Clusters in the Expanse: Understanding and Unbiasing IPv6 Hitlists

Oliver Gasser
Technical University of Munich
gasser@net.in.tum.de

Quirin Scheitle
Technical University of Munich
scheitle@net.in.tum.de

Pawel Foremski
IITiS PAN
pjf@iitis.pl

Qasim Lone
Grenoble Alps University
qasim.lone@univ-grenoble-alpes.fr

Maciej Korczyński
Grenoble Alps University
maciej.korczynski@univ-grenoble-alpes.fr

Stephen D. Strowes
RIPE NCC
sdstrowes@gmail.com

Luuk Hendriks
University of Twente
luuk.hendriks@utwente.nl

Georg Carle
Technical University of Munich
carle@net.in.tum.de

ABSTRACT

Network measurements are an important tool in understanding the Internet. Due to the expanse of the IPv6 address space, exhaustive scans as in IPv4 are not possible for IPv6. In recent years, several studies have proposed the use of target lists of IPv6 addresses, called IPv6 hitlists.

In this paper, we show that addresses in IPv6 hitlists are heavily clustered. We present novel techniques that allow IPv6 hitlists to be pushed from quantity to quality. We perform a longitudinal active measurement study over 6 months, targeting more than 50 M addresses. We develop a rigorous method to detect aliased prefixes, which identifies 1.5 % of our prefixes as aliased, pertaining to about half of our target addresses. Using entropy clustering, we group the entire hitlist into just 6 distinct addressing schemes. Furthermore, we perform client measurements by leveraging crowdsourcing.

To encourage reproducibility in network measurement research and to serve as a starting point for future IPv6 studies, we publish source code, analysis tools, and data.

CCS CONCEPTS

• **Networks → Network structure**; **Naming and addressing**;

KEYWORDS

IPv6, Hitlist, Clustering, Aliasing, Entropy

ACM Reference Format:
Oliver Gasser, Quirin Scheitle, Pawel Foremski, Qasim Lone, Maciej Korczyński, Stephen D. Strowes, Luuk Hendriks, and Georg Carle. 2018. Clusters in the Expanse:, Understanding and Unbiasing IPv6 Hitlists. In *2018 Internet Measurement Conference (IMC '18), October 31-November 2, 2018, Boston, MA, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3278532.3278564

IMC '18, October 31-November 2, 2018, Boston, MA, USA

ACM ISBN 978-1-4503-5619-0/18/10…$15.00
https://doi.org/10.1145/3278532.3278564

1 INTRODUCTION

Internet scanning has a rich history of generating insights for security, topology, routing, and many other fields. Advances in software and link speeds in recent years allow the entire IPv4 Internet to be easily scanned in just a few minutes [2, 24, 42]. However, scanning the expanse of the entire IPv6 Internet is infeasible due to its size, which is magnitudes above both what can technically be sent or stored, and what is an ethical volume of queries to be sent to a system or network. Therefore, state-of-the-art IPv6 Internet scanning resorts to the methods used in the early days of IPv4 Internet scanning, i.e., using lists of target IP addresses, so-called hitlists, which served as a representative subset of the IPv4 address space [19, 21, 27].

The IPv6 address space also comes with unique, different challenges to such hitlists. First, hitlists can be biased (i.e., not representative of the Internet as a whole) due to imbalanced Autonomous System (AS) and prefix representations or IP address aliasing. Second, due to similarly large allocation sizes, a single network—or even a single machine[1]—can easily overwhelm a hitlist with countless IP addresses. Third, addresses might be used only for very brief periods of time, as there is no pressure for re-use. Thus, a key quality of IPv6 hitlists is not the count of IP addresses, but responsiveness and balance over ASes and prefixes. In this paper, we systematically tackle these challenges by:

Comprehensive Address Discovery: The first step in unbiasing a hitlist is creating a *comprehensive* hitlist, for which we draw IP addresses from a multitude of state-of-the-art sources, cf. Section 3.

Clustering by Entropy: To discover and understand clusters in the expanse of the IPv6 space, we leverage entropy analysis of IPv6 addresses. This helps to determine addressing schemes and aggregate clusters, which we explore in Section 4.

De-Aliasing: To reduce the potential impact of aliased prefixes—i.e., a single machine responding to all addresses in a possibly large prefix—we postulate and implement a rigorous method for aliased prefix detection, which we present in Section 5.

Longitudinal Stability Probing: To find reliably responsive addresses, we conduct longitudinal scans for our hitlist across several protocols. As expected, we find only a fraction of discovered

[1] We have indeed seen a web server responding to an entire /32 prefix, i.e., 2^{96} addresses.

IP addresses to actually respond to probing, which is an important filtering criterion for curation of an unbiased hitlist (cf. Section 6).

Diversifying Address Population: We also evaluate three orthogonal methods to push the frontier of IPv6 hitlists: generating addresses using Entropy/IP [33] and 6Gen [56] (in Section 7), leveraging reverse DNS records [29] (in Section 8), and crowdsourcing client IP addresses (in Section 9).

Plotting for Exploratory Analysis: Visualizing IPv6 datasets is challenging, as IPv4 approaches, such as describing the entire address space with a Hilbert curve, do not scale for IPv6. We present a new plotting technique that works with selective input, e.g., prefixes announced in BGP, instead of visualizing the entire address space. We explain how to interpret these plots in Section 3, and use it throughout the paper to give an intuitive view of our data.

Publication and Sharing: Our open sharing of a reliable IPv6 hitlist has already supported various scientific studies [5, 7, 16, 33–35, 39, 44, 66, 68, 69]. We also publicly share daily snapshots of the curated and unbiased IPv6 hitlist created in this work.

Throughout our work, we aim to adhere to highest of ethical standards (cf. Section 10.1), and aim for our work to be fully reproducible. We share code and data at:

https://ipv6hitlist.github.io

2 PREVIOUS WORK

Recent studies find that there are hundreds of million active IPv4 addresses [14, 20, 62, 79]. This densely populated IPv4 space is well suited for brute-force measurement approaches by scanning the complete address space [2, 24]. The IPv6 space, however, is extremely sparse. We survey previous work on targeting the sparse IPv6 space and compare our work with the state of the art in Table 1.

DNS Techniques: The DNS was long known to be a possible source for IPv6 addresses [40, 76]. Strowes proposes using the *in-addr.arpa* IPv4 rDNS tree to gather names that may be resolved to IPv6 addresses [73], on an assumption that naming is common between protocols. This discovered 965 k IPv6 addresses located in some 5531 ASes, many of which (56.7 %) were responsive. More recently, Fiebig et al. walked the rDNS tree to obtain 2.8 M IPv6 addresses [29, 30]. They did not probe these addresses, leaving the question of responsiveness open. Borgolte et al. find IPv6 addresses by NSEC-walking DNSSEC signed reverse zones [17].

Table 1: Comparing this work with four previous works (ordered chronologically) based on the following metrics: number of addresses from public sources (#publ.), BGP prefixes (#pfx.), ASes; addresses from private sources (#priv.); include client addresses (Cts), perform active probing (Prob.), perform aliased prefix detection (APD).

Previous work	#publ.	#pfx.	#ASes	#priv.	Cts	Prob.	APD
Gasser et al. [36]	2.7 M	5.8 k	8.6 k	149 M	✓	✓	✗
Foremski et al. [33]	620 k	<100[1]	<100[1]	3.5 G	✓	✓	✗
Fiebig et al. [29]	2.8 M	n/a[2]	n/a	0	✓	✗	✗
Murdock et al. [56][3]	1.0 M	2.8 k	2.4 k	0	✓	✓	○
This work	55.1 M	25.5 k	10.9 k	0	✓	✓	✓

1: 15 networks, with few prefixes and ASes. 2: 582 k /64s. 3: Responsive addresses.

In this work, we evaluate the responsiveness of IPv6 rDNS as a source and find rDNS IPv6 addresses a valuable addition to a hitlist.

Structural Properties: The IPv6 address space may be sparsely populated, but address plans inside networks tend to indicate structure. Recent related work leverages the structure of IPv6 addressing schemes to find new addresses. Ullrich et al. use rule mining to find a few hundred IPv6 addresses [75]. With Entropy/IP, Foremski et al. present a machine learning approach which trains on collected IPv6 addresses to build an addressing scheme model and generate new addresses [33]. In this work, we refine Entropy/IP to generate IPv6 addresses for probing, and we introduce a new entropy clustering technique. Similar to efforts leveraging dense address areas in IPv4 [47], Murdock et al. presented 6Gen to find dense regions in the IPv6 address space and generate neighboring addresses [56]. In this work, we use 6Gen to generate addresses based on our hitlist and compare its performance with Entropy/IP. Murdock et al. also performed a basic variant of aliased prefix detection (APD), which we extend in this work. Plonka and Berger harness the structure from IPv6 address plans to allow large datasets to be shared [59].

Hitlists: Gasser et al. [36] assemble a hitlist from a multitude of sources. The vast majority of their 149 M IPv6 addresses, however, are obtained from non-public passive sources. We build upon their approach, but exclusively use publicly available sources in order to make our work reproducible. Recently, Beverly et al. analyze the IPv6 topology using large-scale traceroutes, leveraging Gasser et al.'s public hitlist among other sources [16].

Crowdsourcing: There are a few studies that have leveraged crowdsourcing platforms to perform network measurements [45, 50, 77]. We build on prior work by Huz et al. [45], who used the Amazon Mechanical Turk platform to test broadband speeds and IPv6 adoption. Compared to Huz et al.'s 38 IPv6 addresses collected in 2015, we find many more.

3 IPV6 HITLIST SOURCES

We leverage a variety of sources, for which we provide an overview in Table 2. Our guiding principle in selecting sources was that they should be *public*, i.e., accessible to anyone for free, in order to make our work reproducible and to allow fellow researchers to deploy variations of our IPv6 hitlist. We consider data with an open and usually positive access decision process as public, such as Verisign's process to access zone files. We also aim to have balanced sources, which include servers, routers, and a share of clients. The sources we leverage are as follows:

Domain Lists: Described in [7, 35, 66], with a total of 212 M domains from various large zones, resolved for AAAA records on a daily basis, yielding about 9.8 M unique IP addresses. This source also includes domains extracted from blacklists provided by Spamhaus [72], APWG [10], and Phishtank [58], which leverage 8.5 M, 376 k, and 170 k domains, respectively.

FDNS: A comprehensive set of forward DNS (FDNS) ANY lookups performed by Rapid7 [61], yielding 2.5 M unique addresses.

CT: DNS domains extracted from TLS certificates logged in Certificate Transparency (CT), and not already part of domain lists, which yields another 16.2 M addresses.

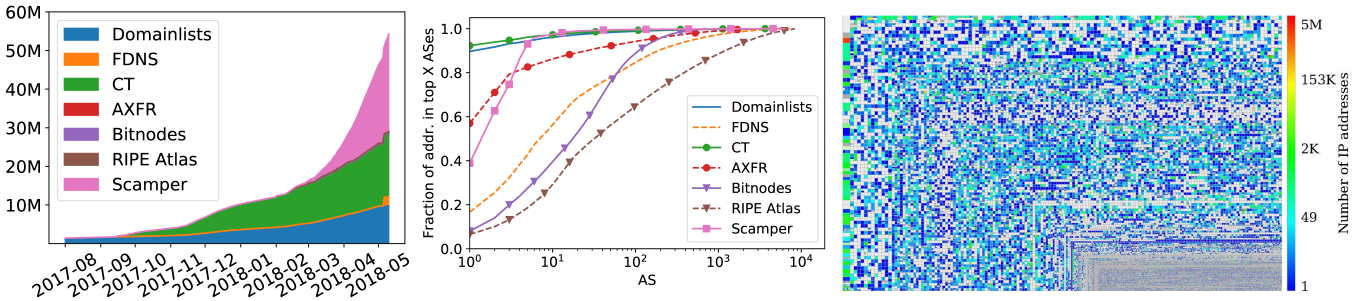
AXFR and TLDR: IPv6 addresses obtained from DNS zone transfers (AXFR) from the TLDR project [53] and our own AXFR

Table 2: Overview of hitlist sources, as of May 11, 2018.

Name	Public	Nature	IPs	new IPs	#ASes	#PFXes	Top AS1	Top AS2	Top AS3
DL: Domain Lists[1]	Yes	Servers	9.8 M	9.8 M	6.1 k	10.3 k	89.7 % ★	2.0 % ●	1.5 % ■
FDNS: Rapid7 FDNS	Yes	Servers	3.3 M	2.5 M	7.7 k	13.6 k	16.7 % ★	8.9 % ▲	6.7 % ✣
CT: Domains from CT logs[2]	Yes	Servers	18.5 M	16.2 M	5.3 k	8.7 k	92.3 % ★	1.6 % ✣	0.8 % ★
AXFR: AXFR&TLDR	Yes	Mixed	0.7 M	0.5 M	3.2 k	4.7 k	57.0 % ★	14.0 % ●	8.3 % ■
BIT: Bitnodes	Yes	Mixed	31 k	27 k	695	1.4 k	8.0 % ★	6.0 % ■	6.0 % ▲
RA: RIPE Atlas[3]	Yes	Routers	0.2 M	0.2 M	8.4 k	19.1 k	6.6 % ✣	3.5 % ★	3.1 % ✣
Scamper	–	Routers	26.0 M	25.9 M	6.3 k	9.8 k	38.9 % ★	23.8 % ●	12.0 % ■
Total			58.5 M	55.1 M	10.9 k	25.5 k	45.4 % ★	18.4 % ★	11.5 % ●

1: Zone Files, Toplists, Blacklists (partially with NDA); 2: Excluding DNS names already included in Domain Lists; 3: Traceroute and ipmap data

★Amazon, ●Host Europe, ■Cloudflare, ▲Linode, ✣DTAG, ★ProXad, ●Hetzner, ■Comcast, ▲Swisscom, ✣Google, ★Antel, ●Versatel, ■BIHNET



(a) Cumulative runup of IPv6 addresses.

(b) AS distribution for hitlist sources.

(c) Hitlist addresses mapped to 56 k BGP prefixes.

Figure 1: Hitlist sources runup, AS distribution as CDF, and zesplot.

transfers. Obtained domain names are also resolved for AAAA records daily. This source yields 0.5 M unique IPv6 addresses.

Bitnodes: To gather client IPv6 addresses, we use the Bitnodes API [78], that provides current peers of the Bitcoin network. Although this is the smallest source, contributing 27 k unique IPv6 addresses, we still find it valuable as it also adds client addresses.

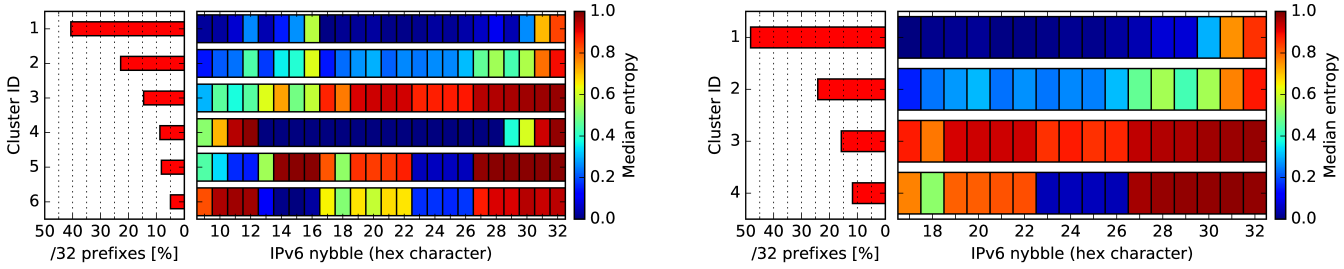
RIPE Atlas: We extract all IPv6 addresses found in RIPE Atlas traceroutes, as well as all IPv6 addresses from RIPE's ipmap project [63], which adds another 0.2 M addresses. These are highly disjoint from previous sources, likely due to their nature as routers.

Scamper: Finally, we run traceroute measurements using scamper [51] on all addresses from other sources, and extract router IP addresses learned from these measurements. This source shows a very strong growth characteristic, with 25.9 M unique IP addresses.

We accumulate all sources, i.e., IP addresses will stay indefinitely in our scanning list. We may revisit this decision in the future, and remove IP addresses after a certain window of unresponsiveness.

Address Runup: Figure 1a shows the cumulative runup of sources over time. First, we can see a strong growth of IPv6 addresses in all sources: typically an increase by factors of 10–100 over the course of a year. Second, DNS-based sources—likely revealing server addresses—together with traceroute addresses obtained from scamper dominate the overall dataset. As we find scamper's explosive growth peculiar, we conduct a closer investigation, which reveals that 90.7 % of those IP addresses are SLAAC addresses, i.e., marked by `ff:fe`. The vendor codes in MAC addresses gained from those routers indicate that they are typically home routers: 47.9 % ZTE and 47.7 % AVM (Fritzbox), followed by 1.2 % Huawei with a long tail of 240 other vendors. This shows that our source includes mainly home routers and CPE equipment. Depending on the type of study, it may be desirable to include or exclude these CPE devices.

zesplot: To explore these large amounts of data, we present visualizations called *zesplots*. A zesplot visualizes IPv6 prefixes, each represented as a rectangle. It does not show the entire IPv6 address space, only the prefixes provided as input. It uses a space-filling algorithm based on *squarified tree-maps* [18], extended with recursive properties. It starts out by filling a vertical row with rectangles, then a horizontal row, then a vertical row again, and so on. The prefixes are ordered based on {prefix-size,ASN}, so that large prefixes are plotted in the top-left corner of the graph, and smaller prefixes in the bottom-right corner, keeping similarly sized prefixes from the same AS adjacent. Consequently, a prefix will be in the same spot in different zesplots, as long as the input prefixes are the same. Axes have no meaning in a zesplot. More-specific subprefixes are plotted in the top half of that prefix's rectangle (a prefix with many more-specifics might result in a gray rectangle when zoomed-out). A white rectangle means no addresses are present in that prefix. For example, in Figure 1c, we find a bright red /19 in the top-left, /32s around the center of the plot, and very specific prefixes like /127s in the bottom-right part. In addition to the list of prefixes, the tool



(a) Fingerprints of full addresses, $\mathbf{F}^{9}_{32}$. **(b) Fingerprints of IIDs, $\mathbf{F}^{17}_{32}$.**

Figure 2: /32 prefixes clustered using entropy fingerprints.

takes a list of addresses, and colors the prefix rectangles based on the number of addresses that belong to that prefix. With these colors, one can quickly spot prefixes which are possibly over-represented in the dataset, or verify certain assumptions (e.g., larger prefixes cover more addresses than smaller prefixes). Depending on the dataset, an *unsized* zesplot might provide extra insights on e.g., clusters of prefixes: in these plots, all rectangles are equivalently sized, and the prefix size is only used for sorting. We publish the zesplot tool [43] aiming at more applications in measurement research.

Input Distribution: When evaluating the AS distribution for each source in Figure 1b, we see stark differences, e.g., for domainlists and CT only a handful of ASes make up a large fraction of addresses, compared to the more balanced RIPE Atlas source. In addition, we analyze the distribution of hitlist addresses to BGP prefixes using zesplot in Figure 1c. We cover half of all announced BGP prefixes, but we find that some prefixes contain unusually large numbers of addresses, which we investigate in Section 5.

Comparison with DNSDB: Our hitlist covers 12.9 % of IPv6 addresses, 69.4 % of ASes, and 48.7 % of BGP prefixes belonging to AAAA records stored in DNSDB [28]. The majority of "missing" addresses belong to large CDN operators, which is probably due to DNSDB collecting passive DNS data globally, versus active probing from a few locations. Moreover, the majority of addresses in our hitlist are not in DNSDB, especially for infrastructure of ISP operators. We find our hitlist and DNSDB to be complementary. We do not add DNSDB to our sources as it is not publicly available.

Going Beyond: Besides the aforementioned daily scanned sources, we also conduct in-depth case studies on three disjunct input sources: newly learned IPv6 addresses using 6Gen and Entropy/IP in Section 7, rDNS-walked IPv6 addresses in Section 8, and crowdsourced client IPv6 addresses in Section 9.

4 ENTROPY CLUSTERING

We introduce a method for grouping IPv6 networks by similar entropy in their addresses and evaluate it on our IPv6 hitlist.

Let S be a set of IPv6 addresses in a particular network, e.g., a /32 prefix, a BGP prefix, or an AS. The set may be a random sample, but with at least 100 addresses. We introduce the following notation:

$$S = \{ \mathbf{A}_1, \cdots, \mathbf{A}_i, \cdots, \mathbf{A}_n \},\ n \geq 100 \tag{1}$$

$$\mathbf{A}_i = (x^i_1, \cdots, x^i_j, \cdots, x^i_{32}), \tag{2}$$

$$x^i_j \in \Omega = \{\text{`0'}, \cdots, \text{`}f\text{'}\}, \tag{3}$$

where $\mathbf{A}_i$ is an address in that network: a sequence of 32 nybbles, i.e., hex characters. Let X_j be a discrete random variable on Ω representing nybble j across S, and have an empirical probability mass function $\hat{P}(X_j)$. Next, we compute the normalized Shannon entropy of X_j, which we call an entropy fingerprint $\mathbf{F}^a_b$:

$$\mathbf{F}^a_b = (H(X_a), \cdots, H(X_j), \cdots, H(X_b)) \tag{4}$$

$$H(X_j) = \frac{1}{4} \cdot - \sum_{\omega \in \Omega} \hat{P}(X_j = \omega) \cdot log\hat{P}(X_j = \omega). \tag{5}$$

where a and b are the first and the last considered nybbles, respectively. Note that if $H(X_j) = 0$, then nybble j is constant; if $H(X_j) = 1$, then all its values are equally probable.

We repeat the above for each network, obtaining a dataset of fingerprints. Next, we run the k-means algorithm on the obtained dataset to find clusters of networks with similar fingerprints. We use the well-known elbow method to find the number of clusters, k, plotting the sum of squared errors (SSE) for $k = \{1, \cdots, 20\}$:

$$SSE(k) = \sum_{c \in C_k} \sum_{\mathbf{E} \in c} d^2(\mathbf{E}, \overline{\mathbf{c}}) \tag{6}$$

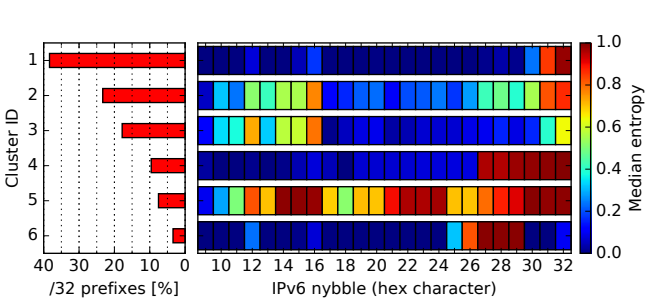
where C_k is the set of clusters obtained for given k, $\overline{\mathbf{c}}$ is the cluster mean, and d^2 is the squared Euclidean distance. We select the k for which we see an "elbow" in the plot, i.e., the point where increasing k does not yield a relatively large reduction in SSE.

Finally, we summarize each cluster graphically with its median entropy on each nybble and with its relative popularity.

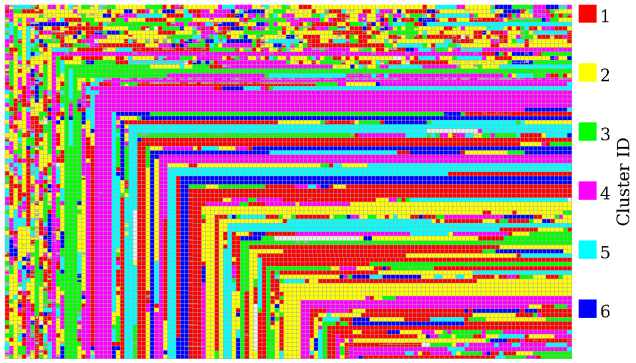
4.1 Results

We present the results of entropy clustering on /32 prefixes in our hitlist in Figure 2, for two different fingerprint lengths. Each plot shows cluster popularity on the left-hand side, and the median entropy of each nybble on the right-hand side. Clusters are represented by rows, ordered top-down by their popularity.

Most notably, in Figure 2a, we identify just 6 clusters of full address fingerprints. The most popular cluster has entropy ≈ 0 on all nybbles but a few at the end of network and interface identifiers (IIDs). It is likely an artifact of a common practice to treat these parts of IPv6 addresses as simple counters. The second most popular pattern is similar, but uses more nybbles and introduces more structure. In cluster 3, we see prefixes with pseudo-random IIDs, manifested in high entropy, which makes predicting addresses in such networks impossible. Finally, we see MAC-based IIDs in clusters 5 and 6, where nybbles 23-26 are likely just `ff:fe`.



(a) /32 prefixes with addresses responding to UDP/53, clustered using entropy fingerprints of full addresses, F^9_{32}.



(b) 22 k BGP prefixes colored using their F^9_{32} clusters.

Figure 3: Entropy clustering for DNS responsive hosts and cluster distribution in BGP prefixes.

In Figure 2b, we focus on IIDs only, i.e., we limit fingerprints to nybbles 17-32, and find just 4 clusters. Again, the majority of evaluated networks use IIDs as counters, as visible in clusters 1 and 2. This is expected and common, e.g., for pools of servers, but shows potential for probabilistic scanning. We see the impact of SLAAC in clusters 3 and 4: pseudo-random and MAC-based, respectively.

Finally, we use entropy clustering throughout the paper to help with the interpretation of our results. First, we try to better understand the IPv6 scanning results detailed in Section 6. In Figure 3a, we present clusters with addresses that respond to a UDP/53 (DNS) scan. We find that most clusters exhibit low entropy on all but a few nybbles. This phenomenon makes probabilistic scanning for IPv6 DNS servers easy, which we demonstrate in Section 7.

In Figure 3b, we show how clusters are distributed to BGP prefixes. The prefixes are ordered by their length and origin AS, but the box size is static instead of based on the prefix length, which helps in pattern spotting. We plot only prefixes belonging to ASes with more than 100 addresses. With the order running from top-left (larger prefixes) to bottom-right (smaller prefixes), we find that the mix of clusters is more heterogeneous for larger prefixes. Within the smaller prefixes, we observe greater consistency. Most of the large, equally colored chunks are equally sized prefixes in a single AS. This hints at operators using the same addressing scheme, or deploying equivalent equipment or setups, in their prefixes.

4.2 Discussion

We highlight that entropy clustering is different from Entropy/IP [33]. Although both algorithms use Shannon entropy, our method finds high-level patterns *across* networks, whereas Entropy/IP finds low-level patterns *within* a network. However, the methods are complementary: for instance, entropy clustering can help in spotting networks susceptible to probabilistic scanning, while Entropy/IP can generate a hitlist for each of them.

We also stress that entropy clustering is not confined to /32 prefixes, or to the fingerprint lengths we presented in the paper. Note that /32 prefixes are commonly the smallest blocks assigned to IPv6 networks [3, 9, 11, 48, 64]. Using different entropy clustering parameters allows the different particularities of IPv6 deployment to be investigated. We provide supplemental results obtained from clustering based on ASes, BGP prefixes, and other fingerprints online. Similarly, using other clustering tools will lead to different results. We present a baseline set of parameters and tools based on common practice, leaving other options for future work.

In summary, entropy clustering is a new technique that simplifies Internet studies. Instead of treating IPv6 as an opaque and expansive addressing space, we can visualize actual addressing patterns and their popularity in one picture. Our results have implications for scanning the IPv6 Internet. For instance, the clusters in Figure 2 suggest people in general use IPv6 addresses in a limited number of ways, and that many IPv6 address nybbles are easy to predict. On the other hand, cluster *popularities* represent our hitlist—and to some extent, IPv6 hitlists in general—rather than the Internet.

5 ALIASED PREFIX DETECTION

Aliased network prefixes, i.e., prefixes under which each possible IP address replies to queries, were already found when conducting IPv4 measurements [6]. For IPv6 measurements, however, aliased prefixes pose a much more significant challenge as they can easily contribute vast numbers of addresses that map to the same server, e.g., through the `IP_FREEBIND` option in Linux. This feature is already in use by CDNs [52], and was identified as a challenge in previous works [29, 56]. Aliased prefixes can artificially inflate the number of IP addresses within a hitlist (e.g., enumerating a /96 prefix can add 2^{32} addresses), and introduce significant bias into any studies using these hitlists. Given this, we want to populate our hitlist only with valuable addresses, i.e., addresses belonging to different hosts and having balanced prefix and AS distributions. This requires reliable detection and removal of aliased prefixes, for which we introduce a rigorous method in the following.

Similar to [29, 56], our method has its roots in the concept that a randomly selected IP address in the vast IPv6 space is unlikely to respond. Thus, when probing randomly selected addresses, a prefix can be classified as aliased after a certain number of replies have been received. Murdock et al. [56] send three probes each to three random addresses in every /96 prefix. Upon receipt of replies from all three random addresses, the prefix is determined as aliased. In the following, we describe how we improve efficiency and effectiveness of this approach in several ways.

Table 3: Example of IPv6 fan-out for multi-level aliased prefix detection. We generate one pseudo-random address in `2001:0db8:407:8000::/64` for each of the 16 subprefixes, i.e., `2001:0db8:407:8000:[0-f]/68`.

2001:0db8:0407:8000::/64
2001:0db8:0407:8000:0151:2900:77e9:03a8
2001:0db8:0407:8000:181c:4fcb:8ca8:7c64
2001:0db8:0407:8000:23d1:5e8e:3453:8268
⋮
2001:0db8:0407:8000:f693:2443:915e:1d2e

Alias detection needs to fulfill two criteria to scale: (1) detection must be low-bandwidth, with a small number of packets required per network, (2) detection must function for end hosts, not only routers, which excludes many alias detection techniques.

5.1 Multi-Level Aliased Prefix Detection

For our daily scans, we perform multi-level aliased prefix detection (APD), i.e., detection at different prefix lengths. This is in contrast to previous works that use static prefix lengths, e.g., /96.

To determine whether a prefix is aliased, we send 16 packets to pseudo-random addresses within the prefix, using TCP/80 and ICMPv6. For each packet we enforce traversal of a subprefix with a different nybble. For example, to check if `2001:db8:407:8000::/64` is aliased, we generate 1 pseudo-random address for each 4-bit subprefix, `2001:db8:407:8000:[0-f]000::/68`. See Table 3 for a visual explanation. Using this technique we ensure that (1) probes are distributed evenly over more specific subprefixes and (2) pseudo-random IP addresses, which are unlikely to respond, are targeted. For each probed prefix we count the number of responsive addresses. If we obtain responses from all 16 probed addresses, we label the prefix as aliased.

We run the aliased prefix detection on IPv6 addresses that are either BGP-announced or in our hitlist. The former source allows us to understand the aliased prefix phenomenon on a global scale, even for prefixes where we do not have any targets. The latter source allows us to inspect our target prefixes more in-depth.

For BGP-based probing, we use each prefix as announced, without enumerating additional prefixes. For our hitlist, we map the contained addresses to all prefixes from 64 to 124, in 4-bit steps. We limit APD probing to prefixes with more than 100 targets for two reasons: First, efficiency, as APD probing requires 32 probes (16 for ICMPv6 and TCP/80, respectively). Second, impact, as prefixes with less than 100 probes can only distort our hitlist in a minor way. We exempt /64 prefixes from this limitation so as to allow full analysis of all known /64 prefixes. We use 47.4 M probes to guarantee complete coverage of all /64 prefixes and 49.2 M probes in total.

As we perform target-based APD at several prefix lengths, the following four cases may occur:

(1) Both more and less specific are aliased
(2) Both more and less specific are non-aliased
(3) More specific aliased, less specific non-aliased
(4) More specific non-aliased, less specific aliased

The first two cases depict the "regular" aliased and non-aliased behaviors, respectively. The third case is more interesting as we observe divergent results based on the prefix length that we query. One example is a /96 prefix which is determined as being non-aliased, with only 9 out of 16 /100 subprefixes determined as aliased. This case underlines the need for our fan-out pseudo-random aliased prefix detection. Using purely random addresses, all 16 could by chance fall into the 9 aliased subprefixes, which would then lead to incorrectly labeling the entire /96 prefix as aliased. The fourth case is an anomaly, since an aliased prefix should not have more specific non-aliased subprefixes. One reason for this anomaly is packet loss for subprefix probes, incorrectly labeling the subprefix as non-aliased.

We analyze how common the fourth case is in our results and investigate the reasons. On May 4, 2018, we detect only eight such cases at the prefix lengths /80, /116, and /120:

The /80 prefix shows 3 to 5 out of the 16 possible responses over time. The branches of responding probes differ between days, with no discernible pattern. We suspect this prefix is behind a SYN proxy [25], which is activated only after a certain threshold of connection attempts is reached. Once active, the SYN proxy responds to every incoming TCP SYN, no matter the destination.

The /116 prefix consistently shows 15 out of 16 probes being answered on consecutive days, even though a less specific prefix was classified as aliased. Moreover, the 15 probes answer with the same TCP options on consecutive days. The non-responding probe is always on the `0x0` branch, so we believe the subprefix is handled differently and not by an aliased system. In fact, comparing the paths of the different branches reveals that the `0x0` branch is answered by an address in a different prefix. The DNS reverse pointer of this address hints at a peering router at DE-CIX in Frankfurt, Germany. This /116 anomaly underlines the importance of the multi-level aliased prefix detection, since there are in fact small non-aliased subprefixes within aliased less specific prefixes.

The case of six neighboring /120 prefixes manifests less consistently than the previously described phenomenon. The branches that lack responses change from day to day, as well as from prefix to prefix. Subsequent manual measurements show that previously unresponsive branches become responsive. The root cause is most likely ICMP rate limiting, which explains the seemingly random responding branches. We try to counter packet and ICMP-rate limiting loss as explained in Section 5.2.

After the APD probing, we perform longest-prefix matching to determine whether a specific IPv6 address falls into an aliased prefix or not. This ensures we use the result of the most closely covering prefix for each IPv6 address, which creates an accurate filter for aliased prefixes. If a target IP address falls into an aliased prefix, we remove it from that day's ZMapv6 and scamper scans.

5.2 Loss Resilience

Packet loss might cause a false negative, i.e., an aliased prefix being incorrectly labeled as non-aliased. To increase resilience against packet loss, we apply (1) cross-protocol response merging and (2) a multi-day sliding window.

As we are probing all 16 target addresses on ICMPv6 and TCP/80, IP addresses may respond inconsistently. Our technique hinges

Table 4: Impact of sliding window on unstable prefix count.

Sliding window	0	1	2	3	4	5
Unstable prefixes	65	26	22	14	14	13

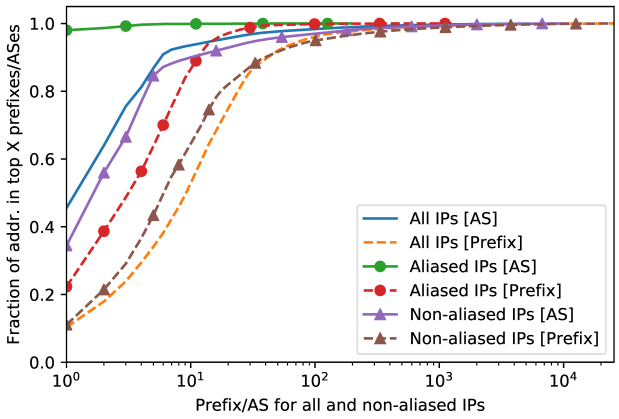


Figure 4: Prefix and AS distribution for aliased, non-aliased, and all hitlist addresses.

on the fact that it is unlikely for a randomly chosen IP address to respond at all, so we treat an address as responsive even if it replies to only the ICMPv6 or the TCP/80 probe. While this greatly stabilizes our results, we still see high-loss networks, which would fail automatic detection, but could be manually confirmed as aliased.

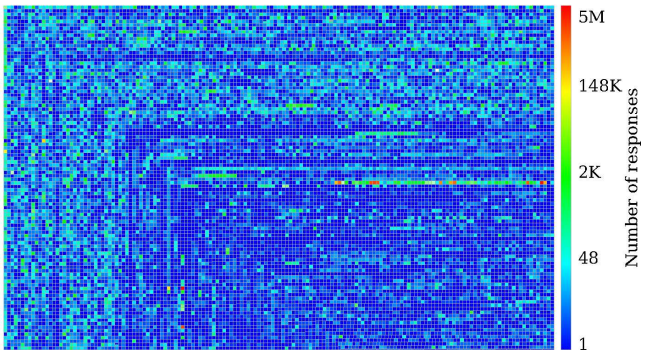
To further tackle these, we introduce a sliding window over several past days, and require each IP address to have responded to any protocol in the past days. As prefixes may change their nature, we perform this step very carefully, and aim for a very short sliding window to react to such changes as quickly as possible.

To find an optimum, we compare the number of days in the sliding window to the number of prefixes that are unstable, i.e., change the nature of stable and unstable over several days. We show the data in Table 4, which confirms that with a sliding window of just 3 days, we can reduce the number of unstable prefixes by almost 80 %, while only adding a small delay for prefixes that change their nature. With the final sliding window of 3 days, only 14 of the 909 aliased prefixes as of May 1, 2018 show an unstable nature.

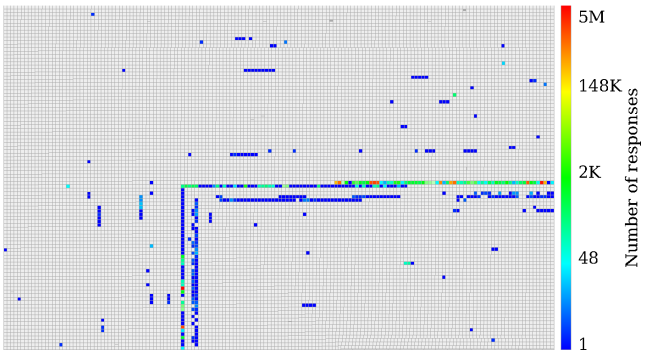
5.3 Impact of De-Aliasing

We apply the aliased prefix detection (APD) filtering daily and analyze the impact on our hitlist for May 11, 2018. Before filtering, there are a total of 55.1 M IPv6 addresses on our hitlist. After identifying aliased prefixes, 29.4 M targets (53.4 %) remain. With the unfiltered hitlist we cover 10,866 ASes and 25,465 announced prefixes. Removing aliased prefixes reduces our AS coverage by only 13 ASes, and lowers prefix coverage by 3.2 % to 24,648 prefixes.

We show the AS and prefix distribution for aliased, non-aliased, and all IPv6 addresses in Figure 4. By comparing AS distributions we find aliased prefixes as heavily centered on a single AS (Amazon). In consequence, the AS distribution for non-aliased prefixes is flatter than the population as a whole. The picture changes for prefix distributions: targets in non-aliased prefixes are now slightly more top-heavy compared to the general population. One of the reasons is that the vast majority of aliased IP addresses are within 189 /48 prefixes announced by Amazon, which are the shortest detected aliased prefixes. This results in shifting down the prefix distribution of the general population.



(a) 16 k prefixes without aliased prefix detection.



(b) 461 (3.0 % of 16 k) detected aliased prefixes.

Figure 5: Responses to ICMP Echo requests.

To further visualize the effect of filtering aliased prefixes, we compare Figure 5a with Figure 5b. In these figures, equally sized boxes are plotted for every prefix, ordered by prefix length and ASN. We find that aliasing barely occurs in the shortest prefixes, plotted in the top left corner of the plot. Towards the lower right end—i.e., the longer prefixes—we find some groups of aliased /32 prefixes, but most eye-catching are the groups forming the big "hook". These are all /48 prefixes, with the majority belonging to Amazon (the "outer" hook) and Incapsula (the "inner" hook). We also see that filtering these prefixes is effective: the brightly colored Amazon aliased prefixes comprise a large share of the input set.

5.4 Fingerprinting Aliased Prefixes

The reason for not scanning aliased prefixes is that each contained IP address is assumed to belong to the same host, and consequently to display the same properties. We investigate this assumption by deploying fingerprinting techniques in our probes. Employing a ZMap module that supports the TCP Options header [65], we send a set of commonly supported fingerprinting options (`MSS-SACK-TS-WS`), setting MSS and WS to 1 to trigger differing replies [68].

Fingerprinting is a fuzzy and challenging technique, which is why we carefully evaluate results and consider them indicative rather than conclusive. We consider this as a case study for validation, and do not feed the results back to scanning.

Below, we investigate replies from 20,692 /64 prefixes classified as aliased, for which all of our 16 APD probes to TCP/80 succeeded on May 11, 2018. We first analyze TTL values of response packets. Previous work found that TTL values cannot be expected to be constant per prefix or even IP address [13, 67]. TTL inconsistencies can stem from routing changes, TTL-manipulating middleboxes, or other on-path effects. We quickly re-appraise this sentiment with our data, and can confirm that 5970 of our 20,692 prefixes offer inconsistent TTL values. We hence replace the raw TTL metric with the likely initial TTL (iTTL) value chosen by a host.

iTTL: Rounding the TTL value up to the next power of 2 results in the iTTL value [13, 46, 55]. Using iTTL, we find only 6 prefixes with inconsistent behavior, all caused by 22 IP addresses responding with differing iTTL values to our 2 consecutive probes. These addresses belong to only 2 /48 prefixes, announced by 2 not commonly known ASes. As the iTTL can be only one of four values (32, 64, 128, or 255), we use differing iTTL as a negative indicator for APD: many different iTTL values suggest a non-aliased prefix. With only 0.03 % of inconsistent prefixes, we remain confident in our APD filtering.

Optionstext: We next evaluate a metric used by [15, 68] that translates TCP options into a string, which preserves the order of options and padding bytes, but not the option values. For example, the string `MSS-SACK-TS-N-WS` would represent a packet that set the Maximum Segment Size, Selective ACK, Timestamps, a padding byte, and Window Scale options. Although we found 99.5 % of responsive hosts to choose that set of options, we identify 104 prefixes that return differing sets of TCP options, an unlikely behavior for a prefix aliased to the same machine.

WScale and WSize: The next metric is TCP window size and TCP window scale option. These options are also not necessarily expected to stay constant, as changes in host state can lead to advertising varying window sizes. However, a consistent window across all IP addresses in a prefix again raises our confidence in our technique. We find 1068 prefixes with inconsistent TCP window sizes, and 105 prefixes with inconsistent window scale options: a sizable number, but amounting to only ≈ 5 % of our aliased prefixes.

MSS: Like iTTL, the TCP Maximum Segment Size is used as a negative indicator. Inconsistent MSS values are determined for 1030 prefixes. This behavior is, as for previous metrics, typically caused by individual IP addresses sending differing MSS sizes.

Timestamps: Finally, we evaluate TCP Timestamps, as suggested by [15, 68]. Although TCP Timestamps offer highly discriminative features, they cannot reliably identify a prefix as non-aliased with only 2 probes per host, due to the variety of possible behaviors, e.g., randomized start values. They can, however, strengthen our confidence that an aliased prefix is behaving consistently. We hence run the following checks: (1) whether all hosts send the same (or missing) timestamps, (2) whether timestamps are monotonic for the whole prefix, and (3) whether the receive timestamp and remote TCP timestamp have a regression coefficient $R^2 > 0.8$. This tests for a global linear counter, strongly hinting that the queried IP addresses belong to the same machine [15, 68]. If any of these three tests succeeds, we consider a prefix to offer consistent behavior, which is a strong indicator for aliasing.

Table 5: Fingerprinting 20.7 k aliased prefixes: inconsistent prefixes per test, in total, and total consistent.

Test	Incs.	Σ Incs.	Σ Cons.
iTTL	6	6	20,686
Optionstext	104	110	20,581
WScale	105	215	19,515
MSS	1030	1175	19,513
WSize	1068	1186	19,506
Timestamps	n/a[1]	n/a[1]	13,202

[1] A failed timestamping test does not indicate an inconsistent prefix, but an indecisive one.

Table 6: Validation: aliased prefixes are considerably less inconsistent and pass many more consistency checks, including timestamping.

Scan type	Incons.	Cons.	Indec.
Non-aliased prefixes	50.4 %	23.8 %	25.8 %
Aliased prefixes	5.1 %	63.8 %	31.1 %

We find 13,202 of the 20,692 prefixes to exhibit a consistent behavior. Given that all Linux machines since kernel 4.10 would fail our tests as they randomize initial timestamps per <SRC-IP, DST-IP> tuple [68], we consider this to be a quite high indicator that our APD probing does indeed find aliased IP addresses. Note that due to many valid scenarios, a failed timestamping test does not make a prefix inconsistent, but is simply indecisive as to whether a prefix may or may not be aliased.

Table 5 shows statistics for all performed consistency tests. Excluding TCP Timestamps, all tests combined find only 1186 inconsistent prefixes. Interestingly, > 90 % of those are caused by hosts showing surprisingly inconsistent behavior to our 2 probe packets. For example, we found 22 hosts responding with distinctively different iTTL values (64 vs. 255) in direct order. We also see hosts responding with different sets of TCP options or option values. Many of these are hosted at CDNs, and might be TCP-level proxies to other services, which could explain time-variant fingerprints.

Validation: To verify our methodology, we run the same tests on prefixes considered *non*-aliased after several days of APD. For direct comparison, we only choose 2940 /64 prefixes with ≥16 responding IP addresses. As shown in Table 6, we find 1481 (50.4 %) to fail at least one of our tests, a considerably higher share than the 5.1 % among aliased prefixes. Additionally, we find only 699 prefixes (23 %) to pass our high-confidence test of consistent timestamping behavior, compared to 63.8 % in aliased prefixes.

Looking deeper into the 699 allegedly non-aliased prefixes that pass our high-confidence test, we find 99 where the whole prefix sends the same TCP timestamp, 509 with strictly monotonic timestamp behavior, and 91 passing our R^2 test. As same and strictly monotonic TCP timestamps are unlikely to happen by chance, we investigate why those prefixes were not detected as aliased. We

find two root causes which can cause this scenario: (1) prefixes may be aliased at subprefixes longer than /64, which we only probe if more than 100 IP addresses are responsive in that prefix—only 167 of the 699 prefixes qualified for probing longer subprefixes than /64. (2) prefixes may contain many IPv6 addresses specifically bound to one machine, but without binding the full prefix to the machine. This would cause a set of our random probes to fail, and hence the prefix would be detected as not aliased. We consider the remaining 167 prefixes to fall in this category.

From this deep-dive, we conclude that those 699 prefixes that pass our timestamping test are indeed highly likely to have all their active IP addresses bound to the same machine, but without representing >100 IP addresses (532 prefixes), or without binding the full prefix (167 prefixes). Therefore, our APD detection algorithm does not detect these as aliased. This discrepancy stems from the facts that (1) our APD, by not probing low-density prefixes, may give some false negatives, and (2) the "same machine" test in our validation also holds true for prefixes with many IPv6 addresses specifically bound to one host, while our APD aims to find prefixes where *all* IP addresses are bound to the same host.

As we consider this validation step merely an informative case study, we do not consider these problematic.

In sum, our validation step shows that (1) our tests are discriminatory and (2) aliased prefixes offer far less diverse configurations, with many more cases believed to be the same machine.

5.5 Comparison to Murdock et al.'s Approach

In order to assess our APD approach we quantitatively compare it to Murdock et al.'s [56]. As Murdock et al. perform alias detection on a best-effort basis by probing addresses in prefixes with a static prefix length of /96, we expect our approach to find more aliased prefixes. This is in fact the case as we find 992.6 k hitlist addresses residing in aliased prefixes which are not detected by Murdock et al. On the other hand, Murdock et al.'s approach classifies only 1.4 k hitlist addresses as aliased which we deem non-aliased.

Additionally, we compare the bandwidth requirements of our APD approach to Murdock et al.'s. As our approach works on multiple prefix levels where at least 100 targets are present, we focus on the most likely aliased prefixes. Consequently, we send probes to a total of 50.1 M IPv6 addresses to determine aliased prefixes in our hitlist. Using Murdock et al.'s static /96 prefixes, more than twice as many IPv6 addresses (113.8 M) are probed.

To summarize, our approach finds 992.6 k more hitlist addresses in aliased prefixes compared to Murdock et al.'s approach and at the same time probes less than half the number of IP addresses.

6 ADDRESS PROBING

We generate IPv6 targets and probe these targets' responsiveness each day. First, we collect addresses from our hitlist sources. Second, we preprocess, merge, and shuffle these addresses in order to prepare them as input for scanning. Third, we perform aliased prefix detection to eliminate targets in aliased prefixes. Fourth, we traceroute all known addresses using scamper [51] to learn additional router addresses. Fifth, we use ZMapv6 [74] to conduct responsiveness measurements on all targets. We send probes on ICMP, TCP/80, TCP/443, UDP/53, and UDP/443 to cover the most common services [36]. We repeat this process each day to allow for longitudinal responsiveness analysis.

6.1 Responsive Addresses

We first evaluate responsive addresses based on their corresponding BGP prefix.

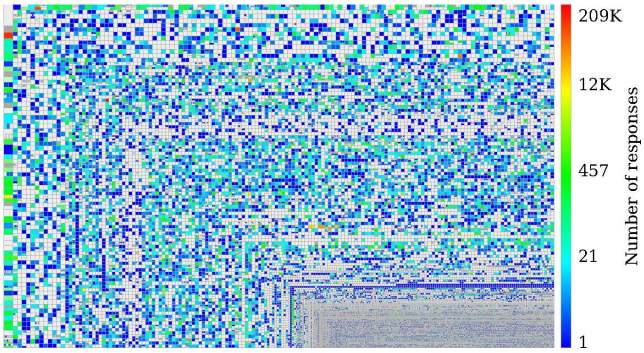


Figure 6: All 56 k BGP prefixes, colored based on the number of responses to ICMP Echo requests on May 11, 2018.

Overall, our hitlist contains 1.9 M responsive IPv6 addresses, spread over 21,647 BGP prefixes covering 9968 different ASes.

Figure 6 shows non-aliased ICMP-responsive addresses per BGP prefix. We see that most prefixes are covered with dozens to hundreds of responsive targets, whereas a few prefixes contribute 12 k or more responsive addresses. The plot is, in terms of colors, strikingly similar to the input set visualized in Figure 1c (note however that the range of the scale in the response plot is smaller). This tells us that for most of the prefixes in our input set, we indeed see responses, and only few return no responses at all. A possible explanation for prefixes with a sizable number of addresses in the input set ending up blank in the response plot is the dropping of ICMP echo requests at a border router.

6.2 Cross-protocol Responsiveness

We analyze the cross-protocol responsiveness of our probes, to understand what kind of IPv6 hosts are responding to our probes. In Figure 7 we show the conditional probability of responsiveness between protocols, i.e., if protocol X is responding, how likely is it that protocol Y will respond. We compare our findings for IPv6 to Bano et al. [14] who performed a similar analysis for IPv4.

We find that if an IPv6 address responds to any of the probes, there is at least a 89 % chance of the same IP address also responding to ICMPv6. The ICMP correlation in IPv6 is higher compared to IPv4, where we see values as low as 73 %. Since ICMPv6 is an integral part of IPv6, it should not be simply blocked in firewalls [22], which makes it more likely that hosts are responding to ICMPv6 compared to its IPv4 counterpart.

Additionally, we see correlations between UDP/443, TCP/443, and TCP/80. More specifically, if an address is responsive to QUIC (UDP/443), it has a likelihood of 98 % to be also providing HTTPS and HTTP services. HTTPS servers are 91 % likely to provide an

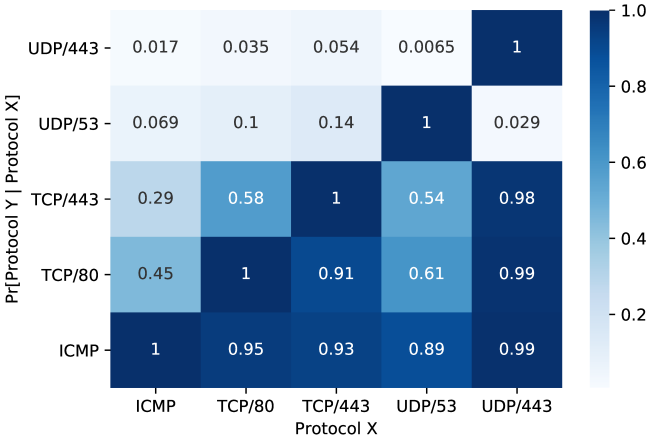


Figure 7: Conditional probability of responsiveness between services.

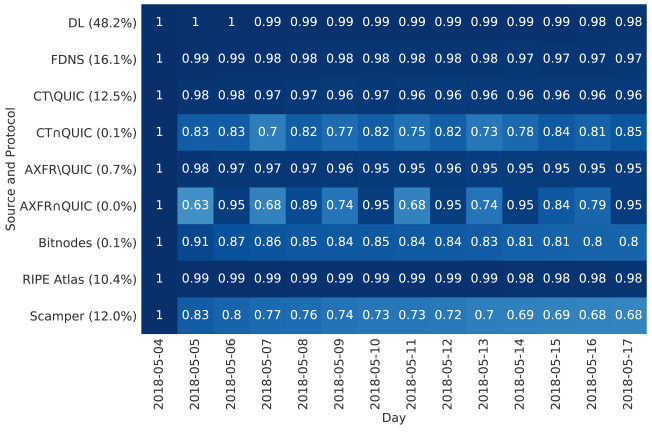


Figure 8: Responsiveness over time, split up by hitlist source and, for special cases, also probed protocol.

HTTP service as well, e.g., to offer a forwarding service to the secure version of a web page. Note that the reverse correlation (HTTP → HTTPS → QUIC) is far less pronounced. Compared to the HTTPS → HTTP correlation of 91 % in IPv6, we see only a 72 % correlation in IPv4 [14].

Analyzing DNS (UDP/53) correlation shows mostly similar results in IPv6 as in IPv4. One exception is the lower correlation to HTTPS in IPv6 (54 %) compared to IPv4 (78 %).

6.3 Longitudinal Responsiveness

To analyze address responsiveness over time, we probe an address continuously even if it disappears from our hitlist's daily input sources. We evaluate longitudinal responsiveness over two weeks as depicted in Figure 8. As a baseline for each source we take all responsive addresses on the first day.

We find that IPv6 addresses from domain lists (DL), FDNS, and RIPE Atlas answer quite consistently over the 14 day period, with all three sources losing only a few percentages of addresses. CT and AXFR sources overall reach a similarly stable response rate; their QUIC response rates, however, fluctuate more heavily and are therefore depicted separately. We investigate this phenomenon and find that more than 80 % of fluctuating addresses are located in two prefixes: Akamai and HDNet. We suspect that these companies are testing the deployment of QUIC on some of their systems or that our measurements are caught by a rate limiting mechanism, resulting in flaky response behavior. Moreover, sources which include clients or CPE devices such as Bitnodes and Scamper lose 20 % and 32 % of the responding hosts, respectively.

7 LEARNING NEW ADDRESSES

In addition to acquiring IPv6 addresses through domain names and other sources, we can also detect addressing schemes, and leverage those patterns to learn previously unknown addresses.

7.1 Methodology

To generate previously unknown addresses, we feed our hitlist into a re-implementation of Entropy/IP [4, 32, 33], and a pre-release version of 6Gen [56]. For this work, we improve the address generator of Entropy/IP by walking the Bayesian network model exhaustively instead of randomly. The improved generator lets us focus on more probable IPv6 addresses, under a constrained scanning budget.

First, we use all addresses in non-aliased prefixes to build a seed address list. Excluding aliased prefixes avoids generating addresses in prefixes where all addresses are responsive, and thus artificially distorting the response rate. Second, we split the seed address list based on ASes, as we assume similar addressing patterns within the same AS. We limit the eligible ASes to those with at least 100 IPv6 addresses to increase the probability of 6Gen and Entropy/IP identifying patterns. Third, we take a random sample of at most 100 k IPv6 addresses per AS to use as input for 6Gen and Entropy/IP. The capped random sample ensures that we provide a balanced input for each AS. Fourth, we run Entropy/IP and 6Gen with the capped random sample as input to generate 1 M addresses for each AS separately. Fifth, we again take a random sample of at most 100 k of all generated addresses per AS for 6Gen and Entropy/IP, respectively. The capped random sample ensures that ASes with more generated addresses are not overrepresented. Sixth, we perform active measurements to assess the value of generated addresses.

7.2 Learned Addresses

Entropy/IP generates 118 M addresses. Of those, 116 M are routable new addresses not yet in our hitlist. 6Gen produces slightly more addresses, 129 M, of which 124 M are new and routable. In total, we learn 239 M new unique addresses. Interestingly, there is very little overlap between 6Gen's and Entropy/IP's generated addresses: only 675 k addresses are produced by both tools, which equals to 0.2 % of all generated addresses.

7.3 Responsiveness of Learned Addresses

We probe the responsiveness of all 239 M learned addresses on ICMP, TCP/80, TCP/443, UDP/53, and UDP/443. 785 k IPv6 addresses respond to our probes, which corresponds to a response rate of 0.3 %. This low response rate underlines the challenges of finding new responsive addresses through learning-based approaches.

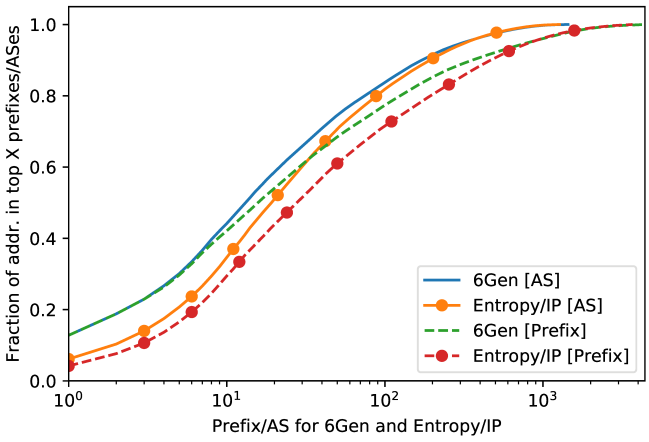


Figure 9: Prefix and AS distribution for responsive addresses generated with 6Gen and Entropy/IP.

Comparing the responsiveness of addresses generated by 6Gen to Entropy/IP, we find that 6Gen is able to find almost twice as many responsive addresses: 489 k vs. 278 k. Our response rate for 6Gen is a lower bound: due to 6Gen's design, choosing the top generated addresses instead of random sampling would likely yield an even higher response rate.

In addition, both Entropy/IP and 6Gen found the same 17 k responsive addresses. The response rate of overlapping addresses generated by both tools is therefore 2.5 %, which is an order of magnitude higher than the general learned address population's 0.3 %. This demonstrates that Entropy/IP and 6Gen find complementing sets of responsive IPv6 addresses, with a small overlap of targets that are more likely to respond. Thus, it is meaningful to run multiple address generation tools even on the same set of input addresses.

Table 7: Top 5 responsive protocol combinations for 6Gen and Entropy/IP.

ICMP	TCP/80	TCP/443	UDP/53	UDP/443	6Gen	Entropy/IP
✓	✗	✗	✗	✗	66.8 %	41.1 %
✓	✓	✓	✗	✗	9.2 %	12.3 %
✗	✗	✗	✓	✗	7.3 %	23.1 %
✓	✓	✗	✗	✗	4.9 %	3.4 %
✓	✓	✓	✗	✓	3.2 %	6.1 %

When analyzing the top 5 protocols for responsive learned addresses in Table 7, we find particular differences between 6Gen and Entropy/IP. Two thirds of 6Gen responsive addresses answer to ICMP only, which is the case for only four out of ten Entropy/IP responsive addresses. On the other hand, Entropy/IP responsive hosts are three times more likely to be DNS servers (UDP/53). Moreover, 6Gen responsive hosts are half as likely to be QUIC-enabled web servers (ICMP, TCP/80, TCP/443, and UDP/443) compared to Entropy/IP. This shows that 6Gen and Entropy/IP not only discover mostly non-overlapping addresses, but also different types of *populations* of responsive hosts.

Finally, we compare ASes and prefixes of responsive addresses for both tools. 6Gen discovers responsive hosts in 1442 ASes, while Entropy/IP does in 1275 ASes. Interestingly, responsive hosts in 384 ASes are found by only one of the tools, i.e., either 6Gen or Entropy/IP. In Figure 9, we show the prefix and AS distributions of responsive hosts. Entropy/IP's distribution is a bit less top-heavy compared to 6Gen's, where the top 2 responsive ASes make up almost 20 % of all addresses. Although there is some overlap in the top 5 ASes, 6Gen features more ISPs, like Sky Broadband, Google Fiber, and Xs4all Internet. In contrast, Entropy/IP's top ASes contain more CDNs and Internet services.

To summarize, 6Gen and Entropy/IP find few overlapping responsive addresses, but mostly in overlapping ASes. The services offered by these hosts differ considerably. Therefore, both tools have their advantages in finding specific addresses and populations. We suggest running both tools to maximize the number of found responsive addresses.

8 RDNS AS A DATA SOURCE

In addition to the sources described in Section 3, we investigate the usefulness of IPv6 rDNS entries for active measurements. As shown by previous work, rDNS walking can be a source for IPv6 addresses [29, 30]. While IPv6 rDNS addresses were used to, e.g., find misconfigured IPv6 networks [17], we are not aware of studies evaluating overall responsiveness. Since walking the rDNS tree to harvest IPv6 addresses is a large effort and puts strain on important Internet infrastructure, we classify this source as "semi-public", compared with sources such as the Alexa Top 1 M list, which is available for download.

We use IPv6 rDNS data provided by Fiebig et al. [30] to perform active measurements and compare the results against other hitlist sources. Analyzing the overlap and structure of IPv6 addresses obtained from rDNS, we find a very small intersection with our hitlist. Of the 11.7 M addresses from rDNS, 11.1 M are new. The prefix distribution of rDNS and hitlist addresses is quite similar, as shown in Figure 10. The AS distribution is even more balanced for rDNS addresses compared to the hitlist. Therefore, the addition of rDNS data to the hitlist input would not introduce a bias at the prefix or AS level.

Next, we perform active measurements to compare the response rate of the rDNS population to the hitlist population. Before the active measurement, we filter 2.1 M unrouted addresses and 13.1 k addresses residing in aliased prefixes (see Section 5) from the rDNS addresses. The response rate for the hitlist with only non-aliased prefixes is generally similar to the response rate of rDNS. The rDNS ICMP response rate is higher: 10 % compared to the hitlist's 6 %. On the other hand, we receive slightly fewer HTTP(S) responses for rDNS, at 2 % (1 %) against the hitlist's 3 % (2 %).

To ensure that responding rDNS addresses are not mostly client addresses, we first analyze the top ASes. As can be seen in Table 8, the top responsive ASes in the rDNS data are hosting and service providers, i.e., mostly servers (especially in the TCP/80 measurement). Next, we look for IPv6 SLAAC's distinct `ff:fe` sequence and evaluate the hamming weight of IIDs for responsive rDNS addresses, as an additional indicator for clients. We find between 6 % and 9 % SLAAC addresses with `ff:fe`. The IID hamming weight

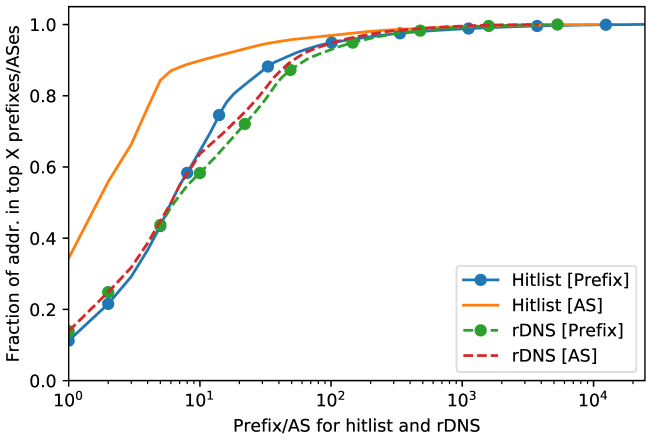


Figure 10: Prefix and AS distribution for hitlist and rDNS input data.

Table 8: Top 5 rDNS ASes in input and responsive via ICMP and TCP/80.

#	Input		ICMP		TCP/80	
1	Comcast	12.5 %	Online S.A.S.	19.6 %	Google	12.8 %
2	AWeber	10.2 %	Sunokman	17.8 %	Hetzner	10.1 %
3	Yandex	9.8 %	Latnet Serviss	8.7 %	Freebit	6.8 %
4	Belpak	6.2 %	Yandex	7.9 %	Sakura	6.5 %
5	Sunokman	6.1 %	Salesforce	5.3 %	TransIP	5.0 %

(i.e., number of bits set to 1) can be used to infer the presence of clients with privacy extensions enabled [36]. The rDNS IID hamming weight does not suggest that the rDNS set contains a large client population, especially for TCP/80, where 60 % of addresses have a hamming weight of six or smaller.

To conclude, the responsive part of the rDNS data source adds a balanced set of IPv6 addresses. We therefore suggest adding rDNS data as input to the IPv6 hitlist.

9 CLIENT IPV6 ADDRESSES

The majority of IPv6 addresses that we have collected so far belong to servers or routers rather than clients. In this section, we use crowdsourcing platforms to collect client IP addresses. Our aim is to begin studying the following questions: Does crowdsourcing serve as an adequate source of residential IPv6 addresses, and might we be able to use addresses gathered through crowdsourcing in IPv6 hitlists? We refer the reader to Section 10.1 describing the ethical consideration of this study.

9.1 Experimental Setup

To investigate our approach of collecting IPv6 client addresses, we use two crowdsourcing platforms and leverage the *test-ipv6.com* [26] code to gather IPv6 addresses.

We set up a web server and integrate it into the crowdsourcing environment to operate similarly to the study by Huz et al. [45]. We use Amazon Turk (Mturk) [8] and Prolific Academic (ProA) [60] to run our experiments and allocate a budget of $150 per platform. We add a limitation for only one submission per user per platform and select the minimum amount of money that can be set for each assignment: $0.01 for Mturk and $0.12 for ProA. We run the experiment between April 23 and May 23, 2018. During this time, 5781 users participate from Mturk, compared to 1186 from ProA. We make our setup and source code available to the community [49].

9.2 Crowdsourcing Participants

Table 9 shows the distribution of measurements per platform, including AS [12] and country mappings [54]. We find about 31 % of Mturk and 20.6 % of ProA users with IPv6 enabled. One reason for Mturk's higher IPv6 ratio might be their customer base: Mturk is more popular in the US and India [45, 50], both countries with considerable IPv6 adoption [41].

Moreover, 31.5 % of IPv6 ASes are overlapping between platforms, although we do not find any common addresses.

A large part of our IPv6 clients participate from a small number of ISPs. The top 3 ASes are Comcast (31.1 %) and AT&T (13.2 %), and the Indian ISP Reliance (7.8 %). Comparing IPv6 clients with IPv4 clients we find the latter to be more diverse, where the top 5 providers constitute only 30 % of all IPv4 clients in our dataset.

Table 9: Client distribution in crowdsourcing study.

	IPv4	IPv6	$ASes_4$	$ASes_6$	$\#cc_4$	$\#cc_6$
Mturk	5707	1787	842	73	93	22
ProA	1176	245	272	48	33	21
Unique	6862	2032	983	92	98	29

9.3 Client Responsiveness

Once the user submits the results, we send an ICMPv6 echo request and traceroute to each IPv6 address every 5 minutes.

We find that only 352 (17.3 %) of IPv6 addresses respond to at least one ICMPv6 echo request. The majority of IPv6 addresses we gathered from residential networks do not respond to these probes.

To investigate whether the low response rate was an artifact of the devices (e.g., privacy extension address cycling, users disconnecting), or network policy, we locate the set of RIPE Atlas probes situated in the same ASes as our crowdsourced client addresses. RIPE Atlas probes generally respond to echo requests they receive.

We select 1398 RIPE Atlas probes with IPv6 connectivity inside these ASes and, having confirmed they were online, send traceroutes to those probes. As many as 641 (45.8 %) probes respond to our queries. Since RIPE Atlas probes will respond by design, these 45.8 % indicate an upper bound of possible crowdsourcing responses. It is likely that users' systems are running local firewalls which will reduce the response rate further. This is indicative only, and deserves further study.

We also find that for 20 % of clients, the last responsive hop is different from the destination AS, which indicates filtering by ISPs.

The low responsiveness from the RIPE Atlas probes and crowdsourcing clients suggests inbound filtering. RFC 7084 [71] leaves it open for the user to decide to either have an "outbound only"

or an "open" configuration, allowing all internally and externally initiated ICMPv6 connections for IPv6 CPEs. Future work is needed to understand the reasons for the deployment of "outbound only" policy for ICMPv6 in residential networks.

Client addresses that respond to our ICMPv6 requests are likely to be less stable than server IP addresses. Only 7 IPv6 addresses from the total of 352 responsive addresses remain active for the entire month. 19 % of IPv6 addresses are active for less than an hour, while 39.4 % of addresses are active for 8 hours or less. Moreover, addresses with dynamic behavior i.e., appearing and disappearing multiple times during the study, had a mean uptime of approximately 8 hours and median uptime of 3 hours per day.

We conclude that crowdsourced addresses provide additional targets for IPv6 client studies. Only a fraction of them are, however, responsive to incoming probes. Future work is needed to get more representative data and to understand the extent to which filtering is performed. Finally, active measurements targeting crowdsourcing addresses need to be performed swiftly after address collection, as the responsive client population is quickly shrinking.

10 MEASUREMENT PRACTICES

In our study we follow measurement best practices by conducting scans in an ethical way and publishing data and code for reproducibility in research.

10.1 Ethical Considerations

Before conducting active measurements we follow an internal multi-party approval process, which incorporates proposals by Partridge and Allman [57] and Dittrich et al. [23]. We assess whether our measurements can induce harm on individuals in different stakeholder groups. As we limit our query rate and use conforming packets, it is unlikely that our measurements will cause problems on scanned systems. We follow scanning best practices [24] by maintaining a blacklist and using dedicated servers with informing rDNS names, websites, and abuse contacts. During our six months of active scans, we received four emails asking for more information on the conducted measurements.

Prior to deploying the crowdsourcing study we discussed the subject with the university ethics committee. We agreed with the committee that IPv6 addresses collected in the crowdsourcing study will be excluded from public datasets. In addition, we informed participants that they could opt out of the active probing. The ethics committee concluded that our experiments did not constitute research on human subjects, as we have no reasonable way of mapping collected IPv6 addresses to individuals, and gave the study formal permission.

10.2 Reproducible Research

To encourage reproducibility in network measurement research [1, 70], we make data, source code, and analysis tools publicly available [37]. This includes zesplot [43], entropy clustering [31], new Entropy/IP generator [32], crowdsourcing documentation [49], and results from daily runs of the IPv6 hitlist, including the list of aliased prefixes [38]. Some of the figures in this paper are clickable, offering additional insights, in-depth analyses, and interactive graphs. This data can serve as a valuable starting point for future IPv6 studies.

11 DISCUSSION

In this section we summarize challenges in generating hitlists and discuss lessons learned for future work.

Time-to-Measurement: We collect IPv6 addresses from a variety of sources containing server, router, and client addresses. Our analysis shows that server IPv6 addresses are more responsive and stable in comparison to CPE and client devices. As a result, when using an IPv6 hitlist as an input for a specific measurement study, researchers need to consider the time-to-measurement: client devices need to be measured within minutes to obtain sensible response rates, whereas servers remain responsive over weeks.

Hitlist Tailoring: To reduce bias when conducting IPv6 measurements, we generally advise to strive for an evenly balanced hitlist across prefixes and ASes, and to remove addresses in aliased prefixes. Depending on the research goal, researchers can pivot from an even address distribution to a stronger focus on certain address types (e.g., HTTPS web servers). Depending on the type of study it may also be desirable to include or exclude specific data sources. For example, studies analyzing hosting providers can use server addresses, while for residential networks, researchers can focus on sources containing mainly CPE and client addresses.

Unresponsive Addresses: While our focus on responsive addresses is reasonable for a hitlist which is used as direct input for a measurement study, there might be scenarios where also unresponsive addresses could be of value. Unresponsive addresses can be used to understand addressing schemes inside a prefix. They can also be used as an input for address learning algorithms (e.g., Entropy/IP or 6Gen) which might then output responsive addresses.

IPv6 Hitlist Service: In order to help future IPv6 measurements we provide daily IPv6 hitlists and a list of aliased prefixes at:

https://ipv6hitlist.github.io

Providing an easy-to-use hitlist service has several advantages for IPv6 research: (1) While conducting measurements on all known IPv6 addresses is possible in one-off measurements, high-frequency periodic measurements might require a focus on a subset of responsive addresses. We provide lists of responsive addresses which can be directly used in IPv6 measurement studies. (2) With the increasing adoption of IPv6, the number of all publicly known IPv6 addresses is bound to increase as well, making it increasingly challenging to perform a full sweep of all known addresses in the future. (3) Reducing the number of measurement targets for the multitude of IPv6 measurement studies is considered good Internet citizenship [47]. (4) By providing historical data we enable other researchers to study the evolution of IPv6 deployment.

12 CONCLUSION

In this work, we leveraged a multitude of sources to create the largest IPv6 hitlist to date, containing more than 50 M addresses. We found that about half of these addresses reside in aliased prefixes and showed, using clustering, that there are only six prevalent IPv6 addressing schemes. Using longitudinal measurements we identified protocols and sources which are less stable over time. We used and extended state-of-the-art tools to generate new addresses, finding that they provide complementary address sets. We will keep running daily IPv6 measurements to provide valuable hitlists.

Acknowledgments: We would like to thank Austin Murdock for providing a pre-release version of 6Gen and Tobias Fiebig for the IPv6 rDNS data. The authors would like to thank the anonymous reviewers and our shepherd Robert Beverly for their valuable feedback. This work was partially funded by the German Federal Ministry of Education and Research under the projects X-Check (16KIS0530), DecADe (16KIS0538), and AutoMon (16KIS0411).

REFERENCES

[1] ACM. Artifact Review and Badging. https://www.acm.org/publications/policies/artifact-review-badging.
[2] David Adrian, Zakir Durumeric, Gulshan Singh, and J Alex Halderman. Zippier ZMap: Internet-Wide Scanning at 10 Gbps. In *WOOT*, 2014.
[3] AFRINIC. IPv6 Address Allocation and Assignment Policy. https://www.afrinic.net/library/policies/122-afpub-2004-v6-001.
[4] Akamai Technologies Inc. Entropy/IP open-source implementation. https://github.com/akamai/entropy-ip.
[5] Rafael Almeida, Osvaldo Fonseca, Elverton Fazzion, Dorgival Guedes, Wagner Meira, and Ítalo Cunha. A Characterization of Load Balancing on the IPv6 Internet. In *Passive and Active Measurement Conference*, 2017.
[6] Lance Alt, Robert Beverly, and Alberto Dainotti. Uncovering Network Tarpits with Degreaser. In *Annual Computer Security Applications Conference*, 2014.
[7] Johanna Amann, Oliver Gasser, Quirin Scheitle, Lexi Brent, Georg Carle, and Ralph Holz. Mission Accomplished? HTTPS Security after DigiNotar. In *ACM Internet Measurement Conference*, 2017.
[8] Amazon. Mechanical Turk. https://www.mturk.com/.
[9] APNIC. APNIC guidelines for IPv6 allocation and assignment requests. https://www.apnic.net/about-apnic/corporate-documents/documents/resource-guidelines/ipv6-guidelines/.
[10] APWG. APWG: Cross-industry Global Group Supporting Tackling the Phishing Menace. http://antiphishing.org.
[11] ARIN. Your First IPv6 Request. https://www.arin.net/resources/first_ipv6_request.html.
[12] Hadi Asghari. PyASN Python module. https://github.com/hadiasghari/pyasn.
[13] Michael Backes, Thorsten Holz, Christian Rossow, Teemu Rytilahti, Milivoj Simeonovski, and Ben Stock. On the Feasibility of TTL-based Filtering for DRDoS Mitigation. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, 2016.
[14] Shehar Bano, Philipp Richter, Mobin Javed, Srikanth Sundaresan, Zakir Durumeric, Steven J. Murdoch, Richard Mortier, and Vern Paxson. Scanning the Internet for Liveness. *SIGCOMM Computer Communication Review*, 48(2), May 2018. doi:10.1145/3213232.3213234.
[15] Robert Beverly and Arthur Berger. Server Siblings: Identifying Shared IPv4/IPv6 Infrastructure via Active Fingerprinting. In *Passive and Active Measurement Conference*, 2015.
[16] Robert Beverly, Ramakrishnan Durairajan, David Plonka, and Justin P Rohrer. In the IP of the Beholder: Strategies for Active IPv6 Topology Discovery. In *ACM Internet Measurement Conference*, 2018.
[17] Kevin Borgolte, Shuang Hao, Tobias Fiebig, and Giovanni Vigna. Enumerating Active IPv6 Hosts for Large-scale Security Scans via DNSSEC-signed Reverse Zones. In *IEEE Symposium on Security and Privacy*, 2018.
[18] Mark Bruls, Kees Huizing, and Jarke J Van Wijk. Squarified treemaps. In *Data visualization*. 2000.
[19] Randy Bush, James Hiebert, Olaf Maennel, Matthew Roughan, and Steve Uhlig. Testing the reachability of (new) address space. In *SIGCOMM Workshop on Internet Network Management*, 2007.
[20] Censys. ICMP Echo Request Full IPv4 Scan Results. https://censys.io/data/0-icmp-echo_request-full_ipv4.
[21] Kimberly Claffy, Young Hyun, Ken Keys, Marina Fomenkov, and Dmitri Krioukov. Internet mapping: from art to science. In *Conference For Homeland Security. Cybersecurity Applications & Technology*, 2009.
[22] E. Davies and J. Mohacsi. Recommendations for Filtering ICMPv6 Messages in Firewalls. RFC 4890 (Informational), May 2007. doi:10.17487/RFC4890.
[23] David Dittrich et al. The Menlo Report: Ethical Principles Guiding Information and Communication Technology Research. *US DHS*, 2012.
[24] Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. ZMap: Fast Internet-wide Scanning and Its Security Applications. In *USENIX Security*, 2013.
[25] W. Eddy. TCP SYN Flooding Attacks and Common Mitigations. RFC 4987 (Informational), August 2007. doi:10.17487/RFC4987.
[26] Falling-Sky project. Test your IPv6. https://github.com/falling-sky/source/wiki.
[27] Xun Fan and John Heidemann. Selecting representative IP addresses for Internet topology studies. In *ACM Internet Measurement Conference*, 2010.
[28] Farsight Security. DNSDB. https://www.farsightsecurity.com/solutions/dnsdb/.
[29] Tobias Fiebig, Kevin Borgolte, Shuang Hao, Christopher Kruegel, and Giovanni Vigna. Something from Nothing (There): Collecting Global IPv6 Datasets from DNS. In *Passive and Active Measurement Conference*, 2017.
[30] Tobias Fiebig, Kevin Borgolte, Shuang Hao, Christopher Kruegel, Giovanni Vigna, and Anja Feldmann. In rDNS We Trust: Revisiting a Common Data-Source's Reliability. In *Passive and Active Measurement Conference*, 2018.
[31] Pawel Foremski. Entropy clustering tool. https://github.com/pforemski/entropy-clustering.
[32] Pawel Foremski. New Entropy/IP generator. https://github.com/pforemski/eip-generator.
[33] Pawel Foremski, David Plonka, and Arthur Berger. Entropy/IP: Uncovering Structure in IPv6 Addresses. In *ACM Internet Measurement Conference*, 2016.
[34] Oliver Gasser, Felix Emmert, and Georg Carle. Digging for Dark IPMI Devices: Advancing BMC Detection and Evaluating Operational Security. In *Workshop on Traffic Monitoring and Analysis*, 2016.
[35] Oliver Gasser, Benjamin Hof, Max Helm, Maciej Korczynski, Ralph Holz, and Georg Carle. In Log We Trust: Revealing Poor Security Practices with Certificate Transparency Logs and Internet Measurements. In *Passive and Active Measurement Conference*, 2018.
[36] Oliver Gasser, Quirin Scheitle, Sebastian Gebhard, and Georg Carle. Scanning the IPv6 Internet: Towards a Comprehensive Hitlist. In *Workshop on Traffic Monitoring and Analysis*, 2016.
[37] Oliver Gasser, Quirin Scheitle, Pawel Foremski Qasim Lone, Maciej Korczyński, Stephen D. Strowes, Luuk Hendriks, and Georg Carle. Data and Analysis at TUM University Library. https://mediatum.ub.tum.de/1452739. doi:10.14459/2018mp1452739.
[38] Oliver Gasser, Quirin Scheitle, Pawel Foremski Qasim Lone, Maciej Korczyński, Stephen D. Strowes, Luuk Hendriks, and Georg Carle. IPv6 Hitlist Website. https://ipv6hitlist.github.io.
[39] Oliver Gasser, Quirin Scheitle, Benedikt Rudolph, Carl Denis, Nadja Schricker, and Georg Carle. The Amplification Threat Posed by Publicly Reachable BACnet Devices. *Journal of Cyber Security and Mobility*, 2017.
[40] F. Gont and T. Chown. Network Reconnaissance in IPv6 Networks. RFC 7707 (Informational), March 2016. doi:10.17487/RFC7707.
[41] Google. Per-Country IPv6 adoption. https://www.google.com/intl/en/ipv6/statistics.html.
[42] Robert Graham. MASSCAN: Mass IP port scanner. https://github.com/robertdavidgraham/masscan, 2013.
[43] Luuk Hendriks. zesplot: IPv6 visualisation based on squarified treemaps. https://github.com/zesplot/zesplot.
[44] Luuk Hendriks, Ricardo de Oliveira Schmidt, Roland van Rijswijk-Deij, and Aiko Pras. On the potential of ipv6 open resolvers for ddos attacks. In *International Conference on Passive and Active Network Measurement*, pages 17–29. Springer, 2017.
[45] Gokay Huz, Steven Bauer, Robert Beverly, et al. Experience in using MTurk for Network Measurement. In *SIGCOMM C2B(I)D Workshop*, 2015.
[46] Cheng Jin, Haining Wang, and Kang G Shin. Hop-Count Filtering: An Effective Defense Against Spoofed DDoS Traffic. In *ACM Computer and Communications Security Conference*, 2003.
[47] Johannes Klick, Stephan Lau, Matthias Wählisch, and Volker Roth. Towards Better Internet Citizenship: Reducing the Footprint of Internet-wide Scans by Topology Aware Prefix Selection. In *ACM Internet Measurement Conference*, 2016.
[48] LACNIC. IPv6 Address Allocation and Assignment Policies. http://www.lacnic.net/684/2/lacnic/4-ipv6-address-allocation-and-assignment-policies.
[49] Qasim Lone. Documentation and Code for IPv6 Crowdsourcing Study. https://github.com/qblone/crowdsourcing-ipv6.
[50] Qasim Lone, Matthew Luckie, Maciej Korczynski, Hadi Asghari, Mobin Javed, and Michel van Eeten. Using Crowdsourcing Marketplaces for Network Measurements: The Case of Spoofer. In *Network Traffic Measurement and Analysis Conference*, 2018.
[51] Matthew Luckie. Scamper: a Scalable and Extensible Packet Prober for Active Measurement of the Internet. In *ACM Internet Measurement Conference*, 2010.
[52] Marek Majkowski. Abusing Linux's firewall: the hack that allowed us to build Spectrum. https://blog.cloudflare.com/how-we-built-spectrum/.
[53] Matthew Bryant. TLD AXFR transfers. https://github.com/mandatoryprogrammer/TLDR.
[54] Maxmind. GeoLite 2 database. https://www.maxmind.com/en/home.
[55] Ayman Mukaddam, Imad Elhajj, Ayman Kayssi, and Ali Chehab. IP Spoofing Detection Using Modified Hop Count. In *Advanced Information Networking and Applications Conference*, 2014.
[56] Austin Murdock, Frank Li, Paul Bramsen, Zakir Durumeric, and Vern Paxson. Target Generation for Internet-wide IPv6 Scanning. In *ACM Internet Measurement Conference*, 2017.
[57] Craig Partridge and Mark Allman. Ethical Considerations in Network Measurement Papers. *Communications of the ACM*, 2016.
[58] PhishTank. A Nonprofit Anti-phishing Organization. http://www.phishtank.com.
[59] David Plonka and Arthur W. Berger. kIP: a Measured Approach to IPv6 Address Anonymization. http://arxiv.org/abs/1707.03900, 2017.

[60] Prolific. Prolific Academic. https://www.prolific.ac/.
[61] Rapid7 Project Sonar. Forward DNS Data. https://opendata.rapid7.com/sonar.fdns_v2/.
[62] Philipp Richter, Georgios Smaragdakis, David Plonka, and Arthur Berger. Beyond counting: new perspectives on the active IPv4 address space. In *ACM Internet Measurement Conference*, 2016.
[63] RIPE NCC. IPMap. https://ftp.ripe.net/ripe/ipmap/.
[64] RIPE NCC. IPv6 Address Allocation and Assignment Policy. https://www.ripe.net/publications/docs/ripe-655.
[65] Quirin Scheitle. TCP Options module for ZMap. https://github.com/tumi8/zmap/blob/master/src/probe_modules/module_tcp_synopt.h.
[66] Quirin Scheitle, Taejoong Chung, Jens Hiller, Oliver Gasser, Johannes Naab, Roland van Rijswijk-Deij, Oliver Hohlfeld, Ralph Holz, Dave Choffnes, Alan Mislove, and Georg Carle. A First Look at Certification Authority Authorization (CAA). *ACM SIGCOMM Computer Communication Review*, 2018.
[67] Quirin Scheitle, Oliver Gasser, Paul Emmerich, and Georg Carle. Carrier-Grade Anomaly Detection Using Time-to-Live Header Information. http://arxiv.org/abs/1606.07613, 2016.
[68] Quirin Scheitle, Oliver Gasser, Minoo Rouhi, and Georg Carle. Large-scale Classification of IPv6-IPv4 Siblings with Variable Clock Skew. In *Network Traffic Measurement and Analysis Conference*, 2017.
[69] Quirin Scheitle, Oliver Gasser, Patrick Sattler, and Georg Carle. HLOC: Hints-Based Geolocation Leveraging Multiple Measurement Frameworks. In *Network Traffic Measurement and Analysis Conference*, 2017.
[70] Quirin Scheitle, Matthias Wählisch, Oliver Gasser, Thomas C. Schmidt, and Georg Carle. Towards an Ecosystem for Reproducible Research in Computer Networking. In *ACM SIGCOMM Reproducibility Workshop*, 2017.
[71] H. Singh, W. Beebee, C. Donley, and B. Stark. Basic Requirements for IPv6 Customer Edge Routers. RFC 7084 (Informational), November 2013. doi:10.17487/RFC7084.
[72] Spamhaus. The Spamhaus Project. https://www.spamhaus.org.
[73] Stephen D. Strowes. Bootstrapping Active IPv6 Measurement with IPv4 and Public DNS. http://arxiv.org/abs/1710.08536, 2017.
[74] TUM. ZMapv6: Internet Scanner with IPv6 capabilities. https://github.com/tumi8/zmap.
[75] Johanna Ullrich, Peter Kieseberg, Katharina Krombholz, and Edgar Weippl. On reconnaissance with IPv6: a pattern-based scanning approach. In *Availability, Reliability and Security Conference*, 2015.
[76] Peter van Dijk. Finding v6 hosts by efficiently mapping ip6.arpa. https://web.archive.org/web/20170603234058/http://7bits.nl/blog/posts/finding-v6-hosts-by-efficiently-mapping-ip6-arpa, March 2012.
[77] Matteo Varvello, Jeremy Blackburn, David Naylor, and Konstantina Papagiannaki. EYEORG: A Platform For Crowdsourcing Web Quality Of Experience Measurements. In *ACM Conference on Emerging Networking EXperiments and Technologies*, 2016.
[78] Addy Yeow. Bitnodes API. https://bitnodes.earn.com/.
[79] Sebastian Zander, Lachlan LH Andrew, and Grenville Armitage. Capturing Ghosts: Predicting the Used IPv4 Space by Inferring Unobserved Addresses. In *ACM Internet Measurement Conference*, 2014.