Beware of IPs in Sheep's Clothing: Measurement and Disclosure of IP Spoofing Vulnerabilities

Alden Douglas Hilton
Brigham Young University

Follow this and additional works at: https://scholarsarchive.byu.edu/etd

Part of the Computer Sciences Commons

BYU ScholarsArchive Citation
https://scholarsarchive.byu.edu/etd/9280

This Thesis is brought to you for free and open access by BYU ScholarsArchive. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of BYU ScholarsArchive. For more information, please contact ellen_amatangelo@byu.edu.
Beware of IPs in Sheep’s Clothing: Measurement and Disclosure of IP Spoofing Vulnerabilities

Alden Douglas Hilton

A thesis submitted to the faculty of Brigham Young University in partial fulfillment of the requirements for the degree of Master of Science

Casey Deccio, Chair
Kent Seamons
Daniel Zappala

Department of Computer Science
Brigham Young University

Copyright © 2021 Alden Douglas Hilton
All Rights Reserved
ABSTRACT

Beware of IPs in Sheep’s Clothing: Measurement and Disclosure of IP Spoofing Vulnerabilities

Alden Douglas Hilton
Department of Computer Science, BYU
Master of Science

Networks not employing destination-side source address validation (DSAV) expose themselves to a class of pernicious attacks which could be prevented by filtering inbound traffic purporting to originate from within the network. In this work, we survey the pervasiveness of networks vulnerable to infiltration using spoofed addresses internal to the network. We issue recursive Domain Name System (DNS) queries to a large set of known DNS servers world-wide using various spoofed-source addresses. In late 2019, we found that 49% of the autonomous systems we tested lacked DSAV. After a large-scale notification campaign run in late 2020, we repeated our measurements in early 2021 and found that 44% of ASes lacked DSAV—though importantly, as this is an observational study, we cannot conclude causality. As case studies illustrating the dangers of a lack of DSAV, we measure susceptibility of DNS resolvers to cache poisoning attacks and the NXNS attack, two attacks whose attack surface is significantly reduced when DSAV in place. We discover 309K resolvers vulnerable to the NXNS attack and 4K resolvers vulnerable to cache poisoning attacks, 70% and 59% of which would have been protected had DSAV been in place.

Keywords: IP spoofing, DNS security, large-scale vulnerability disclosure, network measurement
I first want to acknowledge my wife, Elizabeth. Thank you for your unwavering confidence in me. Thank you for your support, encouragement, and for helping me realize that writing isn’t scary and can even be fun. Thanks go to my parents and siblings as well for their love and support.

I’d also like to acknowledge Dr. Deccio. He was everything I could have hoped for in an advisor: dependable, encouraging, knowledgeable, and compassionate. Thank you for all the time and energy you have invested into helping me succeed.

Thank you to all my lab members, past and present. Thank you all for being good friends and for making the lab a welcoming environment. Trevin especially, thank for being a great partner on all the projects we’ve done together the past few years.

Finally, I gratefully acknowledge the Comcast Innovation Fund for their support of the work that produced this material, DNS-OARC for providing access to the 2019 and 2020 DITL collections, CAIDA for providing access to their infrastructure to carry out our experiment, and the WhoisXMLAPI for providing access to their API. Additionally, I thank Michael Briggs, Trevin Avery, and Robert Richardson for their contributions on the conference paper that led to this work; Robert Richardson for continued his statistical expertise and guidance; and Kent Seamons and Daniel Zappala for their helpful reviews and feedback.
Table of Contents

Title Page i
Abstract ii
Acknowledgments iii
Table of Contents iv
List of Figures vi
List of Tables vii

1 In Preparation: Beware of IPs in Sheep’s Clothing: Measurement and Disclosure of IP Spoofing Vulnerabilities 1

1.I Introduction ......................................................... 2

1.II Background ....................................................... 2

1.II.A Spoofing prevention techniques ................................. 2

1.II.B The Domain Name System .................................... 3

1.III Related Work ..................................................... 4

1.III.A Source Address Validation ................................. 4

1.III.B Notifications .................................................. 4

1.III.C NXNS Attack .................................................. 5

1.IV Measurement Methodology ....................................... 5

1.IV.A DSAV ......................................................... 5

1.IV.B NXNS Measurement ......................................... 6

1.V Methodology Considerations ...................................... 7

1.V.A Comprehensiveness ............................................. 7

1.V.B QNAME Minimization .......................................... 7

1.V.C Human Intervention ............................................ 7

1.V.D Experimental Design .......................................... 7

1.V.E Ethical Considerations ........................................ 8

1.VI DSAV Experiment Results ....................................... 8

1.VI.A Autonomous System Degree .................................. 9
List of Figures

1.1 Distribution of ASes according to (a) number of neighbors and (b) number of prefixes announced. Each bar is subdivided by the number of ASes with or without DSAV. The blue dashed line shows the percentage of each bar that lack DSAV and corresponds to the blue y-scale on the right. .............................. 9

1.2 Recursive resolvers grouped by number of records requested when resolving the NS records. ................................................................. 15
List of Tables

1.I DSAV results for the 10 countries associated with the largest number of ASes. 9

1.II The number of IP addresses or ASes for which at least one spoofed-source reached its target ("Category-Inclusive") or for which a spoofed-source category was the only one to reached its target ("Category-Exclusive"). Percentages represent the fraction of reachable targets. .......................................................... 11

1.III The autonomous systems found to be vulnerable in Round1, broken down by various categories. For each category, the state of the ASes in Round2 is shown (i.e., patched or vulnerable). The percentages in each row are based off the the "Total" column. The percentages might not add to 100% as there were some ASes present in Round1 that were not present in Round2 which would be best classified as "unknown," though we do not explicitly show this category. ................. 13
Chapter 1

In Preparation: Beware of IPs in Sheep’s Clothing: Measurement and Disclosure of IP Spoofing Vulnerabilities

This manuscript has not yet been accepted for publication.
Beware of IPs in Sheep’s Clothing: Measurement and Disclosure of IP Spoofing Vulnerabilities

Alden Hilton, Joel Hirschmann, and Casey Deccio

Abstract—Networks not employing destination-side source address validation (DSA V) expose themselves to a class of pernicious attacks which could be prevented by filtering inbound traffic purporting to originate from within the network. In this work, we survey the pervasiveness of networks vulnerable to infiltration using spoofed addresses internal to the network. We issue recursive Domain Name System (DNS) queries to a large set of known DNS servers world-wide using various spoofed-source addresses. In late 2019, we found that 49% of the autonomous systems we tested lacked DSA V. After a large-scale notification campaign run in late 2020, we repeated our measurements in early 2021 and found that 44% of the ASes lacked DSA V—though importantly, as this is an observational study, we cannot conclude causality. As case studies illustrating the dangers of a lack of DSA V, we measure susceptibility of DNS resolvers to cache poisoning attacks and the NXNS attack, two attacks whose attack surface is significantly reduced when DSA V is in place. We discover 309K resolvers vulnerable to the NXNS attack and 4K resolvers vulnerable to cache poisoning attacks, 70% and 59% of which would have been protected had DSA V been in place.

I. INTRODUCTION

SP oofed Internet traffic is an unfortunate reality of today’s world—and will likely continue to be until all networks block spoofed traffic from leaving their networks (see section II-A). Spoofed traffic facilitates attacks such as the largest-recorded distributed denial-of-service attack to date—the 2.3 Tbps attack reportedly mitigated by Amazon Web Services in early 2020 [1]. Furthermore, spoofing enables access to protected network resources that rely on IP-based controls, such as closed DNS resolvers. We show that such access could allow remote attackers to discover vulnerabilities with otherwise inaccessible systems. Fortunately, while not all spoofed traffic can be reliably identified and blocked once it reaches its destination network, some of it can—a practice we refer to as destination-side source address validation, or DSA V. This practice protects controlled network resources and lowers their potential for use in distributed denial of service attacks. While the effects of spoofing can be mitigated in some cases with protocols that include some form of identity check (e.g., TCP), in other cases, this infiltration creates a vulnerability that can be exploited for surveillance or compromise.

This paper has three main contributions:

1) A longitudinal, large-scale study of the deployment of DSA V.

2) An additional perspective exploring the effectiveness of large-scale vulnerability disclosure.

3) Several case studies that illustrate how a lack of DSA V could be exploited to discover vulnerabilities with otherwise inaccessible systems.

In late 2019, we surveyed 54K networks for DSA V, using methodology that was effective in its detection, yet harmless. We sent spoofed-source packets to these networks, each packet having a source appearing to originate from the network for which it was destined. We observed that about 49% of the networks we surveyed lacked DSA V, allowing our spoofed-source packets into their network. In late 2020, we ran a large-scale notification campaign to alert the administrators of the vulnerable networks of our findings. Then in early 2021, we repeated our analysis, this time surveying 61K networks. For this second survey, we found that the fraction of networks lacking DSA V was lower—approximately 44%. However, we emphasize that as we did not design our notification campaign as a controlled experiment, we cannot definitively attribute the cause of this improvement.

Even more important than the fact that a network can be infiltrated is the impact of the unauthorized access—how it might be exploited to survey or compromise internal systems. Thus, as case studies illustrating the importance of DSA V, we measure susceptibility of recursive DNS resolvers to two types of attacks: DNS cache poisoning and the NXNS attack, a recently discovered denial-of-service vector [3]. With both of these attacks, the only requirement for exploitation is that a DNS query is received by the target as if from a trusted source. The address the query response is returned to is irrelevant, meaning an attacker is free to choose source addresses that are potentially trusted by closed resolvers. DSA V can be used to block these addresses and would prevent closed resolvers from being abused in cache poisoning, NXNS, or other attacks with similar requirements. We discover 399K resolvers vulnerable to the NXNS attack, 70% of which would have been protected had DSA V been in place. We also identify nearly 4,000 DNS servers that were vulnerable to cache poisoning attacks, 59% of which would have been protected had DSA V been in place.

II. BACKGROUND

A. Spoofing prevention techniques

There are two sides to spoofing prevention, which we refer to as origin-side source address validation (OSAV) and

1In the conference paper this work is based off of [2], we listed this number as 62K. In that paper, we considered IPv4 and IPv6 networks separately; in this work, we simply aggregate the data by autonomous system number.
destination-side source address validation (DSA V). OSA V deals with preventing spoofed traffic from leaving its network of origin; DSA V deals with preventing it from entering its destination network.

Best Current Practice (BCP) 38 [4], proposed as a mechanism to prevent denial-of-service attacks that rely on IP address spoofing, is the primary mechanism for performing OSA V. The basic idea is for routers to restrict traffic originating from their networks to the address space allocated for them, thus preventing spoofed traffic from ever leaving its network of origin. This practice prevents networks from become an “unwitting source of an attack” [4]. However, it is important to note that it does not protect the networks employing it from attack; if even one network lacking OSA V exists, traffic with spoofed IP addresses will continue to be a threat.

Spoofed traffic is more difficult to identify once it reaches its destination network—at that point, no path history is known, including its origin. Nonetheless, while not all spoofed traffic can be identified and filtered, some of it can, such as traffic from special purpose IP addresses [5] and the prefixes announced by the destination network itself. There are several mechanisms for accomplishing this, outlined in BCP 84 [6], [7], including access control lists (ACLs) and unicast Reverse Path Forwarding (uRPF). ACLs are manually maintained lists of addresses that are acceptable (or alternatively, unacceptable) and should be allowed (or blocked). ACLs, though perhaps the easiest toconceptualize, can be difficult, or even infeasible, to maintain at scale.

There are several variants of uRPF, each with their own set of tradeoffs, described in [7]. Strict uRPF means to only accept a packet if a route from the destination back to the source exists and the path back to the source would use the same interface the packet was received on. Strict uRPF is often avoided, as it will drop legitimate traffic in the case of asymmetric routing. Loose uRPF only considers the first of the two conditions listed above: that a route from destination back to the source exists. Loose uRPF is seen as “not desirable” [7], as it may miss a significant portion of spoofed traffic, though it does not have the false-positive problem of strict uRPF. Loose uRPF partially satisfies our tests (see section IV-A2), blocking traffic from special use addresses [5], but may not be sufficient to block traffic appearing to originate from one’s own prefixes. There are variants of uRPF that attempt to strike a balance between strict and loose, the latest being Enhanced Feasible-Path uRPF, or EFP-uRPF. EFP-uRPF specifies that if the path back to the source uses an interface that would be used for at least one of the prefixes announced by the originating autonomous system—regardless of whether the source address is in that specific prefix—the packet is accepted.

The choice of specific technique for performing DSA V is very dependent on the network configuration: there is no one-size-fits-all. One case in particular represents an open problem for which no ideal DSA V solution exists. Specifically, some networks operate multiple “island” networks under the same autonomous system number, with no means of communicating with each other other than across the Internet at large. In this case, the AS must allow incoming traffic from its own prefixes; otherwise, the islands would not be able to communicate with each other. Although this scenario represents a legitimate reason for an autonomous system to accept its own prefixes, it nonetheless leaves open the door for spoofed traffic to enter the network. One possible solution is to split the islands into separate autonomous systems, though there is considerable overhead to doing so. For the purpose of this analysis, we consider networks such as this as vulnerable to spoofing attacks.

B. The Domain Name System

The domain name system (DNS) is responsible for translating human-readable domain names to the IP addresses of the servers hosting those domains (e.g., example.com → 192.0.2.10). The DNS is organized hierarchically into zones. Queries for names within a given zone are directed to servers authoritative for that zone, which provide the name-to-IP-address mapping, if there is one. Zone administrators can delegate portions of their namespace, such that they form “children” zones, hosted by other authoritative servers. When a nameserver receives a query for a name in the delegated namespace, it returns one or more NS records, one for each nameserver that is authoritative for the delegated space. Recursive resolvers are servers responsible for navigating this hierarchical namespace; they receive queries from clients and issuing queries to the appropriate authoritative servers—following the NS records—resolving domain names on behalf of end users.

NS delegations come in the form of domain names, such as ns1.example.com. When these names themselves are subdomains of the delegated namespace, then the authoritative servers include in their referral response the IP addresses of the nameservers, known as glue records.

If the glue records are not included, the recursive resolver must initiate a new resolution process to resolve the IP address(es) of the nameserver(s) before it can carry on with the original query. Frequently, multiple nameservers exist for the same zone, e.g., to provide redundancy and extra resilience.

In early 2020, Afek et al. [3] disclosed the NXNS attack, an attack technique that exploits this aspect of the DNS. Prior to their disclosure, many DNS resolver implementations would proactively resolve all NS addresses, even though in theory only 1 is needed. In typical cases, this is not an issue. However, an attacker can exploit this by configuring a zone with as many NS records as possible without including any glue records; in practice, Afek et al. [3] found this number to be 135. Because of factors such as resolvers attempting to obtain both the IPv4 and IPv6 addresses of each NS record in the referral, authoritative servers using DNSSEC to prove the nonexistence of those records, and TCP control packets, the inclusion of 135 fake NS records allows an attacker to force a resolver and an authoritative server of his/her choice to exchange 3,240 packets, degrading the performance of both the recursive resolver and authoritative server. Thus, the NXNS attack allows attackers to achieve an application factor of 1,620 (in terms of packets sent). Afek et al. [3] outline several variants of the attack which collectively can be used...
to overwhelm both recursive resolvers and authoritative servers alike. Patches have been released for most major DNS resolver implementations including BIND, Unbound, Knot Resolver, and PowerDNS Recursor [8]–[11]. The basic idea behind these patches is to limit the number of NS records resolved to a reasonable amount (e.g., 5). Notably, Microsoft did not release a patch for Windows DNS [12].

In this work, we also consider a second attack on the DNS, known as cache poisoning. In this attack, would-be cache poisoners impersonate authoritative servers and force recursive resolvers to accept false IP-to-name mappings, facilitating man-in-the-middle attacks. One of the most salient cache poisoning techniques was disclosed in 2008 [13]. The primary defense against this attack is source port randomization, a practice now considered standard [14].

While both of these attacks can be prevented by applying the appropriate patches and configurations to the resolvers, DSAV can be used to drastically reduce the attack surface of vulnerable resolvers. Best practice is to “close” DNS resolvers, or configure them to accept queries from only a trusted set of clients, such as clients from resolvers’ own networks [15]. Both of these attacks depend on an attacker’s ability to query the resolvers targeted, closed resolvers are more difficult to reach, if the networks hosting the resolvers employ DSAV. If the networks lack DSAV, the attackers are able to spoof addresses from within the same network as the resolvers and bypass the IP-based access control. Thus, in this work, we consider both vulnerability at the resolver-level as well as vulnerability at the network-level.

III. RELATED WORK

A. Source Address Validation

The most prominent example of research relating to the employment of BCP 38 is that of the Spoofer Project [16]. The Spoofer Project employs custom build software that volunteers install in their network. This software attempts to send spoofed packets out of the network directed to servers run by the project. If these packets reach the servers, the project is able to infer that BCP 38 is not in place on the host’s network. Researchers recently reported that at least a quarter of the autonomous systems tested do not prevent spoofed traffic from leaving their networks [17].

The Spoofer Project is also able to test for DSAV, though in a very limited sense. If the custom software is installed on a machine with an Internet facing address (i.e., not behind a NAT), the project can direct spoofed traffic towards the clients and determine if it arrives at the host. However, the requirement to not be installed behind a NAT severely limits the scope of the results pertaining to DSAV. That said, in 2019, the researchers reported that at least two thirds of the autonomous systems (ASes) tested did not employ DSAV [17].

In work performed concurrently with (and independently of) our late-2019 measurement, Korczyński, et al. [18], tested networks for source address validation using a methodology similar to ours. They issued queries to every IP address in the IPv4 space, in each case spoofing the source IP address just higher than the selected destination. In contrast, our objectives focused on exploring the variety of ways in which a lack of DSAV might be discovered. Our methodology differs primarily in that 1) we use several diverse, spoofed-source IP addresses for each destination, rather than only the next sequential IP address; 2) the selection of target IP addresses used in our research consist of those that generate query activity at the root servers; and 3) our study includes both IPv4 and IPv6. Our results show that there are advantages to both the current methodology and that used by Korczyński, et al. In particular, the sheer breadth of the IPv4 address space scanned by Korczyński, et al., resulted in more overall hits than our targeted approach. The diversity of spoofed sources used in our experiment uncovered resolvers—and ASes—that would not have otherwise been identified using only a same-prefix source. Nonetheless, the overall percentage of measured ASes with reachable IPv4 targets is consistent between the two studies, within 1%: 48.78% vs. 49.34%. Finally, in the current paper, we extend our analysis to survey and identify vulnerabilities of internal systems, as case studies of our methodology.

B. Notifications

There is a growing body of literature addressing the issue of carrying out effective large-scale vulnerability notifications. Many different challenges have been highlighted, such as finding the best communication channel, finding the correct points of contact, and establishing trust in the notification itself.

Communication channels that have been explored include emails directed to contacts found through Whois [19]–[27], emails directed to default email addresses [19], [21], [22], [25], [26], and notifications disseminated through regional CERTs [22], [24], [28]. Note that this list of communication channels is not comprehensive; other methods have been used but aren’t listed here as they are application-specific and don’t fit our use case (e.g., notifications sent through Google Search Console [27]). Of these options, emailing Whois addresses appears to be most effective. This is according to Li et al. who show that directly contacting resource owners is more effective than contacting regional CERTs [24] and Cetin et al. [25] found that over 80% of emails to default addresses bounce.

Many different variables have been explored for establishing trust in emails, such as the level of detail included, the email format (e.g., HTML vs plain-text), and the apparent reputation of the sender’s address. We detail some relevant findings from these studies in sections VII-A2 and VII-A3, where we describe our notification design.

The general consensus of all these studies is that large-scale notifications have a modest, yet statistically significant effect of remediation. Given this finding, it is the hope that with further research into notification design, this effectiveness can be increased. In this study, we explore one dimension of notification design that as far as we are aware, has yet to be explored—that of emailing multiple points of contact per organization. Doing so increases the chances of the email being received, but at the same time, increases the potential burden on the recipients.
C. NXNS Attack

As far as we are aware, only one study has looked at this vulnerability [3]; no follow-up work has been completed to date. In their work, they outlined the threat model and detailed tests performed to validate the model. In contrast, in our study, we measure the prevalence of resolvers still vulnerable to the attack roughly a year after its disclosure.

IV. MEASUREMENT METHODOLOGY

A high-level outline of our experiment is as follows:

- December 2019: Round 1 DSAV and source port randomization measurements
- October 2020: Email notifications
- March 2021: Round 2 DSAV, source port randomization, and NXNS measurements

The methodology for carrying out the two measurements is described in the remainder of this section. The methodology for the notification campaign is described in section VII.

A. DSAV

We ran two experiments testing for DSAV: the first between November and December 2019 and the second in March 2021. We will refer to the first as Round1 and the second as Round2. The basic method for detecting a lack of DSAV is the same for both experiments. We sent DNS queries with spoofed source addresses to known DNS resolvers worldwide. Each resolver was sent unique, dynamically-generated queries for domains under our control; as such, if we observed a corresponding query at our authoritative server, we were able to determine that our spoofed queries successfully infiltrated the network.

1) DNS Servers (Targets): For both experiments, the set of resolvers measured were found in the “Day in the Life” collection sponsored by the DNS Operations, Analysis, and Research Center (OARC) [29]. The DITL consists of traffic collected at participating root servers over a 48-hour period. As in general, at some point in the resolution process, all resolvers must query the root servers, the DITL provides a rich, highly-representative set of recursive resolvers. Round1 used DITL data collected in April 2019; Round2 used data collected May 2020. The Round1 data set consists of 11,204,889 IPv4 addresses and 784,777 IPv6 addresses, corresponding to 54,185 autonomous systems. The Round2 data set consists of 15,646,911 IPv4 addresses and 1,569,667 IPv6 addresses, corresponding to 60,618 autonomous systems. The Round2 data set consists of 15,646,911 IPv4 addresses and 1,569,667 IPv6 addresses, corresponding to 60,618 autonomous systems. 4,371,182 (36%) IP addresses and 49,854 (92%) ASes present in Round1 were also present in Round2. Previous research has shown that there is churn in IP addresses of DNS resolvers—specifically, open resolvers—over time [30]; thus, it is unsurprising that only 36% of the resolvers from Round1 were still present in Round2.

2) Spoofed Sources: Each destination received several queries, enumerating the different type of source addresses that should be blocked at the network border. The following categories of spoofed sources were used:

- **Same-ASN**: randomly selected IP addresses from prefixes advertised by the same autonomous system (as the target).
- **Same-prefix**: a random IP address from the same /24 (IPv4) or /64 (IPv6) prefix as the target.
- **Destination-as-source**: The destination IP address itself.
- **Bogon addresses**: Addresses from reserved address space (e.g., [5]). The specific addresses used are listed later in this section.

The Same-ASN addresses and the Bogon addresses together form a pool of addresses that are used for each destination in the autonomous system. When generating source addresses for each autonomous system, we considered IPv4 and IPv6 separately. For example, if a given AS had both IPv4 and IPv6 addresses in the DITL, two separate pools would be generated, one with IPv4 addresses and the other with IPv6 addresses.

The method for generating the Same-ASN addresses varies slightly between the two measurements. For Round1, to generate the IPv4 addresses, we divided all the IP address space originating from an AS into 24-bit prefixes. From each /24 prefix, we selected, at random, a single IP address. In both cases, the first and the last IP addresses were excluded from selection because of their reserved status in a /24 subnet. The resulting IP addresses formed the set of other-prefix addresses for any target IP address announced by that AS. Because some ASes had a prohibitively large number of /24 prefixes, we limited our selection to 97 prefixes\(^2\). For the IPv6 addresses, we used a more targeted methodology to identify more realistic client addresses, rather than blindly probing the sparsely-populated IPv6 address space. The addresses were drawn at random from the /64s of addresses observed in an IPv6 hit list [31]. One address per /64 was selected, limiting the selection to the first 100 addresses within the /64, excluding the first two. If more than 97 /64s were present in the hit list, a random sample was selected. This means that for autonomous systems not present in the hit list, no Same-ASN addresses were generated, a limitation that affected roughly 15% of the ASes with IPv6 address space. We had thought we had accounted for these cases, but due to an error in our code, this limitation was not corrected until Round2.

For Round2, 20 Same-ASN addresses were generated for each autonomous system. Notably, this is significantly less than the first experiment, where up to 97 addresses were generated for each. We lowered the number of addresses used to reduce the traffic generated as a result of the experiment and to facilitate a shorter experiment duration (see IV-A5). To generate the Same-ASN IPv4 addresses, addresses were selected at random from the entire space announced by the AS. For the IPv6 addresses, the Same-ASN addresses were first drawn at random from the /64s of addresses observed in an IPv6 hit list [31]. One address per /64 was selected (again, limiting the selection to the first 100 addresses in the /64, excluding the first two). If more than 20 /64s were present in the hit list, a random sample was selected. If less than 20 /64s were present, the remaining addresses were drawn at random from the entire space announced by the AS. This was

---

\(^2\)The number 97 was chosen when we had three other spoofed-source categories in mind, such that the maximum number of source IP addresses we would use for a given target would be an even 100. However, we ended up adding another source IP address to our experiment, such that at most 101 sources would be used to query a given target.
done by dividing the space in /64 prefixes, selecting /64s at random, selecting random addresses from the /64s (limiting the selection within the /64s to the first 100 addresses in the /64, excluding the first two), and repeating the process until the 20 address had been selected.

Note that for both rounds, this methodology does not guarantee that the Same-ASN addresses will not be in the same /24 or /64 of any of the destination addresses. However, when processing the results, for any given successful source-destination pair, if the source and destination are both within the same /24 or /64, we classify the resulting hit as Same-prefix.

We also generate a random Same-prefix address on a per-destination basis. For each destination IPv4 address, we select an address within its /24, excluding the first and last addresses. For the Round1 IPv6 addresses, we selected addresses within the first 100 addresses of its /64, excluding the first two addresses. For the Round2 IPv6 addresses, we expanded the selection to include the first 200 addresses of the /64, again excluding the first two addresses. The Same-prefix queries guaranteed that each destination received at least one query appearing to be from the same /24 or /64 as itself.

The bogon addresses used vary slightly between the two experiments. For Round1 IPv4 addresses, we used 192.168.0.10 (private) and 127.0.0.1 (loopback). For the IPv6 addresses, we used fc00::10 (unique local) and ::1 (loopback). For Round2, we used these same addresses with just one exception, substituting fd00::10 for fc00::10. The use of fc00::100 for Round1 was a mistake, albeit a mild one. Though it does represent a “valid” bogon address—falling under undefined space—it lacks the L-bit (i.e., locally allocated) and is not what we had intended, hence the substitution of fd00::10 for Round2. Additionally, we added 192.0.2.10 and 2001:db8::10 (documentation) for Round2.

In summary, for Round1, each destination received queries from up to 101 addresses: 97 Same-ASN, 2 Bogon addresses, the Same-prefix query, and the Destination-as-source query. For Round2, each destination received queries from up to 25 addresses: 20 Same-ASN, 3 Bogon addresses, the Same-prefix query, and the Destination-as-source query.

Autonomous system lookups for the 2019 dataset were performed using the RIPEstat API. For the second experiment, we transitioned to using the Routeviews prefix to AS mappings dataset published by CAIDA [32] as it provided a more efficient way of performing the lookups.

3) Query Names: For both experiments, we encoded the spoofed source, the query destination, and a timestamp into the qname requested. This encoding scheme ensured the uniqueness of each query as well as allowed us to determine definitively which spoofed source addresses reach their target destinations. We encoded the query names used in our experiment according to the following template: ts_src_dst_kw.dns-lab.org, where ts is the timestamp the query was sent, src is the spoofed-source IP address, dst is the target IP address, and kw is a keyword associated with the current experiment.

For Round1, we configured our servers to return NXDOMAIN for all queries associated with the experiment. This was an oversight which caused us to miss queries from resolvers configured to use QNAME minimization (see section V-B). We corrected this oversight for the Round2 by configuring our authoritative server to return NOERROR for all queries under the domain used for the experiment.

4) Follow-Up Queries: For both experiments, a process monitored traffic at our authoritative servers. This process would initiate a set of follow-up queries in real time the first—and only the first—time a query associated with a given destination address arrived at our servers. The follow queries included:

- IPv4- and IPv6-only: two sets of 10 queries that elicited queries over IPv4 and IPv6, respectively.
- Open resolver: non-spoofed-source query.
- TCP: a query that elicited an additional query over TCP to our servers.

The IPv4- and IPv6-only queries were elicited by using query names in DNS domains that were only delegated to IPv4 addresses or IPv6 addresses, respectively. The TCP query was elicited by issuing a query for which the authoritative server would always respond with the truncation (TC) bit set. A truncated response causes the recursive resolver to issue its query again over TCP [33].

For the second experiment, we added a query that tested for susceptibility to the NXNS attack. For details on this follow-up test, see section IV-B.

5) Query Execution: The second experiment was completed much faster than the first. We intended to complete Round1 in 4 weeks, though it actually took 7 due to unexpected interruptions (such as a power outage). We designed Round2 to require a week to complete. This quicker run time was made possible by lowering the number of spoofed sources per destination. Round1 ran from November 6 and December 27, 2019. Round2 ran from March 11 to March 19, 2021. Both measurements were initiated from a network that lacked OSAV (BCP 38 [4]). The absence of OSAV in our client’s network was a requirement for effectively testing OSAV. For both measurements, we engineered it such that the initial queries destined for each target address were distributed uniformly throughout the entire week, easing any potential burden our experiment might have posed on any one address.

B. NXNS Measurement

We next engineered a custom authoritative server that synthesized NS records based on the destination address that was encoded into the query, allowing us to track how many NS records each resolver requested. We configured this server to return 20 NS records. We validated this server’s ability to identify unpatched resolvers by installing multiple versions of common DNS resolver implementations; for each implementation we install one version that was released prior to the NXNS attack disclosure and one version that was released afterwards. The objective here was to observe in a controlled environment how each resolver implementation responds to the NXNS attack. Specifically, we installed the following safe resolvers: BIND 9.16.8, Unbound 1.12.0, Knot Resolver 5.2.0, and PDNS Recursor 4.3.1; we also installed the following...
**vulnerable resolvers:** BIND 9.12.1; Unbound 1.9.4, Windows DNS on Windows Server 2019, Knot Resolver 5.1.0, and PDNS Recursor 4.3.0. Note that we only tested one version of Windows DNS as Windows did not release a patch for this attack [12].

We then tested these resolvers with our custom server. Of the safe resolvers: BIND requested the NS records in sets of 5, requesting both the A and AAAA records simultaneously (a total of 10 records). Unbound requested a random subset of 12 of the 40 possible A/AAAA records. Knot resolver had the same behavior as BIND. PDNS recursor requested 5 records at a time—and only the A records. Of the vulnerable resolvers, BIND, Unbound, and Knot requested all 40 A/AAAA records simultaneously. Windows DNS and PDNS recursor requested all 20 A records simultaneously, but did not request the AAAA records. Note that all resolvers were installed and tested on a machine without IPv6 connectivity. With these tests, we verified that it is possible to observe susceptibility to the NXNS attack from an authoritative server’s perspective.

The test is relatively harmless; at most, 40 queries need to be requested, a nominal amount. The zone used for the experiment is not configured with DNSSEC, which plays a large role in the theoretical amplification factor of the NXNS attack. This configuration, combined with the relatively low number of NS records returned, results in our test only eliciting roughly 1% of the number of queries that theoretically could be elicited by the NXNS attack.

**V. METHODOLOGY CONSIDERATIONS**

Several issues merit our discussion, including the comprehensiveness of our data, QNAME minimization, and potential human intervention.

**A. Comprehensiveness**

DITL data is generally considered representative and has been used in numerous studies. However, it should not be considered to be comprehensive. It is possible that some resolvers did not need to query the root during the collection period, either because they relied on cached records or due to local instances of the root zone [34]. Additionally, it is likely that IP addresses that did represent resolvers at the time of the DITL capture no longer did at the time of our measurements (e.g., due to IP churn). As an alternative to using the DITL, we could have conducted an Internet-wide, comprehensive scan, similar to the work of [18]. However, we elected to use the DITL to avoid the volume of traffic required for a comprehensive scan. Additionally, a comprehensive scan of the IPv6 address space is not feasible. We consider the DITL sufficiently representative for our purposes.

**B. QNAME Minimization**

In an effort to preserve privacy, some modern DNS resolvers avoid sending authoritative servers the full query name (QNAME) and instead only ask for the next unknown label. This is known as QNAME Minimization [35]. In the case of our experiment, before asking for the full QNAME (i.e., ts.src.dst.kw.dns-lab.org), a resolver using QNAME minimization would ask for kw.dns-lab.org, then dst.kw.dns-lab.org, etc. For Round 1, our authoritative servers returned an NXDOMAIN response code in response to any queries related to our experiment. For at least some resolver implementations that implement QNAME minimization, an NXDOMAIN response halts further queries associated with the QNAME. This is because a NXDOMAIN for a given domain name implies that no subdomains (i.e., with additional labels on the left) exist [36].

For Round 1, we observed QNAME-minimized queries from 17,981 (0.16%) of the IP addresses that we targeted with our initial reachability query. For 9,898 (55%) of these IP addresses, we never received a query with the full QNAME. Most notably, they did not include the label with the encoded source address. With no way to identify the source IP address that we used to reach these 9,898 targets, we excluded them from the total number of reachable targets. Nonetheless, we still learned something about the networks from which the QNAME-minimized queries originated. We observed that DNS clients from 2,081 ASNs queried for kw.dns-lab.org (the product of QNAME minimization). Of those, 2,041 (98%) were identified as lacking DSAV, in that we observed queries from these same QNAME-minimizing resolvers or from other (i.e., non-QNAME-minimizing) resolvers. Thus, QNAME minimization did not diminish our DSAV measurement results. Nonetheless, we corrected this oversight for Round 2 by returning a NOERROR response code for all qnames under the experimental domain.

**C. Human Intervention**

To account for potential false positives due to human intervention (e.g., curious analysts resolving the qnames manually), we only considered queries from resolvers that responded to the initial query within 10 seconds of when we sent it. For Round 1, queries for an additional 3,444 IPv4 addresses and 70 IPv6 addresses had a life-time that exceeded our threshold, representing less than 0.1% of addresses, for both protocols. These addresses correspond to 433 ASes; for all but 19 of these, we were able to infer the lack of DSAV through the presence of other resolvers which did query our servers within the 10 second window. For Round 2, 2,595 IPv4 and 85 IPv6 addresses were excluded by this filter, corresponding to 509 ASes. For all but 25 of these, we were able to infer lack of DSAV through the presence of other resolvers.

**D. Experimental Design**

Because we did not design our notification campaign as a controlled experiment (e.g., with randomly assigned treatment groups), we cannot definitively attribute the cause for any differences we observe in the two measurement experiments. Due to time constraints near the time of notification delivery, designing a controlled experiment was not possible. Our priority was delivering the notifications without further delay. Additionally, the slight differences in methodology between the two measurements further complicate determining the exact cause for any differences in results. Regardless, there still
is much to be learned by analyzing the differences between the two datasets.

E. Ethical Considerations

The measurement of network and systems vulnerabilities requires care, both in the activity itself and in the disclosure of the findings. Because of the nature of our research, we consulted various resources for ethical guidance. As our measurement studies did not contain any human elements, we did not consult our institution’s IRB for that portion of our work. However, the notification campaign introduced a human element. Thus, prior to delivering the notifications, we consulted two members of the IRB. It was determined that our work did not constitute human subjects research, and as such, was outside of the IRB’s domain. Additionally, we consulted with individuals from the legal department, the office of research and creative activities, and our own computer science department with respect to the ethics of our research. We likewise reviewed the Menlo Report [37], which holds some of the key guidelines for ethical research in this area. Of the ethics principles outlined in the Menlo Report, those most applicable to our current research are 1) justice, 2) respect for law and public interest, and 3) beneficence.

Regarding justice, our measurements considered all target IP addresses (i.e., from the DITL data) equally; no particular industry, geography, nation state, address space, or protocol was deliberately targeted more than another.

Perhaps the biggest ethical question associated with our research was the legality of measuring another’s network using our methodology. Our measurements crossed interstate boundaries world-wide, each potentially with their own laws regarding unauthorized network access. For example, the United States (U.S.) outlaws any intentional access of non-public, government-owned computer systems, without authorization [38]. We cannot definitively determine whether or not the systems that we measured are non-public nor whether our benign packets even constitute a violation of this statute. In any case, we believe that our methodology is justified because of the benefit it brings in the public interest. Indeed the Menlo Report’s principle of beneficence suggests that the benefits of an experiment should be maximized and the harms minimized. Bringing to light the severity and pervasiveness of the lack of DSAV and the potential for network penetration is extremely valuable to the Internet community. We expect that responsibly publishing our findings will be a catalyst in spreading awareness and taking the necessary action to fill the security gaps identified herein.

The potential harms associated with our experiment might include degradation of service due to our traffic, time spent following up on alerts from Intrusion Detection Systems (IDS), or careless vulnerability disclosure. We took several measures to minimize any negative impact, and even the appearance of abuse. First, we limited both the number and the rate of queries directed towards any given destination, as described in subsubsection IV-A5. Considering query rates at production DNS servers are typically measured in queries per second, and our maximum per-destination rate was on the order of four per day—plus a one-time series of fewer than 30 follow-up queries—the impact of our experiment would have been barely, if at all, noticeable. As one of these follow-up queries was designed to elicit further queries to explore susceptibility to the NXNS attack, the exact number of queries required to resolve the follow-up queries varied on the resolver configuration. In the worst case (i.e., a vulnerable resolver), an additional 40 queries are required—still a nominal volume of queries. Second, the SOA (start of authority) record of the DNS zone corresponding to our query names (see subsubsection IV-A3) included: 1) a RNAME (responsible name) field with an email address with which we could be contacted, e.g., for more information or to opt out; and 2) an MNAME (master server name) field with the domain name of a Web server providing a brief description of this project. The project description included contact and opt-out information. The system from which the queries with non-spoofed sources were sent (see subsubsection IV-A4) also ran a Web server with the same project information.

Our notifications undoubtedly resulted in added load for the administrators that received them. Not even considering the effort required to actually address the vulnerability itself, the mere presence of unsolicited emails in one’s inbox can be taxing. Further exacerbating this issue, some of the responses we received indicated that at least in some cases, the admin was forwarded the same notification multiple times from the other recipients of the notification. While this added load is unfortunate, we ultimately felt the potential good that could be accomplished through the notifications outweighed this cost. We took some efforts to mitigate this cost, such as excluding addresses from our communications that appeared as points of contact for too many organizations. Additionally, our emails clearly indicated who we were and provided instructions on how to stop receiving further communications from us. Reactions to our emails were mixed, but ultimately the positive reactions outweigh the negative ones (see section VII-B2).

VI. DSAV Experiment Results

The pervasive lack of DSAV well exceeded our expectations. For Round1, of the 11,204,889 IPv4 addresses targeted with our experiment, at least 519,447 (4.6%) received and handled one or more of our queries, as indicated by a recursive-to-authoritative query observed at our authoritative DNS servers. Similarly, of the 784,777 IPv6 addresses we targeted, 49,008 (6.2%) recursively handled at least one of our spoofed-source queries. While the figures for target IP addresses are nominal, the number of ASes with affected IP addresses was far more pervasive: 26,695 (49%) of the ASes were vulnerable to infiltration via spoofed-source packets. These numbers represent the lower bound of networks that do not support DSAV.

As for Round2, we found 26,488 networks lacking DSAV (44%). 436,781 IPv4 addresses (2.8%) and 261,105 IPv6 addresses (16.6%) were reachable. Encouragingly, the percentage of vulnerable networks is somewhat smaller than that of Round1. Similarly, the percentage of IPv4 addresses reachable is lower in the second round. Surprisingly, the
percentage of reachable IPv6 addresses is considerably higher the second time. As discussed in section VI-C, the increase in IPv6 addresses reached can largely be attributed to a single autonomous system with over 200K reachable IPv6 addresses. This finding illustrates that as a large portion of DSAV is performed at the autonomous-system-level, results at the IP-level can be volatile—merely a property of the autonomous systems lacking DSAV.

Table I shows our results broken down by country. The top 10 countries with the most autonomous systems are shown. IP-to-country lookups were done using the MaxMind GeoLite2 Country database. Each autonomous system was associated with one or more countries based on the countries listed for its constituent IP addresses. Both data sets included over twice as many United States (US)-based ASes than the next most represented country, Brazil. Nevertheless, the diversity of percentage of ASNs lacking DSAV (i.e., “reachable”) is apparent, with the US being below average and Brazil, Russia, and Ukraine showing that over half of ASes lack DSAV. Notably, for all 10 countries, the percentage of reachable ASes decreased in Round2.

Only considering the autonomous systems that were present for both measurements, in Round1, 51% (25,511 out of 49,854) lacked DSAV. By the same criteria, in Round2, 48% (24,137) lacked DSAV. Using a two sample z-test of proportions, we see that the decrease is significant ($p < 0.00001$). Oddly, 2,410 (9.9%) of the ASes that were not vulnerable in Round1 were vulnerable in Round2. While it is certainly possible that these ASes changed configurations and became vulnerable in the interim between the two measurements, it is perhaps more likely that the changes made to our methodology for Round2 allowed us to reach ASes that we could not reach using Round1’s methodology. Interestingly, only considering the 10,764 ASes present in Round2 but not present in Round1, only 2,351 (22%) lacked DSAV. One possible cause for this is that these ASes had a relatively small presence in the DITL dataset, meaning we targeted fewer addresses—potential resolvers—from them. The average number of addresses targeted in Round2 for ASes that were also in Round1 is 338, the median 7. By contrast, the average number for the new ASes is 32 and the median 2. The lower number of targeted addresses reduces the opportunity we have to observe a lack of DSAV.

### A. Autonomous System Degree

In an attempt to shed some more light on why fewer of the “new” ASes lacked DSAV, we hypothesized that autonomous systems with a higher degree, i.e., ASes with more network neighbors, would be more likely to lack DSAV. Given that the new networks had a smaller presence in the DITL, it is conceivable that they represent less-connected ASes and—according to our hypothesis—could be expected to have greater DSAV deployment. To explore this hypothesis,
using RIPEstat’s API\(^3\), we determined the number of unique neighbors for each autonomous system. Given that the API supports historical lookups, we were able to obtain current data for both measurements. We found valid neighbor counts for all but 2% of the autonomous systems. We divided the ASes into groups based on number of neighbors, then determined the proportions of ASes in each group with DSAV present or lacking. The results are shown in Figure 1a. Indeed, there appears to be a correlation with degree and lack of DSAV. For Round1, on the low extreme, 81% of the networks with only one neighbor lacked DSAV. On the other extreme, 81% of the networks with ten or more neighbors lacked DSAV. This pattern holds for Round2: 35% and 73% of the one-neighbor and ten-plus-neighbor networks lacked DSAV, respectively. Formally, we model the relationship between number of neighbors and DSAV using logistic regression and find that the relationship is significant at \( p < 2e^{-16} \), for both rounds. For every unit increase in number of neighbors, the log odds of lacking DSAV increases by 0.02 (Round1) and 0.01 (Round2). Note that this model assumes that the relationship between the number of neighbors and the log odds is linear.

With regard to the “new” ASes—those present in Round2 but not in Round1—the average number of neighbors is roughly three times smaller than that of the ASes present in both data sets (5 vs 15), though the median is the same for both groups (2). This shift in distribution likely contributes to the greater deployment of DSAV in this group; however, this explanation is ultimately incomplete, as this new group experiences improved rates across the board, with only 19% of the one-neighbor ASes lacking DSAV.

### B. Number of Announced Prefixes

We repeat the analysis of section VI-A, this time looking at the number of prefixes announced by each autonomous system, determining the number of prefixes announced per AS using data published by CAIDA [32]. This analysis is shown in Figure 1b. The number of prefixes announced also correlates with a lack of DSAV. Once again comparing the extremes, for Round1, 32% of the ASes with only 1 or 2 prefixes lack DSAV, as compared to 83% of the ASes with more than 21 prefixes. Similarly, for Round2, 27% and 79% lack DSAV.

Using logistic regression to model the relationship between number of announced prefixes and DSAV, we find that for both rounds, the relationship is significant at \( p < 2e^{-16} \). For every unit increase in number of neighbors, the log odds of lacking DSAV increases by 0.02 (for both rounds). The ASes from Round2 that were not in Round1 also announced fewer prefixes, likely contributing to the low percentage of “new” ASes lacking DSAV. The average number of prefixes announced by these is 5 and the median 2, as compared to 19 and 4 announced by the ASes present in both measurements.

Both autonomous system degree and number of announced prefixes can be interpreted as measures of autonomous system complexity. Given that both of these factors correlate with a lack of DSAV, the data support the notion that the more complex the network, the more difficult it is to deploy DSAV. As discussed in section II-A, there is no one-size-fits-all DSAV solution. Of particular relevance for complex ASes, ACLs are difficult to maintain at-scale and strict uRPF cannot be used in the case of asymmetric routing. Thus, complex autonomous systems are left with few options for blocking spoofed traffic, possibly explaining the correlation we observe in the data. As such, the development of spoofing prevention techniques suitable for complex ASes—such as EFP-uRPF, incidentally standardized between our two measurements—is valuable.

### C. Spoofed Source Effectiveness

We now consider the effectiveness of the various sources that were used, in terms of eliciting DNS activity of their target. All numbers are presented as a fraction of the total reachable targets.

We first analyze the overall fraction of spoofed sources that reached their targets. For Round1, nearly half of all reachable target IP addresses (collectively IPv4 and IPv6) only acted on one or two sources. The median number of spoofed sources with which queries reached IPv4 and IPv6 destinations was 3 and 2, respectively. However, 16% of IPv4 destinations and 9% of IPv6 destinations were reachable using over 50 spoofed sources.

For Round2, 70% of the reachable targets were only reached by one or two sources. The median number of spoofed sources with which queries reached IPv4 and IPv6 destinations was 2 and 1, respectively. However, 24% of IPv4 and 4% of IPv6 addresses were reachable using over half of the spoofed sources.

Next we analyze the effectiveness of the different categories of spoofed sources (see section IV-A2), in terms of DSAV detection. For both measurements, spoofed sources from every category reached at least one IP target. The breakdown of target IP addresses and ASNs reachable by spoofed source category for both measurements is shown in Table II, under the headings “Category-Inclusive.”

For Round1, IPv4, same-asn and same-prefix sources were the most prevalent, reaching 78% and 63% of all IPv4 targets that received spoofed-source queries. Dst-as-src was surprisingly effective, reaching 17% of the reachable targets. Private addresses also had a presence, reaching 3.4% of the targets. Even the loopback address reached one target. For Round2 IPv4, the performance of each category is comparable, with just one exception: the percentage of reachable targets that responded to the same-asn address is lower—57%. This is likely due to the decrease in number of same-asn addresses generated (97 vs 20).

As for Round1 IPv6, Same-prefix and Destination-as-source were the two most effective categories, reaching 84% and 70% of addresses, respectively. Same-ASN addresses still had a significant presence, reaching 45% of addresses. For Round2, Destination-as-source was by far the most effective category, reaching 96% of all IPv6 addresses. The Same-ASN and Same-prefix addresses did not perform well—relative to Round1—only reaching 5% and 12% of the addresses, respectively. This can actually be attributed to skew from a large autonomous system with over 200K reachable IPv6 addresses, 86% of all reachable, 98% of the IPv6 addresses that

\(^3\)https://stat.ripe.net/docs/data_api#asn-neighbours
TABLE II: The number of IP addresses or ASes for which at least one spoofed-source reached its target (“Category-Inclusive”) or for which a spoofed-source-category was the only one to reach its target (“Category-Exclusive”). Percentages represent the fraction of reachable targets.

<table>
<thead>
<tr>
<th>Source Class</th>
<th>IPv4 Addr</th>
<th>IPv6 Addr</th>
<th>ASes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Same ASN</td>
<td>405,018 (78%)</td>
<td>322,073 (45%)</td>
<td>25,781 (97%)</td>
</tr>
<tr>
<td>Same Prefix</td>
<td>337,000 (63%)</td>
<td>40,986 (84%)</td>
<td>24,390 (91%)</td>
</tr>
<tr>
<td>Dist-as-Src</td>
<td>30,853 (12%)</td>
<td>2,098 (4.3%)</td>
<td>3,507 (13%)</td>
</tr>
<tr>
<td>Private/ULA</td>
<td>17,762 (3.4%)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Loopback</td>
<td>1 (0.0%)</td>
<td>106 (0.2%)</td>
<td>27 (0.1%)</td>
</tr>
</tbody>
</table>

(a) Round1 data.

<table>
<thead>
<tr>
<th>Source Class</th>
<th>IPv4 Addr</th>
<th>IPv6 Addr</th>
<th>ASes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Same ASN</td>
<td>172,372 (33%)</td>
<td>2,410 (4.9%)</td>
<td>1,939 (7.3%)</td>
</tr>
<tr>
<td>Same Prefix</td>
<td>90,366 (17%)</td>
<td>3,972 (8.1%)</td>
<td>390 (1.5%)</td>
</tr>
<tr>
<td>Dist-as-Src</td>
<td>13,384 (2.6%)</td>
<td>4,869 (9.9%)</td>
<td>380 (1.4%)</td>
</tr>
<tr>
<td>Private/ULA</td>
<td>2,508 (0.5%)</td>
<td>2,299 (0.5%)</td>
<td>129 (0.5%)</td>
</tr>
<tr>
<td>Loopback</td>
<td>0 (0.0%)</td>
<td>22 (0.0%)</td>
<td>4 (0.0%)</td>
</tr>
</tbody>
</table>

(b) Round2 data.

That said, some DSAV can be performed at the IP-level—for example, should the Destination-as-source query not be blocked at the network border, it could still be blocked at the destination machine. It is notable that the percentage of target-reaching queries that used Destination-as-source was much higher for IPv4 targets (70% and 96%) than that for IPv4 hosts (17% and 18%). We observed in our lab testing that modern Linux kernels drop destination-as-source packets that use IPv4, but IPv6 destination-as-source packets are sent to user space. Thus, if the fraction of reachable Linux targets is equal between the IPv6 and IPv4 realms, we would expect the percentage of IPv6 destination-as-source hits to be higher.

Each category of spoofed source independently contributed to the overall effectiveness of our experiment. Every category resulted in reaching a target that would not otherwise have been reached, even considering all other categories combined. Thus, if we had excluded any category of spoofed addresses from our experiment, our total number of reachable targets would have been lower—for both by IP address and ASN. This is shown in Table II, under the headings “Category-Exclusive.” Notably, had we limited our spoofed sources to addresses within the same IPv4 /24 (or IPv6 /64) as the target, with a source address different from the destination address, for Round1, we would not have discovered 36% of reachable IPv4 addresses, 15% of the IPv6 addresses, or 9.2% of reachable ASes. For Round2, we would not have reached 31% of the reachable IPv4 addresses, 88% of the IPv6 addresses, or 6% of the ASes. While the query with spoofed loopback source was handled by relatively few targets, several IPv6 addresses and ASes would not have resulted in hits without our inclusion of this source category.

VII. NOTIFICATION CAMPAIGN

A. Design

1) To Addresses: The first challenge with a large-scale vulnerability notification campaign is finding the correct points of contact for 27K networks. We initially attempted to find POCs using a bulk Whois data download using the service provided by ARIN [39]. However, using this data, we were only able to find points of contact for 9,779 ASes (37%). Even after supplementing the ARIN data with RIPEstat’s Whois API, we were only able to find points of contact for 15,749 ASes (59%). Eventually, we turned to the WhoisXML API[^4], a commercial service that aggregates network contact information. There is precedent for using their API [22], [25], [26]. Using this API, we were able to find contact information for 21,224 ASes (80%). Considering ARIN’s, RIPE’s, and WhoisXML’s data jointly, we found contact information for 23,227 networks. We were able to find contact info for an additional 1,038 networks (3.8%) by looking at the SOA records of the resolvers we queried.

Unfortunately, the Whois XML API doesn’t actually give results at the autonomous system level. What it does provide is contact information for blocks of IP addresses which are in the autonomous system. Unfortunately, because these points of contact are associated with blocks of IP addresses, not the autonomous system itself, we cannot guarantee that each email will reach an admin associated with the AS; in the worst case, the admin represents a sub-organization within the AS that cannot speak or act for the AS, nor be trusted with sensitive information pertaining to the AS as a whole. To mitigate this limitation, we selected up to 5 points of contact for each organization. This increased the chance that our notification would reach the correct point of contact and decreased the chance that the notification would be left exclusively with a party that should not have received it. As far as we are aware, sending emails to multiple points of contact simultaneously has not been explored in prior work.

We first selected addresses from the WhoisXML API data, selecting in the following order of preference: technical, abuse,
admin, organization, and maintainer. We then supplemented this data with the database downloaded from ARIN and RIPEstat’s Whois API, selecting addresses in the following order of preference: tech, admin, and NOC. Finally, as a last resort, for any ASes with no POCs identified thus far, we performed SOA lookups on the domain names of a sample of the responsive resolvers from each AS. Note that we explicitly excluded dns@cloudflare.com, as it was a very common result of the SOA lookups, likely from resolvers that were outsourced to CloudFlare.

After completing this process, we performed one final step to curate the data. We noticed that many different autonomous systems had points of contact in common. While this is reasonable, there are extreme cases of it. For example, the process described above would have resulted in helpdesk@apnic.net receiving over 1,000 emails from us. Thus, we excluded any addresses that would have received over 4 emails from us. Unfortunately, we did not notice this issue until we had already sent some emails, though we corrected the issue quickly. This resulted in 3 separate addresses receiving 13 emails from us, which again, is unfortunate, though we did not receive any complaints from those addresses. In total, we excluded 881 addresses and prevented the above mentioned addresses from receiving further emails from us.

In summary, we emailed 57,339 different addresses. 89% were found through the WhoisXML API, 1% through ARIN’s database, 8% through RIPE’s API, and 2% through SOA lookups. Each autonomous system received on average 3 emails from us.

2) From Addresses: A unique From address was used for each point of contact. We did this in order to collect data for a different study relating to email security (e.g., SPF [40]). The From addresses followed the following template:

```
<dsav-info@[uid].report.[date].dsav-mail.dns-lab.org>
```

Though we had some concern that the seemingly auto-generated From address might be off-putting to the recipients, prior work has shown that the email address of the notification sender has no effect on patching rates [21]. Additionally, each email included dsav-info@byu.edu as a Reply-To header, clearly identifying our institution. We received no comments on the abnormal From address.

3) Content: The full body of the emails we sent is included in Appendix A. We discuss here several of the decisions we made that resulted in the final design.

a) HTML vs plain-text: Prior work [26] found no significant difference in remediation rates between plain-text and HTML emails, including both HTML emails with and without external images. Thus, with no compelling evidence dictating which to use, we elected to use HTML-formatted emails. This allowed us to include an external image to track which emails were opened. However, we did include a plain-text version for email clients that don’t load HTML.

b) Language: Prior work has established that translating the notification out of English at best has no effect and is possibly detrimental [24], [27]. Thus, we sent all emails in English, regardless of the apparent country of the email domain.

c) Level of detail: Past work indicates that detailed notifications might be more effective than terse messages [24], though the authors note that the difference was not significant. A more recent study found no significant effect from how the message is framed (e.g., a focus on the effect on end users vs a technical focus) [27]. Thus, once again there is no compelling evidence to dictate what level of detail to include. We elected to include the following in our emails:

- A basic description of our methodology;
- Basic examples of how the vulnerability could be exploited;
- The specific number of resolvers within their AS that we were able to reach;
- A very high-level description of how to fix the issue;
- The specific number of resolvers we found that lacked source port randomization (if applicable).

We split our notifications by address family. Thus, if we were able to reach both IPv4 and IPv6 addresses for a given autonomous system, we sent two separate sets of emails to that autonomous system. The subject lines of the emails followed the form, “IPv[4,6] Spoofing Vulnerability AS[asn]”.

d) Inclusion of external report: We included a link to a custom report, specific to each autonomous system, that included the following:

- A link to a frequently asked questions page
- An option to run a quick re-test
- A more detailed description of our methodology
- The specific IP addresses that we were able to reach within their AS, as well as anything we were able to infer about them (operating system and source port randomization)
- The spoofed source addresses that were able to reach each IP address

An example report is included in Appendix B.

Notably, we added the FAQ page, the self-test option, and the specific spoofed sources after sending the emails. Ideally, we would have had these included in the reports from the beginning; however, we severely underestimated the level of response we would receive. As such, we did not anticipate the need for these features and adding them to the reports was the only way of addressing the responses to the emails in a timely manner.

In a prior study which similarly included an optional, detailed report, only 15-26% of the reports were requested [26]. Though we knew we would likely face similar rates, we had significantly more information than we could possibly convey in an email, especially for the larger autonomous systems. Thus, we did our best to balance what we could fit into the email proper and what needed to be relegated to the report.

The self-test option triggered a rerun of the portion of the experiment pertaining to a given autonomous system, though only a subset of the queries were re-sent. This allowed network operators to gain immediate feedback on whether or not any actions they had taken had corrected the issue and to observe

\[\text{Initially, the reports only included the general categories that were successful, e.g., “Same ASN,” “Same prefix”, etc.}\]
TABLE III: The autonomous systems found to be vulnerable in Round1, broken down by various categories. For each category, the state of the ASes in Round2 is shown (i.e., patched or vulnerable). The percentages in each row are based off the the “Total” column. The percentages might not add up to 100% as there were some ASes present in Round1 that were not present in Round2 which would be best classified as “unknown,” though we do not explicitly show this category.

<table>
<thead>
<tr>
<th>Category</th>
<th>Total</th>
<th>Patched</th>
<th>Vulnerable</th>
</tr>
</thead>
<tbody>
<tr>
<td>All vulnerable</td>
<td>26,695</td>
<td>3,784 (14.2%)</td>
<td>21,727 (81.4%)</td>
</tr>
<tr>
<td>Interaction</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No contact found</td>
<td>2,430</td>
<td>372 (15.3%)</td>
<td>1,683 (69.3%)</td>
</tr>
<tr>
<td>Email sent</td>
<td>24,265</td>
<td>3,412 (14.1%)</td>
<td>20,044 (82.6%)</td>
</tr>
<tr>
<td>Email delivered</td>
<td>20,156</td>
<td>2,783 (13.8%)</td>
<td>16,773 (83.2%)</td>
</tr>
<tr>
<td>Email opened</td>
<td>12,633</td>
<td>1,791 (14.2%)</td>
<td>10,529 (83.3%)</td>
</tr>
<tr>
<td>Report opened</td>
<td>7,437</td>
<td>1,165 (15.7%)</td>
<td>6,125 (82.4%)</td>
</tr>
<tr>
<td>Self-test ran</td>
<td>832</td>
<td>259 (31.1%)</td>
<td>564 (67.8%)</td>
</tr>
</tbody>
</table>

| PoC source                     |        |         |            |
| PoCs found via WhoisXML        | 19,605 | 2,643 (13.3%) | 16,962 (84.1%) |
| PoCs found via RIPE or ARIN    | 2,003  | 400 (20.0%) | 1,533 (72.5%) |
| PoCs found via both RIPE/ARIN and WhoisXML | 1,603 | 260 (16.2%) | 1,317 (82.2%) |
| PoCs found via SOA records     | 1,054  | 109 (10.3%) | 778 (73.8%) |

| # of emails                    |        |         |            |
| 0 emails delivered             | 2,430  | 372 (15.3%) | 1,683 (69.3%) |
| 1 email delivered              | 6,775  | 1,147 (16.9%) | 5,240 (77.3%) |
| 2 emails delivered             | 5,735  | 910 (15.9%) | 4,630 (80.7%) |
| 3 emails delivered             | 3,495  | 518 (14.8%) | 2,878 (82.6%) |
| 4 emails delivered             | 3,345  | 441 (13.2%) | 2,842 (85.0%) |
| 5+ emails delivered            | 4,913  | 396 (8.1%) | 4,417 (90.4%) |

what the traffic looked like on their end. The test results became available within a minute of being requested.

B. Results

1) Notification Efficacy: Of the 26,695 ASes we found lacking DSAV, we were able to find contacts for and attempted to email 24,265 (91%) of them. 61% of the emails we sent were accepted for delivery and did not result in a bounce report emailed back to us. Our results indicated that an email was successfully delivered to at least one contact associated with the autonomous system for 20,156 (75%) of the ASes. Via the image embedded into the email body, we were able to infer that at least 12,533 (47%) ASes opened the email. Notably, it is possible to configure one’s email client to not automatically request external resources; thus, it is possible that some recipients opened the email without requesting the embedded image, as evidenced by 100 ASes whose detailed report linked in the email was opened without the embedded image being requested. In total, the detailed reports for 7,437 (28%) ASes were opened and 832 (3%) ASes used the self-test option we added to the reports.

In an effort to gain some insight on the effect of our efforts, we divide the ASes we found vulnerable in Round1 into different sets based on properties such as opening the email and opening the report, then compare the patching rates between these groups. This can be seen in Table III. We look at several different dimensions, such as level of interaction with the notification (labelled “Interaction” in the table), the source used to find the points of contact (“PoC source”), and the number of emails received by the ASes (“# of emails”).

First, looking at the level of interaction with the notifications, all groups had comparable patching rates—between 14 and 16%, with the exception of those that ran the self-test. 31% of those that ran the self-test patched. In an attempt to find a precise answer regarding the effect of the notification emails, we perform a statistical analysis of the data using a logistic regression model. For this model, we separated individual ASes into disjoint sets, based on the most advanced interaction they had with the email. For example, if for a given AS, an email was sent, received, and opened, for the purposes of this test, we only classified them as “Email opened.” The results confirm our prior observations: there was little change in the odds of patching for any category—aside from those who ran the self-test tool. Those that ran the self-test tool were between 140% and 200% more likely to have deployed DSAV for Round2. Note that this does not imply that the inclusion of a self-test tool improves patching rates—as we did not design the notification campaign as a controlled experiment, we cannot make any definitive conclusions here.

We next consider the role the source of point of contact info plays in the notification efficacy. For this analysis, we only consider the emails that were successfully delivered, discounting the emails that were rejected. Interestingly, ASes that we emailed using contacts found exclusively via the WhoisXML API had a lower patching rate—13.5%—than those we contacted using info obtained from ARIN and RIPE—20.0%. While again, we cannot say conclusively why this is, it is possibly due to the fact that the WhoisXML does not provide true AS-level contacts, only the contact info associated with the blocks of IP addresses registered to the autonomous system.

We similarly consider the effect of the number of emails received had on patching rates—once again, only considering emails that were accepted for delivery. As can be in Table III under “# of emails,” the patching rate for ASes for which no emails were delivered is comparable to the overall patching rate. The group with the highest patching rate were ASes that had exactly one email accepted for delivery. Interestingly, the patching rates appears to correlate negatively with the number of emails delivered. There are two possible explanations for this, the first being that the increased volume of emails led the recipients to dismiss our notifications as spam. Alternatively, as the number of emails we sent per AS was directly related to the number of contacts that were available, the ASes that received more emails likely represent more complex autonomous systems—which, as shown in sections VI-A and VI-B, are more likely to lack DSAV. Regardless of the reason, it appears like increasing the contacts emailed per autonomous system is an ineffective method of spurring remediation.

2) Email Responses: The volume of responses to our notifications far exceeded our expectations. In total, we received 1,719 responses to our emails (not counting bounce reports). We worked as fast as we could to respond to these, even developing a frequently asked questions page, a self-test tool, and a more detailed report in order to address the needs many of the responders had in common. We had these new tools
available within 3 days of sending the emails.

We conducted a qualitative analysis of the responses to our emails by developing the following classification system. Each email we received was classified as one of the following: Human-Positive, Human-Neutral, Human-Negative, or Automated. Additionally, any number of the following labels could be applied to each email, as appropriate: Request for more info, Retest request, Have/will patch, Wrong email, Forwarded info, Disapproval of the experiment, Disapproval of the emails, and Other language.

To develop this classification system, two researchers independently read through the emails until we felt we had a good sense of the underlying themes. We then met and proposed an initial classification system. Then each researcher independently coded a random sample of 50 emails. After coding the sample, the researchers met and resolved any differences in how they coded the sample. One coder then coded the remaining emails. Anonymized examples of each of these classes and labels are included in Appendix C.

Of the 1,719 responses we received, 1,091 (63%) were automated. The automated emails largely consisted in notifications of ticket creation (e.g., “A support ticket has now been opened for your request”) and changes in ticket status. Of the 628 human responses, 142 (23%) were positive, 446 (71%) were neutral, and 40 (6%) were negative. Of the human responses, 15 (2.4%) indicated disapproval of our research (e.g., “with whose authority did you conduct the research on my network?”) and 29 (4.6%) indicated disapproval of our emails (e.g., accusations of spam). 200 (32%) of the responders indicated that they either already had or intended to patch the vulnerability. 120 (19%) requested a re-test of their network (e.g., to validate a changed configuration). 256 (41%) requested more information (e.g., on our methodology, mitigation strategies, etc.) 105 (17%) indicated that we had reached the wrong person. 47 (7.5%) indicated that they had forwarded the email to the appropriate contact. 256 (41%) of the emails were in languages other than English, though all but 6 of these were automated.

VIII. CASE STUDY: DNS RESOLVERS

While the knowledge that a network lacks DSAV is valuable in and of itself, in this section we demonstrate how that knowledge might be used by someone with malicious intent to survey or exploit vulnerabilities of internal systems, specifically DNS resolvers.

A. Source Port Randomization

We begin by investigating one of the most prominent DNS resolver vulnerabilities to date: lack of source port randomization [13], [41]. While this vulnerability was disclosed and related patches were distributed nearly 13 years prior to the writing of this paper, we hypothesized that there might be some instance of it in the wild, behind closed doors.

To assess the source port randomization of the reachable targets, we computed the range of source ports for the 10 IPv4 or IPv6 follow-up queries (see section IV-A4) observed for each target IP address. Note that for this analysis, we only consider the resolvers that resolved these queries directly, rather than forwarding them to a third party, an assessment we made by comparing the addresses embedded in the query name against the IP address of the machines querying our server. The source port range is a useful heuristic to identify resolvers that are not randomizing their source ports. It also provides characteristics that are helpful for identifying OS or software, a phenomenon we explore in the conference version of this work [2].

The most serious case of poor source port randomization is no source port variance whatsoever.6 In response to our spoofed-source follow-up queries, in Round1, 3,810 (1.3%) resolvers issued 10 queries to our authoritative servers with no variance in source port! These account for 1,780 (7%) of all ASes lacking DSAV. It would be trivial to poison the cache of these resolvers due to the combination of 1) the capability to induce a query through spoofed-source query and 2) no source port randomization. With a known source port, only the query’s 16-bit transaction ID is left to guess; the search space is reduced from $2^{32}$ (4.3 billion) to $2^{16}$ (65,536). More than half of these resolvers (2,244 or 59%) are closed, meaning that DSAV would reduce their attack potential.

The results for Round2 were very similar. 3,591 (1.3%) resolvers had no variance of source port, representing 1,597 (6%) of the ASes lacking DSAV. 1,699 (47%) are closed, again meaning that DSAV would reduce their threat surface. 2,094 (55%) of the addresses vulnerable in Round1 were also present in the Round2 data. Of these, 1,923 (92%) still lacked any form of source port randomization in Round2. Given that these resolvers were disclosed in our email notifications, this finding is disappointing, though perhaps unsurprising.

B. NXNS Attack

This portion of the analysis only applies to the Round2 data, as we had not implemented the test for Round1. The number of requests made by each resolver when resolving the NS records is shown in Figure 2. As our server returned 20 NS records, we consider any resolver that requested 20 or more records vulnerable. As we verified in a controlled environment, any resolver running patched versions of BIND, Unbound, Knot Resolver, or PowerDNS Recursor will request significantly less than this. By this criteria, 399,430 (57%) resolvers were vulnerable (239,327 IPv4 addresses and 160,103 IPv6). 80% (21,106) of the reachable autonomous systems had at least one vulnerable resolver. Oddly enough, there are some resolvers that requested more than 40 records. The most extreme case was a resolver that requested 80 records; it requested an A, AAAA, A6, and ANY record for each NS record. Resolvers such as these would have even greater amplification potential than accounted for by the theoretical model [3].

We can observe peaks corresponding to the numbers of queries requested by the various resolver implementations during our controlled testing. For example, as BIND and Knot

6This analysis could be expanded to resolvers with ineffective source port randomization rather than just resolvers with static ports, something we do in the conference version of this paper [2]. However, we consider this out of scope of the current work.
Resolver both requested 10 records, it is likely that the large spike at 10 records consists of resolvers running either BIND or Knot Resolver.

Using the open resolver follow-up query, we are able to classify these resolvers as open or closed. 281,076 (70%) of the vulnerable resolvers are correctly configured as closed resolvers. Were the ASes of these resolvers to employ DSAV, these ~300K resolvers would not be able to be used as NXNS-amplifiers by attackers external to their networks.

IX. DISCUSSION

The findings in this paper are non-trivial. We have shown that, in many cases, systems thought to be accessible only by trusted parties can be reached with a minimal amount of effort. We have also seen that it is not simply the reachability of these systems that matters, but the fact that a potentially malicious third party can discover their weaknesses with just a few strategically-formed queries. Finally, we observed both through active measurement and through anecdotal evidence—supported by communications with DNS operators—that there is some complacency with regard to security and maintenance of internal systems. These messages collectively convey a message that security at network borders is—in many instances—false, that old, vulnerable software and configurations yet have a deployment presence in the wild, and that substantial effort will be required to motivate the changes necessary to fix these insecurities. Without such change, internal systems continue to be reachable and potentially more vulnerable to various attacks, including DNS cache poisoning (section VIII-A) and NXNS exploitation (section VIII-B).

Our goal with the current research is not merely to identify the problems with networks and systems, but to spark impetus for widespread change. Increased OSAV adoption (i.e., BCP 38) is certainly part of the solution to source address spoofing in the wild—to prevent both reflection attacks and spoofed-source infiltration. However, OSAV requires the participation of third party networks—those hosting the attackers, not the victims. On the other hand, potential victims of spoofed-source network infiltration can prevent such attacks by configuring their own systems for DSAV.

There is no one-size-fits-all solution for deploying DSAV. The choice of specific technique for performing DSAV is very dependent on the network configuration. Relatedly, we find that network complexity appears to correlate with a lack of DSAV. Networks with a larger number of network neighbors had a high rate of lack of DSAV, similar to networks with a large number of announced prefixes.

Because of the broad impact of our findings, spreading awareness is important, particularly in connection with the publication of this paper. As part of our efforts to spread awareness, we have reached out to the networks we found to be vulnerable. Additionally, we have created a generalized self-test tool that can be used by network operators to test their own networks. This tool is now publicly available and can be accessed at https://dsav-test.byu.edu/.

In this work, we used the domain name system as a means of exploring the deployment of DSAV. However, this is not a DNS-specific vulnerability. Exploring the potential for spoofing-based network intrusion using other network protocols remains a possible avenue for future work.

X. CONCLUSION

In this paper we have investigated an area previously explored very little—the lack of DSAV in networks and systems. We presented a methodology for effectively identifying networks and systems vulnerable to spoofed-source infiltration. In December of 2019 we found that of 54K autonomous systems, 49% lacked DSAV. After a notification campaign run in October 2020, we repeated our measurement and found that of 61K ASes, 44% lacked DAV. Finally, we identified vulnerable DNS software and systems and found a significant number of resolvers vulnerable to DNS cache poisoning and the NXNS attack.

The findings in this paper are significant and can have real impact on Internet security. It is our hope that the results of this study, as well as our efforts to encourage change can make that impact a positive one, providing a stronger defense against spoofing, infiltration, DDoS attacks, and cache poisoning.

ACKNOWLEDGMENT

We gratefully acknowledge the Comcast Innovation Fund for their support of the work that produced this material. DNS-OARC for providing access to the 2019 and 2020 DITL collections, CAIDA for providing access to their infrastructure to carry out our experiment, and the WhoisXML API for providing access to their API. We also acknowledge with gratitude Michael Briggs, Trevin Avery, and Robert Richardson for their contributions on the conference version of this work. We also thank Robert Richardson for his statistical expertise and guidance. Finally, we thank Daniel Zappala and Kent Seamons for their helpful reviews of our work.

REFERENCES


APPENDIX A
EMAIL TEMPLATE

A. Plain-text Version

Below follows the plain-text version of the notifications we sent. Items in bold inside brackets are replaced with the actual values for a given AS. The paragraph regarding source port randomization is omitted completely if a given AS has no such resolvers.

Dear admin of AS[asn],

We are researchers from Brigham Young University’s Internet Measurement and Anti-Abuse Laboratory [1]. During December 2019, we conducted a large-scale Internet measurement experiment. The intent of this experiment was to determine the pervasiveness of networks failing to filter spoofed incoming traffic appearing to originate from within their own networks. This oversight allows attackers to infiltrate the network and impersonate internal resources. This in turn facilitates attacks which could otherwise be prevented, such as DNS cache poisoning or the NXNS attack, a powerful new denial of service technique. Our methodology relied on sending DNS queries with spoofed source addresses to known DNS resolvers. The queries were for domains under our control; as such if we observed a corresponding query at our authoritative server, we were able to determine that our spoofed queries successfully infiltrated the network.

Unfortunately, it appears that AS[asn] is vulnerable to this class of attack. [n] of the known resolvers within your autonomous system acted on our spoofed queries, indicating that our spoofed queries successfully infiltrated the network. Even if these resolvers are configured to resolve queries from any host, this indicates a vulnerability, as the traffic would have been blocked at the network border had DSAV been in place. The solution–DSA V, or Destination-Side Source Address Validation–is to simply filter out incoming traffic that appears to have originated from within the network.

As part of our research, we identified [n] resolvers lacking source port randomization within your autonomous system. This is an extremely dangerous configuration as it leaves the resolvers completely vulnerable to DNS cache poisoning attacks.

Our work has been accepted to the Internet Measurement Conference (IMC) 2020, which will begin near the end of this month on October 27. While we will present our results in-aggregate only and will not specifically mention your autonomous system, we recommend resolving this issue soon before there is greater general awareness of the vulnerability. If you have already resolved these issues, you can safely ignore this message. If you would like additional details about our findings relating to your network, please see “DSA V Report for AS[asn] (IPv[4,6])”: https://imaal.byu.edu/dsav/report/ [anonymous-identifier].

We understand that email messages are sometimes lost or overlooked. For that reason, we will be sending a repeat of this message within a week, to ensure that it is received and the appropriate administrators notified.

Thank you,
IMAAL Research Lab, Brigham Young University

[1] https://imaal.byu.edu

Why am I receiving this email? We selected up to 5 contacts per autonomous system. [Description of how email was found]. If you received this message in error and would rather not receive any further communications relating to this experiment, simply respond to this email and let us know. For any additional questions or concerns, you may email dsav-info@byu.edu.

For the final paragraph, the description depends on the source.

- **WhoisXML API:** “Your email was found through the WhoisXML API. It is likely that your email was used while registering some block of IP addresses within AS[asn].”
- **ARIN database:** “Your email was found through Whois info provided by ARIN. It is likely that your email was used while registering AS[asn].”
- **RIPE:** “Your email was found through Whois info provided by RIPE. It is likely that your email was used while registering AS[asn].”
- **SOA:** “Your email was found by requesting the SOA record associated with one of the DNS resolvers which acted on our spoofed queries.”

As a final note, though the email indicates that we would “be sending a repeat of this message within a week,” we elected not to do so due to the large response we received to the first set of emails.
B. HTML Version

Below is an anonymized example of the HTML-formatted emails that we sent.

---

Your network was found to be vulnerable to infiltration via spoofed-source packets. This email contains a brief description of our experiment and the results specific to your network.

Dear admin of AS[BLANK],

We are researchers from Brigham Young University's Internet Measurement and Anti-Abuse Laboratory [1]. During December 2019, we conducted a large-scale Internet measurement experiment. The intent of this experiment was to determine the pervasiveness of networks failing to filter spoofed incoming traffic appearing to originate from within their own networks. This oversight allows attackers to infiltrate the network and impersonate internal resources. This in turn facilitates attacks which could otherwise be prevented, such as DNS cache poisoning or the NXNS attack, a powerful new denial of service technique. Our methodology relied on sending DNS queries with spoofed source addresses to known DNS resolvers. The queries were for domains under our control; as such if we observed a corresponding query at our authoritative server, we were able to determine that our spoofed queries successfully infiltrated the network.

Unfortunately, it appears that AS[BLANK] is vulnerable to this class of attack of the known resolvers within your autonomous system acted on our spoofed queries, indicating that our spoofed queries successfully infiltrated the network. Even if these resolvers are configured to resolve queries from any host, this indicates a vulnerability, as the traffic would have been blocked at the network border had DSAV been in place. The solution—DSAIV, or Destination-Side Source Address Validation—is to simply filter out incoming traffic that appears to have originated from within the network.

Our work has been accepted to the Internet Measurement Conference (IMC) 2020, which will begin near the end of this month on October 27. While we will present our results in aggregate only and will not specifically mention your autonomous system, we recommend resolving this issue soon before there is greater general awareness of the vulnerability. If you have already resolved these issues, you can safely ignore this message. If you would like additional details about our findings relating to your network, please see DSAIV Report for AS[BLANK].

We understand that email messages are sometimes lost or overlooked. For that reason, we will be sending a repeat of this message within a week, to ensure that it is received and the appropriate administrators notified.

Thank you,
IMAAL Research Lab, Brigham Young University

[1] https://imaal.byu.edu

Why am I receiving this email? We selected up to 5 contacts per autonomous system. Your email was found through the WhoisXML API. It is likely that your email was used while registering some block of IP addresses within AS[BLANK]. If you received this message in error and would rather not receive any further communications relating to this experiment, simply reply to this email and let us know. For any additional questions or concerns, you may email dsav-info@byu.edu.

---

Internet Measurement and Anti-Abuse Laboratory
APPENDIX B
REPORT TEMPLATE

Below is an anonymized example of the reports that were linked in the emails. Clicking “Same ASN” triggers a Javascript function that opens a dropdown listing all the “Same ASN” addresses that reached that IP address.

Destination-Side Source Address Validation (DSAV)
Report for AS[REDACTED] (IPv4)

This page contains a report of the IPv4 DSAV status of AS[REDACTED]. DSAV (destination-side source address validation) is the filtering of spoofed incoming traffic at the network border. AS[REDACTED] was tested and found to be vulnerable. Our methodology consisted of issuing spoofed queries to DNS resolvers within the AS using source IP addresses appearing to originate from the AS itself. Failure to filter spoofed traffic such as this facilitates attacks which could otherwise be prevented, such as DNS cache poisoning.

We ran our experiment in December 2019. Our work has been accepted to the Internet Measurement Conference (IMC) 2020, which will take place late October. While we will present our results in aggregate only and will not specifically mention your autonomous system, we recommend resolving this issue soon before there is greater general awareness of the vulnerability.

For questions questions about this report, please see our Frequency Asked Questions (FAQ) page.

Self-Test Tool

We provide a basic self-test tool. Please note the following:

- The test takes some time to run (~1 minute); please be patient.
- The test will load a one-time report; it will not update the contents of report shown here. In the future, this site and the tester will be more fully-featured, but the moment, it is designed to network administrators see if their network is still vulnerable to spoofing-based intrusion.

Re-Test Now

Internal DNS Resolvers Reached by Spoofed-Source Queries

Each of the IP addresses below acted on our spoofed queries, indicating that our spoofed queries successfully penetrated the network.

5 classes of source addresses were spoofed. These include:

- **Same ASN** Addresses from the same /24 (IPv4) or /64 (IPv6) prefix as the destination (other than the destination itself).
- **Same ASN** Addresses from the same ASN as the destination but outside the /24 or /64 prefix.
- **Private** Addresses from the same /24 (IPv4) or /64 (IPv6) prefix as the destination.
- **Localhost** 127.0.0.1 for IPv4 and ::1 for IPv6.
- **Self** The destination itself used as the source address.
- **Source Port Range** The range of source ports used by the resolver when querying our server and comparing the range used to known ephemeral port selection strategies.

Additionally, for some resolvers we were able to infer the host operating system by either 1) eliciting a TCP-based query to our authoritative servers and analyzing the TCP query using pfctl, or 2) analyzing the range of source ports used by the resolver when querying our server and comparing the range used to known ephemeral port selection strategies.

Finally, we classify each resolver reached as open or closed, based off of whether or not it queried our server in response to an unspoofed query. Note that even reaching an open resolver indicates a lack of DSAV, as the traffic would have been blocked at the network border had DSAV been in place.

<table>
<thead>
<tr>
<th>Destination</th>
<th>Spoofed Sources Reaching Destination</th>
<th>Inferred OS (Source Port Range)</th>
<th>Open</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Same ASN</td>
<td>Windows</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>Same ASN</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Contact

For questions, comments, and additional information, please send email to dsa-vinfo@byu.edu.
### Appendix C
**Response Classification**

Below are examples of actual emails that fit each class/label. [x] indicates content that was removed for anonymity, [...] indicates content that we truncated for brevity.

<table>
<thead>
<tr>
<th>Class/label</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human-Positive</td>
<td>Thank you for doing this research and testing, it’s quite helpful and I hope it is generally recognized as such. Best, [x]</td>
</tr>
<tr>
<td>Human-Neutral</td>
<td>Good day describe in more detail - what exactly do you want? Best regards, [x]</td>
</tr>
<tr>
<td>Human-Negative</td>
<td>Hello, Please stop SPAMING us !!!! best regards, [x]</td>
</tr>
<tr>
<td>Automated</td>
<td>DSAV Project, Thank you for contacting us. This is an automated response confirming the receipt of your ticket. Our team will get back to you as soon as possible. When replying, please make sure that the ticket ID is kept in the subject so that we can track your replies[...]</td>
</tr>
<tr>
<td>Request for more Info</td>
<td>Hello, I’ve received this message from you and I would like to receive further information about the vulnerability you’ve found. I look forward hearing from you. Best regards, [x]</td>
</tr>
<tr>
<td>Disapproval of the emails</td>
<td>Guys, at this rate, what you’re doing is already pissing everyone off with your constant spam. I have this now received 4 times today. Would you tone it down a bit for crying out loud? at this rate, I’m short of clicking to filter your spam out. [x]</td>
</tr>
<tr>
<td>Have/Will Patch</td>
<td>Greetings, I am network engineer for the AS defined in the email you sent below on 10/13/20. I am following up to inform you that we have implemented changes to hopefully resolve issues with this vulnerability. I do not know if you plan to run this experiment or something similar again, but if you do, can you please inform us if you notice this or other vulnerabilities with this AS. Thank You, [x]</td>
</tr>
<tr>
<td>Wrong Email</td>
<td>Hi, Why are we receiving this report ? Our company has nothing to do with this AS number .. [x]</td>
</tr>
<tr>
<td>Forwarded Info</td>
<td>Thanks, I will pass it on to the dns group.</td>
</tr>
<tr>
<td>Disapproval of the experiment</td>
<td>With whose authority did you conduct the research on my network? [x]</td>
</tr>
<tr>
<td>Other language</td>
<td>Dobrý den, Prosím vás, nerozumím anglický, napište český. Děkujeme.</td>
</tr>
</tbody>
</table>