6Sense: Internet-Wide IPv6 Scanning and its Security Applications

Grant Williams, Mert Erdemir, Amanda Hsu, Shraddha Bhat, Abhishek Bhaskar, Frank Li, and Paul Pearce, *Georgia Institute of Technology*

https://www.usenix.org/conference/usenixsecurity24/presentation/williams

This paper is included in the Proceedings of the 33rd USENIX Security Symposium.

August 14–16, 2024 • Philadelphia, PA, USA

978-1-939133-44-1

Open access to the Proceedings of the 33rd USENIX Security Symposium is sponsored by USENIX.

6SENSE: Internet-Wide IPv6 Scanning and its Security Applications

Grant Williams *Mert Erdemir* *Amanda Hsu* *Shraddha Bhat*
Abhishek Bhaskar *Frank Li* *Paul Pearce*

Georgia Institute of Technology
6sense-team@cc.gatech.edu

Abstract

Internet-wide scanning is a critical tool for security researchers and practitioners alike. By exhaustively exploring the entire IPv4 address space, Internet scanning has driven the development of new security protocols, found and tracked vulnerabilities, improved DDoS defenses, and illuminated global censorship. Unfortunately, the vast scale of the IPv6 address space—340 trillion trillion trillion addresses—precludes exhaustive scanning, necessitating entirely new IPv6-specific scanning methods. As IPv6 adoption continues to grow, developing IPv6 scanning methods is vital for maintaining our capability to comprehensively investigate Internet security.

We present 6SENSE, an end-to-end Internet-wide IPv6 scanning system. 6SENSE utilizes reinforcement learning coupled with an online scanner to iteratively reduce the space of possible IPv6 addresses into a tractable scannable subspace, thus discovering new IPv6 Internet hosts. 6SENSE is driven by a set of metrics we identify and define as key for evaluating the generality, diversity, and correctness of IPv6 scanning. We evaluate 6SENSE and prior generative IPv6 discovery methods across these metrics, showing that 6SENSE is able to identify tens of millions of IPv6 hosts, which compared to prior approaches, is up to 3.6x more hosts and 4x more end-site assignments, across a more diverse set of networks. From our analysis, we identify limitations in prior generative approaches that preclude their use for Internet-scale security scans. We also conduct the first Internet-wide scanning-driven security analysis of IPv6 hosts, focusing on TLS certificates unique to IPv6, surveying open ports and security-sensitive services, and identifying potential CVEs.

1 Introduction

IPv4 Internet scanning has transformed security research. Beginning with the debut of Durumeric et al.'s ZMap [25] at USENIX Security 2013, researchers used fast IPv4 Internet scans in more than 700 peer-reviewed papers to: uncover new classes of security weaknesses [23], understand attacker behavior [3], guide the development of new protocols like TLS 1.3 and SMTP-STS [1, 24], improve DDoS defenses and censorship circumvention platforms [42, 68], and uncover in-progress network attacks [26]. Data collected from Internet scans also enabled researchers to notify hundreds of thousands of server operators about vulnerable systems [43].

While fast Internet scanning has helped us understand a range of Internet security questions, the scope of understanding is confined to the 32-bit IPv4 address space. This limitation stems from existing end-to-end tooling employing an *exhaustive* scanning methodology by which they send packets to *every single IPv4 address* on the Internet, totaling billions of packets per scan. This brute force method is unable to scale to IPv6's roughly 340 trillion trillion trillion addresses.

As IPv6 adoption grows, currently comprising more than 40% of Google's traffic [34], it becomes critical to develop tools to extend Internet scanning into the IPv6 space. Moreover, prior work has demonstrated differences in security posture between dual-stacked IPv4 and IPv6 systems [21] along with IPv6-specific security challenges, indicating a need for IPv6 security and vulnerability assessments. Worsening the problem, IPv6 utilization is not uniform across the globe. Tools that are unable to explore IPv6 will fundamentally limit our understanding of security threats and potentially bias results to specific regions, populations, and technologies.

The lack of technology to perform Internet-wide IPv6 scanning has not gone unnoticed. Given the intractability of exhaustive enumeration, IPv6 scanning approaches must be, by necessity, driven by *heuristic* and *learning* methods. Such initial approaches to IPv6 address generation have been undertaken and are numerous [5, 17–19, 29–31, 35, 39, 45, 46, 50, 59, 61, 71–73]; these works largely focused on *Target Generation Algorithms* which build an understanding from a set of known addresses and generalize new addresses to scan.

Unfortunately, these methods have fundamental limitations that preclude their use for large-scale Internet security scans: 1) They are frequently offline algorithms [17–19, 29, 46, 50, 71]; meaning they cannot adapt to online scanning results, which we will show is critical to enumerating hosts. 2) They frequently do not account for IPv6 aliases[1], causing accuracy and hit-rate reporting in prior work to be misleading. We will show that aliasing is a substantial challenge in IPv6 Internet scanning, and surmounting aliasing is key to performing security scans. 3) They are not deployable for security scanning tasks. As we will show in Section 6, existing solutions either generate such large quantities of aliased addresses they are intractable for deployment or produce limited unique active hosts. These challenges make them undeployable by researchers to achieve similar security results as with IPv4.

Orthogonal to prior approach limitations, no concrete metrics have been established for evaluating IPv6 scanning solutions, with multiple papers utilizing similar metrics with different definitions (e.g., "Hits" with and without accounting for aliases). Worse still, commonly used metrics do not account for the unique structure of IPv6 addresses, whereby a single end-site can be assigned a /64 prefix, rather than a full distinct address. Establishing a set of consistent metrics is key to producing effective IPv6 security scans.

We design, deploy, and evaluate 6SENSE, an end-to-end reinforcement learning [67] IPv6 scanning system designed for Internet-wide security scans. 6SENSE leverages IPv6 address and autonomous system (AS) deployment patterns to perform per-AS address generation. 6SENSE finds significantly more hosts than prior IPv6 discovery methods. 6SENSE's design and implementation are driven by six important IPv6 scanning metrics we identify as influencing the efficiency, correctness, diversity, and deployability of IPv6 scanning systems. As a part of our evaluation, we identify and describe these metrics and evaluate available and tractable IPv6 discovery approaches across these metrics, demonstrating that 6SENSE greatly outperforms existing solutions and provides a significantly greater diversity of discovered hosts and networks. We also show 6SENSE is tunable against these metrics, as well as by protocol and port, allowing researchers to focus their scans based on their scanning needs. Lastly, we deploy 6SENSE to perform an initial Internet-wide scanning-driven security analysis of IPv6 hosts to demonstrate the promise of generative IPv6 security scanning.

Specifically, our contributions include:

- Identifying, defining, and evaluating a set of six comprehensive metrics for IPv6 scanning performance that can form the basis of future evaluations of scanning approaches.
- Developing a new generalized approach to IPv6 address generation and scanning, and implementing that approach as 6SENSE. 6SENSE uses reinforcement learning from a set of known addresses to generate a diverse set of candidate IPv6 addresses which it subsequently scans. Of note is 6SENSE's understanding of IPv6 address structure and allocation strategies, techniques lacking from prior approaches. We believe 6SENSE can not only serve as a platform for Internet-scale IPv6 security studies, but 6SENSE's framework and tooling can be iterated on and enhanced to yield continually improving IPv6 Internet scan results.
- Deploying available prior IPv6 generative approaches and evaluating 6SENSE against each approach across our metrics, finding up to 3.6x more hosts and 4x more end-site /64 assignments than prior approaches.
- Performing the first scanning-driven, Internet-wide security analysis of IPv6 hosts. We explore 3 security contexts: TLS Certificates, sensitive exposed internal services, and open ports. We also conduct a qualitative study of insecurity and configuration issues with discovered IPv6 hosts.
- Discovering 81K security-sensitive devices exposed to the Internet, ranging from virtualization tools to enterprise networking hardware to consumer devices.
- Quantifying the potential CVE exposure of discovered devices, finding at least 70 applicable CVEs.

6SENSE has been open-sourced at https://github.com/IPv6-Security/6Sense to suppport IPv6 security research.

2 IPv6 Background

IPv6 addresses are 128 bits. Each consecutive 4 bits has a hexadecimal representation called a nybble. Within an address, the 128 bits are (conceptually) split into Upper-64 bits and Lower-64 bits. Typically, the Upper-64 bits describe an allocation or assignment, although prefix sizes vary [40]. The Lower-64 bits describe a specific host.

Upper-64 bits: Allocation + SSID. An allocation specifies a prefix leased from the Regional Internet Registry (RIR) to an organization. Commonly, this allocation is of size /32 or larger [40]. The remaining bits of the Upper-64 bits are a subnet ID (SSID), varying in usage. An SSID may begin a pattern continuing into the Lower-64 bits, it may be incremental, or it may follow a pattern entirely of its own. Further, SSID similarities within the same allocation vary. We observe that some SSIDs within allocations follow patterns. This is partly from observing incremental and other obviously patterned SSIDs. Additionally, Upper-64 bits, in practice, may correspond to end-sites [63], but may vary in size.

Lower-64 bits. The Lower-64 bits, also known as the interface identifier (IID), are commonly assigned to a single end-site [37]. End-sites often utilize specific patterns in the IID range (e.g., addresses with `::1` Lower 64 Bits are frequently observed), some guided by RFCs [11, 37]. However, some hosts utilize fully randomized IIDs [16, 33]. While fully randomized IIDs are intractable to scan, prior work suggests patterns in IIDs do exist and are common [29].

Aliasing. Aliasing (sometimes called pseudo-dense regions) is a significant challenge for IPv6 measurements [27, 30, 50], and occurs when a large contiguous IPv6 range is fully responsive on all addresses. However, it is infeasible that each

[1] Aliases are impossibly large fully responsive IPv6 prefixes, frequently assigned to a single machine [30, 73]. Such regions are extremely common [30].

address represents a distinct active host due to the number of addresses. Scanning methods optimizing for discovered active addresses may over-dedicate resources to exploring aliased regions, resulting in misinterpreted and poor results. Aggressively scanning aliased prefixes may also present ethical issues by overloading in-path networks. While we present a dealiasing approach in Section 5.3, we note studying the effects of aliasing on security measurements and developing different dealiasing methods remains an open problem. We note dealiasing in the IPv6 scanning context does not typically refer to resolving singular devices with addresses in separate networks. Rather, it focuses on contiguous active regions.

3 Related Work

IPv6 Scanning. ZMap [25] introduced fast Internet-wide IPv4 scanning by comprehensively enumerating the entire IPv4 address space. While this tool revolutionized IPv4 network security, the tremendous size of IPv6—340 trillion trillion trillion addresses—prohibited similar approaches for IPv6 scanning. Instead, recent work introduced IPv6 target generation [29, 50, 64]. Target generation approaches (TGAs) use known IPv6 addresses to deduce new active addresses and can be categorized into Initial Explorations, Generative Tree Approaches, and Machine Learning Approaches.

Initial Explorations: The first IPv6 TGA was introduced by Ullrich et al. [64] and leveraged common static addressing patterns in known prefixes. Subsequently, Foremski et al. [29] presented Entropy/IP which, given known "seed" addresses, produces a statistical model of the dependencies between nybbles. This model generates addresses by varying high-entropy nybbles. 6Gen [50] identifies dense address space regions through seed address clustering and generates within these regions. Liu et al. [46] introduced 6Tree, which builds a tree of addresses initialized by the seeds. It scans the address region for leaf nodes, iteratively merging scanned nodes into larger regions up the tree. Relative to modern approaches, these models are not competitive as they are not adaptive.

Generative Tree-Based Approaches: 6Hit [39] introduced the first online reinforcement learning system as an online extension to 6Tree. 6Hit targets generation towards more active 6Tree nodes and recreates the tree when hitrates plateau. However as it does not dealias, it gravitates towards aliased regions, resulting in ineffective practical scanning.

Separately, DET [61] builds upon 6Tree by regenerating the address tree after fixed increments. DET updates 6Tree to use an entropy-based tree branching method. It splits generation into multiple batches, recreating the tree based on active addresses between each batch. DET is an online approach, but it lacks alias detection and gravitates toward aliased regions. Furthermore, DET's repeated tree reconstruction is heavyweight and slow. 6Graph [71] leverages DET's approach for generating address trees based on entropy, however, it extracts the high-density IPv6 region patterns from leaf nodes offline, and later applies them to generate addresses by sampling within those regions. As it is offline, it cannot adapt over time, including to aliases.

Machine Learning Approaches: Cui et al. [17] presented 6GCVAE, the first deep learning approach, using gated convolutional variational autoencoders. Next, 6VecLM [19] used word embeddings and transformers to generate candidate addresses. Unfortunately, these approaches tended to generate invalid IPv6 addresses (e.g., in unallocated ranges) and degenerating into aliased regions. 6GAN [18] updated these methods by using sequential generative adversarial networks to solve these problems but suffers performance issues.

Other Approaches: Li et al. explored non-generative IPv6 router discovery whereby they leveraged the behavior of some routers to respond to packets addressed to non-existent devices within their subnet with ICMP Destination Unreachable [44]. From these ICMP packets, their tool extracts the last-hop in-path router. This approach is fast as it has no generative or learning components. Where applicable this method is apt; however its leveraging of such control packets from routers limits it to only devices that generate such ICMP Destination Unreachable responses, as well as only to on-path last-hop routers (rather than end hosts). It has not been shown applicable for Internet-wide scanning.

IPv6 Security. Previous work studying IPv6 security policies has been limited by the method of identifying active IPv6 addresses. Czyz et al. identified dual-stack hosts with fingerprinting and found that many applications were more open on IPv6 than on IPv4 [21]. Borgolte et al. analyzed the security of collected IPv6 addresses through DNSSEC reverse zones, finding that the openness of applications varied [7]. Li et al. deployed their ICMP Destination Unreachable method across 13 residential ISPs, and performed security analysis by classifying periphery devices by vendor, identifying device vulnerabilities, DDoS amplification, and routing loops [44].

Prior generative IPv6 scanning work has not directly addressed security, leaving it as future work [50, 73]. Early TGAs only evaluate their performance based on responsive addresses per category (such as from router, server, and client seeds) [29, 64]. Moreover, generative tree-based and machine learning approaches focus solely on discovery performance compared to prior work [17–19, 39, 46, 50, 70, 71]. By using an IPv6 hitlist [30], Song et al. explore the hit-rate change based on different services using various protocols and ports [61]. Security is largely an unexplored field in the generative Internet-wide IPv6 scanning space; we share our initial security explorations in Section 7.

4 IPv6 Scanning Metrics

Given the inability to exhaustively search the IPv6 address space, it is critical to establish metrics to understand the effectiveness of proposed IPv6 scanners at discovering hosts and thus performing security studies. Prior work largely used the overall number of found addresses ("hits") to naively express the utility of approaches [5, 17–19, 29–31, 35, 39, 45, 46, 50, 59, 61, 71–73]. "Hit-rate" extends this metric to balance hits against the number of scan probes sent. While hits are important, they fail to capture crucial dimensions of IPv6 host discovery that are important for security studies. They treat all active addresses equally, although as previously discussed, large-scale IPv6 aliasing produces active addresses that are not distinct meaningful hosts to measure. Meanwhile, optimizing for hits also ignores network or host diversity, potentially resulting in skewed views of the IPv6 Internet.

Here, we propose a broader set of metrics—four positive and two negative—that better encompass the performance of IPv6 scanners. Note, our metrics exclude seed inputs (counted as "duplicates") from results to reflect additive performance. We envision that for different studies, different metrics could be optimized for. For example, surveying infrastructure reachability requires AS diversity, while monitoring residential router vulnerabilities may require more active Upper-64s.

[+] Number of Hits. Counting hits is useful as a first approximation of an IPv6 scanner's potential, and prior work primarily relied upon this metric [29, 30, 39, 50, 61, 73]. However, as discussed, hits do not effectively convey host or network diversity and can be skewed by addresses in aliased regions. Thus we define this metric as the number of active IPv6 addresses found in a given scan (excluding seeds) minus those identified as in aliased regions.

[+] Number of New /64 Subnets with Hits. To capture the diversity of end-sites found by an IPv6 scanning approach, we consider the number of non-seed /64 subnets with non-aliased hits. Per RFCs [63], /64s are the most specific prefix for end-site assignment. While /64s may not always delimit end-sites, allocation policies specifically recommend against assigning more specific prefixes [57]. Thus, examining /64 subnets can provide us with additional evaluation of diversity not captured by counting hits. We acknowledge that /64s are not necessarily static, as prior work has characterized the mobility of IPv6 addresses and prefixes [52, 58]. However, we maintain that even with address and prefix churn, counting /64s remains a valuable evaluation metric.

[+] Number of Newly Generated /64 Subnets. To quantify the diversity of IPv6 scanner exploration, even if it fails to find hits in certain subnets, we also consider how many new /64 subnets are generated (beyond seed /64s). While this metric may not directly translate into improved hits, we believe it provides intuition on how a method explores the space.

[+] Number of ASes with Hits. Here, we consider network diversity by quantifying the number of distinct ASes found with a non-seed non-aliased hit. We can map addresses to ASes using the Routeviews Prefix-to-AS mappings Dataset (pfx2as) [8]. By including AS counts alongside /64 counts, we highlight two distinct perspectives on the diversity of discovered addresses. We note that ASes are not a perfect representation of network diversity [32], and there is potential to refine or expand this metric based on routing information.

[-] Number of Duplicate Addresses Generated. Generative approaches to IPv6 scanning have a key problem: repeatedly generating the same address. Such duplicate addresses result in lost generative budget and repeated scans of the same address. Thus, here we quantify the number of IPv6 addresses generated more than once by a system. We treat generating seed addresses as a form of duplicate. Although this metric does not directly correlate to our other discovery metrics, counting duplicates generated is important to evaluate the effectiveness and efficiency of a scanning method.

[-] Aliased Hits. Aliased IPv6 regions are a significant problem for IPv6 scanning (Section 2), and further work is necessary to fully understand their impact, as well as develop sophisticated strategies to identify them and account for them in scanning metrics. In the context of our metrics, we deploy our online dealiasing approach and quantify the number of active IPv6 addresses found within aliased regions. In conjunction with our other metrics, this metric allows us to fully evaluate method diversity and effectiveness.

5 Methodology

The space of approaches to IPv6 scanning is vast; prior work tried numerous heuristic, statistical, and learning regimes. We now present an overview of our design, motivated by past work and the need for a deployable system for security studies.

Starting point: Generalizing from Known Addresses. IPv4 systems simply enumerate addresses; IPv6 needs a different approach. TGAs largely generalize from known addresses, or "seeds." We adopt this approach but expand it to also understand patterns in how addresses are *allocated*.

Utilizing Domain-Specific Knowledge: Address Structure. As previously discussed, an IPv6 address typically has 3 components: allocation, SSID, and Lower-64 bits. Unlike prior approaches, we use this domain-specific knowledge in our method (to optimize address discovery), with different approaches for each section of the address space.

Efficient Exploration: Allocation-Aware, Metric-Driven Online Learning. We take an online reinforcement learning approach to generalize known network allocations into generated addresses. This allows us to update our learned representation in real-time, a key to improving system performance. Moreover, this design allows us to explore regions and optimize our learning toward specific metrics.

Leverage End-Host Behavior: Lower-64 Bits. We leverage domain-specific knowledge of common Lower-64 bit patterns (e.g., routers at ::1, IPv4 embedding, MAC encodings, incrementing addresses) to iteratively explore Lower-64

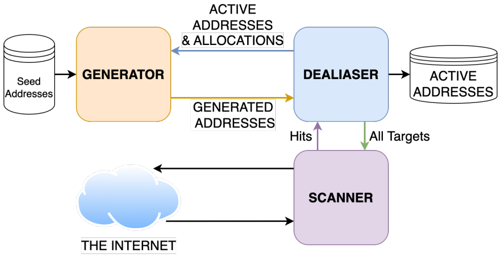

Figure 1: 6SENSE overview. 6SENSE consists of three components: Address Generator (Figure 2), Dealiaser (Figure 3), and Scanner. The Generator passes generated full addresses to the Dealiaser to perform online alias resolution (see Figure 2). The Dealiaser performs both offline and online dealiasing. The Dealiaser then interacts with the scanner to perform Internet scans of IPv6 addresses. Each component is broken down in more detail in subsequent figures. Sub-component figures are color-coordinated with their parent.

bits patterns. We pair this with our learned representation of allocations and SSIDs to produce candidate addresses to scan.

Online Dealiasing. Since aliased regions are likely to "sinkhole" scanners optimizing for hits, discovering hosts within aliased regions provides limited value to researchers. Known lists of aliased regions provide an important resource to preemptively avoid aliases, but cannot account for changes in allocations or new aliases. To effectively scan the IPv6 space, it is critical for scanning systems to include online dealiasing.

5.1 6SENSE Overview

6SENSE, shown in Figure 1, is an end-to-end online IPv6 scanning system. It consists of three core components: Address Generator, Dealiaser, and Scanner. We summarize and subsequently describe the system in this section.

Scanning IPv6 begins with address generation. We leverage IPv6 address assignment and allocations to generate active addresses efficiently. We handle the Upper-64 bits and Lower-64 bits of addresses separately, leveraging domain-specific knowledge of how addresses are allocated and utilized. This distinction is based on RFC guidance that specifies the most specific allocation that should be given to an end-site is a /64, with the end-site deciding how to assign to hosts within the /64 based on various schemes [63].

6SENSE samples known IPv6 allocations using a weighted Reinforcement Learning update technique (Section 5.2.1). It sends these variable-length allocations to an LSTM [38] to generate candidate Upper-64 bits of addresses (Section 5.2.2). Finally, 6SENSE provides these Upper-64 bits to an iterative Lower-64 bits generator that uses domain knowledge to pair Upper-64 bits to likely active Lower-64 bits (Section 5.2.3).

After generation, 6SENSE provides addresses to the Dealiaser (Section 5.3). The Offline Dealiaser performs prefix matching to exclude known aliased addresses before active probing (Section 5.3.1). Addresses then proceed to the Online Dealiaser (Section 5.3.2) and the Scanner (Section 5.4). The Scanner interacts dynamically with the Online Dealiaser to determine if addresses lie in aliased regions by sending additional probes to addresses near active IPs.

Finally, dealiased active addresses are returned to the Generator, which uses them to retarget generation toward active regions. This adaptive system allows 6SENSE to discover active regions sparsely represented in input datasets, far outperforming prior static methods. By adjusting *how* we use active addresses to retrain generation, we can vary the target of generation based on customized metrics. This, coupled with the modular scanning approach, allows our model to adapt to individual use cases (Section 5.2.1).

5.2 Address Generation

Address generation is broken down into three distinct parts. First, 6SENSE generates candidate IPv6 allocations. These are blocks of IPv6 addresses allocated in input datasets. Next, it extends those allocations to a full Upper-64 bits, comprising the subnet prefix of each address. Finally, it generates specific Lower-64 bits based on observed patterns.

5.2.1 Allocation Generation

6SENSE's allocation generation uses the Explore-Exploit Reinforcement Learning paradigm [62]. Exploration maximizes the Breadth of generated results by exploring all allocations present in the seed dataset, and exploitation maximizes results in known successful regions (Depth) by weighting more heavily towards active prefixes. Since exploitation is overloaded in security contexts, we use the terms: Breadth–Depth.

We specifically focus our search on seed allocations since vast quantities of the IPv6 address space are unallocated [20], and many other allocated regions are unused [40].

Breadth. For the first N_s addresses, 6SENSE generates equally from each allocation in the seeds. At the end of this phase, 6SENSE uses scan results to discard allocations without active addresses. It weights the remaining allocations proportional to their hit-rates.

The optimal N_s varies per port/protocol. To further generalize our approach, we introduce an automated threshold discovery mechanism to determine N_s. The goal of exploring is to find as many active regions (allocations) as possible before moving to the Depth phase. Thus, we base our stopping criteria on the rate of new allocations found. We sample a fixed-size ($1M$) block of addresses during exploration, and after each block b, estimate the scaled gradient G_b of new active allocations A_b using the difference: $G_b = \frac{A_b - A_{b-1}}{A_b}$. Breadth completes when G_b drops below a tunable threshold.

To properly balance AS discovery across ports, we set the tunable completion threshold to 0.005 for experiments

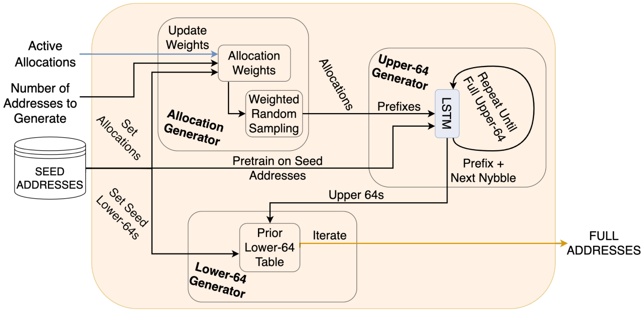

Figure 2: 6SENSE's Address Generator. 6SENSE uses weighted allocation sampling to target specific known IPv6 allocations. Once allocations are selected, it uses an LSTM trained on known subnet utilization patterns to create a candidate Upper-64 bits. From this, known Lower-64 bit patterns are iterated upon to create full candidate 128-bit addresses.

in Section 6. Optimal thresholds vary with the number of addresses sampled, but 0.005 allows proper balancing of AS and Allocation discovery across 100M IPs.

Depth. In subsequent iterations, 6SENSE samples in batches of N_i addresses. It weights sampling based on hit-rates over the past $3N_i$ iterations. We chose a N_i of $1M$ for ICMP. To reflect their expected reduced volume of active addresses, we chose $5M$ for the other ports and protocols tested in Section 6.4. In this way, 6SENSE moves away from allocations where it ceases finding hits or exhausts discoverable addresses. This phase is crucial, as it allows 6SENSE to avoid exhausting its scanning budget in unused or undiscoverable prefixes.

5.2.2 From Allocations to Full Upper-64 bits

As an end-site can be assigned up to a /64, full Upper-64 bits are more likely determined by the allocation and the organization it is leased to [63]. Thus, we cannot rely on fixed patterns for generation. Instead, our Upper-64 generator must discover new statistically significant patterns within the Upper-64 bits. These patterns account for organization-specific and generalizable Upper-64s. Because the Lower-64 bits are not fixed, we may wish to sample an Upper-64 multiple times, so our sampling mechanism must account for the frequency we sample the same Upper-64 bits within an allocation.

Sequential Modeling suites this task well, while also capturing positional dependencies within addresses [14]. Sampling Upper-64 bits sequentially (from allocations) captures the frequency of active seed Upper-64 bits per allocation and Upper-64 bit assignment patterns. 6SENSE uses an LSTM for its relative simplicity and speed. The LSTM takes as input an allocation and sequentially predicts each next nybble, up to 64 bits. The nybble granularity aligns with prior work [17–19, 29, 39, 46, 50, 59, 61, 71, 72]. We sample from our LSTM in batches of 50K to balance time and memory.

By generating using a deep learning model, we characterize the Upper-64 bit patterns in terms of allocation-specific patterns, generalizable patterns across allocations (Upper-64 bit patterns in one allocation may be referenced by the LSTM weights for another allocation with less data), and the frequency of Upper-64 bits within an allocation. This also makes 6SENSE robust to sparsely represented regions in the seed addresses. For these regions, the LSTM generates more seed Upper-64 bits (since only these Upper-64 bits appear in that region in the seed dataset) allowing us to efficiently explore different Lower-64 bits within these known active Upper-64s. This per-allocation structure allows us to parallelize address *generation* across GPUs, which we discuss in Section 6.6.

For training, we ran our LSTM with a batch size of 3200, a learning rate of 0.001, and two LSTM layers of sizes: 512, and 256. This structure balances LSTM predictive power with generative speed. Our training dataset consists of every prefix within our active seed dataset between the allocation length and 15 nybbles (60 bits) with masked variable allocation lengths. The labels for each prefix are the next nybble within the Upper-64 bits. We train for 50 epochs and use a dropout of 0.2 per iteration, to prevent overfitting. While we want 6SENSE to roughly emulate seed patterns, overfitting can decrease diversity in the addresses by only generating seed Upper-64 bits and not exploring new ones.

5.2.3 Lower-64 bits Generation

There are numerous well-defined Lower-64 bit patterns within IPv6 addresses. Many common patterns, however, include a bottom-up counting strategy, with active Lower-64 bits having similar lowest bits. The similarity of the low bits means an exhaustive iterative search around the lowest bits of the Lower-64 bits is both efficient and fruitful.

6SENSE first extracts a list of previously seen Lower-64 bits for each allocation, ordered by frequency. While sampling, each time it sees a repeated Upper-64, 6SENSE generates a new Lower-64 in the following order:

1. **Prior Lower-64 bits:** The Lower-64 bits of the Upper-64 bits' allocation in order of frequency in the seed dataset, skipping Upper/Lower pairs already present in the seeds.
2. **`::1`:** Due to the prevalence of `::1` in known configurations and observed IPv6 addresses, 6SENSE tries `::1`, if not already present in the prior Lower-64 bits.
3. **Iterative:** Iteration over the bits of each Lower-64 in the Upper-64's allocation. 6SENSE tries all Lower-64 bits in the allocation, counting to modulo 16, to cover each nybble.

Sampling Lower-64 bits by allocation leverages organization-level patterns, while iterating prevents duplicates and uses domain knowledge of the most common patterns. Finally, by trying `::1` we leverage behaviors common across allocations.

We note this technique is relatively simple, but as shown in Section 6, it is both efficient and powerful. We expect future work can further mine this portion of the address space.

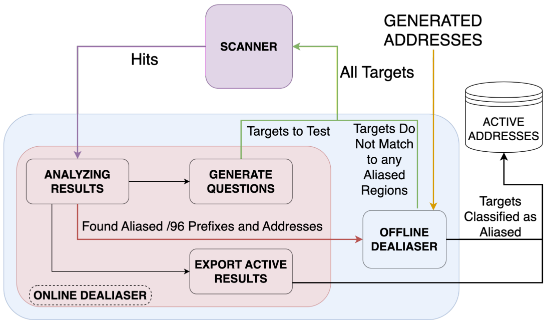

Figure 3: 6SENSE's dealiaser. The system contains both an offline dealiaser that identifies previously discovered regions, and an online dealiaser that deduces if hits are aliased.

5.3 Detecting and Resolving Aliases

The tension between maximizing hits and aliased regions creates a unique challenge. Generators blindly optimizing for hits primarily generate from aliased regions if they are not identified. To prevent this, 6SENSE identifies aliased addresses both before and after scanning to prevent generators from over-allocating later scanning resources to them.

6SENSE's multi-tiered dealiaser, shown in Figure 3, consists of two components. First, we classify aliased addresses using alias lists from prior work [31]. Second, any active address passing prefix-based alias detection is sent to an online dealiaser. This structure leverages known aliases and density-based alias detection, without substantial speed decreases.

5.3.1 Prefix-Based Dealiaser

Offline dealiasing is a useful first step to reducing the overhead of online approaches. 6SENSE's offline dealiaser is implemented in Golang, and it uses the list of aliased prefixes provided by Gasser et al. [31]. However, to ensure we do not filter dealiased addresses or inadvertently discover false positives, we first dealias each prefix using the online dealiasing approach described in Section 5.3.2 at each prefix's granularity. We initialize 6SENSE with a list of mentioned aliases stored in a modified radix tree, where aliased prefixes are represented as leaf nodes divided by bit level. After the initial tree construction, 6SENSE performs alias lookups by sending candidate IPs to the Offline Dealiaser in batches to match the longest prefix. If the target IP does not match an alias, it is sent to the scanner and the Online Dealiaser.

5.3.2 Online Dealiaser

The online dealiaser is designed to work with an IPv6 scanner by determining if a target IP address is in an aliased region at a given prefix. We adapt and modify an offline algorithm proposed by Murdock et al. [50]. We chose a /96 prefix granularity for the online dealiasing. A more specific prefix level may incorrectly label dense regions as aliased, while a less specific prefix level may miss fine-grained aliasing. However, one can change parameters to perform dealiasing under different granularities and thresholds for labeling a region as aliased. To account for loss in our dealiasing, our generator discards allocations where over 50% of active addresses are aliased, providing a lower bound on our findings.

When we encounter a responsive prefix in a batch for the first time after offline dealiasing, the online dealiaser generates 3 more random IP addresses (per [50]) in that prefix and queues them in the scanner. After waiting for scanning to complete, we check how many responses were received for each prefix. If at least 2 out of the 3 addresses for a given prefix responded to our probes, we label that prefix aliased. Otherwise, we label it dealiased. Then, the resulting aliased and dealiased data is returned to the offline deliaser to update the tree, and to the generator for online weight updates.

Unlike prior work, we select a majority threshold (2 out of 3 instead of 3 out of 3) as a conservative lower-bound approach to reduce the probability of exploring aliased regions (false negatives), potentially at the expense of excluding non-aliased regions (false positives). This tradeoff is apt since the probability of finding *any* address in a non-aliased region randomly is low, while the cost of exploring aliased regions is high. We discuss this further in Section 6.2.

5.4 Scanner

Although there are existing IPv6 scanning tools available [5, 30], they lack important functionality, such as blocklisting (ZMapv6, Yarrp6) and packet verification (Yarrp6). Further, ZMapv6 is simply the IPv4 version of ZMap with IPv6 support added on, leading to a sprawling code base containing not applicable features (e.g., packet generation) as well as configuration and usage issues. To address these concerns, we developed a new IPv6-purpose-built scanner, inspired by ZMap [25] but purpose-built for interacting with generative IPv6 systems. We implemented our scanner in Golang, which has been used for other network measurement tools [41, 74].

The system has various packet probing modules (e.g., ICMP Echo Reply scan, TCP SYN scan, and UDP DNS scan), packet validation, input/output parsing, blocklisting (not available in prior IPv6 scanners), monitoring, and feedback facilities. We perform packet validation to confirm received packets are in response to our scan by embedding validation bits in the probe packet [25]. These are generated by hashing the source and destination addresses with a key randomly generated at scan initiation (with the SHA256 algorithm). The scanner then statelessly regenerates and verifies the validation bits.

5.5 Datasets

Seed Dataset. A key facet of 6SENSE is the generalization and learning of IPv6 patterns based on a comprehensive input

Dataset Name	Pop.	Unique	De-Aliased	Active
Censys Certs	Servers	320,328	128,497	75,253
IPv6 Hitlist	All	6,763,519	6,746,480	5,866,234
CAIDA DNS Names	Servers	345,776	268,233	141,019
Cisco Umbrella	Servers	130,742	44,808	35,157
Rapid7 FDNS	Servers	15,015,809	5,154,197	1,651,805
Scamper	Routers	333,811	325,559	25,169
Majestic Million	Servers	135,290	24,134	18,446
Alexa Top 1 Million	Servers	95,375	9,028	7,196
Tranco	Servers	292,090	84,268	64,181
SecRank	Servers	109,686	59,744	35,688
Over All Sources	All	21,278,453	11,104,852	6,349,455

Table 1: Overview of our input seed dataset. We categorize the address types from each dataset and across address status.

set of existing, known active IP addresses. Table 1 presents our dataset. To build this dataset we aggregate potentially active IPv6 addresses from a variety of sources. We begin with the public IPv6 hitlist created by Gasser et al. [30, 73], and augment it with IPv6 addresses collected with Scamper [9]. Furthermore, we collect domain names from CAIDA's DNS [10], Rapid 7 Forward DNS [56], and Censys's dataset of X.509 certificates accessed in December 2022 [12, 22]. For completeness, we additionally use domain top lists: Cisco's Umbrella [65], The Majestic Million [48], Tranco [55], SecRank [69], and an archival version of the Alexa Top 1 Million [2]. For each domain, we use ZDNS [41] to resolve it to IPv6 addresses and add these addresses to our dataset.

To refine the dataset, we dealias and scan all addresses. Active addresses include the total number of active, dealiased addresses on ports TCP80, TCP443, UDP53, or ICMPv6. We perform online and offline dealiasing described in Section 5.3 to identify aliases, including those missed by prior work.

ASes and Allocations. To distill our input seed dataset into ASes and allocations, we need a source of such information. We use the CAIDA Routeviews Prefix to AS mappings [8] to gather prefixes and their corresponding ASes. We use these prefixes as our allocations. In total, we found 49,364 allocations in our seed dataset.

For generation, we revise our prefixes. As our Upper-64 bit generator is designed to generate the next nybble of a prefix, we need starting allocation prefixes in multiples of 4. Thus, for generation, we expand the allocations into all possible allocation values to the closest nybble. We additionally expand any allocation with a prefix less specific than /32 by enumerating all prefixes of size /32 within the less specific prefix. We highlight that adding these additional prefixes allows us to generate a more representative number of Upper-64s per network (Section 5.2) in the case of prefixes that are less specific than a /32. We also emphasize that the nature of our dataset makes the effect of this expansion on performance insignificant. Our dataset [8] is created with routing data. Prior work [40] illustrated that routes with a measurable active IP are most likely to be of size /32 or /48, and rarely less-specific than a /32. In the case of prefixes that are not at nybble boundaries, we expand to a maximum of 8 new prefixes (e.g., when the prefix is 3 bits away from a nybble boundary; i.e., /37 expands to /40), making any over-representation of these networks and performance overhead negligible.

5.6 Ethics

We begin by following the ethics of scanning outlined in the ZMap paper [25]. Namely, we: 1) randomize all scans and packets generated at all phases to ensure network infrastructure is not stressed, 2) operate at a low scanning rate of 10,000 packets per second, total, across all networks (discussed further subsequently), 3) use well-identified (via DNS, whois, and webpages) machines to perform scans, 4) provide parties an opportunity to opt-out of scans, and 5) maintain a blocklist of IP addresses who were opted out of scanning.

Further, we look to recent censorship measurement work [6, 53, 54] which articulate a framework, based on the Belmont [28] and Menlo [4] reports, for reasoning about the risks of Internet measurement. Specifically, the notions of justice, beneficence, and respect for law and public interest. From this framework, we devise methods and procedures that seek to: 1) ensure infrastructure or machines are not stressed or damaged, 2) seek to minimize risk, and 3) ensure that those who bear the risk of measurements are also the beneficiaries of said measurements. The latter point is particularly apt in the context of security studies, as those who bear the risk of these measurements—end hosts—are specifically the individuals who will receive direct benefits from understanding the security risks and associated vulnerabilities, via our disclosure of these discoveries to CERTs and ISPs.

IPv6 Scanning Speed. A unique problem in IPv6 Internet scanning is aliasing (Section 2), whereby vast continuous address ranges map to a single or handful of devices. Thus a generative IPv6 scanner could direct a large portion of its scanning budget towards a single device, believing it to be an entire network. This raises the question of what is an ethical speed at which to scan the IPv6 Internet. Our work seeks to identify and eliminate aliased regions (Section 5.3), potentially at the cost of alias false positives (Section 6.2). However alias classification errors are possible and scanning incurs some inherent risk. Returning to the framework above, our goal is to minimize potential risk; we propose (and adhere to within this work) a scan-rate of 10,000pps. Such a limit ensures that the overall scan-rate of the system does not exceed approximately 5-7Mbps, a value significantly below many modern Internet connections (e.g., the classification of broadband [15]). This value is such that even if aliases are explored and our randomization of IPs is poor, our system does not cause a Denial-of-Service (DoS) event. We note that such efforts do not eliminate risk, rather, our work seeks to minimize risk such that the benefits of large-scale Internet measurement exceed those inherent risks (beneficence), while ensuring those that bear the risk are also those that benefit from the results (justice).

We believe that scanning at higher speeds can be ethical if specific attention is paid toward the reliable identification of IPv6 aliases—such a problem is not yet well studied, and worthy of exploration. To this end we note that our system has the capability of running significantly faster (Section 6.6), but we strongly discourage such speeds until such time that better dealiasing techniques exist or other safety mechanisms are in place. We note that a bad-actor *could* utilize our system to launch DoS attacks. Such an outcome is possible, but not rational; our system is a strictly worse DoS tool than even random address selection due to the significant performance overhead of machine-learning-based packet generation.

IPv6 Scanning Best Practices. Summarizing our principles above and prior work [25,66], we suggest (and document how we adhere to) the following practices for IPv6 Scanning:

- Clearly identify all scanning activity with DNS records, webpages, and where possible IP WHOIS information that identifies the researchers and provides a means of contact.
  - *All machines used for scanning have forward and PTR DNS records that indicate the machine is used for network scanning and identify our organization. The scanning IPs/domains contain webpages that explain our activity and provide an email address within our organization to contact for more information or to opt-out.*
- Respond to questions openly and promptly, and provide a means to opt-out of future measurements.
  - *The webpage hosted on each scanning IP/domain includes a contact email at our organization. We respond to all emails promptly (within one business day) and immediately exclude requested ranges from scanning. We received three contacts during the course of this project, responded to each, and none requested opt-ing out.*
- Work with local network operators to ensure research activity does not disrupt normal operation, and work with CERTs or similar entities to remediate discovered vulnerabilities.
  - *We worked with our local IT unit to place our scanning machines outside of infrastructure that could be impacted by scanning, and will be continually passing discovered vulnerable IPs to our local CERT.*
- Where possible, consider the use of public datasets rather than performing active measurements from scratch.
  - *We used existing datasets as seeds, but given the nascent nature of this work, no such public dataset exists.*
- Attempt to identify and remediate aliases during scanning.
  - *We develop new methods to identify and remediate aliases, discussed in Section 5.3.2.*
- Rate-limit, distribute, and randomize all scanning activity such that errors in alias detection do not result in DoS.
  - *As discussed above, we rate-limit all scanning activity to 10,000pps, and fully randomize all scanning activity.*

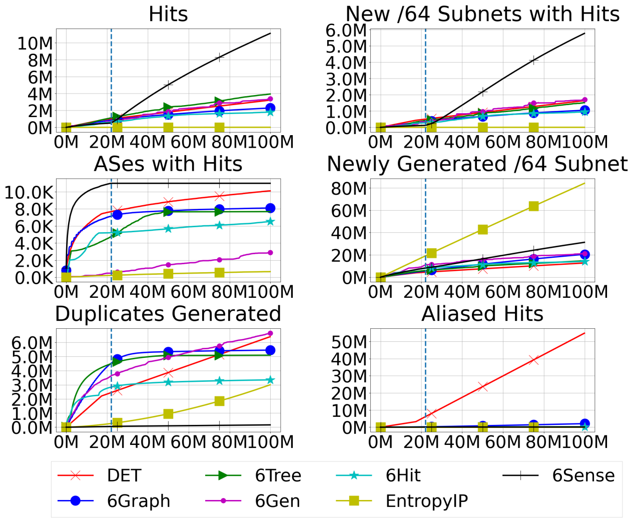

Figure 4: ICMP results across our six metrics, comparing 6SENSE to other approaches. For all figures, the x-axis is the cumulative number of addresses generated/scanned, and the y-axis is the cumulative sum of the given metric. The vertical line represents when 6SENSE transitions from Breadth to Depth based on the diversity gradient. 6SENSE finds 2.8x more hits than any other method, 3.3x more /64s than any other method, and does so with nearly no duplicates or aliases.

6 Evaluation

For our evaluation of 6SENSE, we use the metrics from Section 4 to show how well our system can find hosts in the IPv6 space. Our evaluation focuses on comparing 6SENSE to the six best previously proposed address discovery methods.

6.1 Evaluation Setup

All systems were executed and tested on a 64-core AMD EPYC 7452 processor with 512GB of ram and an NVIDIA A40 GPU with 48GB of GPU Ram. For the evaluation, our scanning system operated 10,000 packets-per-second (pps). All hosts were scanned from a purpose-built academic scanning network in North America.

Limitations of prior methods. While this section focuses on comparing 6SENSE to 6 prior methods, we note that other methods exist. Broadly, we found that prior methods not evaluated here were either closed-source with insufficient description to reproduce [35,60,72], or could not generate or execute in a reasonable amount of time [17–19, 70]. For example, when using 6Forest [70] to generate 100M IPs, it ran for over two weeks before exhausting memory. Additionally, prior work [73] also encountered challenges running 6GAN [18]. Resource limitations for training and generation across models limited us to evaluating 100M IPs. We also filter AS12322 due to prolific aliasing that does not conform to traditional sizing. We note prior work found significant portions of their hits in this AS [61], likely pointing to substantial aliasing.

6.2 Dealiasing Thresholds

Prior generative scanning that attempted dealiasing generated 3 random packets to a given /96 prefix, and required all 3 sent packets to return hits for the prefix to be classified as aliased [50]. During the development of 6SENSE, we noted that some aliased /96 prefixes displayed inconsistent and lossy behavior resulting in alias detection *false negatives*. These regions could then in turn produce large numbers of aliased hits given the online nature of the system.

To understand the effect of this number-of-hits threshold on alias detection, we conducted a small-scale study of 10M generated addresses (distributed randomly across all seed allocations) where we varied our alias detection threshold between requiring 1 to 3 (out of 3) generated packets to be hits in order to classify a range as aliased. Across 190K total active /96s we find a threshold of 3 gives us 4,255 aliased /96s. Decreasing to 2 results in the classification of 17 additional potentially aliased prefixes. Decreasing to 1 results in 33 additional potentially aliased prefixes.

For each additional prefix evaluated at both thresholds of 1 and 2, we perform manual investigation. We find that the majority of discovered prefixes belong to a single AS, AS28573, with behavior that may denote either aliases at smaller granularity than /96 or dense regions. Excluding this AS, we find that at 2 packets, we identify 6 additional true aliased, but lossy, regions; repeated additional probes could resolve the alias. At thresholds of both 1 and 2 addresses, we find 1 region that we suspect is dense.

For the remainder of our experiments we select a threshold of 2 out of 3 packets, which we believe provides an acceptable trade-off between controlling for lossy-but-aliased regions, and alias detection false-positives excluding a handful of dense regions. We note the problem of alias resolution is complex, and deserves significant further research.

6.3 ICMP Results

Previous IPv6 work [30, 73] indicates that ICMP echo requests are most likely to be responsive if any other protocol (e.g. TCP80, TCP443, or UDP53) is responsive. Thus, we begin our evaluation by comparing 6SENSE to DET, 6Tree, 6Hit, 6Graph, 6Gen, and Entropy/IP on ICMP echo scans. We note that performing a comparison against these frequently incomplete solutions required significant effort, including developing custom toolchains and scanners to deploy them.

Our comparison results are shown in Figure 4. After 100M probes, **6SENSE finds 2.8x more hits than the prior highest-performing discovery method**. It does so while also discovering 3.3x more new /64 subnets—a metric more likely to represent new end-sites on the Internet. It does this efficiently, generating minimal duplicates and aliases. 6SENSE has similar AS coverage to other methods. As our method focuses on address discovery via known ASes, we expect a plateau of discovered ASes matching the input dataset, although even this outperforms most prior work. Note while Entropy/IP generates significantly more Upper 64s, it performs very poorly, as it generates many IPs in unassigned/unallocated networks.

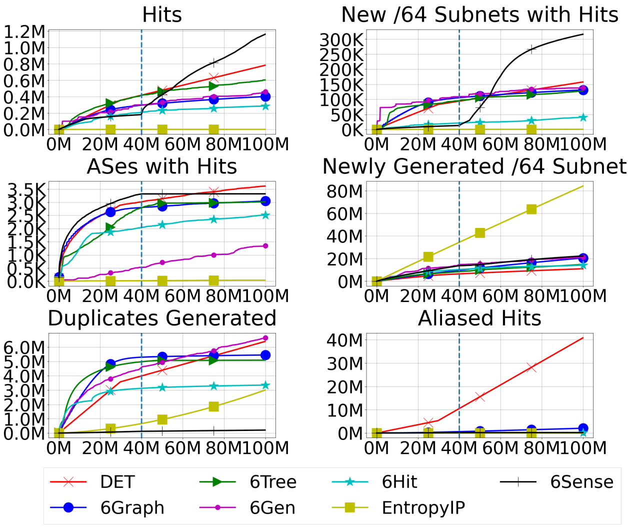

Figure 5: TCP443 results across metrics and approaches. 6SENSE finds 1.4x more hits and 2x more /64s than any other method, with minimal duplicates and aliases.

A key observation is the substantial number of duplicates and aliases generated by other methods. Without 6SENSE's online approach to dealiasing as a first-order concern, most hits from these tools are duplicates or aliases, yielding poor or incorrect security results. Another interesting observation is that across each of the positive metrics, a different non-6SENSE tool yielded the best results (behind 6SENSE), indicating that, pre-6SENSE, no single alternative approach was optimal for IPv6 ICMP Internet-wide scanning.

6.4 TCP443, TCP80, and UDP53

Prior generative work focused *exclusively* on finding responsive hosts using ICMP. While ICMP is important, many security applications require identifying hosts responsive to specific ports. Thus we perform a similar evaluation to ICMP except we focus on completing three-way handshakes for TCP443 and TCP80 and finding DNS servers on UDP53.

Figure 5 shows the results for finding responsive hosts on TCP443. After 100M probes, 6SENSE continues to yield better results on both Hits and New Upper-64s, finding **1.4x more hits and 2x more active /64s than other methods**, doing so efficiently with minimal duplicates and aliases. We note that our transition from Breadth to Depth, while optimal for new hosts, produced slightly worse AS coverage. We explore

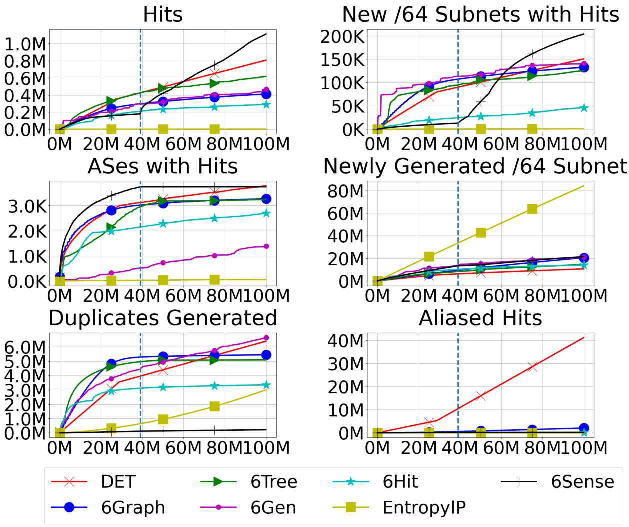

Figure 6: TCP80 results across metrics and approaches. 6SENSE finds 1.3x more hits and 1.3x more active /64s than other methods, and does so more efficiently with minimal duplicates and aliases.

this trade-off further in Section 6.8, finding AS coverage can improve at the expense of raw hosts. Section 7.1 explores this scan's TLS certificates.

Figure 6 shows the results of identifying responsive hosts on TCP80. After 100M probes, 6SENSE continues to yield better results for both Hits and New Upper-64s, efficiently finding **1.3x more hits and 1.3x more active /64s than other methods** with minimal duplicates and aliases. We note that our transition from Breadth to Depth, does not yield reduced AS coverage compared to other approaches. These results are broadly similar to those for TLS443.

Figure 7 shows results for 6SENSE and other methods identifying responsive DNS resolvers on UDP53. In this evaluation, we generate a valid DNS request for `akamai.com` (a domain not expecting censorship [54]) and consider a hit to be a host returning any valid DNS response, as our goal is understanding if the host speaks the protocol. We observe 6SENSE performs substantially better in identifying UDP53 hosts than any other method, and its performance gap is larger than for ICMP and TCP443. After 100M probes, **6SENSE finds 3.6x more hits and 4x more active /64s than other methods**. On UDP53, 6SENSE experiences a dramatic improvement when transitioning from Breadth to Depth, signifying the identification of dense resolver regions. Interestingly, aliasing is limited across methods on UDP53, with models only finding tens of thousands of aliased hits. We are uncertain why UDP53 exhibits less aliasing than ICMP and TCP protocols.

Overall Discovery. Combining all 6SENSE's output, we found 11,882,633 Hits and 6,128,152 new active Upper-64s, substantially more than any prior IPv6 generative scanning method on a 100M budget, shown in Table 2.

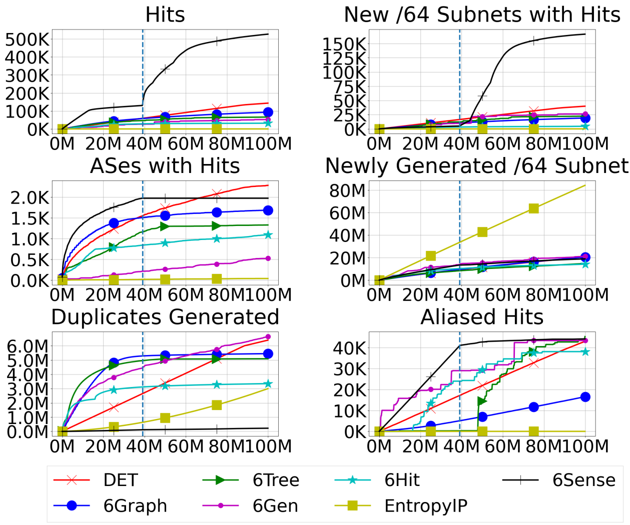

Figure 7: UDP53 results across metrics and approaches. 6SENSE finds 3.6x more hits and 4x more active /64s than other methods, with minimal duplicates and aliases.

	ICMP	TCP80	TCP443	UDP53	Total
/128s	11,118,330	1,113,150	1,162,222	526,606	11,882,633
/64s	5,776,637	203,948	316,372	166,573	6,128,152

Table 2: Overall number of new active IPv6 hosts (/128s) and /64s 6SENSE discovers across all experiments.

6.5 Lower-64 Analysis

Beyond hits and new end-sites discovered, it is also important to understand the diversity of Lower-64 bit patterns (IIDs) uncovered and how those were identified during our Lower-64 generative phases. We begin by examining this in 3 phases of 6SENSE's Lower-64 generation as described in Section 5.2.3: Phase 1 (using seed IIDs, split out into ::1 or other IIDs); Phase 2 (trying ::1 if not seen already), and Phase 3 (iterative sampling). Table 3 shows these results across our 4 scan experiments. We notice across ports that iterative sampling provides the greatest number of active IIDs (at over 60% for TCP80, TCP443, and UDP53 and almost 40% for ICMP). Similarly, ::1 takes up a large portion, followed finally by seed IIDs from the same allocation. Phase 2, expectedly, has minimal effect as ::1 is quite common within seed datasets.

We explore discovered IIDs further with Figure 8, a CDF

	Phase 1, ::1	Phase 1, Other	Phase 2	Phase 3
ICMP	35.76%	24.37%	0.06%	39.79%
TCP80	14.91%	9.33%	0.03%	75.71%
TCP443	24.30%	7.91%	0.04%	67.74%
UDP53	29.84%	3.50%	0.03%	66.61%

Table 3: Breakdown of discovered addresses by 6SENSE Lower-64 generation phase, across each experiment.

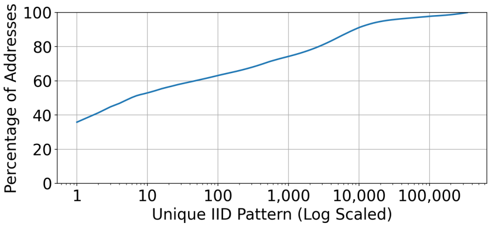

Figure 8: CDF of the discovered Lower-64 (IID) Patterns across hits in our ICMP experiment.

	6SENSE Total	6SENSE Non-Dealias.	DET	E/IP	6Gen	6Hit	6Tree	6Graph
10k pps	9,503	6,023	2,376	5,799	8,494	8,291	7,982	6,944
Unconst.	118,604	72,098	3,217	26,123	97,087	91,010	91,324	23,062

Table 4: Scan speed (pps) of 6SENSE and prior systems. We show 6SENSE's total scan rate and the rate excluding dealiasing probes (as online dealiasing consumes some scan budget). Constrained to 10k pps, 6SENSE is over 2x faster than DET, the closest competitor in hits, and commensurate with other methods. Unconstrained 6SENSE is 22x faster than DET.

of the discovered IID patterns across hits in our ICMP experiment. The most common pattern was ::1 (per Table 3), appearing for 35% of addresses. While the most common 100 IID patterns account for 65% of addresses, 6SENSE discovers many other patterns, with the remaining 35% comprising 342,594 unique addresses. Thus, 6SENSE discovers a diverse set of addresses both learning observed common patterns while also discovering new ones.

6.6 Runtime Evaluation

Table 4 shows the generation and scanning speed of 6SENSE and prior methods when constrained to 10k pps and when unconstrained (discussed next). When constrained, 6SENSE devotes a sizable portion of its scan budget to online dealiasing, unlike most other systems which do not perform online dealiasing. Still, when compared to DET, the closest competitor to 6SENSE in overall hit-performance, 6SENSE generates and scans over twice as fast, and is commensurate with other prior systems (which yield fewer hits).

As 6SENSE is a learning system, its runtime is divided between training, occurring only once per dataset, and execution (the generative and scanning process per scan). Scans can be run numerous times and for varying lengths on a single trained model. In the constrained 100M experiment, 6SENSE trains once for 4.27hr and then scans 100M generated IPs in 4.61hr. Factoring training into overall execution time, 6SENSE is faster than DET (8.88hr vs 11.69hr), but slower than other approaches; repeated and larger scans amortize this training cost and result in 6SENSE performing better over time. We note that slower execution, especially at slower scan speeds, is expected, given that 6SENSE is a deep learning solution.

	Hits	New Upper-64s	No-Send Speed
6SENSE	11,132,715	6,007,234	72.1K pps
No LSTM	4,881,122	-	84.5K pps
No Online Dealiasing	5,634,095	2,488,028	71.4K pps

Table 5: Ablation study of 6SENSE's LSTM and online dealiaser. We evaluate the yield of each ablation, as well as the system's speed in a No-Send simulated experiment.

	DET	6SENSE: Depth	6SENSE: Breadth
ICMP	10,109	10,998	12,503
TCP80	3,787	3,749	4,536
TCP443	3,707	3,319	3,970
UDP53	2,283	1,973	2,410

Table 6: Diversity of ASes, across each experiment, for DET, 6SENSE's Breadth and 6SENSE's Depth phase. We find optimizing towards an AS-diversity metric increases the overall AS coverage by 14–22%, depending on the protocol.

Unconstrained Speed and Parallelization. While we conducted all our experiments at a rate of 10k pps (per our ethical framework in Section 5.6), here we explore how fast our approach can run without an artificial rate limit. To explore this question while respecting our ethical framework, we modified 6SENSE such that: 1) we remove rate-limit constraints, 2) we do not actually transmit packets, and 3) we simulate a uniformly distributed hit-rate of 11% on generated packets, and 0.1% on detected aliases. Both of these values are based on our ICMP results in Section 6.3. This approach allows us to observe how our system scales without the ethical challenges of scanning the Internet at high speed. We further modify our experimental setup such that we parallelize the *generation* LSTM component of 6SENSE across multiple GPUs. The design of 6SENSE affords straightforward parallelization as the LSTM can be copied across devices, and generation can be performed in parallel, isolated to individual prefixes. We then evaluate both default and parallelized 6SENSE, utilizing 8 NVIDIA A40 GPUs (Section 6.1).

Default 6SENSE can scale to 26,917pps, more than 2.6x of our intentionally limited evaluation. Meanwhile, parallelized 6SENSE can achieve 72,098pps, or more than 7.2x our intentionally limited evaluation. The sub-linear speedup originates from only parallelizing generation, not learning nor updates. Table 4 shows the unconstrained performance of 6SENSE compared to other systems. When unconstrained, 6SENSE generates and scans 22x faster than DET, but slower than some other systems (which yield less hits). While we expect further optimizations are possible, we stress that generative scanning tools should not be operated at such speeds without significantly more exploration in effective and safe dealiasing.

6.7 Ablations

We next investigate how much of 6SENSE's yield is attributable to our Upper-64 generation techniques, Alloca-

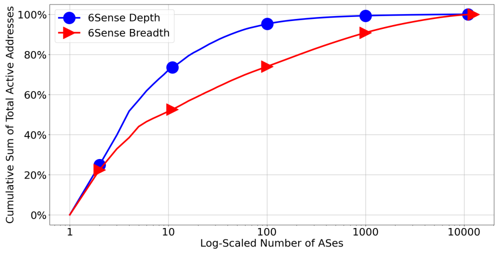

Figure 9: CDF of the distribution of ASes for ICMP experiments, optimizing for hits (Depth) vs diversity (Breadth). X-axis is log scaled. Optimizing for diversity increases the number of ASes by 14%, with a more uniform distribution.

tion sampling+Lower-64 generation techniques, and online dealiasing approach. To understand this, we evaluate three versions of 6SENSE on ICMP with 100M probes: 1) full 6SENSE, 2) a version of 6SENSE where we replace our Upper-64 generating LSTM with random sampling from our seed dataset's /64s within each allocation ("No LSTM"), and 3) 6SENSE without online dealiasing (but with the full LSTM). For our experiment without online dealiasing, we still perform offline dealiasing of the seed dataset, consistent with all other experiments. We additionally evaluate each version's speed through a similar No-Send simulation as in Section 6.6.

Table 5 depicts the performance of each version, evaluating the Hits and New Upper-64s in the ICMP experiments, and the No-Send simulated speed. Removing the LSTM results in only 4.8M active addresses compared to 6SENSE's 11.1M (and inherently does not yield new Upper-64s). Thus, the hits contributed by the LSTM are not from only repeating Upper-64s in the seeds, but also from its discovery of new Upper-64s. Removing online dealiasing results in 5.6M hits across 2.5M Upper-64s, down from 6SENSE's 11.1M and 6M, respectively. Note that our system without online dealiasing still outperforms other models. From our No-Send simulated speed analysis, we find that removing the LSTM component speeds up the system by roughly 17%, and the online dealiaser has negligible impact on No-Send speed.

6.8 Optimizing for Metrics: Breadth vs. Depth

To understand how our diversity gradient approach to Breadth vs. Depth allows us to balance between hits and AS diversity, we performed an alternative evaluation of 6SENSE on ICMP, TCP80, TCP443, and UDP53. In this alternative version, we focused on optimizing solely for AS diversity by staying in the Breadth phase for an entire generative run. Table 6 shows the results compared to DET, which had the best AS diversity behind 6SENSE. This experiment reveals that we can increase AS coverage by 14–22%, depending on the protocol.

This demonstrates the customizability of our weighted allocation search structure. Adjusting the threshold and metric allows us to tune the model for different metrics. The weighting structure lends itself to further customizations, e.g., weighting towards allocations where we have yet to find addresses.

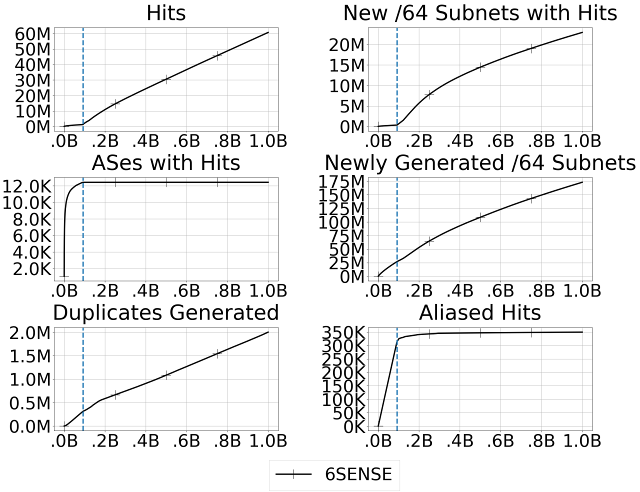

Figure 10: ICMP results across metrics and approaches for 1 Billion Addresses. 6SENSE finds 60.7M active IPs in 22M Upper 64s with minimal duplicates and aliases.

Depth vs Breadth Host Composition. We next look at the composition of each 6SENSE strategy. The overall composition of the hosts 6SENSE discovered in our prior evaluation is shown in Figure 9. We find that 6SENSE discovers hosts across a wide variety of ASes. Cloudflare (AS 13335, a CDN), Sky (AS 5607, a residential broadband provider), and Claro (AS 28573, a mobile provider) are the top network providers, indicating a mix of both servers and client addresses. Returning to the trade-off between Breadth and Depth, we also explore the host composition of 6SENSE in Breadth mode (optimizing for AS diversity), also shown in Figure 9. While the total number of ASes increases by 14%, we see that the composition of hosts is significantly more uniform, with the top ASes comprising fewer total hits. This increase in compositional diversity is compelling, indicating that optimizing toward specific metrics based on the scanning application is both feasible and fruitful.

6.9 Large Scale ICMP Scan

Finally, Figure 10 shows the results of using 6SENSE to identify responsive hosts on ICMP for a large scale scan of 1 Billion addresses. At 1 Billion addresses, 6SENSE continues to find active hosts, with 60.7M active IPs in 22M Upper 64s. For this experiment we set our G_b threshold to 0.0005, 10x lower than for 100M, and similarly our Depth N_i to 10M to account for the 10x increase in scan budget. We notice this leads to similar AS findings, 12.4K, as the Breadth method in Section 6.8, since this threshold leads to a similar exploration cutoff point of 92M (compared to 100M in Section 6.8).

Active Hosts	Total Certs	Unique Certs	New Certs
1,165,518	805,617	157,690	109,128

Table 7: Observed certificates across the new IPv6 addresses discovered by 6Sense. To calculate new certificates we compare our discovered dataset with Censys, finding 109,128 certificates that have not been previously observed via scans, name-based crawls, or certificate transparency logs.

Category	**Example**	**Count**
Consumer Routers/Modems	DLink	78,532
	Fritz	978
	Hitron	626
	Ubiquiti	90
	Zyxel	73
Security Tools	OPNsense	23
	Fortinet	19
	Sangfor	14
	HillStone	4
Virtualization Tools	Kubernetes	52
	VMWare	19
Enterprise Switches	Brocade	64
	Cisco	60
	Lenovo	1
Printers	HP	351
	Lexmark	5
Unknown	-	28,217

Table 8: Case study of security-specific TCP443 device types.

7 Security Analysis

To demonstrate the utility of 6Sense, we perform the first Internet-wide scanning-driven IPv6 host security study, analyzing 4 dimensions: 1) TLS certs, 2) Exposed internal services, 3) Applicable CVEs, and 4) Common open ports.

7.1 TLS Certificates

For all IPv6 TCP443 hosts 6Sense discovered (Section 6.4), we use ZGrab2 [74] to connect and negotiate a TLS connection. We record connection metadata, certificate information, and payload data. Table 7 presents our measurement results. Across 1.2M hosts active on port 443, we collected certificates from 806K hosts, making up 158K unique certificates.

To understand the additive value of IPv6 scans vs. IPv4 scans, we compare all certificates we collected to the certificate dataset in Censys [22], one week after collection. Censys catalogs certificates through Internet-wide IPv4 scans of 3,500 TCP ports (including TCP443) on a rolling basis [13]. It additionally incorporates certificates from domain name scans (e.g., domain top lists) and certificate transparency logs. Thus, observed certificates not found in the Censys dataset are most likely IPv6-specific and from a unique population not found via other methods. From this analysis, we discover 109K never-before-seen certificates in the wild, pointing to a unique population of devices that are observable only in IPv6. Section 7.2 show these are largely end-user devices.

Device Category	**Device Name**	**CVEs**
Switches	Cisco WS-C3650	10
	Brocade ICX 7450	7
	Lenovo EN4093R	1
Routers	D-Link DIR-853/ET	12
	D-Link M15/R15	1
	EdgeMax (various)	1
	ZyXEL VMG3925	7
	ZyXEL VMG8825	2
	ZyXEL EX3301-T0	3
	AVM Fritz!Box	12
Printers	HP M479	3
	HP Officejet 3830	5
	HP Officejet 4650	2
	HP Officejet Pro 8600	2
	HP LaserJest M15w	6
	HP Deskjet 5730	1
Total	-	70

Table 9: *Vulnerability Analysis.* Potential CVEs for devices discovered by 6Sense.

Among the identified new certificates none are browser-trusted, unsurprisingly, as browser-trusted certificates must appear in approved certificate transparency logs [36], and Censys incorporates all such logs. Examining the **purported** issuers: the common issuers are D-Link, a producer of consumer routers (78,533 unique certificates), and Let's Encrypt (10,230 unique certificates). DigiCert (1,407 unique certificates) is the next most common issuer, followed by a diverse set, including Amazon, Microsoft, Cisco, VMWare, and hobbyist projects. We stress that since these are not trusted certificates, the identity of these issuers cannot be trusted. We further explore this population of discovered devices next.

7.2 Population Case Studies

We perform a case study of the IPv6 hosts that 6Sense newly discovered to better understand what devices and services are exposed. We began by exploring all purported issuers on certificates from TCP443, manually selecting issuers of security interest, such as consumer routers, enterprise switches, and VM services. For hosts with certificates issued by such issuers, we crawled the website landing page hosted on TCP443. We manually created webpage fingerprints for several types of devices and applied them to characterize a subset of the hosts.

Table 8 presents an overview of our findings. We broadly classify our discovered devices into the following categories: (1) Routers, (2) Security Tools, (3) Virtualization Tools, (4) Enterprise Switches, and (5) Printers.

We found publicly exposed login pages for all devices found under the categories of routers, switches, security and visualization tools. Such devices are especially vulnerable if misconfigured (i.e., default credentials are still in use). For many printers, we found publicly exposed configuration landing pages that required no authentication. Such configuration pages give the ability to: (1) perform various actions using the

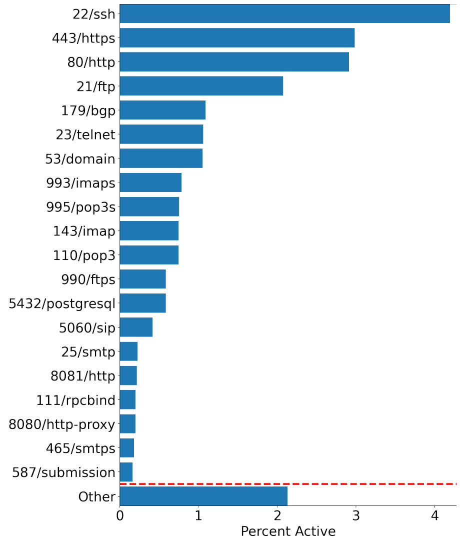

Figure 11: Top 20 most common ports from a random sample of discovered ICMP active hosts. SSH is the most common open service, with numerous sensitive networking (e.g., BGP) and security (e.g., PostgreSQL, rpcbind) services exposed.

printer (print/scan), (2) change the settings of the printer, (3) update device firmware, (4) view printer history, or (5) infer sensitive information such as physical location.

Broadly, we discovered a large number of Internet-exposed configurations and login pages on tens-of-thousands of devices traditionally not observable on the open Internet. These include routers, modems, printers, enterprise switches, virtualization tools, and security products. These services, particularly those of virtual machine services, enterprise network hardware, and security tools, represent significant security threats whereby an attacker can compromise key infrastructure if weak or default credentials are in use. Exposure of such devices has two potential causes: (1) They depend on IPv6 non-discoverability for security, which clearly does not hold, or (2) these devices have IPv6-specific misconfigurations. Section 8 discusses this further.

7.3 CVE Analysis

From the devices discovered from certificates, we sample from each category to perform an in-depth manual analysis to extract device-specific information. This process involved visiting the hosts manually in a browser to understand: (1) the type of content hosted on these hosts and their purpose, (2) any discernible product-specific information such as device name. We explicitly **did not** perform any form of automated vulnerability testing for ethical reasons; we do not own the devices, do not have consent, nor do we have confidence that such testing is safe as it could crash or damage the devices.

We were able to successfully extract device name and model for: (1) routers belonging to D-Link, ZyXEL, and Hitron, (2) commercial switches belonging to Cisco, Brocade, and Lenovo, and (3) printers belonging to HP. For extracted devices, we then used a combination of different vulnerability databases [49,51] to obtain CVEs associated with each device, presented in Table 9. We find a total of 70 CVEs associated with the specific devices, indicating *potential* vulnerability, as we did not explicitly attempt to exploit these vulnerabilities.

IPv4 vs. IPv6 Differences. During our manual analysis, we repeatedly discovered landing pages for printers that appeared to be connected to WiFi access-points via an IPv4 address in the private range (devices behind a NAT) but were publicly accessible through IPv6. This points to a specific configuration challenge where consumers and/or their devices broadly utilize NAT as a security tool, discussed more in Section 8.

7.4 Port Scans

To understand the population and potential security implications of our discovered hosts, we used Nmap [47] to conduct TCP port scans of the top 100 ports (according to Nmap) across 6Sense's ICMP hits. Given Nmap's limited speed, we scan a random 0.1% sample of our discovered addresses.

Figure 11 presents an overview of our results. We find that the most common open TCP service is SSH, on over 4% of discovered hosts, followed by HTTP and HTTPS (∼3% each), and FTP (∼2%). Such a profile may be indicative of consumer network devices. Interestingly, services such as BGP also appear, suggesting non-consumer network equipment. More concerning is the presence of open PostgreSQL, rpcbind, and other security-sensitive services, commonly exploited in the past. These observations reinforce those made in Section 7.2, that misconfigurations and reliance on IPv6 non-discoverability are prevalent.

7.4.1 IPv4 Comparisons

We now compare the distribution of popular open ports between our IPv6 results and IPv4. To do this, we collect the IPv4 port distributions from Censys [22] for the 100 ports we scanned in IPv6. Figure 12 shows the results of this comparison. To properly compare IPv4 and IPv6, we need to normalize the volume of hosts represented in each dataset, as there is no effective way of comparing exhaustive scans to IPv6 generative scans. We do this by examining the total fraction each protocol contributed to the set of ports found.

We show the top 20 ports found for IPv4 vs. the top 20 ports found for IPv6 with this normalization. We notice significant differences in the distribution and top ports between IPv4 and IPv6. While both have high concentrations at 443 and 80, IPv6 weights more heavily towards 22/ssh. This suggests a higher concentration of discoverable devices with SSH open. Notably, IPv6 shows much higher 179/bgp open ports, suggesting more discoverable non-consumer networking devices. Alternatively, IPv4 finds significantly greater distributions of 8443/https and 8008/http and sees 5060/sip more often. Other

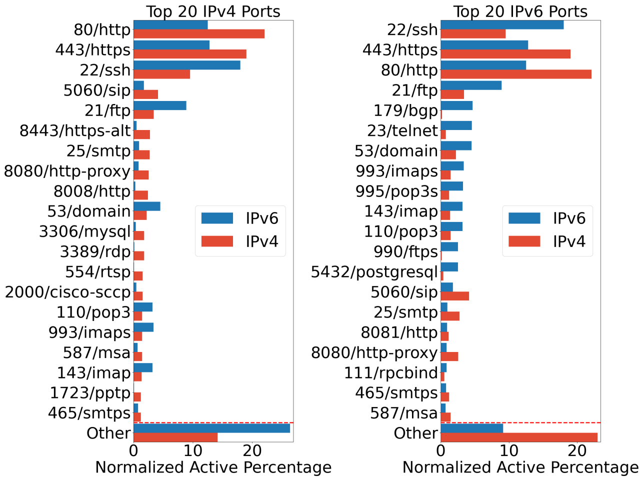

Figure 12: Top 20 most common ports for IPv4 vs. IPv6. Data is normalized by cumulative total number of active ports.

differences seem less intuitive, such as the higher prevalence of 3306/mysql in IPv4 vs. 5432/postgresql in IPv6 and the higher 25/smtp in IPv4 vs. 993/imaps and 995/pop3s in IPv6.

It is unclear whether these fundamental differences stem from a different distribution of discoverable devices in IPv6, difficulties in comparing such differing protocols, or differences in the fundamental distributions of IPv6 and IPv4 devices. Prior work [21] performed IPv4 to IPv6 comparisons on dual-stack hosts and found similar results with higher proportions of SSH and BGP services open on IPv6 compared to IPv4. Our results combined with these prior observations point to the possibility of fundamentally different scanning-accessible populations with different security properties, validating the need for 6Sense and showing its security potential.

8 Concluding Discussion

We developed 6Sense, an end-to-end IPv6 scanning system that significantly outperforms existing IPv6 host discovery methods along multiple metrics. 6Sense's design overcomes limitations of prior approaches, affording practical large-scale IPv6 Internet scans. We demonstrated its utility by conducting a security analysis of IPv6 hosts at scale, uncovering several serious concerns. Here, we distill key lessons learned through developing 6Sense, as well as directions for future work.

Metrics and Use-Cases Matter. Our work shows that based on the metrics used and the specific protocol, IPv6 scanning solutions can vary significantly in terms of both performance and end-result. While we defined six initial scanning metrics, future work must carefully consider the appropriate metrics for different security studies, based on how those metrics impact dataset biases and end-to-end results.

Further Innovation. While 6Sense enabled us to conduct the first scanning-based security study of IPv6 hosts, there is still room for algorithmic improvements, particularly in exploring allocations and generating candidate Lower-64 bits. We hope 6Sense can serve as a framework for the community to adopt and improve IPv6 scanning tools and datasets.

Dealiasing is not solved. While we present an initial aliasing solution, we note that the actual problem of IPv6 aliasing is significant, and requires a more sophisticated dynamic solution that is able to differentiate dense regions from aliased regions, robust to network effects, and adaptive to alias size.

Indications of IPv6 Security Problems. Our results uncover serious security issues across the IPv6 address space. We speculate that these issues arise from three potential causes. First, differences in firewall and service configurations between IPv4 and IPv6 result in device operators inadvertently leaving IPv6 services exposed [21]. Second, the vast nature of the IPv6 space has led to a "security through obscurity" mindset, where device operators assume that their devices are protected by simply being undiscoverable. This clearly does not hold, as 6Sense demonstrates. Third, the widespread deployment of NAT for IPv4 masked extensive end-host configuration issues, now exposed in IPv6 since hosts are no longer hidden behind NATs. All told, these findings motivate the need for substantial investment in IPv6 network security.

9 Acknowledgements

This work was supported by a Georgia Tech Research Institute Independent Research and Development grant, funding from Cisco Systems, ONR award N00014-23-1-2080, NSF CNS award 2319315, NSF CAREER award 2239183, and by NSF Graduate Research Fellowship DGE-2039655. The authors also thank Zakir Durumeric for his valuable feedback.

References

[1] David Adrian, Karthikeyan Bhargavan, Zakir Durumeric, Pierrick Gaudry, Matthew Green, et al. Imperfect forward secrecy: How diffie-hellman fails in practice. In *ACM CCS*, 2015.

[2] Alexa. Alexa top 1 million. http://s3.amazonaws.com/alexa-static/top-1m.csv.zip.

[3] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J Alex Halderman, et al. Understanding the mirai botnet. In *USENIX Security Symposium (USENIX Security)*, 2017.

[4] Michael Bailey, David Dittrich, Erin Kenneally, and Doug Maughan. The menlo report. *IEEE Security & Privacy*, 2012.

[5] Robert Beverly, Ramakrishnan Durairajan, David Plonka, and Justin P. Rohrer. In the IP of the beholder: Strategies for active IPv6 topology discovery. In *Internet Measurement Conf.*, 2018.

[6] Abhishek Bhaskar and Paul Pearce. Many roads lead to rome: How packet headers influence DNS censorship measurement. In *USENIX Security Symposium (USENIX Security)*, 2022.

[7] Kevin Borgolte, Shuang Hao, Tobias Fiebig, and Giovanni Vigna. Enumerating active ipv6 hosts for large-scale security scans via dnssec-signed reverse zones. *IEEE S&P*, 2018.

[8] CAIDA. Routeviews prefix to as mappings dataset for ipv4 and ipv6, 2023. https://www.caida.org/catalog/datasets/routeviews-prefix2as/.

[9] CAIDA. scamper. https://catalog.caida.org/software/scamper, 2023.

[10] The caida ucsd ipv6 dns names dataset - 12/19/2021, 2023. https://www.caida.org/catalog/datasets/ipv6_dnsnames_dataset/.

[11] B. Carpenter and S. Jiang. Significance of IPv6 Interface Identifiers. RFC 7136, 2014.

[12] Censys. Bulk data. https://search.censys.io/data.

[13] Censys. Internet scanning intro. https://support.censys.io/hc/en-us/articles/360059603231-Censys-Internet-Scanning-Intro.

[14] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS Workshop on Deep Learning*, 2014.

[15] Federal Communications Commission. Compare broadband availability in different areas, 2021. https://broadband477map.fcc.gov/#/area-comparison?version=jun2021&tech=acfosw&speed=25_3&searchtype=county.

[16] Alissa Cooper, Fernando Gont, and Dave Thaler. Security and Privacy Considerations for IPv6 Address Generation Mechanisms. RFC 7721, 2016.

[17] Tianyu Cui, Gaopeng Gou, and Gang Xiong. 6GCVAE: Gated Convolutional Variational Autoencoder for IPv6 Target Generation. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2020.

[18] Tianyu Cui, Gaopeng Gou, Gang Xiong, Chang Liu, Peipei Fu, and Zhen Li. 6GAN: IPv6 Multi-Pattern Target Generation via Generative Adversarial Nets with Reinforcement Learning. In *International Conf. on Comp. Comms. (ICCC)*, 2021.

[19] Tianyu Cui, Gang Xiong, Gaopeng Gou, Junzheng Shi, and Wei Xia. 6veclm: Language modeling in vector space for ipv6 target generation. In *Machine Learning and Knowledge Discovery in Databases: Applied Data Science Track: European Conference (ECML PKDD)*, 2021.

[20] Jakub Czyz, Mark Allman, Jing Zhang, Scott Iekel-Johnson, Eric Osterweil, and Michael Bailey. Measuring ipv6 adoption. *SIGCOMM Comput. Commun. Rev.*, 2014.

[21] Jakub Czyz, Matthew Luckie, Mark Allman, Michael Bailey, et al. Don't forget to lock the back door! a characterization of ipv6 network security policy. In *Network and Distributed Systems Security (NDSS)*, 2016.

[22] Zakir Durumeric, David Adrian, Ariana Mirian, Michael Bailey, and J. Alex Halderman. A search engine backed by internet-wide scanning. In *ACM Conference on Computer and Communications Security (CCS)*, 2015.

[23] Zakir Durumeric, David Adrian, Ariana Mirian, James Kasten, Elie Bursztein, Nicolas Lidzborski, Kurt Thomas, Vijay Eranti, Michael Bailey, and J Alex Halderman. Neither snow nor rain nor mitm... an empirical analysis of email delivery security. In *Internet Measurement Conference (IMC)*, 2015.

[24] Zakir Durumeric, Frank Li, James Kasten, Johanna Amann, Jethro Beekman, Mathias Payer, Nicolas Weaver, David Adrian, Vern Paxson, Michael Bailey, and J. Alex Halderman. The matter of heartbleed. In *Internet Measurement Conf.*, 2014.

[25] Zakir Durumeric, Eric Wustrow, and J Alex Halderman. Zmap: Fast internet-wide scanning and its security applications. In *USENIX Security Symposium (USENIX Security)*, 2013.

[26] Brown Farinholt, Mohammad Rezaeirad, Paul Pearce, Hitesh Dharmdasani, Haikuo Yin, Stevens Le Blond, Damon McCoy, and Kirill Levchenko. To catch a ratter: Monitoring the behavior of amateur darkcomet rat operators in the wild. In *IEEE symposium on security and privacy (SP)*, 2017.

[27] 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 (PAM)*, 2017.

[28] National Commission for the Protection of Human Subjects of Biomedical and Behavioral Research. The belmont report - ethical principles and guidelines for the protection of human subjects of research, 1979.

[29] Pawel Foremski, David Plonka, and Arthur Berger. Entropy/ip: Uncovering structure in ipv6 addresses. 2016.

[30] Oliver Gasser, Quirin Scheitle, Pawel Foremski, Qasim Lone, Maciej Korczyński, Stephen D Strowes, Luuk Hendriks, and Georg Carle. Clusters in the expanse: Understanding and unbiasing ipv6 hitlists. In *Internet Measurement Conf.*, 2018.

[31] 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.

[32] Petros Gigis, Matt Calder, Lefteris Manassakis, George Nomikos, Vasileios Kotronis, Xenofontas Dimitropoulos, Ethan Katz-Bassett, and Georgios Smaragdakis. Seven years in the life of hypergiants' off-nets. In *ACM SIGCOMM*, 2021.

[33] Fernando Gont, Suresh Krishnan, Dr. Thomas Narten, and Richard P. Draves. Temporary Address Extensions for Stateless Address Autoconfiguration in IPv6. RFC 8981, 2021.

[34] Google. Ipv6 statistics, 2023. https://www.google.com/intl/en/ipv6/statistics.html.

[35] L. I. Guo, H. E. Lin, SONG Guanglei, WANG Zhiliang, YANG Jiahai, L. I. N. Jinlei, and G. A. O. Hao. IPv6 active address discovery algorithm based on multi-level classification and space modeling. *Journal of Tsinghua University*, 2021.

[36] Josef Gustafsson, Gustaf Overier, Martin Arlitt, and Niklas Carlsson. A first look at the ct landscape: Certificate transparency logs in practice. In *Passive and Active Measure.*, 2017.

[37] R. Hinden and S. Deering. IP Version 6 Addressing Architecture. RFC 4291, 2006.

[38] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997.

[39] Bingnan Hou, Zhiping Cai, Kui Wu, Jinshu Su, and Yinqiao Xiong. 6Hit: A Reinforcement Learning-based Approach to Target Generation for Internet-wide IPv6 Scanning. In *International Conf. on Comp. Comms. (ICCC)*, 2021.

[40] Amanda Hsu, Frank Li, and Paul Pearce. Fiat lux: Illuminating ipv6 apportionment with different datasets. In *ACM SIGMETRICS*, 2023.

[41] Liz Izhikevich, Gautam Akiwate, Briana Berger, Spencer Drakontaidis, Anna Ascheman, Paul Pearce, David Adrian, and Zakir Durumeric. Zdns: A fast dns toolkit for internet measurement. In *Internet Measurement Conference*, 2022.

[42] Lukas Krämer, Johannes Krupp, Daisuke Makita, Tomomi Nishizoe, Takashi Koide, Katsunari Yoshioka, and Christian Rossow. Amppot: Monitoring and defending against amplification ddos attacks. In *RAID*, 2015.

[43] Frank Li, Zakir Durumeric, Jakub Czyz, Mohammad Karami, Michael Bailey, Damon McCoy, Stefan Savage, and Vern Paxson. You've got vulnerability: Exploring effective vulnerability notifications. In *USENIX Security Symposium*, 2016.

[44] Xiang Li, Baojun Liu, Xiaofeng Zheng, Haixin Duan, Qi Li, and Youjun Huang. Fast ipv6 network periphery discovery and security implications. In *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2021.

[45] Mengfan Liu, Yi Guo, Liancheng Zhang, and Ming Hu. A Survey on Methodologies and Techniques for IPv6 Network Alias Resolution. In *IEEE ICCC*, 2021.

[46] Zhizhu Liu, Yinqiao Xiong, Xin Liu, Wei Xie, and Peidong Zhu. 6Tree: Efficient dynamic discovery of active addresses in the IPv6 address space. *Computer Networks*, 2019.

[47] Gordon Fyodor Lyon. *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*. Insecure, 2009.

[48] The majestic million, 2023. https://majestic.com/reports/majestic-million.

[49] Mitre. Cve report. https://cve.report/.

[50] Austin Murdock, Frank Li, Paul Bramsen, Zakir Durumeric, and Vern Paxson. Target generation for internet-wide ipv6 scanning. In *Internet Measurement Conference (IMC)*, 2017.

[51] Laurent Durnez Nicolas Crocfer. Opencve. vulnerability database. https://www.opencve.io/welcome.

[52] Ramakrishna Padmanabhan, John P. Rula, Philipp Richter, Stephen D. Strowes, and Alberto Dainotti. Dynamips: Analyzing address assignment practices in ipv4 and ipv6. In *Emerging Networking EXperiments and Technologies (coNEXT)*, 2020.

[53] Paul Pearce, Roya Ensafi, Frank Li, Nick Feamster, and Vern Paxson. Augur: Internet-wide detection of connectivity disruptions. In *IEEE Symposium on Security & Privacy*, 2017.

[54] Paul Pearce, Ben Jones, Frank Li, Roya Ensafi, Nick Feamster, Nick Weaver, and Vern Paxson. Global measurement of DNS manipulation. In *USENIX Security Symposium*, 2017.

[55] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. Tranco: A research-oriented top sites ranking hardened against manipulation. *Network and Dist. System Security Symp.*, 2018.

[56] Rapid7 forward dns, 2023. https://opendata.rapid7.com/sonar.fdns_v2/.

[57] RIPE. Best current operational practice for operators: Ipv6 prefix assignment for end-users - persistent vs non-persistent, and what size to choose, 2023. https://www.ripe.net/publications/docs/ripe-690.

[58] Erik Rye, Robert Beverly, and K C Claffy. Follow the scent: Defeating ipv6 prefix rotation privacy. In *ACM Internet Measurement Conference (IMC)*, 2021.

[59] Guanglei Song, Lin He, Zhiliang Wang, Jiahai Yang, Tao Jin, Jieling Liu, and Guo Li. Towards the Construction of Global IPv6 Hitlist and Efficient Probing of IPv6 Address Space. In *International Symposium on Quality of Service (IWQoS)*, 2020.

[60] Guanglei Song, Jiahai Yang, Lin He, Zhiliang Wang, Guo Li, Chenxin Duan, Yaozhong Liu, and Zhongxiang Sun. Addrminer: A comprehensive global active ipv6 address discovery system. In *USENIX Annual Technical Conference (ATC)*, 2022.

[61] Guanglei Song, Jiahai Yang, Zhiliang Wang, Lin He, Jinlei Lin, Long Pan, Chenxin Duan, and Xiaowen Quan. DET: Enabling Efficient Probing of IPv6 Active Addresses. *IEEE/ACM Transactions on Networking*, 2022.

[62] Jonathan Sorg, Satinder Singh, and Richard L. Lewis. Variance-based rewards for approximate bayesian reinforcement learning. In *Uncertainty in Artificial Intelligence (UAI)*, 2010.

[63] G. Huston T. Narten and L. Roberts. IPv6 Address Assignment to End Sites. RFC 6177, 2011.

[64] Johanna Ullrich, Peter Kieseberg, Katharina Krombholz, and Edgar Weippl. On Reconnaissance with IPv6: A Pattern-Based Scanning Approach. In *International Conference on Availability, Reliability and Security*, 2015.

[65] Cisco umbrella popularity list, 2023. http://s3-us-west-1.amazonaws.com/umbrella-static/index.html.

[66] Jeroen Van Der Ham. Ethics and internet measurements. In *IEEE Security and Privacy Workshops (SPW)*, 2017.

[67] Marco A Wiering and Martijn Van Otterlo. Reinforcement learning. *Adaptation, learning, and optimization*, 2012.

[68] Eric Wustrow, Colleen M Swanson, and J Alex Halderman. Tapdance: End-to-middle anticensorship without flow blocking. In *USENIX Security Symposium (USENIX Security)*, 2014.

[69] Qinge Xie, Shujun Tang, Xiaofeng Zheng, Qingran Lin, Baojun Liu, Haixin Duan, and Frank Li. Building an open, robust, and stable Voting-Based domain top list. In *USENIX Security Symposium (USENIX Security)*, 2022.

[70] Tao Yang, Zhiping Cai, Bingnan Hou, and Tongqing Zhou. 6forest: An ensemble learning-based approach to target generation for internet-wide ipv6 scanning. In *INFOCOM*, 2022.

[71] Tao Yang, Bingnan Hou, Zhiping Cai, Kui Wu, Tongqing Zhou, and Chengyu Wang. 6graph: A graph-theoretic approach to address pattern mining for internet-wide ipv6 scanning. *Computer Networks*, 2022.

[72] Gang Zheng, Xinzhong Xu, and Chao Wang. An Effective Target Address Generation Method for IPv6 Address Scan. In *IEEE ICCC*, 2020.

[73] Johannes Zirngibl, Lion Steger, Patrick Sattler, Oliver Gasser, and Georg Carle. Rusty clusters? dusting an ipv6 research foundation. In *Internet Measurement Conference (IMC)*, 2022.

[74] ZMap. Zgrab. https://github.com/zmap/zgrab2.