Detecting Search Lists in Authoritative DNS

Andrew Simpson
Verisign, Inc.
12061 Bluemont Way
Reston, VA 20190
1-703-948-4228
asimpson@verisign.com

ABSTRACT

Early research of name collisions has postulated that search list interaction drives some portion of the DNS requests that have been observed for non-existent name spaces. In this paper we will explore the possibility that search lists can result in queries for DNS namespaces that do not currently exist. The paper will systematically exhibit use cases that can trigger search list interactions and then explain how this appears in root DNS traffic. The use cases will stem from illustrations of widely deployed HTML that inevitably trigger stub resolvers to append search suffixes to queries users do not even realize they are issuing. The result of this analysis is a method for understanding namespaces where collisions resulting in information leakage may exist.

1. INTRODUCTION

"The purpose of search list processing is to aid users by automatically mapping explicit query names to intended Fully Qualified Domain Names (FQDNs) through iterative (but structured) exploration of the DNS namespace." **(McPherson & Kumari, 2013)**

The iterative nature of search list processing means that local stub resolvers are designed to take queries for host names that do not resolve publicly or do not look like fully qualified hostnames and append the local search list to ensure the local network is not capable of providing a host that can answer for the name. In theory, this behavior should be contained within networks that are authoritative to answer these types of queries. The reality, however, is that there are many situations in which queries with local network details are requested against the public DNS system. RFC 1535 **(Gavron, 1993)** states "in any event where a '.' exists in a specified name it should be assumed to be a fully qualified domain name (FQDN) and SHOULD be tried as a rooted name first."

The end user use of search lists has been identified both as a likely cause of queries that reach public DNS servers for names that do not exist publicly and also as a possible source of harmful namespace collisions. "Concern arises when subdomain (e.g., www.corp) that normally is expanded iteratively using search list processing is delegated as a new namespace (i.e., within a new gTLD in the global Internet root). This delegation causes the queried name to resolve earlier in the resolution process and later stages of the search list processing to not be applied. Specifically, if a user has always expected www.corp to result in a response for www.corp.example.com, then an intermediate answer from a newly delegated gTLD will cause internal clients to resolve to registrant of www.corp (assuming the resource record being queried exists) rather than proceeding through the search list, as they may have always done before" **(McPherson & Kumari, 2013)**.

For the purpose of this paper, a search list is considered a set of search suffixes. A search suffix is a host such as "example.com" from above that will be appended by a users stub resolver to look on a local network for the host a user is trying to reach. These suffixes can each be iteratively applied to a user requested string. The rules for applying each suffix vary from one operating system (OS) to the next, and some of these details will be discussed later.

Given the volume of queries that may have originated from this type of interaction, there have been several community proposals to better capture and characterize this type of behavior. "Using Test Delegations from the Root Prior to Full Allocation and Delegation" **(Huston, Kolkman, Sullivan,**

Kumari, & Michaelson, 2013) proposes that triggering end users to request hosts that should invoke search suffixes, possibly by using large ad networks, it may be possible collect statistics about where various search suffixes are deployed. The proposed technique would provide details about specific networks, how they are configured, and how they employ and respond to various search list interactions by simultaneously triggering requests for queries against an established host that serves as a control. While this would be very informative, there is a considerable amount of setup required to get broad scale data.

This paper will illustrate how, by leveraging existing HTML references that trigger DNS prefetching, it is possible to get a good picture of the number of deployed search suffixes that exist within TLDs that have not been delegated. We accomplish this by studying DNS lookup patterns in root DNS traffic data that haves been compiled at DNS-OARC **(White, 2013)**.

2. BACKGROUND

Huston et al. proposed that one way to stimulate queries "is to embed measurement in web content, using a scripting language." The scripting language does provide a clean way to control exactly what gets queried when. However, establishing a web server that users will request your scripts from and then getting them to actually download and exercise the scripts will take time and money. The alternative we propose is to leverage the broken state of the web and end user browsers that aggressively prefetch any host name they see in HTML data, to catalog where we suspect search list interaction is happening and then to use existing or captured DNS data to characterize the current state.

Browser prefetching, or pre-resolving, is a technique employed by many popular browsers to speed up end user performance **(Krishnan & Monrose, 2011)**. Modern browsers equipped to do this will scrape HTML content that end users are viewing and preemptively trigger DNS queries for the hosts that are discovered. This primes the users' local cache for the possibility that, if a user attempts to visit that host, the DNS result has already been obtained. From a global DNS perspective, the impact of prefetching can be felt as total query volumes go up, but this behavior can also provide a wealth of information for study.

In the event that a website has a link to a hostname that doesn't exist, a browser will attempt to prefetch this bad link. If it is a normally formed hostname, it will be requested first against the public DNS. When a non-existent response is returned, the local resolver will have the opportunity to iterate through its search list and append each search suffix to check the local zone for the name. If the hostname is not well-formed, for example if it does not have a dot anywhere in the hostname, the local resolver will immediately append the search list to the query. Depending on the user's configuration and the network they are currently sitting in, this query will either get captured locally or be sent out across the public DNS. If the search suffix is for a TLD that is not delegated, the query will likely end up at the root server for resolution with little chance of a cache hit if the exact string being "searched" for has not previously been queried by users under that recursive.

It is appropriate to point out that different browsers have different implementations for determining what needs to be prefetched. Teasing apart the nuanced differences between these systems would be a very valuable exercise, but is out of the scope of what is being currently being evaluated. The method described here proves this to be true in some modern installed systems. Illustrations will be primarily from Mac OS X running Chrome even though others were tested (Ubuntu 13.04 running Firefox and Windows 7 running Chrome were tested and showed similar results). The primary point is that this behavior is expected to happen and can be used as a possible interpretation of root activity, particularly for TLDs that are not delegated.

The approach outlined here includes a technique for detecting where we expect browsers to begin prefetching in order to understand search list strings that are actively deployed. We then quantify the number search suffixes and some of the characteristics of the network leveraging it exhibits by observing DNS traffic in root DNS data.

3. APPROACH

Determining Where Prefetching with Search List Suffixes Should Occur

The Verisign Internet Profile Service is a platform used internally by Verisign to study the health of the overall domain industry. This project crawls a small amount of content from every domain that permits it and analyzes request traffic from the DNS resolution process. The IPS corpus includes roughly 700 million web pages. The crawl process gives Verisign the opportunity to build detailed statistics about linking relationships between domain names. These linking statistics can help isolate domains that have links to hosts that would not logically exist in the public space. We can then confirm that these invalid links are likely to be requested by studying the amount of traffic directed towards sites with the invalid links on them. The set of links that this process discovers becomes the core set of strings that can be used to identify search strings.

Studying Authoritative Traffic Data for Search List Interaction

Authoritative DNS traffic can now provide information about the search suffixes that may be in use. Theoretically, any authoritative server can characterize the search suffixes in use under the zones it is authoritative for but the results presented here only characterize the trends visible in root traffic for TLDs that do not exist. By scanning the host names that are requested, it is possible to identify all requests that have one of a target set of host components believed to trigger search list interactions. The data used to produce these results is from the NXD root traffic for applied for gTLDs from the ICANN new gTLD program. The data used is from the 2012 and 2013 DITL Collision data. The DNS-OARC Collisions group, including members Roy Hooper of Demand Media and Kevin White of JAS Global Advisors, mined a large amount of the raw DITL data into smaller and more manageable files based on the new proposed TLD strings. These files were separated by TLD and by year, allowing for easy analysis. The details of the files and their format can be found on the DNS-OARC site **(White, 2013)**. These two years were selected in order to focus on data where there were enough major browser vendors that had implemented DNS prefetching.

Characterizing The Network Emitting Search List Queries

Once the queries that are likely to have search suffixes appended have been extracted, we will illustrate how to leverage this data to further characterize the network from which they originated. We identify the search strings and then use them to group sets of queries that are likely to have originated from the same search string. The remaining details about the hosts requested out of that network and the sources for those queries can help determine activity on the network and further infer the number of entities that may depend on a particular search suffix.

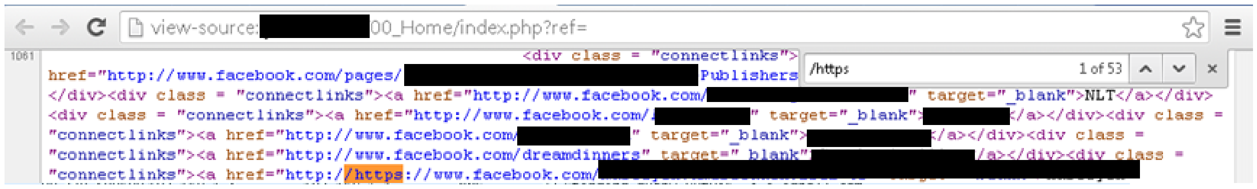
4. EXPERIMENTATION

To prove that this behavior does in fact happen, an OSX system running the latest version of Google Chrome was instrumented to record the DNS queries being issued. One website identified as having a number of broken links was used to illustrate the broad range of prefetches that could trigger search list interactions. The progression below shows in Figure 1 how a page was visited with the browser and shows a sampling of the subsequent DNS queries that were issued without any clicks from the end user. Figures 2 and 3 pair HTML code that contained a link with a screen capture from the PCAP that illustrates the stub resolver queries with the search suffix appended.

Figure 1: Live example of a domains that was identified as having a number of invalid links that should trigger prefetching and was loaded in a Chrome browser

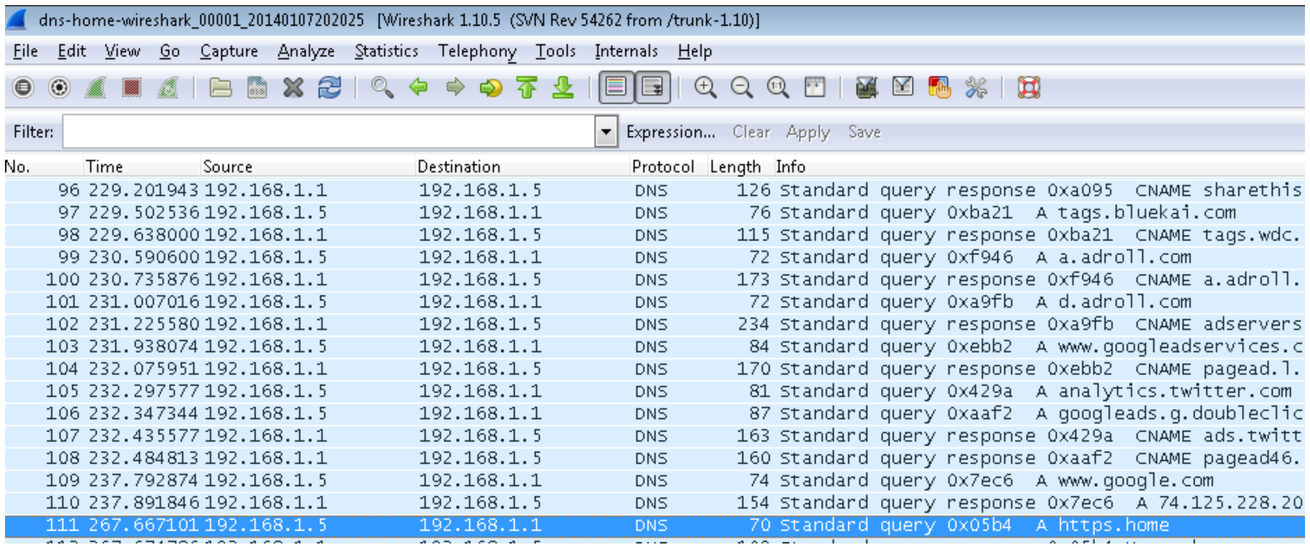
[redacted]e.com/00_Home/index.php?ref=
[redacted]
PUBLISHERS
E-NEWSLETTERS CONTACT PERMISSIONS
FAQ SITE MAP STORE LOCATOR
title, author or keyword
search
ENGLISH ESPAÑOL
DESKTOP MOBILE
ERRORS/ISSUES ?

Figure 2: HTML Example from main page that has a link to the hostname "https"



```
                              <div class = "connectlinks">
href="http://www.facebook.com/pages/[redacted]Publishers
</div><div class = "connectlinks"><a href="http://www.facebook.com/[redacted]" target="_blank">NLT</a></div>
<div class = "connectlinks"><a href="http://www.facebook.com/[redacted]" target="_blank">[redacted]</a></div><div class =
"connectlinks"><a href="http://www.facebook.com/[redacted]" target="_blank">[redacted]</a></div><div class =
"connectlinks"><a href="http://www.facebook.com/dreamdinners" target="_blank"[redacted]/a></div><div class =
"connectlinks"><a href="http://https://www.facebook.com/[redacted]
```

Figure 3: PCAP taken while page was loading showing that "home" was appended to "https" and sent out as a DNS query



```
No.  Time       Source       Destination  Protocol Length Info
 96 229.201943 192.168.1.1  192.168.1.5  DNS      126 Standard query response 0xa095  CNAME sharethis
 97 229.502536 192.168.1.5  192.168.1.1  DNS       76 Standard query 0xba21  A tags.bluekai.com
 98 229.638000 192.168.1.1  192.168.1.5  DNS      115 Standard query response 0xba21  CNAME tags.wdc.
 99 230.590600 192.168.1.5  192.168.1.1  DNS       72 Standard query 0xf946  A a.adroll.com
100 230.735876 192.168.1.1  192.168.1.5  DNS      173 Standard query response 0xf946  CNAME a.adroll.
101 231.007016 192.168.1.5  192.168.1.1  DNS       72 Standard query 0xa9fb  A d.adroll.com
102 231.225580 192.168.1.1  192.168.1.5  DNS      234 Standard query response 0xa9fb  CNAME adservers
103 231.938074 192.168.1.5  192.168.1.1  DNS       84 Standard query 0xebb2  A www.googleadservices.c
104 232.075951 192.168.1.1  192.168.1.5  DNS      170 Standard query response 0xebb2  CNAME pagead.l.
105 232.297577 192.168.1.5  192.168.1.1  DNS       81 Standard query 0x429a  A analytics.twitter.com
106 232.347344 192.168.1.5  192.168.1.1  DNS       87 Standard query 0xaaf2  A googleads.g.doubleclic
107 232.435577 192.168.1.1  192.168.1.5  DNS      163 Standard query response 0x429a  CNAME ads.twitt
108 232.484813 192.168.1.1  192.168.1.5  DNS      160 Standard query response 0xaaf2  CNAME pagead46.
109 237.792874 192.168.1.5  192.168.1.1  DNS       74 Standard query 0x7ec6  A www.google.com
110 237.891846 192.168.1.1  192.168.1.5  DNS      154 Standard query response 0x7ec6  A 74.125.228.20
111 267.667101 192.168.1.5  192.168.1.1  DNS       70 Standard query 0x05b4  A https.home
```

5. DATA

Broken Links

The Verisign Internet Profile Service identified 1,590,388 domains globally that contain a link or other source reference to host names that do not have valid top level domains. The most common hostnames identified in html are listed in Table 1.

Table 1: Invalid full hostnames referenced in public HTML

Hostname	Domain Count
http	**392,800**
static.mywebstats	**296,550**
Localhost	**138,246**

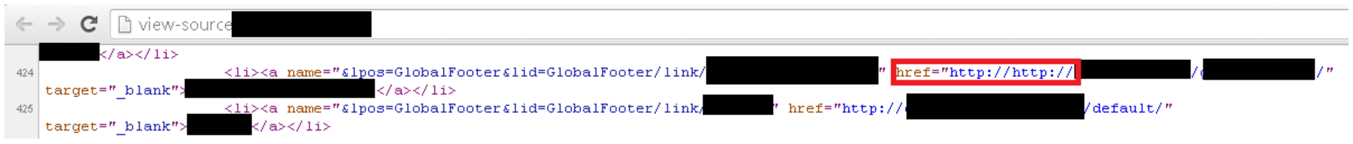
https	**42,035**
Index	**36,938**
H	**34,450**
www.mijndomein.nlhttp	**31,660**
www.	**30,279**
www.http	**20,600**
www	**16,136**
www.daily.co.ukproducts	**15,542**
Facebook	**11,307**
None	**10,440**
www.edju	**9,621**
A	**6,802**
N	**6,524**
Website	**6,521**
Google	**6,236**
Images	**5,911**
Admin	**4,485**

The invalid links are only useful for detecting search suffixes if people are visiting these sites so that prefetching can occur by legitimate end users. From Table 1, the links to “static.mywebstats” all originate from the same “under construction” style parking page and have very few users visiting them. By contrast, the bare references to “http” and “https” are perfect for this study for two reasons:

1. They do not have any dots and fall into a specific search convention specified for single label hosts. When a DNS request is made for single label hosts, the operating system will resolve them first with the search string appended **(Huston, New gTLD Concerns: Dotless Names and Name Collisions, 2013) (SAC064, 2014)**. When a browser discovers these host names being referenced in an HTML document sends a request to the local stub resolver for the single label host.
2. References to these hosts appear in very popular websites that get hit by millions of users each month. In fact, 9 of the domains ranking in Quantcast’s US Top 1,000 names **(Quantcast Corporation, 2014)** have links to these bare strings.

The references to html or https as a hostname typically comes from code that looks like this “http://http://”. Figure 4 has an example, with certain identifying features stripped out.

Figure 4: HTML from a popular sites Global Footer that contains link to "html" host



view-source[redacted]

```
[redacted]</a></li>
            <li><a name="&lpos=GlobalFooter&lid=GlobalFooter/link/[redacted]" href="http://http://[redacted]/[redacted]/"
target="_blank">[redacted]</a></li>
            <li><a name="&lpos=GlobalFooter&lid=GlobalFooter/link/[redacted]" href="http://[redacted]/default/"
target="_blank">[redacted]</a></li>
```

As further proof that this activity routinely occurs, our results revealed that there were more than 250K queries, amongst all queries for an undelegated but applied for TLD, where the first label in DNS qname matched the pattern of “http(s)?” in 2013 DITL data. We make the assumption that a significant portion of the queries prefixed with those strings are a result of browsers prefetching from sites carrying links like those that we have illustrated.

For those reasons, the method used to generate our results about where search suffixes have been appended is very simple. Any request the root servers received where the first label in the qname is either http or https is immediately followed by a search suffix.

The 2013 DITL data shows that 794 of the applied for strings have at least one search suffix that can be detected. Table 2 contains two examples of queries that would indicate a search list is in use.

Table 2: Example DITL queries that likely originate from sources that have appended a search list

Date	Time	TLD	SLD	Transport	Root Server	Query Type	Query
5/28/2013	**04:41.4**	**sampleTLD**	**Vip**	**udp**	**a-root**	**A**	**http.vip.sampleTLD**
5/29/2013	**20:27.6**	**sampleTLD**	**Vip**	**udp**	**m-root**	**A**	**http.vip.sampleTLD**

In the example packets from Table 2, sampleTLD has been used to anonymize the actual TLD studied. Throughout the remainder of the paper, sampleTLD, which will be used for the detailed examples, always represents the same TLD from root data. For the purpose of this study, we will broadly assume that any query where the first label “http” or “https” is the result of a browser attempting to prefetch that label as a result of ill formed HTML links and the suffix appended after that label is the users local search string “vip.sampleTLD”. Not surprisingly, the TLD that this behavior is most commonly observed in is “.home”. In the 2013 DITL collection, there were 214,794 queries that matched this pattern with 1,129 distinct search suffixes appearing to be appended. “.Corp” queries had more distinct suffixes appear in their overall dataset even though there were fewer total queries. Table 3 has the top TLDs and the number of unique search suffixes identified.

Table 3: Top TLDs from 2013 DITL matching search list query pattern

TLD	Search Suffix Queries	Unique Search Strings
home	**214,794**	**1,129**
corp	**38,226**	**3,183**
site	**9,370**	**190**
network	**8,364**	**198**
cisco	**8,099**	**15**
box	**5,718**	**61**
iinet	**3,874**	**8**
office	**3,157**	**374**
global	**2,671**	**183**
google	**2,270**	**17**
ads	**2,104**	**265**
samsung	**2,079**	**6**
inc	**1,987**	**189**
group	**1,884**	**269**

casa	**1,528**	**22**
business	**1,460**	**10**
dev	**1,082**	**99**
prod	**960**	**66**
unicorn	**916**	**3**
orange	**904**	**6**

If we further break down sampleTLD traffic from 2013 DITL data, we can now determine what percentage of the traffic for the sampleTLD can be attributed to the search list interaction. Using the described methodology, we have identified more than 150 search suffixes that account for nearly 90% of the more than 12M queries that were observed. Table 4 breaks this traffic down in a generic way that shows the strings are being requested by a large volume of IP addresses each.

Table 4: Anonymized Search Strings and the Percent of Traffic They Contributed

Search Suffix	Percent Total	IP Addresses Querying
No Search Suffix	**10.8%**	**>50K**
student.sampleTLD	**6.4%**	**>100K**
corp.root.sampleTLD	**5.2%**	**>50K**
corp.sampleOrg1.sampleTLD	**4.4%**	**>50K**
sampleOrg2.sampleTLD	**4.2%**	**>10K**
res.sampleOrg3.sampleTLD	**3.5%**	**>10K**
sampleOrg4.SampleTld	**2.6%**	**>10K**
sampleDiv1.sampleOrg5.sampleTLD	**2.5%**	**>1K**

6. FINDINGS

Traffic Not Associated with an Identified Search Suffix

More than 14% of the non-search traffic contains an "invalid" or non-letter/digit/hyphen based second level domain. 25% of the traffic can still be identified as having a protocol based discovery label, as discussed in earlier studies such as DNS-SD, ISATAP, WPAD or Microsoft Active Directory **(Verisign Labs, 2013)** in the third label. This is an indication that while this approach is informative and characterizes a large number of the suffixes active in a particular namespace, it is not 100% effective at identifying all internal "search suffixes." A more complete algorithm to derive search suffixes would permit each of these protocols to also define namespace barriers **(Verisign Labs, 2013)**.

Additional Details from Traffic with an Identified Search Suffix

There are 626,088 distinct labels appearing just before the search suffix in the example top-level domain that was studied for this experiment. The most common label to appear in the query immediately preceding the string that has been identified as a search suffix is "_mscds". This label can be described as "a Microsoft-specific subdomain which enables location of domain controllers that have specific roles in the Active Directory domain or forest. Resource records for the DNS root domain of a new Active Directory forest are stored in a _msdcs zone instead of a subdomain, and that zone is stored in the forest-wide application directory partition." **(Microsoft, 2011)** The second most common string appearing immediately before the search suffix is "wpad". Wpad typically represents a protocol that is outlined in the internet draft "Web Proxy Auto-Discovery Protocol" **(Gauthier, Cohen, Dunsmuir, & Perkins, 1999)**. The protocol is designed to allow web client software to automatically discover the URL of a configuration URL (CURL) that instructs a client which proxy cache servers are available for use.

Additional labels appearing under the search suffixes range from other protocols to end user machine names like “masmith-lt” which could possibly be “Michael A. Smith’s Laptop.”

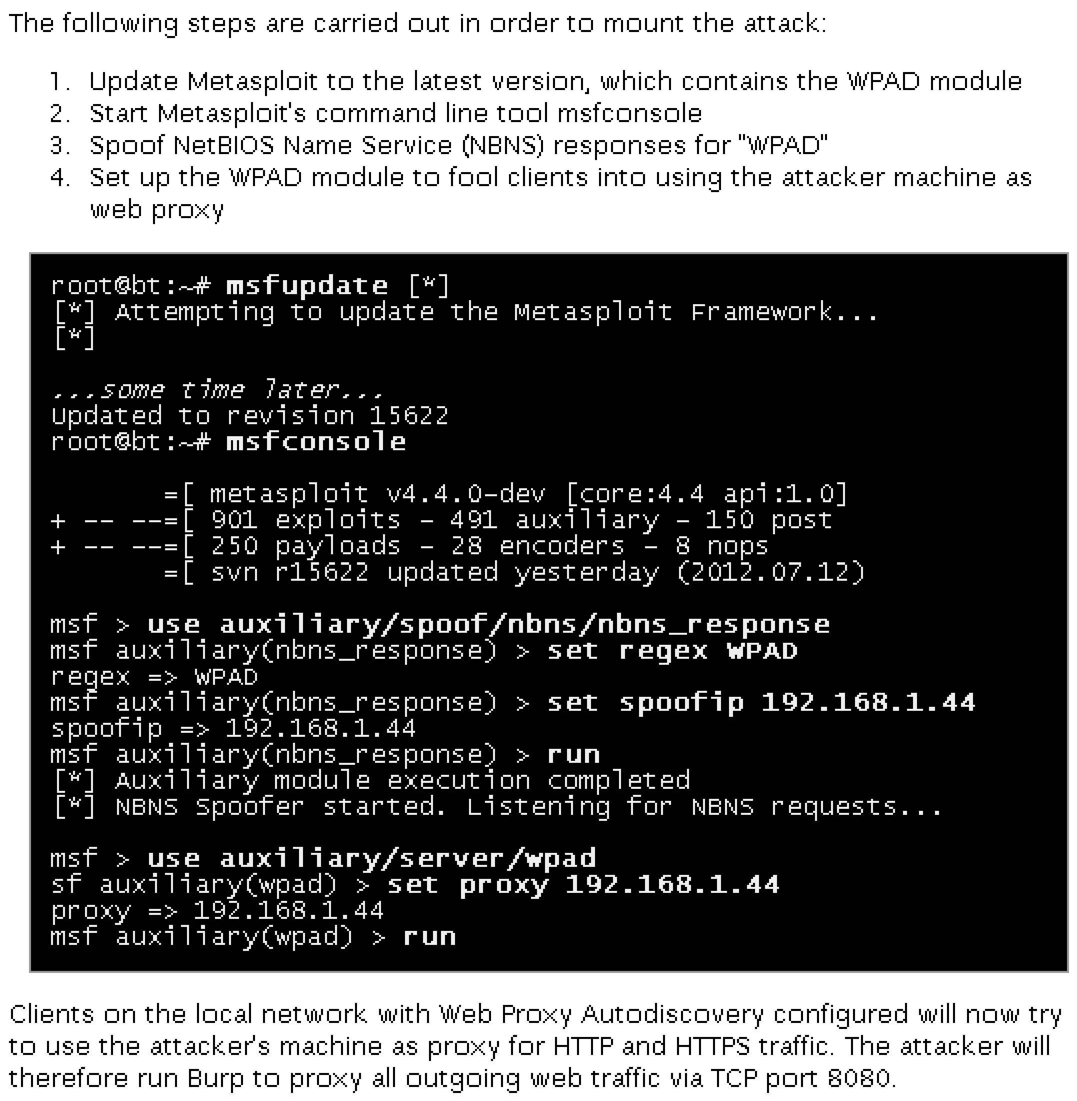
Table 5: Top Labels found to be preceding search strings in queries under example TLD

Label Preceding Search Suffix	Number of Times Observed
_msdcs	**1,434,729**
wpad	**415,608**
com	**359,103**
isatap	**232,838**
_tcp	**221,146**
_sites	**155,337**
org	**151,351**
sms_slp	**109,079**
kr	**96,220**
local	**93,859**
net	**65,469**
fihp-avi01	**61,829**
mena	**53,678**
lnsrvxx-xxx01	**51,941**
ussrvxx-xxx01	**51,459**
corproot	**50,318**

7. ILLUSTRATION OF COLLISION RISK

Now that there is an established pattern that can identify search suffixes, it is possible to evaluate one of the “what if” scenarios that could come out of a collision associated with delegating these previously non-existent entities. If a non-affiliated party takes control of a domain that is currently receiving traffic from someone else’s search suffix, they can begin to respond authoritatively to the end users querying it. Returning to the WPAD protocol in particular, all but 3 of the more than 150 search suffixes referenced in Table 4 had at least one request for wpad under the identified search suffix in the 2013 DITL data. WPAD can easily be exploited if the queries get answered by the wrong party **(Hjelmvik, 2012)** **(Bugher, 2008)**. Figure 5 is a snippet from a WPAD man-in-the-middle exploit using Metasploit (http://www.metasploit.com/).

Figure 5: Hjelmvik Illustration of how to setup WPAD exploit


The following steps are carried out in order to mount the attack:

1. Update Metasploit to the latest version, which contains the WPAD module
2. Start Metasploit's command line tool msfconsole
3. Spoof NetBIOS Name Service (NBNS) responses for "WPAD"
4. Set up the WPAD module to fool clients into using the attacker machine as web proxy

```
root@bt:~# msfupdate [*]
[*] Attempting to update the Metasploit Framework...
[*]

...some time later...
Updated to revision 15622
root@bt:~# msfconsole

       =[ metasploit v4.4.0-dev [core:4.4 api:1.0]
+ -- --=[ 901 exploits - 491 auxiliary - 150 post
+ -- --=[ 250 payloads - 28 encoders - 8 nops
       =[ svn r15622 updated yesterday (2012.07.12)

msf > use auxiliary/spoof/nbns/nbns_response
msf auxiliary(nbns_response) > set regex WPAD
regex => WPAD
msf auxiliary(nbns_response) > set spoofip 192.168.1.44
spoofip => 192.168.1.44
msf auxiliary(nbns_response) > run
[*] Auxiliary module execution completed
[*] NBNS Spoofer started. Listening for NBNS requests...

msf > use auxiliary/server/wpad
sf auxiliary(wpad) > set proxy 192.168.1.44
proxy => 192.168.1.44
msf auxiliary(wpad) > run
```

Clients on the local network with Web Proxy Autodiscovery configured will now try to use the attacker's machine as proxy for HTTP and HTTPS traffic. The attacker will therefore run Burp to proxy all outgoing web traffic via TCP port 8080.

In the example above, the exploit is being conducted from a wireless network where the attacker is in the same LAN. However in the situation where a new TLD has been delegated for which queries are already hitting the public root, it will be possible to answer them outside of the users network to direct them to a public web server. If that web server responds with a CURL that directs users to a public web proxy controlled by an attacker, the end user's private information could be exposed. For example, the more than 100K IP addresses requesting student.sampleTLD from Table 4 could be provided configuration to use a malicious web proxy.

These are known exploits that the current mitigation strategy for name collisions are not prepared to handle. The primary mitigation strategy proposed in "*Mitigating the Risk of DNS Namespace Collisions*" is to use a 120-day period of controlled interruption to direct queries to 127.0.53.53 **(JAS Global Advisors, 2014)**. This will effectively break users who are currently using their local search suffix to discover proxy servers. For the users whose machines are currently looking for, but not leveraging, WPAD, the controlled interruption period will not change anything. This means they will not know their search suffix could soon be controlled by someone with malicious intent. As an attacker who successfully provides back their proxy server to unsuspecting end users, "you have total control over their web activity. You can read everything they do, everything that the server responds with, etc. You can redirect any web traffic transparently to your own fake sites. You can even modify the responses from real sites on-the-fly with scripts" **(Bugher, 2008)**.

8. CONCLUSIONS

Ultimately, we have provided a systematic analysis of how we can observe and characterize the impact of search lists from root DNS traffic. By using the simple techniques we have provided, it is possible to better understand the number of unique entities that may be reliant on a particular string even before it is delegated. Once any string is delegated, the authoritative operator will have access to all of this information in real-time. It will be possible for them to instrument their server to collect more timely information on this topic than what is currently available at OARC. More current data may present the insights needed to determine which substrings below their level of authority are safe to delegate and which ones require some form of risk mitigation to make sure that any dependent parties do not collide with one another.

9. Works Cited

Aboba, B., & Cheshire, S. (2002, 11). *Dynamic Host Configuration Protocol (DHCP) Domain Search Option*. Retrieved 02 25, 2014, from IETF Network Working Group: http://tools.ietf.org/html/rfc3397

Bugher, G. (2008, 01 11). *WPAD: Internet Explorer's Worst Feature.* Retrieved 02 27, 2014, from Perimeter Grid: http://perimetergrid.com/wp/2008/01/11/wpad-internet-explorers-worst-feature/

Gauthier, P., Cohen, J., Dunsmuir, M., & Perkins, C. (1999, 07 28). *Web Proxy Auto-Discovery Protocol*. Retrieved 02 27, 2014, from IETF Internet Draft: http://tools.ietf.org/html/draft-ietf-wrec-wpad-01

Gavron, E. (1993). *A Security Problem and Proposed Correction With Widely Deployed DNS Software*. Retrieved from http://tools.ietf.org/html/rfc1535

Hjelmvik, E. (2012, 07 12). *WPAD Man in the Middle*. Retrieved 02 27, 2014, from Netresec: http://www.netresec.com/?page=Blog&month=2012-07&post=WPAD-Man-in-the-Middle

Huston, G. (2013, 11 12). *New gTLD Concerns: Dotless Names and Name Collisions*. Retrieved 02 25, 2014, from RIPE Labs: https://labs.ripe.net/Members/gih/dotless-names

Huston, G., Kolkman, O., Sullivan, A., Kumari, W., & Michaelson, G. (2013, 10 19). *Using Test Delegations from the Root Prior to Full Allocation and Delegation*. Retrieved 02 04, 2014, from http://tools.ietf.org/html/draft-kolkman-root-test-delegation

ICANN SSAC. (2014, 02 13). *SSAC Advisory on DNS "Search List" Processing*. Retrieved 02 25, 2014, from SAC064: http://www.icann.org/en/groups/ssac/documents/sac-064-en.pdf

JAS Global Advisors. (2014, 02 26). *Mitigating the Risk of DNS Namespace Collisions*. Retrieved 02 27, 2014, from ICANN: https://www.icann.org/en/about/staff/security/ssr/name-collision-mitigation-26feb14-en.pdf

Krishnan, S., & Monrose, F. (2011). *An Empirical Study of the Performance, Security and Privacy Implications of Domain Name Prefetching*. Chapel Hill: University of North Carolina.

McPherson, D., & Kumari, W. (2013, 09 27). *On DNS Search List Processing: Perhaps the Most Misunderstood Stable of DNS Resolution*. Retrieved 02 04, 2014, from http://forum.icann.org/lists/comments-name-collision-05aug13/pdfswejx3rLKE.pdf

Microsoft. (2011, 05 27). *How DNS Support for Active Directory Works*. Retrieved 02 27, 2014, from Microsoft TechNet: http://technet.microsoft.com/en-us/library/cc759550(WS.10).aspx

Quantcast Corporation. (2014, 02 08). *Top Ranking United States Websites*. Retrieved 02 08, 2014, from Quantcast: https://www.quantcast.com/top-sites

SAC064. (2014, 02 13). *SSAC Advisory on DNS "Search List" Processing*. Retrieved 02 25, 2014, from ICANN SSAC: http://www.icann.org/en/groups/ssac/documents/sac-064-en.pdf

Verisign Labs. (2013, 09 15). *ICANN's Proposal to Mitigate Name Collision Risks – .CBA Case Study*. Retrieved 02 26, 2014, from Verisign Inc.: https://www.verisigninc.com/assets/report-cba-analysis.pdf

Verisign Labs. (2013, 08 22). *New gTLD Security, Stability, Resiliency Update: Exploratory*. Retrieved 02 25, 2014, from Verisign Labs Technical Reports: http://techreports.verisignlabs.com/docs/tr-1130008-1.pdf

White, K. (2013, 10 25). *2013 Collisions Project DITL Analysis | DNS-OARC*. Retrieved 02 07, 2014, from DNS-OARC: https://www.dns-oarc.net/node/332

.