Identity Trail: Covert Surveillance Using DNS

Saikat Guha and Paul Francis

Cornell University, Ithaca NY 14853, USA
{saikat, francis}@cs.cornell.edu

Abstract. The Domain Name System (DNS) is the only globally deployed Internet service that provides user-friendly naming for Internet hosts. It was originally designed to return the same answer to any given query regardless of who may have issued the query, and thus all data in the DNS is assumed to be public. Such an assumption potentially conflicts with the privacy policies of private Internet hosts, particularly the increasing numbers of laptops and PDAs used by mobile users as their primary computing device. IP addresses of such devices in the DNS reveal the host's, and typically the user's, dynamic geographic location to anyone that is interested without the host's knowledge or explicit consent. This paper demonstrates, and measures the severity of an attack that allows anyone on the Internet to covertly monitor mobile devices to construct detailed user profiles including user identity, daily commute patterns, and travel itineraries. Users that wish to identify their private hosts using user-friendly names are locked into the DNS model, thus becoming unwitting victims to this attack; we identify a growing number of such dynamic DNS users (two million and climbing), and covertly trail over one hundred thousand of them. We report on a large scale study that demonstrates the feasibility and severity of such an attack in today's Internet. We further propose short-term and long-term defenses for the attack.

1 Introduction

The Domain Name System (DNS) is a core Internet infrastructure that maps user-friendly mnemonics to non-user-friendly IP addresses. The DNS resolves IP addresses for both *public services*[1] like Google, as well as *private services*[2] such as Alice's personal laptop. The DNS does not distinguish between the scope of the services it resolves.

As stated in RFC 4033 [1], the DNS was originally designed with the assumption that the DNS will return the same answer to any given query regardless of who may have issued the query, and that all data in the DNS is thus public. The DNS does not provide any authorization mechanism or other means of differentiating between inquirers. Indeed, DNS nameservers do not even know the IP address of the querying host. Network DoS attackers exploit this shortcoming to learn the IP address of the victim and overwhelm the victim's link

[1] Services available to everyone e.g. www.google.com
[2] Services available to a small group of people e.g. alice.dyndns.org

to the Internet. This paper identifies a different attack whereby merely learning the IP address of the victim can result in a breach of privacy defined as contextual-integrity [2].

An IP address implicitly encodes a wealth of information about the host. The address identifies the host's ISP (university, corporation, residential broadband ISP etc.), and the geographical location of the host to within a nearby city [3]. This information is available in public WHOIS databases, reverse DNS entries [4], and commercial databases [5]. While leaking the geographic location of static Internet hosts (e.g. a corporate webserver) may not be a critical privacy concern, leaking the dynamic geographic location of private, and often mobile hosts (e.g. employee laptops) may comprise an unacceptable privacy breach. The DNS does not capture this difference between public and private hosts.

Private services that wish to use user-friendly names are consequently forced to make their existence and location visible to everyone on the Internet. Alice, for instance, may wish to run a private FTP server on her personal laptop that she can use to transfer files to and from her laptop whether she is at work or at home. Since the laptop may often acquire a different IP address each time she connects, Alice is forced to relearn her address each time. Alternatively, Alice can configure her laptop to update the DNS with its latest address such that she can use a stable user-friendly DNS name in her FTP client. By putting her address in the DNS for *her own personal use*, however, Alice unwittingly reveals her geographic location to *anyone* on the Internet.

DynDNS [6], No-IP [7], TZO [8], and many other online services [9–11] cater to individuals such as Alice that wish to register DNS names and dynamically update their IP address mapping. Update clients are available for a wide range of platforms including not only Windows, Linux and MacOS, but also cellphones, pocket PCs and embedded devices [12]. While commercial services like Akamai [13] provide IP address confidentiality of the origin service by redirecting all data through Akamai proxies, it is typically targeted at large public websites (Google, Microsoft etc.) and not small private services such as Alice's. Users such as Alice are entirely dependent on the dynamic DNS services mentioned earlier for user-friendly names, with services such as DynDNS boasting of a growing user-base of two million users [14].

As we report later, a majority of these dynamic DNS names are for mobile private hosts. Common services running on these hosts include private FTP and web servers, BitTorrent trackers, and web-cams. Even though each of the services may enforce access control policy at the service level, simply registering the host's address in the DNS leaks private information to anyone who cares to learn it. The service owner must therefore chose either user-friendly DNS naming or privacy.

This privacy issue was not considered one way or another in the design of the DNS. In a largely static Internet, the IP address does not divulge much private information and thus the issue has not been important in the past. In the present Internet, however, with mobile private hosts, the lack of confidentiality of private IP addresses published into the DNS is a privacy risk as mobile hosts,

unlike static hosts, reveal the dynamic geographic location of the user. This is exacerbated by current trends where laptops are surpassing desktops in retail sales [15].

Overall this paper makes three contributions.

1. We identify an attack that allows an attacker to covertly monitor a victim's location at any given moment, and over time build a detailed profile of the victim including the victim's identity, daily commute patterns, and trip itineraries.
2. We demonstrate the feasibility of such an attack on the Internet today by performing surveillance on over a hundred thousand users without raising any alarms and report the depth of private information gleaned.
3. We propose short-term fixes to the DNS that can be deployed today to mitigate this attack, and discuss a long-term solution for secure name resolution for private services on the Internet.

The rest of the paper is organized as follows. Section 2 presents an overview on the DNS, dynamic DNS services and related work in DNS security. Section 3 presents our attack. Section 4 reports on a covert surveillance experiment of over a hundred thousand Internet users. Section 5 discusses short-term and long-term measures to defend against such an attack. Section 6 concludes the paper and outlines future work.

2 DNS Overview and Related Work

The DNS describes the architecture and infrastructure for name resolution on the Internet [16]. The namespace is hierarchical for user-friendliness and ease of administration. Each subtree of the namespace (called *zone* or *domain*) is managed by designated nameservers referred to as the *authoritative nameservers* for that domain. An authoritative nameserver responds with the IP address (or other information in the DNS) for a DNS name in its domain when queried. If the DNS name lies outside the nameserver's domain, the server can forward the query to another nameserver that may be able to answer authoritatively (called *recursive querying*), and forward the response back to the original inquirer. Alternatively, the server can simply return the address of the next server to query (called *non-recursive querying*). A server that performs recursive querying may cache the results of any queries that it forwards and respond to subsequent queries for the same DNS name from its cache thereby improving performance and reducing the load on the authoritative nameservers. Clients OS network stacks typically implement the minimum functionality necessary to send the query to a recursive nameserver; ISPs and public services provide recursive nameservers that customers can use for all queries.

The DNS does not enforce any access control policy before name resolution. In particular, the authoritative nameserver typically does not learn the identity of the host performing the name resolution. This is because a recursive

DNS query contains the address of only the nameserver performing the recursive lookup, and not the address of the host on whose behalf the query is being performed. Furthermore, when a response is cached and a query answered from the cache, the authoritative nameserver does not learn that a lookup took place. Any host on the Internet can query for the IP address of another host through a recursive nameserver without revealing the original inquirer's identity to the authoritative nameserver.

Dynamic DNS services such as DynDNS administer multiple zones within which a user can create a DNS name (e.g. alice.dynalias.org, or bob.homeip.net). The user configures a background DynDNS client application to run on their laptop and update the laptop's IP address with the service every time the IP address changes. DNS name creation and dynamic updates of the associated IP address are performed over HTTP and protected with HTTP-based authentication methods. The nameservers for these dynamic DNS services, however, are typical DNS nameservers that cannot authenticate the source of a DNS query.

In addition to resolving the DNS name to an IP address, the DNS provides inverse resolution that maps an IP address to a canonical DNS name for that host. The canonical name for an IP address is assigned by the host's ISP. For example, the DNS name alice.dyndns.org may resolve to 192.0.2.1 in the address block allocated to Acme Inc. The reverse DNS resolution for the same IP, however, may return host-342.acme.org as the canonical name for the host. Anyone can find the ISP-assigned canonical name for the host that a DNS name points to.

Past work in securing DNS can be classified into two categories. The first category deals with protecting the DNS from outages and DoS attacks. CoDNS [17] and CoDoNS [18] use a peer-to-peer substrate for DNS queries in order to improve resiliency against failures and shield authoritative nameservers from flash crowds and DoS attacks, but otherwise allow anyone to resolve the DNS name for any host. The second category of past work deals with the integrity of DNS responses. DNSSEC [1] provides data and origin authentication of DNS data. ConfiDNS [19] provides better integrity of non-DNSSEC responses in CoDNS. Gabrilovich et al. [20] identify homograph attacks against the DNS where a user can be tricked to resolve a look-alike DNS name instead of the intended DNS name, and offer some potential solutions. None of these systems authenticate the source of the DNS query, and consequently cannot defend against the attack identified in this paper.

Proposed replacements for the DNS allow an individual to learn the victim's routing and addressing information without the victim's explicit consent. DOA [21] unconditionally resolves an endpoint identifier to a stack of addresses revealing the route to the destination. UIA [22] allows endpoint addresses to be resolved as long as the endpoint can be named; the ability to name another endpoint is transitive and is likely to be universal for global communication. Neither of the proposed approaches are designed to protect the confidentiality of a private host's address.

3 Identity Trail Attack

The attack intends to track a victim's location covertly. The 9-line attack code is listed in Fig. 1. The attack consists of logging a DNS lookup (line 6) and IP address to geolocation result (line 8) every hour. The geolocation uses a public service [5] that returns the city, province, and country of the host as well as the canonical hostname for the IP address. The attack assumes that the attacker knows the DNS name of the victim's laptop; while this is easy to arrange for a boss spying on their employee or a spouse spying on their significant other, to demonstrate the attack at scale we discuss in the next section how we learned the names of tens of thousands of potential victims. The attack uses only public services such as the DNS and IP geolocation as they were designed. The attack does not require superuser privileges, nor does it need to send any packets to the host being monitored. Finally, thousands of hosts can be monitored in parallel. The extent of private information that can be extracted using this attack is explored in the next section.

```
1: #!/bin/bash
2: HOST=victim.dynalias.net
3: GEO='http://www.ippages.com/simple/'
4: FLD='hostname,ip,city,state,country'
5: while sleep 3600; do
6:     IP=$(host $HOST | awk '{print $4}')
7:     date
8:     curl -s "$GEO?get=$FLD&ip=$IP"
9: done
```

Fig. 1. BASH shell script tracking the location of `victim.dynalias.net` every hour.

4 Attack Validation

In order to determine the feasibility and severity of launching the described attack over the Internet today, we monitored the mobility of over a hundred thousand Internet users. The attack involved discovering names of potential victims and monitoring their IP address for extended periods of time without being detected. Finally, we analyzed the mobility patterns in the trace to profile our victims' daily commute patterns, business and personal trip itineraries and, in some cases, even the identities of the victim.

4.1 Discovering DNS Names

We discovered 36,011 potential victims through a variety of methods. In our first experiment, we targeted users of the DynDNS service by first performing

Google and Yahoo! searches for all 65 DynDNS-controlled domains under which users can register DNS names. We found 4351 DNS names (far fewer than we expected) mentioned in web pages, mailing list archives, USENET posts, and other publicly searchable forums. In our second experiment, we performed a dictionary scan of four of the most popular DynDNS domains. Our dictionary consisted of 24,289 combinations of common first and last names and initials. The scan successfully resolved 31,660 DNS names with a success rate of up to 39% for the dyndns.org domain; the high success rate suggest similarities with [23] where the authors find that registered userids, in many cases, match first or last names of the user. The scan was performed from 40 hosts in 5 hours and rate-limited to a conservative aggregate of 5 packets per second distributed over 5 DynDNS nameservers to avoid triggering potential DDoS[3] responses. Only 9 of the hosts discovered in the dictionary scan were also returned by the online search engines suggesting almost all the names discovered by the dictionary scans are for private services. To verify this hypothesis, we performed a third experiment where we used Nmap [24] to scan a subset of 100 hosts discovered to determine the services provided. We discovered that 50% run HTTP servers, 21% run FTP servers, and 11% run the Windows File-and-Printer sharing service; the FTP and Windows services requires authentication while the default HTTP homepage is usually devoid of content and hyperlinks (possibly privacy by obscurity, where the actual content is located at a secret URL on the server).

Overall we make the following three observations from the above experiments. (1) The services discovered are intended for private use based on service authentication, lack of advertisement on public Internet forums, and lack of content on default landing pages. (2) An attacker can covertly discover a large number of potential victims without triggering alarms. (3) Poorly chosen DNS names registered by DynDNS users, in some cases, leak the name of the user for a mobile host.

4.2 Monitoring Hosts

In our first surveillance experiment, we monitored 18,720 hosts from July 20, 2006 to August 8, 2006 and found evidence of deliberate user mobility. The monitoring load was rate-limited to 1 packet per second and did not raise any alarms at DynDNS or at the source (to the best of our knowledge). Figure 2 is a screenshot of a summer road-trip taken by user M as tracked by our application. M's name resolves to a Seattle IP on 7/20. It subsequently resolves to Port Angeles WA on 7/21, and continues down a southern route along the west-coast through Otis OR, Smith River CA, Garberville CA, Los Angeles CA, Los Alamos CA, and Garden Grove CA at 1–2 day intervals. M then resolves to Las Vegas, NV for 3 days starting the night of 8/2. Finally, M appears to drive north through Montana back home to Saskatoon, Canada on 8/8, which is where he was resolved to on 11/7 as well. Based on reverse DNS lookups, M logs in through local broadband ISPs except on 7/29 when M logs in through

[3] Distributed Denial of Service

Fig. 2. Tracking a user's summer road-trip through the DNS

a dial-up ISP whose proxy servers are located in Reston, Virginia; fundamental limitations of geolocalization pertaining to proxies are explained in [3]. M runs a local firewall configured to filter all inbound packets. Unfortunately, we were unable to disambiguate M's real identity enough to contact him for verification.

In another trace (not illustrated), user S is geolocated to San Mateo CA until 7/30. S is subsequently resolved to Hyderabad, India for the week commencing Monday 7/31 through Sunday 8/6. For a few hours on 7/30 and 8/6, S is geolocated to a Singapore address suggesting that S flew across the pacific on these dates. In yet another trace, user K is geolocated to Vancouver Canada until 7/26, and then to Ottawa Canada after that. In this case, we were able to identify K through the webpage (hosted on the mobile host) and contact him over email for positive confirmation of the correctness of our tracking. That said, we failed to contact the users for an overwhelming majority of the hostnames tracked due to the private nature of services run by them; in particular, as mentioned previously, FTP servers when present required non-anonymous user authentication, HTTP servers delivered blank default pages (except for user K for example) and so on.

Confirmation from user K above notwithstanding, we verified the correctness of our application in our second surveillance experiment where we tracked the authors of this paper and compared their traces to known real-world data. Figure 3 plots the mobility of one of the authors from August 18, 2006 to November 2, 2006. All the information in the figure was gathered by performing geolocalization and reverse DNS lookups for the IP address. Geolocation within the United States was correct to within 100 mi, and in Italy was correct to within 250 mi. The daily commute patterns are usually correct to within one or two hours, but in some instances when the author did not turn the tracked laptop on

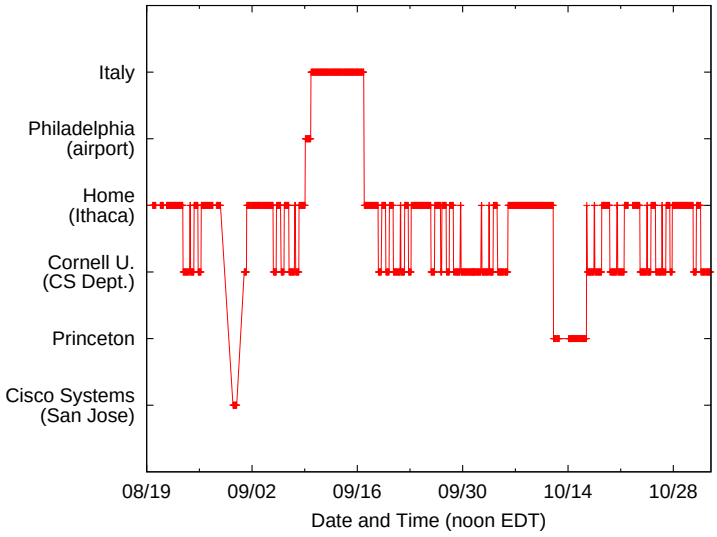

Fig. 3. Tracking a user's daily commute and travel through the DNS

for several hours after commuting, the trace is inaccurate. The trace, however, does accurately capture university holidays, work related trips and one airport layover. Based on the reverse DNS lookups suggesting the user's affiliation to the Cornell University Computer Science Department, the trip to Cisco Systems Inc. in San Jose, the overlap of the Italy trip with SIGCOMM 2006, a popular data communications conference in Italy during the same period, and the published proceedings of the conference, there was enough information to narrow down the identity of the person tracked to within two people in the Cornell CS department. Additional public information available on the department homepage yielded a unique match.

In our third surveillance experiment, we follow a random sampling of 118,000 DNS names for 77 days beginning August 14, 2006. Anonymized update records were obtained from logs kept by DynDNS. We filter out updates where consecutive IP addresses for a user belong to the same /24 subnet, or belong to the same ISP and are geolocated to the same city in order to discount many DHCP-related updates. Figure 4 plots the number of unique cities, provinces, countries and networks that mobile users were resolved to in rank order. To account for the geolocation errors observed in the previous experiment, we cluster geolocations within $\pm 0.75°$ latitude and longitude of each other to a single location, which is plotted as a separate curve in the figure. As evident from the figure, the median number of updates across all users was 64; in the median case a user logged in from IP addresses geolocated to 15 cities, and 3 provinces over the course of our measurement. The number of users connected to more than one ISP was 15,055,

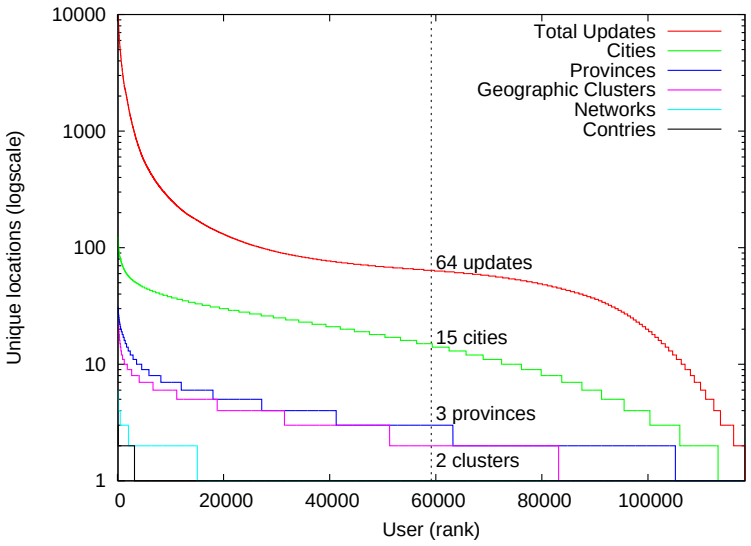

Fig. 4. Tracking the mobility of 118,000 users

while that geolocated to more than one country was 3,210. The median number of geographically distinct clusters a host was geolocated to was 2.

While the data above is indicative of user mobility, we are quick to point out the limitations of our measurement.

1. First, we do not exclude dynamic DNS entries potentially caused by botnets. In [25], the authors claim that botnet operators use Dynamic DNS services to relocate the "command and control" (C&C) server to dodge efforts by local authorities shutting down identified C&C servers. While such activities may explain the outliers in our sample (hosts that resolve to a hundred cities in only 77 days), we do not investigate such outliers any further.
2. Second, we use a proprietary commercial service for geolocation. The accuracy of such services is biased towards country-level and province-level geolocation more than city-level geolocation based on anecdotal evidence, although city-level geolocation is typically accurate to a within a nearby city. Recent systems such as Constraint-based geolocation [26] and Octant [27] have been proposed to improve geolocalization accuracy, however, the coverage provided by these services was not sufficient for our dataset.
3. Third, a host switching between different network providers in one city may be geolocated to different nearby cities based on the providers' access routes. Conversely, a mobile host accessing the network from different nearby cities through the same provider may be geolocated to a common nearby city.
4. Fourth, a physically static host behind an ISP that uses the same address space for a geographically diverse region, particularly in combination with traditional NATs [28], may be geolocated to different cities in that region.

Clustering geolocations based on proximity helps mitigate the effect of these artifacts, but does not completely eliminate them. Nevertheless, based on our (rather noisy) geolocation data, we believe that between 70% (based on clustering) to 91% (based on actual geolocations) of the users in our DynDNS-log dataset logged in from at least two different locations[4].

Overall we make the following four observations. (1) A majority of users updating their IP address on DynDNS are mobile users. (2) The IP address can accurately track a user to within a few hundred miles. (3) The extent of private information leaked over time is potentially significant enough to reconstruct trip itineraries, daily commute patterns, and in some cases, narrow down the identity of the user. (4) This entire surveillance operation can be performed covertly by any attacker on the Internet.

5 Solutions

Fundamentally, the Internet lacks an access-controlled name resolution service for private hosts. The DNS has been shoehorned to fill this need, however, as illustrated by the attack described in this paper, it is ill suited for this purpose. Particularly, the DNS lacks access control policy enforcement for name resolution. At present, any attacker on the Internet can track the mobility of any DNS-named host at any given time. In the short-term, there is no substitute for DNS for naming private endhosts, however, nameservers can use heuristics to limit the scope of the attack without requiring any client modifications. In the long-term, however, the core issue of name resolution policy of private services must be tackled.

5.1 Short-Term Defense

The DNS does not provide any mechanism for a host to specify access control policies for its own name. One fix would be to encourage users to pick obscure DNS names and restrict their dissemination to trusted parties. While this approach would hinder our dictionary attack, somewhat counterintuitive is that the obscurity provided by this approach is nullified when combined with DNSSEC. The DNSSEC [1] standard currently *requires* that the entire list of zone names be revealed to all in order to provide signed proof of non-existence of names that do not exist. An attacker can therefore simply enumerate all the names in a DNSSEC secured zone to discover potential victims.

A different fix would be for the authoritative nameserver to restrict host lookup to a white-list of recursive nameservers authorized for that host. The white-list may be set statically by the user, dynamically by the update application, or heuristically by the nameserver based on query patterns. Deploying a white-list does not require changes to the DNS protocol or client applications.

[4] Where "location" includes different providers (residential ISP, public wi-fi, office etc.) in the same city, as well as different cities.

This approach relies on access control of ISPs' recursive nameservers, which is typically enabled in response to recent attacks [29]. The approach protects against attackers on networks far from the victim, but not from attackers on the same network as the victim since both would have access to a common ISP recursive nameserver, which would be on the white-list.

A flawed way to add access control to the DNS protocol itself is to encrypt responses at the nameserver so only authorized users may decrypt them. Users of a private service could configure shared secrets or public keys with the authoritative nameserver. The nameserver could then encrypt the IP address using symmetric or asymmetric encryption [30, 31] based on the attack model. The key management required for such a solution that properly takes into account the distributed caching and recursion inherent in DNS as well as complex endhost privacy policies, however, is likely to be a challenge.

Another flawed approach is to require application-level proxies for private services. For example, a private service could subscribe to, and advertise the address of, a public proxying service which would proxy application data between the endpoints. Akamai [13] offers such a commercial HTTP and FTP proxy service. The cost of providing such a service is significant as the service proxies application data unlike the DNS, which only participates in initial name resolution. As a result, the proxy service must be well-provisioned and massively replicated. While a commercial model for using such a proxy service exists for large commercial services (e.g. Google, Microsoft, CNN etc.) it is not obvious whether one exists for smaller private services (e.g. Alice's private FTP service). Furthermore, operating at the application-level in the middle of the network, such data proxies constrain innovation at the network edges as the middle must be upgraded to support each new application — a consequence of violating the end-to-end underpinnings of the Internet [32]. A better approach combines the *off-path*[5] aspect of DNS with the privacy preserving aspect of proxying, as described below.

5.2 Long-Term Defense

Modifying the name resolution process to identify the inquirer and to directly involve the access control policies of the private host provides access-controlled and scalable name resolution in the long run. Consider a name resolution architecture where Alice's laptop registers its IP address with a *registrar*. The registrar plays the role similar to that played by an authoritative nameservers in the DNS. In order to communicate with Alice, Bob contacts Alice's registrar. Instead of returning Alice's IP address to Bob, however, Alice's registrar *proxies* Bob's query to Alice's IP address and proxies Alice's responses to Bob[6]. The registrar conceals Alice's IP address from Bob. Alice can conduct an identification

[5] A service not along the application data path, but rather off to the side such that application data packets do not have to pass through it.

[6] Similar to SIP [33], except endhosts do *not* depend on the registrars to perform authentication. Registrars simply provide a proxied communication channel that conceals the IP addresses involved.

protocol over this proxied path, and reveal her IP address to Bob if he is granted access. While this proxy-based end-to-end name resolution architecture doesn't allow for in-network caching, it has been shown that caching the heavy-tail of DNS queries [34] for private hosts is of little value in the first place [35].

Proxy-based end-to-end secure name resolution complements emerging Internet architectures for private hosts. In [36], we propose an Internet architecture where endpoints negotiate protocol stacks, configuration parameters and coordinate the opening of NAT/firewall ports given a third party to proxy and mediate connection-setup messages; the registrar used for access-controlled name resolution can provide this proxy service for private hosts.

More research is needed, however, to better understand the privacy-performance trade-off for the DNS as well as for end-to-end secure name-resolution in the context of public and private services. For public services, it may be the case that there is no need to replace the DNS with the more heavy-weight approach as policy usually allows anyone to learn the IP address. For private services, on the other hand, it may be the case that the need for secure name resolution outweighs the latency added by the proxy mechanism.

6 Conclusions and Future Work

This paper presents an attack on mobile users that dynamically register their IP address in the DNS. The attack allows any attacker on the Internet to covertly glean private information about the user including daily commute patterns and itineraries for trips. The paper demonstrates the ease with which hundreds of thousands of vulnerable users can be monitored without their knowledge. This information could easily be logged for later use should the attacker later learn or infer the identity of the user. The root cause of the attack lies in the lack of access control for DNS name resolution combined with an increasing number of mobile Internet users. This paper suggests a short-term patch to existing DNS services that restricts the scope of the attacker, and a more long-term solution that involves end-to-end secure name resolution for private services on the Internet. The proposed solutions are preliminary in that we do not have a lot of field experience with them. One goal of this paper is to draw the attention of the community to attacks on the DNS that stem from design assumptions in the original Internet architecture that are no longer valid.

Because this attack exploits the public nature of IP addresses for private hosts, this paper additionally suggests that it is appropriate to ask whether the DNS with its focus on public services should be supplemented with a secure name resolution service for private hosts.

We certainly do not answer this question—indeed we have not fully implemented the end-to-end name resolution architecture, much less experimented with it on a broad scale and studied its security provisions. We do believe, however, that secure name resolution for private services deserves debate within the research community.

Acknowledgements

The authors would like to acknowledge the support of Jeremy Hitchcock and the DynDNS team for this study. We would also like to thank the anonymous reviewers for their helpful suggestions, including the one about interactions with the DNSSEC enumeration vulnerability.

References

1. Arends, R., Austein, R., Larson, M., Massey, D., Rose, S.: RFC 4033: DNS Security Introduction and Requirements (March 2005)
2. Nissenbaum, H.: Privacy as contextual integrity. Washington Law Review **79**(1) (February 2004) 119–158
3. Padmanabhan, V.N., Subramanian, L.: An investigation of geographic mapping techniques for internet hosts. In: Proceedings of the SIGCOMM '01, San Diego, CA (August 2001)
4. Spring, N., Mahajan, R., Anderson, T.: Quantifying the causes of path inflation. In: Proceedings of the SIGCOMM '03, Karlsruhe, Germany (August 2003)
5. The Privacy Ecosystem: IPPages – IP Address properties of your Internet Connection
6. Dynamic Network Services, Inc.: DynDNS – A free DNS service for those with dynamic IP addresses
7. Vitalwerks Internet Solutions, LLC.: No-IP – Dynamic DNS, Static DNS for Your Dynamic IP
8. Tzolkin Corporation: TZO.com – Dynamic DNS Services for your Dynamic or Static IP Address
9. Deerfield dot com: DNS2GO – Dynamic DNS Services for your IP Address
10. CanWeb Internet Services Ltd.: DynIP – Dynamic DNS Service
11. GravityFree: DtDNS – Your Complete DNS Solution
12. Dynamic Network Services, Inc.: DynDNS: Third Party Clients – keep IP address current, use with all DNS services
13. Akamai Technologies, Inc.: Akamai: How it works
14. Dynamic Network Services, Inc.: (2006) Private communications.
15. Kanellos, M.: Notebooks pass desktops in U.S. retail (February 2006) ZDNet News.
16. Mockapetris, P., Dunlap, K.: Development of the domain name system. In: Proceedings of the SIGCOMM '88, Stanford, CA (August 1988)
17. Park, K., Pai, V.S., Peterson, L., Wang, Z.: CoDNS: Improving DNS performance and reliability via cooperative lookups. In: Proceedings of the Sixth Symposium on Operating Systems Design and Implementation (OSDI 2004), San Francisco, CA (December 2004)
18. Ramasubramanian, V., Sirer, E.G.: CoDoNS: The Design and Implementation of a Next Generation Name Service for the Internet. In: Proceedings of SIGCOMM'04, Portland, OR (August 2004)
19. Poole, L., Pai, V.S.: ConfiDNS: Leveraging Scale and History to Improve DNS Security. In: Proceedings of WORLDS'06, Seattle, WA (November 2006)
20. Gabrilovich, E., Gontmakher, A.: The homograph attack. Communications of the ACM **45**(2) (February 2002) 128
21. Walfish, M., Stribling, J., Krohn, M., Balakrishnan, H., Morris, R., , Shenker, S.: Middleboxes no longer considered harmful. In: Proceedings of the OSDI '04, San Francisco, CA (December 2004)

22. Ford, B., Strauss, J., Lesniewski-Laas, C., Rhea, S., Kaashoek, F., Morris, R.: Persistent personal names for globally connected mobile devices. In: Proceedings of the OSDI '06, Seattle, WA (November 2004)
23. Perkowitz, M., Doorenbos, R.B., Etzioni, O., Weld, D.S.: Learning to understand information on the internet: An example-based approach. Journal of Intelligent Information Systems **8**(2) (2004) 133–153
24. Gordon Lyon: Nmap Security Scanner
25. Dagon, D., Gu, G., Zou, C., Grizzard, J., Dwivedi, S., Lee, W., Lipton, R.: A Taxonomy of Botnets. In: Proceedings of CAIDA DNS-OARC Workshop, San Jose, CA (July 2005)
26. Gueye, B., Ziviani, A., Crovella, M., Fdida, S.: Constraint-based geolocation of internet hosts. IEEE/ACM Transactions on Networking **14**(6) (December 2006) 1219–1232
27. Wong, B., Stoyanov, I., Sirer, E.G.: Octant: A comprehensive framework for the geolocalization of internet hosts. In: Proceedings of the NSDI '07, Cambridge, MA (May 2007)
28. Srisuresh, P., Egevang, K.: RFC 3022: Traditional IP Network Address Translator (Traditional NAT) (January 2001)
29. US-CERT: The Continuing Denial of Service Threat Posed by DNS Recursion (v2.0)
30. Bellare, M., Desai, A., Jokipii, E., Rogaway, P.: A concrete security treatment of symmetric encryption. FOCS **00** (1997) 394
31. Boneh, D., Gentry, C., Waters, B.: Collusion resistant broadcast encryption with short ciphertexts and private keys. In: Proceedings of the CRYPTO '05, Santa Barbara, CA (August 2005)
32. Saltzer, J.H., Reed, D., Clark, D.D.: End-to-end arguments in system design. ACM Transactions on Computer Systems **2**(4) (1984) 277–288
33. Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., Schooler, E.: RFC 3261: SIP Session Initiation Protocol (June 2002)
34. Jung, J., Sit, E., Balakrishnan, H., Morris, R.: DNS Performance and Effectiveness of Caching. In: Proceedings of SIGCOMM Internet Measurement Workshop, San Francisco, CA (November 2001)
35. Breslau, L., Cao, P., Fan, L., Phillips, G., Shenker, S.: Web Caching and Zipf-like Distributions: Evidence and Implications. In: Proceedings of INFOCOM'99, New York, NY (March 1999) 126–134
36. Guha, S., Francis, P.: An End-Middle-End Approach to Connection Establishment. In: Proceedings of SIGCOMM'07, Kyoto, Japan (August 2007)