the ability to detect hijacks near real-time. To that end, a concrete next step is to trigger reactive DNS measurements on certificate issuance. The reactive DNS measurement data can then be

cross referenced with historical deployment maps to flag any suspicious certificate issuance. Using follow-on reactive measurements, we can then infer a hijack by checking if changes to nameserver delegations were transient.

6.2 Final Thoughts

It is important to note that the work in this dissertation is largely possible due to publicly available data sets. While projects like ICANN CZDS and OpenINTEL have been instrumental in illuminating the DNS ecosystem, the granularity of measurement mostly misses short-time scale anomalies. Notably, short-time scale anomalies — both benign and malicious — are largely invisible to existing measurement infrastructure. This gap in visibility is well-understood by sophisticated adversaries who time their attacks accordingly. For instance, zone files and active DNS measurements largely miss the hijacks identified in Chapter 5. As such, this dissertation motivates the need to address the lack of transparency in DNS by adopting a system, such as certificate transparency, that logs all potential state-impacting changes, to effectively identify security threats, vulnerabilities, and harmful behavior.

Bibliography

[1] Donald E. Eastlake 3rd and Aliza R. Panitz. Reserved Top Level DNS Names. RFC 2606, June 1999. https://rfc-editor.org/rfc/rfc2606.txt.

[2] Josh Aas, Richard Barnes, Benton Case, Zakir Durumeric, Peter Eckersley, Alan Flores-López, J. Alex Halderman, Jacob Hoffman-Andrews, James Kasten, Eric Rescorla, Seth Schoen, and Brad Warren. Let's Encrypt: An Automated Certificate Authority to Encrypt the Entire Web. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, pages 2473–2487, New York, NY, USA, 2019. Association for Computing Machinery.

[3] Joe Abley, Brian Dickson, Warren "Ace" Kumari, and George G. Michaelson. AS112 Redirection Using DNAME. RFC 7535, May 2015. https://rfc-editor.org/rfc/rfc7535.txt.

[4] Danny Adamitis, David Maynor, Warren Mercer, Matthew Olney, and Paul Rascagneres. DNS Hijacking Abuses Trust In Core Internet Service, April 2019. https://blog.talosintelligence.com/2019/04/seaturtle.html.

[5] AFRINIC. AFRINIC ratifies "Lame Delegations in the AFRINIC reverse DNS" Policy, 2019. https://afrinic.net/lame-delegations-in-afrinic-reverse-dns-policy-ratified.

[6] AFRINIC. Lame Delegations Statistics, 2020. https://stats.afrinic.net/lamerdns/.

[7] Alexa. Top 1M Sites, May 2020. https://www.alexa.com/topsites.

[8] Eihal Alowaisheq, Siyuan Tang, Zhihao Wang, Fatemah Alharbi, Xiaojing Liao, and XiaoFeng Wang. Zombie Awakening: Stealthy Hijacking of Active Domains through DNS Hosting Referral. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, CCS '20, pages 1307–1322, New York, NY, USA, 2020. Association for Computing Machinery.

[9] Eihal Alowaisheq, Peng Wang, Sumayah Alrwais, Xiaojing Liao, XiaoFeng Wang, Tasneem Alowaisheq, Xianghang Mi, Siyuan Tang, and Baojun Liu. Cracking the Wall of Confinement: Understanding and Analyzing Malicious Domain Take-downs. In *Proceedings of The Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, USA, 2019. Internet Society.

[10] APNIC. Lame DNS Reverse Delegation, 2020. https://www.apnic.net/manage-ip/manage-resources/reverse-dns/lame-dns-reverse-delegation.

[11] ARIN. Recommended Draft Policy ARIN-2014-5: Remove 7.2 Lame Delegations, 2014. https://www.arin.net/vault/policy/proposals/2014_5.html.

[12] Comodo Certification Authority. Comodo SSL Single DV Certificate, January 2022. https://ssl.comodo.com/comodo-ssl-dv-trial.

[13] Richard Barnes, Jacob Hoffman-Andrews, Daniel McCarney, and James Kasten. Automatic Certificate Management Environment (ACME). RFC 8555, March 2019. https://www.rfc-editor.org/info/rfc8555.

[14] David Barr. Common DNS Operational and Configuration Errors. RFC 1912, February 1996. https://rfc-editor.org/rfc/rfc1912.txt.

[15] Tara Siegel Bernard, Tiffany Hsu, Nicole Perlroth, and Ron Lieber. Equifax Says Cyberattack May Have Affected 143 Million in the U.S., September 2017. https://www.nytimes.com/2017/09/07/business/equifax-cyberattack.html.

[16] Henry Birge-Lee, Yixin Sun, Anne Edmundson, Jennifer Rexford, and Prateek Mittal. Bamboozling Certificate Authorities with BGP. In *27th USENIX Security Symposium*, USENIX Security 18, pages 833–849, Baltimore, MD, August 2018. USENIX Association.

[17] Benjamin Braun. Investigating DNS Hijacking Through High Frequency Measurements. Technical report, UC San Diego, 2016. https://escholarship.org/uc/item/8tm5c7r7.

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

[19] CAIDA. Routeviews prefix to as mappings dataset for ipv4 and ipv6, 2020. http://www.caida.org/data/routing/routeviews-prefix2as.xml.

[20] CAIDA. Inferred AS to Organization Mapping Dataset, 2021. https://www.caida.org/data/as_organizations.xml.

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

[22] Censys. Censys Bulk Data Access, March 2021. https://censys.io/data.

[23] Alberto Cerpa and Jeremy Elson. Internet Content Adaptation Protocol (ICAP). RFC 3507, April 2003. https://rfc-editor.org/rfc/rfc3507.txt.

[24] Stuart Cheshire and Marc Krochmal. Special-Use Domain Names. RFC 6761, February 2013. https://rfc-editor.org/rfc/rfc6761.txt.

[25] Cloudflare. BGP leaks and cryptocurrencies, April 2018. https://blog.cloudflare.com/bgp-leaks-and-crypto-currencies/.

[26] DNS Coffee. DNS Coffee, 2020. https://dns.coffee.

[27] Let's Encrypt Community. Why no CRL URL in the certificate?, January 2017. https://community.letsencrypt.org/t/why-no-crl-url-in-the-certificate/25686.

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

[29] Matt Dahl. Widespread DNS Hijacking Activity Targets Multiple Sectors, January 2019. https://www.crowdstrike.com/blog/widespread-dns-hijacking-activity-targets-multiple-sectors/.

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

[31] Digicert. Domain Control Validation (DCV) Methods, November 2021.

[32] Digicert. What's The Difference Between DV, OV & EV SSL Certificates?, January 2022. https://www.digicert.com/difference-between-dv-ov-and-ev-ssl-certificates.

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

[34] DomainTools. Iris Investigation Platform - Passive DNS, January 2022. https://www.domaintools.com/products/iris.

[35] Zakir Durumeric. Censys Search 2.0, April 2021. https://support.censys.io/hc/en-us/articles/360060941211-Censys-Search-2-0-Official-Announcement.

[36] Digital Element. NetAcuity IP Geolocation Data, January 2021. https://www.digitalelement.com/geolocation/.

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

[38] Let's Encrypt. Challenge Types - DNS-01 Challenge, November 2021. https://letsencrypt.org/docs/challenge-types/.

[39] Entrust. What is a SAN (Subject Alternative Name) and how is it used?, March 2019. https://www.entrust.com/blog/2019/03/what-is-a-san-and-how-is-it-used/.

[40] ICANN GAC. Principles for Delegation and Administration of ccTLDs Presented by Governmental Advisory Committee, February 2000. https://archive.icann.org/en/committees/gac/gac-cctldprinciples-23feb00.htm.

[41] Gandi. How to Turn On Transfer Lock for a Domain, January 2022. https://docs.gandi.net/en/domain_names/transfer_out/transfer_lock.html.

[42] Google. Broadening HSTS to secure more of the Web, September 2017. https://security.googleblog.com/2017/09/broadening-hsts-to-secure-more-of-web.html.

[43] Google. Next steps toward more connection security, April 2017. https://blog.chromium.org/2017/04/next-steps-toward-more-connection.html.

[44] Google. A safer default for navigation: HTTPS, March 2021. https://blog.chromium.org/2021/03/a-safer-default-for-navigation-https.html.

[45] The Guardian. WikiLeaks hacked as OurMine group answers "hack us" challenge, August 2017. https://www.theguardian.com/technology/2017/aug/31/wikileaks-hacked-ourmine-group-julian-assange-dns-attack.

[46] Muks Hirani, Sarah Jones, and Ben Read. Global DNS Hijacking Campaign: DNS Record Manipulation at Scale, January 2019. https://www.mandiant.com/resources/global-dns-hijacking-campaign-dns-record-manipulation-at-scale.

[47] Scott Hollenbeck. Extensible Provisioning Protocol (EPP). RFC 5730, August 2009. https://rfc-editor.org/rfc/rfc5730.txt.

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

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

[50] Rebekah Houser, Shuai Hao, Zhou Li, Daiping Liu, Chase Cotton, and Haining Wang. A Comprehensive Measurement-based Investigation of DNS Hijacking. In *40th International Symposium on Reliable Distributed Systems (SRDS)*, pages 210–221, Chicago, IL, USA, 2021. IEEE.

[51] ICANN. IANA Report on the Delegation of the .TEL Top-Level Domain, 2007. https://www.iana.org/reports/2007/tel-report-22jan2007.html.

[52] ICANN. Transfer Report for `tel`, 2017. https://www.iana.org/reports/tld-transfer/20170503-tel.

[53] ICANN. Centralized Zone Data Service, 2021. https://czds.icann.org.

[54] ICANN Security and Stability Advisory Committee (SSAC). SSAC Advisory on Private Use TLDs, September 2020. https://www.icann.org/en/system/files/files/sac-113-en.pdf.

[55] Andrew J. Kalafut, Minaxi Gupta, Christopher A. Cole, Lei Chen, and Nathan E. Myers. An Empirical Study of Orphan DNS Servers in the Internet. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, IMC '10, pages 308–314, New York, NY, USA, 2010. Association for Computing Machinery.

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

[57] Brian Krebs. Does your domain have a Registry Lock?, January 2020. https://krebsonsecurity.com/2020/01/does-your-domain-have-a-registry-lock/.

[58] Ivan Kwiatkowski and Pierre Delcher. DarkHalo after SolarWinds: the Tomiris connection, September 2021. https://securelist.com/darkhalo-after-solarwinds-the-tomiris-connection/104311/.

[59] LACNIC. Lame Delegation Policy, 2020. https://www.lacnic.net/686/2/lacnic/6-lame-delegation-policy.

[60] Ben Laurie, Adam Langley, Emilia Kasper, Eran Messeri, and Rob Stradling. Certificate Transparency Version 2.0. RFC 9162, December 2021. https://www.rfc-editor.org/info/rfc9162.

[61] Baojun Liu, Chaoyi Lu, Haixin Duan, Ying Liu, Zhou Li, Shuang Hao, and Min Yang. Who Is Answering My Queries: Understanding and Characterizing Interception of the DNS Resolution Path. In *27th USENIX Security Symposium (USENIX Security 18)*, pages 1113–1128, Baltimore, MD, August 2018. USENIX Association.

[62] Daiping Liu, Shuai Hao, and Haining Wang. 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)*, pages 1414–1425, Vienna, Austria, 2016. ACM.

[63] Keyu Man, Zhiyun Qian, Zhongjie Wang, Xiaofeng Zheng, Youjun Huang, and Haixin Duan. DNS Cache Poisoning Attack Reloaded: Revolutions with Side Channels. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, CCS '20, pages 1337–1350, New York, NY, USA, 2020. Association for Computing Machinery.

[64] Robert McMillan and Dustin Volz. Suspected Russian Hack Extends Far Beyond SolarWinds Software, Investigators Say, January 2021. https://www.wsj.com/articles/suspected-russian-hack-extends-far-beyond-solarwinds-software-investigators-say.

[65] Warren Mercer and Paul Rascagneres. DNSpionage Campaign Targets Middle East, November 2018. https://blog.talosintelligence.com/2018/11/dnspionage-campaign-targets-middle-east.html.

[66] Paul Mockapetris. Domain Names - Concepts and Facilities. RFC 1034, November 1987. https://rfc-editor.org/rfc/rfc1034.txt.

[67] Paul Mockapetris. Domain Names - Implementation and Specification. RFC 1035, November 1987. https://rfc-editor.org/rfc/rfc1035.txt.

[68] Elizabeth Montalbano. Thousands of Malicious npm Packages Threaten Web Apps, February 2022. https://threatpost.com/malicious-npm-packages-web-apps/178137/.

[69] Vasileios Pappas, Zhiguo Xu, Songwu Lu, Daniel Massey, Andreas Terzis, and Lixia Zhang. Impact of Configuration Errors on DNS Robustness. In *Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM, pages 319–330, New York, NY, USA, 2004. ACM.

[70] Amreesh Phokeer, Alain Aina, and David Johnson. DNS Lame Delegations: A Case-Study of Public Reverse DNS Records in the African Region. In *Proceedings of the 8th EAI International Conference on e-Infrastructure and e-Services for Developing Countries — AFRICOMM*, Ouagadougou, Burkina Faso, December 2016. ICANN, European Alliance for Innovation.

[71] Dave Piscitello. Conficker Summary and Review, May 2010. https://www.icann.org/en/system/files/files/conficker-summary-review-07may10-en.pdf.

[72] Audrey Randall, Enze Liu, Ramakrishna Padmanabhan, Gautam Akiwate, Geoffrey M. Voelker, Stefan Savage, and Aaron Schulman. Home is Where the Hijacking is: Understanding DNS Interception by Residential Routers. In *Proceedings of the 21st ACM Internet Measurement Conference*, IMC '21, pages 390–397, New York, NY, USA, 2021. Association for Computing Machinery.

[73] GoDaddy Representative. Personal Communication, June 2020.

[74] Eric Rescorla. HTTP Over TLS. RFC 2818, May 2000. https://rfc-editor.org/rfc/rfc2818.txt.

[75] Eric Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446, August 2018. https://www.rfc-editor.org/info/rfc8446.

[76] Rick Lamb. DNSSEC Deployment Report, 2022. http://rick.eng.br/dnssecstat/.

[77] Artur Romao. Tools for DNS debugging. RFC 1713, November 1994. https://rfc-editor.org/rfc/rfc1713.txt.

[78] Scott Rose, Matt Larson, Dan Massey, Rob Austein, and Roy Arends. DNS Security Introduction and Requirements. RFC 4033, March 2005. https://www.rfc-editor.org/info/rfc4033.

[79] Sectigo. crt.sh - Certificate Search, January 2022. https://crt.sh/.

[80] Andreas Sfakianakis. On Sea Turtle campaign targeting Greek governmental organisations, February 2020. https://www.linkedin.com/pulse/sea-turtle-campaign-targeting-greek-governmental-andreas-sfakianakis/.

[81] Aftab Siddiqui. KlaySwap – Another BGP Hijack Targeting Crypto Wallets, February 2022. https://www.manrs.org/2022/02/klayswap-another-bgp-hijack-targeting-crypto-wallets/.

[82] Internet Society. What Happened? The Amazon Route 53 BGP Hijack to Take Over Ethereum Cryptocurrency Wallets, April 2018. https://www.internetsociety.org/es/blog/2018/04/amazons-route-53-bgp-hijack/.

[83] Raffaele Sommese, Mattijs Jonker, Roland van Rijswijk-Deij, Alberto Dainotti, K.C. Claffy, and Anna Sperotto. The Forgotten Side of DNS: Orphan and Abandoned Records. In *Proceedings of the 2020 Workshop on Traffic Measurements for Cybersecurity (WTMC)*, Virtual Event, 2020. IEEE.

[84] Raffaele Sommese, Giovane C. M. Moura, Mattijs Jonker, Roland van Rijswijk-Deij, Alberto Dainotti, K. C. Claffy, and Anna Sperotto. When parents and children disagree: Diving into DNS delegation inconsistency. In *Proceedings of the International Conference on Passive and Active Network Measurement (PAM)*, pages 175–189, Virtual Event, 2020. Springer, Springer International Publishing.

[85] Sooel Son and Vitaly Shmatikov. The Hitchhiker's Guide to DNS Cache Poisoning. In *Security and Privacy in Communication Networks*, pages 466–483, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[86] Cisco Talos. Sea Turtle keeps on swimming, finds new victims, DNS hijacking techniques, July 2019. https://blog.talosintelligence.com/2019/07/sea-turtle-keeps-on-swimming.html.

[87] The Washington Post. The New York Times Web site was taken down by DNS hijacking. Here's what that means., August 2013. https://www.washingtonpost.com/news/the-switch/wp/2013/08/27/the-new-york-times-web-site-was-taken-down-by-dns-hijacking-heres-what-that-means/.

[88] United States of America v Zhang et al. Case No 13CR3132-H, Indictment (superseding), June 2017. https://www.justice.gov/opa/press-release/file/1106491/download.

[89] Roland van Rijswijk-Deij, Mattijs Jonker, Anna Sperotto, and Aiko Pras. A High-Performance, Scalable Infrastructure for Large-Scale Active DNS Measurements. *IEEE Journal on Selected Areas in Communications (JSAC)*, 34(6):1877–1888, 2016.

[90] Verisign. Registry Lock Service, January 2022. https://www.verisign.com/en_US/channel-resources/domain-registry-products/registry-lock/index.xhtml.

[91] Verizon. 2021 Data Breach Investigations Report. https://enterprise.verizon.com/resources/reports/2021-data-breach-investigations-report.pdf, May 2021.

[92] VirusTotal. VirusTotal update_mfa.exe Details, October 2021. https://www.virustotal.com/gui/file/8900cf88a91fa4fbe871385c8747c7097537f1b5f4a003418d84c01dc383dd75/.

[93] Dustin Volz. DNC Says Russia Tried to Hack Into its Computer Network Days After 2018 Midterms, January 2019. https://www.wsj.com/articles/dnc-says-russia-tried-to-hack-into-its-computer-network-days-after-2018-midterms.

[94] Lan Wei and John Heidemann. Whac-A-Mole: Six Years of DNS Spoofing. Technical report, University of Southern California, 2020. https://arxiv.org/pdf/2011.12978.pdf.

[95] Florian Weimer. Passive DNS Replication, April 2005. https://www.first.org/conference/2005/papers/florian-weimer-paper-1.pdf.