Search in the Expanse: Towards Active and Global IPv6 Hitlists

Bingnan Hou*, Zhiping Cai*, Kui Wu†, Tao Yang* and Tongqing Zhou*
*College of Computer, National University of Defense Technology, China
†Department of Computer Science, University of Victoria, Canada
E-mails: {*houbingnan19, zpcai, yangtao97, zhoutongqing*}*@nudt.edu.cn, wkui@uvic.ca*

IEEE INFOCOM 2023 - IEEE Conference on Computer Communications | 979-8-3503-3414-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/INFOCOM53939.2023.10229089

***Abstract*—Global-scale IPv6 scan, critical for network measurement and management, is still a mission to be accomplished due to its vast address space. To tackle this challenge, IPv6 scan generally leverages pre-defined seed addresses to guide search directions. Under this general principle, however, the core problem of effectively using the seeds is largely open. In this work, we propose a novel IPv6 active search strategy, namely HMap6, which significantly improves the use of seeds, w.r.t. the marginal benefit, for large-scale active address discovery in various prefixes. Using a heuristic search strategy for efficient seed collection and alias prefix detection under a wide range of BGP prefixes, HMap6 can greatly expand the scan coverage. Real-world experiments over the Internet in billion-scale scans show that HMap6 can discover 29.39M unique /80 prefixes with active addresses, an 11.88% improvement over the state-of-the-art methods. Furthermore, the IPv6 hitlists from HMap6 include all-responsive IPv6 addresses with rich information. This result sharply differs from existing public IPv6 hitlists, which contain non-responsive and filtered addresses, and pushes the IPv6 hitlists from quantity to quality. To encourage and benefit further IPv6 measurement studies, we released our tool along with our IPv6 hitlists and the detected alias prefixes.**

***Index Terms*—IPv6, Internet-wide scan, IPv6 hitlists.**

I. Introduction

Internet-wide scan is an essential building block in network measurement. In the past, asynchronous scan tools like ZMap [1] and Masscan [2] have drastically enhanced our capability of conducting Internet-wide network surveys. Nevertheless, these tools are not effective when applied for IPv6 because the huge address space of IPv6 renders a comprehensive scan nearly impossible. Therefore, IPv6 Internet-wide measurement resorts to lists of target IP addresses, so-called hitlists, which serve as a representative subset of the IPv6 address space [3].

The existing published hitlists promote the research of efficient IPv6 scan strategy since the common belief is that the addresses (also called seeds) from the hitlists can provide useful information in the search for active IPv6 addresses. Specifically, previous works usually used a subset of the IPv6 hitlists (e.g., Gasser's IPv6 hitlists [3]) to generate target addresses and utilized existing high-speed scanners (e.g., ZMapv6 [1]) to probe these targets.

Though prior works have improved the discovery rate of active IPv6 addresses, they used a small probing budget[1] or a biased seedset because their target generation algorithms are too complex to handle a large-scale budget or seedset in normal-capacity servers. For example, studies of [4] and [5] only used a seedset of less than 100K addresses with a budget of less than 10M probes. The above fact results in two pitfalls in existing works: (1) Even with a high discovery rate, a small budget can only return a small number of discovered active addresses. (2) The target addresses generated from a biased seedset will have limited coverage since the seeds largely influence the search directions. Consequently, the problem of obtaining large-scale and global-wide active IPv6 addresses still remains largely open. We are thus motivated to develop a simple yet effective active scan strategy for large-scale and global-wide active IPv6 address discovery.

In this work, we propose a novel IPv6 active search strategy, called HMap6, to achieve efficient large-scale scans. Compared with the state-of-the-art approaches, HMap6 greatly improves the utilization of seed address for target address generation from the perspective of marginal benefit. To be more specific, HMap6 (1) combines a heuristic method using BGP prefixes for seed address collection and (2) performs alias prefix detection. The first feature can effectively increase the number and the coverage of the scans, and the second feature can significantly improve probing efficiency since alias prefixes are very common in IPv6 network [3]. With alias prefix detection, HMap6 only scans the addresses under the non-aliased prefixes. Note that scanning alias prefixes would waste a considerable amount of probe resources as these addresses may be assigned to the same physical device or CDN.

In summary, HMap6 is a high-efficiency IPv6 prober written in C++ and is portable to any UNIX-like platform. We released the source code of HMap6 along with our hitlists (i.e., discovered active addresses) and detected alias prefixes at https://hbn1987.github.io/HMap6.github.io. Our main contributions include:

- We develop HMap6, a novel active address search strategy for efficient global-wide IPv6 scan. HMap6 obtains the initial seed addresses in various ways, i.e., integrating public seedsets with heuristically collected seeds under the announced BGP prefixes. Equipped with the seed addresses, it can quickly generate large-scale, likely responsive active addresses in various prefixes for subsequent scans.
- HMap6 includes a novel heuristic method for active

[1]Probing budget is defined as the number of allowed probing attempts.

TABLE I
COMPARING HMAP6 WITH FIVE PREVIOUS WORKS.

Scan strategy	Generation algorithm	Time complexity	Seed number	Budget used
6Gen [6]	AHC	$O(n^3)$	100K	1M
6Tree [7]	DHC	$O(nlogn)$	400K	1500M
6Forest [8]	DHC+EL	$\Omega(nlogn)$	1400K	500M
6Hit [4]	DHC+RL	$\Omega(nlogn)$	100K	10M
6Gan [5]	GANs	$-^*$	50K	0.05M
HMap6	BHC	$O(nlogn)$	**5689K**	**7808M**

*The time complexity of GANs model is difficult to estimate.

address (seed) discovery, which combines a simple yet efficient de-aliasing algorithm in the active search process to get more valuable seeds with various prefixes. Experimental results show that our heuristic seed discovery method can find seeds in various prefixes quite different from the existing published open-source hitlists.
- HMap6 can be easily deployed on commodity computers or even laptops. Real-world experiments over IPv6 Internet on a commodity computer show that compared with the state-of-the-art solutions, HMap6 discovered 29.39M unique /80 prefixes with active addresses in billion-scale scans, which is an 11.88% improvement over the state-of-the-art solutions.
- We push the existing published IPv6 hitlists from quantity to quality. Our published hitlists contained $12.64\times$ more /80 prefixes with all-responsive addresses than Gasser's IPv6 hitlists, along with a large number of newly-found alias prefixes.

II. RELATED WORK

Target generation algorithms mainly focus on utilizing seeds to generate address targets to scan. They assume that the address space with high-density seeds is more likely to have undiscovered active addresses. 6Gen [6] uses the agglomerative hierarchical clustering (AHC) algorithm to expand each seed as a cluster center to generate the target addresses by maintaining the maximal seed density and the minimal scale. 6Tree [7] utilizes the divisive hierarchical clustering (DHC) algorithm to form a space tree from seeds' structure to divide the IPv6 address space into nodes/subspaces, and it generates target addresses based on the seed density of each node. 6Forest [8] also uses the DHC algorithm to generate the scan subspaces. The difference is that it combines an ensemble learning (EL) method to remove the outlier addresses in the seeds, i.e., addresses with different addressing patterns in the seed cluster. 6Hit [4] introduces a dynamic adjustment strategy into the scan process using a reinforcement learning (RL) algorithm. Yet, the way that the preordering scan results guide the subsequent search directions destroys the asynchronism of scanning and thus reduces the probing speed. 6Gan [5] generates target addresses with generative adversarial nets (GANs), which utilize multiple generators to learn IPv6 addressing patterns determined by the seeds.

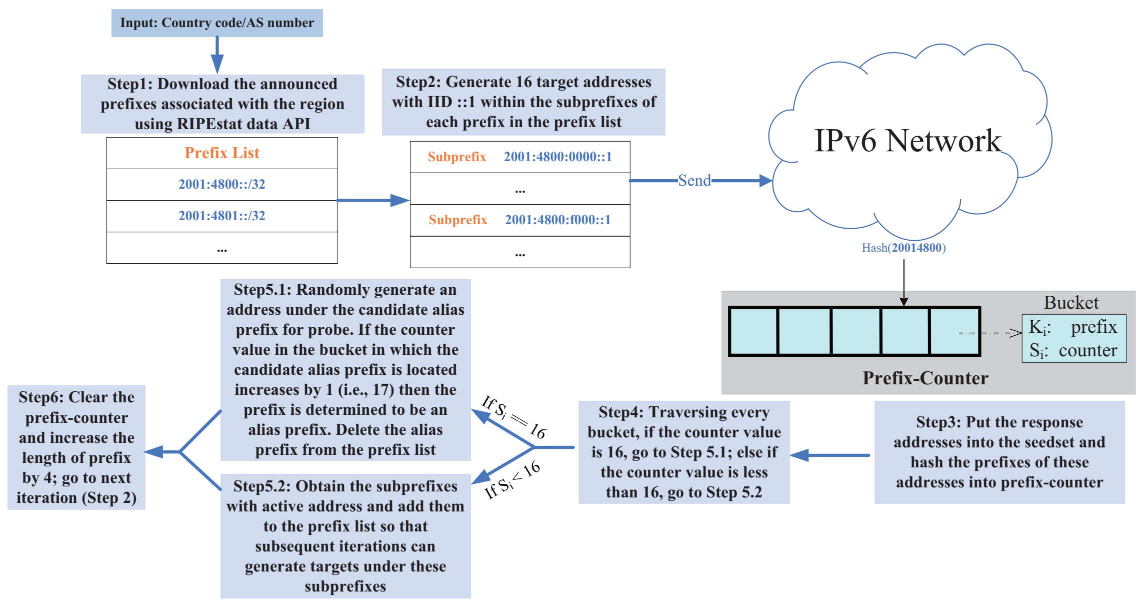
We survey the previous works on scan practicability and compare them with HMap6 in Table I. The *seed number* and the *budget used* in the table indicate the maximum number of seeds and the maximum budget in a scan in these previous works, respectively. We can see that 6Gen and 6Gan used a small seedset (i.e., less than 100K seeds) because their algorithm complexity is too high to handle large-scale seedsets, especially on regular-capacity servers. 6Hit used a small-budget (i.e., 10M probes) scan since its dynamic search strategy renders a complete asynchronous scan slow. Though 6Tree and 6Forest can perform a relatively large-scale scan, the DHC-based target generation strategy performs poorly when the budget increases. This is because the DHC algorithm, which only uses downward splitting to generate the scan subspaces, does not fully utilize the seeds' information.

In this work, HMap6 utilizes bidirectional hierarchical clustering (BHC) that improves and integrates AHC and DHC to make full use of the seeds to generate more promising scan subspaces. HMap6 has low algorithm complexity (i.e., $O(nlogn)$ where n indicates the seed number) to process large seedsets, and its static scan strategy can easily carry out large-scale budget scans.

Alias prefix detection is the process of determining if IP addresses are assigned to the same physical device or CDN. Aliased prefixes, namely, prefixes under which each possible IP address replies to queries, are very common in IPv6. Through the IP_FREEBIND option in Linux, this feature can be easily deployed in CDNs [3]. Note that this feature severely impacts dynamic target generation algorithms (e.g., [4]), as they will get more feedback when scanning the aliased areas, making the following-on scans fall into the "alias trap".

Gasser et al. [3] proposed APD, a prefix detection approach especially suitable for large-scale alias prefix resolution. APD is based on the observation originally from [6] that a randomly-selected IP address in the vast IPv6 space is unlikely to respond. So it generates 16 random addresses under the prefix, and if all of these addresses respond, then the prefix is judged to be an alias prefix. By utilizing APD, Gasser published a wide-range alias prefix list, which has been regarded as the ground truth by several previous works [5], [9]. In this work, we used a variant of the APD algorithm to find a large number of new alias prefixes under the announced BGP prefixes during seed address collection.

IPv6 Hitlists catalog IPv6 addresses and alias prefixes on the Internet. A hitlist is commonly used as the seed addresses for the target generation algorithms, and an alias list is usually used to indicate the prefixes that should not be over-scanned. The largest public hitlist and alias list from Gasser et al. [3] collect a wide range and a large number of IPv6 addresses and alias prefixes from multiple sources (e.g., crowdsourcing platform [10], Bitnodes API [11], RIPE IPMap [12], Scamper [13]). However, a large proportion of addresses in the hitlist are unresponsive due to various reasons such as passive collection and obsolescence. Compared with the existing published hitlist, the hitlists collected by active probing using HMap6 found more widespread and all-responsive addresses.



Input: Country code/AS number
Step1: Download the announced prefixes associated with the region using RIPEstat data API
Prefix List
2001:4800::/32
2001:4801::/32
...
Step2: Generate 16 target addresses with IID ::1 within the subprefixes of each prefix in the prefix list
Subprefix 2001:4800:0000::1
...
Subprefix 2001:4800:f000::1
...
Send
IPv6 Network
Hash(20014800)
Bucket
K_i: prefix
S_i: counter
Prefix-Counter
Step3: Put the response addresses into the seedset and hash the prefixes of these addresses into prefix-counter
Step4: Traversing every bucket, if the counter value is 16, go to Step 5.1; else if the counter value is less than 16, go to Step 5.2
If S_i = 16
If S_i < 16
Step5.1: Randomly generate an address under the candidate alias prefix for probe. If the counter value in the bucket in which the candidate alias prefix is located increases by 1 (i.e., 17) then the prefix is determined to be an alias prefix. Delete the alias prefix from the prefix list
Step5.2: Obtain the subprefixes with active address and add them to the prefix list so that subsequent iterations can generate targets under these subprefixes
Step6: Clear the prefix-counter and increase the length of prefix by 4; go to next iteration (Step 2)

Fig. 1. Steps of heuristic seed collection.

III. The Details of HMap6

The high-level ideas of HMap6 include **1) collecting diverse seed addresses**, by heuristically probing the announced BGP prefixes to obtain active seed addresses distributed under various prefixes; **2) generating promising subspaces**, by utilizing the structure information of the seeds to form the subspaces for asynchronous scan; **3) avoiding alias scan**, by blacklisting public and our detected alias prefixes to avoid scanning these alias prefixes.

A. Collection of Seed Addresses

The target generation algorithms for IPv6 scan require the input of the seed addresses to specify the direction of the scan, so the seeds play a critical role in active search. Previous studies have found a positive correlation between the number of seeds and the number of newly-discovered active hosts [6], [7]. To this end, we explore an efficient seed address collection method in various BGP prefixes to complement the existing public IPv6 hitlists.

Fig. 1 shows the steps of our seed collection method:

- **Step 1**: HMap6 uses the RIPEstat data API [14] to form the prefix list, which consists of prefixes associated with a country/region or AS.
- **Step 2**: HMap6 sends 16 packets to the addresses with interface ID (IID) ::1 within the subprefixes of each announced prefix. For each packet, it enforces the traversal of a subprefix with a different nybble. For example, given the announced prefix 2001:4800::/32, we generate one pseudo-random address for each 4-bit subprefix with IID ::1, namely 2001:4800:[0-f]000::1. The reason for selecting addresses with IID ::1 is that addresses with small IID often occupy the largest proportion in the collected addresses [15]. In other words, they are more likely to be active.
- **Step 3**: HMap6 puts the response addresses into the seedset and hashes the prefixes of these addresses into a prefix counter. HMap6 uses the prefix counter in the receiving thread to count the number of active addresses under the prefix. The prefix counter consists of buckets that are dynamically allocated, and there are two fields in each bucket: one to store the prefix (i.e., K_i field with initial value none) and the other for counting (i.e., S_i field with initial value 0). When a reply arrives, HMap6 extracts its prefix and hashes it into the prefix counter. It stores the prefix in K_i and increases the value in S_i by one where i indicates the index of the prefix in the prefix counter.
- **Step 4**: After hashing all arrived replies into the prefix-counter, HMap6 traverses every bucket to check the value in S_i.
- **Step 5:** De-alias.
  - (5.1) If S_i equals 16, then K_i is regarded as a candidate alias prefix. This is because the target under each subprefix within the prefix has a response, which is consistent with the alias prefix detection condition of the APD [3] algorithm. The main difference is that APD generates a random address under each subprefix, while HMap6 generates the address with IID ::1 under each subprefix. However, an address with IID ::1 is more likely to be active than a randomly generated address [15]. To this end, we add a qualification filter to determine the alias prefix. We randomly generate an address under the candidate alias prefix. Only when this address is reachable (i.e., S_i equals 17), the candidate alias prefix is considered as an alias prefix. Then, we delete the alias prefix from the prefix list to avoid

further probing the alias addresses.

  – (5.2) If S_i is less than 16, then HMap6 obtains the subprefixes with active addresses and adds them to the prefix list so that subsequent iterations can generate targets under these active and non-alias prefixes.

- **Step 6:** HMap6 clears the prefix-counter, increases the length of the prefix by 4 (i.e., one nybble) and goes to **Step 2** for the next iteration.

The above iteration terminates when the prefix length exceeds 80. Using the above heuristic search strategy and de-alias algorithm, HMap6 ensures that (1) probes are distributed evenly over more specific subprefixes to allow more balanced addresses sampling in one region; (2) the de-alias algorithm effectively avoids the waste of probing resources caused by excessive scan of the alias prefixes; and (3) the asynchronous design (i.e., sending and receiving packets on separate threads) and the use of prefix-counter greatly improve the efficiency of seed collection and alias resolution.

B. Subspace Generation

An IPv6 address can be represented as a hexadecimal string with 32 nybbles. Alternatively, we can consider an IPv6 address as a vector in a 32-dimensional space (i.e., address space). The value in each dimension of the vector is an integer in $[0, 15]$ (i.e., the value range of a nybble). Following the convention, we use the wildcard symbol "*" to denote a nybble which can be any value in $[0, 15]$ (i.e., a variable dimension). Target generation algorithms partition the address space into regions/clusters/subspaces for the probers to scan. In the following, we use regions, clusters, and subspaces interchangeably.

Existing solutions [4], [6], [7] adopt a unidirectional hierarchical clustering algorithm to group seeds into a hierarchical "tree" of clusters, each cluster representing a region in the address space. This clustering algorithm can use either agglomerative or divisive clustering, depending on whether the hierarchical decomposition is formed in a bottom-up (merging) or top-down (splitting) approach.

As shown in Fig. 2, the agglomerative hierarchical clustering (AHC) algorithm starts by instantiating with a cluster for each seed ($c_i, i \in [1, 5]$). Each iteration calculates the vector distance between clusters and merges two clusters with the minimum distance. In contrast, the divisive hierarchical clustering (DHC) algorithm adopts a top-down clustering approach. It starts with a single cluster containing all seeds. After each iteration, it splits the clusters from the left-most variable dimension (i.e., dimension with a value of "*"). It stops when each seed is in its singleton cluster or the cluster can no longer split (i.e. only one variable dimension left).

HMap6 adopts both AHC and DHC algorithms to determine the regions since no clear evidence indicates which algorithm is better. In addition, HMap6 reduces the algorithm's complexity compared with the alternatives. For example, 6Gen [6] uses a more complex AHC algorithm that does not explicitly merge similar clusters. Instead, each cluster grows independently, and it allows a seed address to belong to multiple clusters. This AHC algorithm makes 6Gen accurately calculate the number of seeds in the clusters but increases the time complexity to $O(n^3)$, where n denotes the number of seeds. The DHC used by 6Tree [7] splits the clusters to only contain a single variable dimension. This increases the number of splits and the computation overhead compared to HMap6, which splits the clusters into subclusters with a specified variable dimension (i.e., $\mathcal{D}$, where $\mathcal{D} \geq 1$).

The pseudocode of scan subspace generation is shown in Algorithm 1. HMap6 uses the improved AHC and DHC algorithms to generate the subspaces (Lines 1-2). In contrast to the alternative that terminates target generation with a pre-specified budget, HMap6 terminates target generation with a pre-specified size of subspace $\mathcal{D}$, that is, target generation stops when seed addresses can no longer form a subspace with variable dimension $\mathcal{D}$ through hierarchical clustering. For example, the yellow nodes in Fig. 2 are all scan subspaces generated when $\mathcal{D}$ is 2. Instead of using a pre-defined budget to terminate target generation, HMap6 uses $\mathcal{D}$ to indicate the size of the scan range expanded by the seed. Note that the number of clusters reflects the effective scan direction provided by the seeds when $\mathcal{D}$ is determined.

Algorithm 1 Scan Subspace Generation

```
Input: The seedset $\mathcal{C}$, the specified variable dimension $\mathcal{D}$, and
    the alias prefix list $A$;
Output: The target subspaces $S$;
 1: $S = AHC(\mathcal{C}, \mathcal{D})$;
 2: $S.Append(DHC(\mathcal{C}, \mathcal{D}))$;
 3: for $s \in S$ do
 4:     if $s \in A$ then
 5:         $S.Remove(s)$;
 6:     end if
 7: end for
 8: function $AHC(\mathcal{C}, \mathcal{D})$        ▷ Compute the minimum
    distance between seeds and clusters/subspaces, merge the
    seeds into the nearest clusters, and update the scope of the
    clusters so that they can cover the newly-added seeds.
 9: end function
10: function $DHC(\mathcal{C}, \mathcal{D})$        ▷ Start at
    the root cluster/subspace covering all seeds. Then, split the
    root cluster into subclusters, starting with the first variable
    dimension on the left, and continue until the number of
    variable dimensions is reduced to $\mathcal{D}$.
11: end function
```

C. Avoiding Alias Scan

Scanning alias prefixes or honeypots is a vast waste of probing resources. In particular, dynamic search strategies (e.g., 6Hit) tend to overscan alias prefixes or honeypots because they will get more replies/rewards from these areas. HMap6 uses three mechanisms to avoid overly scanning alias prefixes or honeypots. Firstly, HMap6 does not select seeds under the known alias prefixes to reduce the probability of generating

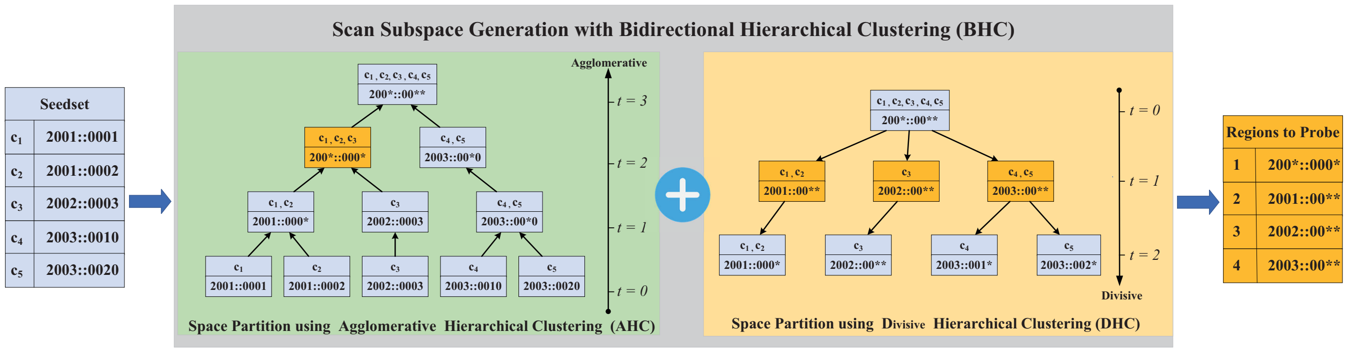


Fig. 2. A toy example of scan subspace generation using hierarchical clustering. Only the generated subspaces with a variable dimension of 2 (i.e., $\mathcal{D} = 2$) are specified as output in the legend.

alias subspaces for scanning. Secondly, HMap6 expands the detection range of alias prefixes by referring to the APD algorithm in heuristic seed collection. Thirdly, HMap6 deletes the subspaces under the alias prefixes to avoid scanning the known alias prefixes (Lines 3-7 in Algorithm 1).

In addition, to make the scan cover as many different prefixes as possible on a limited budget, HMap6 does not scan the low-order addresses under a prefix. Specifically, HMap6 does not scan the subspaces with all the variable dimensions under the /80 prefix (e.g., subspace of 2001::0***) such that the HMap6's scan can form a more comprehensive and unbiasing IPv6 hitlist.

In the probing process, HMap6 leverages the idea from Yarrpv6 [16] and ZMap [1] to conduct asynchronous scan. Meanwhile, HMap6 uses the symmetric RC5 block cipher with a 32-bit block size [17] to randomize the sequence of the target addresses, which splits the probing load and minimizes the impact of rate-limiting.

IV. Performance Evaluation

In this section, we first evaluate our heuristic seed collection method by analyzing the differences between the collected seeds and alias prefixes with Gasser's hitlist and alias prefix list, respectively. Then, we compare the performance of HMap6's search strategy with representative alternatives in terms of discovery rate and time overhead. All experiments were carried out in a real IPv6 network with 200 Mbps bandwidth. We did Internet-wide scan with a Linux machine (CPU: Intel Xeon Platinum 8369 @ 3.3GHz, Memory: 16GB).

A. Results of Heuristic Seed Collection

We utilize HMap6's heuristic search strategy for seed collection using the ICMPv6 probe, which is considered less obtrusive than UDP or TCP probes. Our country/region-level seed collection covers 231 countries/regions that have announced BGP prefixes. We compare our seed collection results (i.e., H_{ICMPv6}) with the ICMPv6 responsive addresses in Gasser's hitlist (i.e., G_{ICMPv6}) published on January 15th, 2022, as shown in Table II. Note that our heuristic seed collection sets a termination condition, i.e., the length of the announced prefixes and the added prefixes with responsive addresses

TABLE II
Comparison between our heuristic seed collection results with Gasser's hitlist using ICMPv6 probe.

Country	Seedset	Seed number	Ratio	Involved ASes	Budget cost
India	H_{ICMPv6}	7.92K	0.23%	216	1.67M
	G_{ICMPv6}	3.31K	0.15%	241	-
America	H_{ICMPv6}	1372.40K	39.07%	1401	7.69M
	G_{ICMPv6}	135.93K	6.21%	2386	-
China	H_{ICMPv6}	12.48K	0.36%	115	4.28M
	G_{ICMPv6}	1439.55K	65.74%	323	-
Brazil	H_{ICMPv6}	33.72K	0.96%	3278	6.87M
	G_{ICMPv6}	14.81K	0.68%	2907	-
Japan	H_{ICMPv6}	1.98K	0.06%	160	0.58M
	G_{ICMPv6}	22.99K	1.05%	250	-
Germany	H_{ICMPv6}	8.81K	0.25%	571	2.51M
	G_{ICMPv6}	196.89K	8.99%	978	-
Mexico	H_{ICMPv6}	1.06K	0.03%	56	0.24M
	G_{ICMPv6}	2.25K	0.10%	67	-
United Kingdom	H_{ICMPv6}	3.58K	0.10%	309	6.31M
	G_{ICMPv6}	14.84K	0.68%	524	-
Vietnam	H_{ICMPv6}	65.88K	1.88%	54	7.44M
	G_{ICMPv6}	1.68K	0.08%	61	-
France	H_{ICMPv6}	7.46K	0.21%	187	4.03M
	G_{ICMPv6}	145.56K	6.65%	381	-
Others	H_{ICMPv6}	1997.10K	56.68%	3915	50.58M
	G_{ICMPv6}	211.92K	9.68%	6379	-
Total	H_{ICMPv6}	3.51M	100%	10134	92.21M
	G_{ICMPv6}	2.19M	100%	14091	-

iterates to 80. For example, our simple heuristic search strategy only spent a budget of 0.24M to reach the end condition of the maximum prefix length (i.e., 80) in Mexico. We list the top 10 countries with the largest IPv6 users (according to the IPv6 country deployment report [18]) in detail, while "Others" and "Total" indicate the rest of the countries/regions that are not listed in the table and the total country-by-country scan results, respectively. To demonstrate the distribution of collected seeds, we list the number of involved ASes using CAIDA's IP to AS dataset [19].

Although our heuristic seed collection strategy uses a small probing budget (i.e., 92.21M in total), the collected seeds cover a wide range of prefixes and ASes as it searches widely across the announced BGP prefixes. Of note, the seed number also indicates the number of different /80 prefixes as we only select one address under a /80 prefix in our heuristic scan results

TABLE III
THE NUMBER OF DETECTED ALIAS PREFIXES COMPARED WITH GASSER'S ALIAS LIST IN VARIOUS PREFIX LENGTH.

Alias list	Prefix length $\in [28, 32]$	$\in (32, 64]$	$\in (64, 120]$
H_{alias}	85	2092	2588
G_{alias}	224	63891	658
$H_{alias} - G_{alias}$	68	1978	2588

and Gasser's hitlist. We can see that the number of ASes our heuristic scan covered is in the same order of magnitude as that covered by Gasser's hitlist. In particular, our heuristic strategy found more ASes in some countries (e.g., Brazil), and most of the seeds we collected in different countries did not appear on Gasser's hitlist. This result shows that our heuristic seed collection method efficiently finds many new addresses and prefixes that supplement Gasser's hitlist seeds.

We also identified many alias prefixes during seed collection. We verified these identified alias prefixes, and almost all of them meet the alias prefix determination conditions of APD [3], indicating that our alias prefix resolution method with active search is also effective. Table III lists the number of alias prefixes we collected (i.e., H_{alias}) compared to Gasser's alias prefix list (i.e., G_{alias}) with prefix length between 28 and 120. Note that $H_{alias} - G_{alias}$ indicates our newly-discovered alias prefixes, which are not listed in Gasser's alias list. We can see that our heuristic search strategy found 4634 newly-discovered alias prefixes ranging from /28 to /120, including 105 large ones with a prefix length of 28 and 32, consuming a total budget of just 92.21M ICMPv6 probes.

In summary, HMap6 has found many new alias prefixes and has collected a seedset that has a much broader coverage than Gasser's seedset.

B. Active Discovery Performance

We compare the performance of HMap6 and the state-of-the-art target generation methods. We performed Internet-wide scan at the maximum probing rate of 100Kpps (probes per second). This is because (1) our prober can easily achieve this speed (HMap6 can achieve a maximum of 163.66 Kpps probing speed with a ≈34.67% CPU utilization in our commodity computer), and (2) our network administrator did not permit us to use a higher probing rate. Note that this probing rate was also used in the Yarrp [20] and FlashRoute [21] measurement studies.

Seedset. The seedset consists of seeds collected by our heuristic approach and responsive addresses in Gasser's hitlists published on January 15, 2022. The characteristics of the seedsets are shown in Table IV, where S_{ICMPv6} indicates the seedset formed by all the responsive addresses with the specified protocol, e.g., $S_{ICMPv6} = H_{ICMPv6} \cup G_{ICMPv6}$. We can see that our seedsets are quite big and widely distributed across countries/regions.

Metrics. We define three evaluation metrics to compare the performance of HMap6 with the representative alternatives. The ***discovery number*** refers to the number of de-aliased

TABLE IV
CHARACTERISTICS OF THE SEEDSETS.

Seedset	# Seed	# Country/region	# AS
S_{ICMPv6}	5.69M	200	16435
$S_{UDP/53}$	2.58M	197	13476
$S_{TCP\_ACK/80}$	0.43M	148	4638

newly-discovered unique /80 prefixes. Specifically, all replies exclude the addresses listed in the seedset, as well as networks and addresses in the publicly curated list of aliases from Gasser et al. [3]. The ***discovery rate*** indicates the proportion of de-aliased newly-discovered active addresses/prefixes in the total budget. The term ***marginal benefit*** refers to the incremental newly-discovered addresses/prefixes with additional probing budgets.

Advantage of HMap6's subspace generation method. We show the number of subspaces with various variable dimensions generated by each scan subspace generation (SGA) algorithm, as shown in Table V. Note that the number of dimensions indicates the number of variable dimensions of the generated subspace. The more variable dimensions of the subspace, the larger the subspace. For example, the number of dimensions of subspace 2001::00** is 2, and scanning this subspace will consume 256 probing budgets. It can be seen that the number of subspaces of each dimension generated by the DHC algorithm is the smallest due to its top-down hierarchical splitting method, in which the root node/space contains all the seeds and then the space splits along the direction towards the leaf nodes. This approach tends to wrap as many seeds as possible in fewer generated subspaces. This explains why DHC performs well on a small budget since the generated subspaces can be packed with more seeds on a small budget. Recall that the higher the seed density, the greater the probability that the subspace contains other active addresses in density-driving target generation algorithms.

The advantages of AHC over DHC are prominent when the scan budget is large. For example, when using seedset S_{ICMPv6}, the AHC algorithm generates 110.64K 4-dimensional subspaces, which will consume a 7.25B probing budget, while the DHC algorithm only generates 17.16K 4-dimensional subspaces (which will consume a 1.18B budget). In other words, when the budget exceeds 1.18B, the DHC-based search strategies (e.g., 6Tree and 6Hit) will have to scan subspaces with dimensions greater than 4. The exponential growth of the subspace size will greatly reduce its seed density, leading to a sharp decline in the discovery rate.

The BHC algorithm used by HMap6 generates the most subspaces in each dimension because the subspace it generates is a union of subspaces generated by (modified) AHC and DHC algorithms. In other words, *HMap6's BHC algorithm achieves the maximum seed utilization, enabling it to have a better discovery rate regardless of the budget size.*

To validate the above analysis, we run billion-scale scans to compare the total number of newly discovered /80 prefixes with active addresses of each strategy, as shown in Table VI.

TABLE V
THE NUMBER OF SUBSPACES WITH VARIOUS DIMENSIONS GENERATED BY EACH SUBSPACE GENERATION ALGORITHM USING THE THREE SEEDSETS.

SGA \ # Dimensions	Generation with S_{ICMPv6} 1	2	3	4
AHC	475.44K	398.32K	180.93K	110.64K
DHC	33.15K	32.10K	26.19K	17.16K
BHC	477.11K	404.20K	195.56K	119.15K

SGA \ # Dimensions	Generation with $S_{UDP/53}$ 1	2	3	4
AHC	175.36K	137.66K	74.87K	55.91K
DHC	18.80K	15.44K	18.26K	12.19K
BHC	175.98K	142.22K	87.19K	62.72K

SGA \ # Dimensions	Generation with $S_{TCP\_ACK/80}$ 1	2	3	4
AHC	6.1K	10.60KK	15.72K	12.12K
DHC	0.49K	2.21K	3.35K	2.19K
BHC	6.62K	11.26K	16.62K	13.17K

TABLE VI
THE SCAN RESULTS OF EACH SEARCH STRATEGY.

Seedset	Total budget	Search strategy	Discovery number	Discovery rate
S_{ICMPv6}	7.81B	HMap6	25.47M	**0.33%**
		6Gen	22.76M	0.29%
		6Hit	15.08M	0.19%
		6Forest	12.98M	0.17%
		6Tree	12.91M	1.17%
$S_{UDP/53}$	4.11B	HMap6	8.35M	**0.20%**
		6Gen	7.89M	0.19%
		6Hit	6.24M	0.15%
		6Forest	4.11M	0.10%
		6Tree	4.19M	0.10%
$S_{TCP\_ACK/80}$	0.86B	HMap6	4.14M	**0.48%**
		6Gen	3.65M	0.42%
		6Hit	3.57M	0.41%
		6Forest	3.47M	0.40%
		6Tree	3.37M	0.39%
Total	12.78B	HMap6	29.39M	**0.23%**
		6Gen	26.27M	0.21%
		6Hit	17.40M	0.14%
		6Forest	14.98M	0.12%
		6Tree	14.90M	0.12%

The reason we choose this scale budget as the scan termination condition is that the marginal benefit of each strategy drops to <0.46 hit per 100 probes using the three seedsets. In other words, the utilization of seeds is almost complete on this scale budget (i.e., the budget of this volume makes sure that all strategies can scan all the generated subspaces with a variable dimension less than or equal to 4). We can see that HMap6 found the most number of addresses using each seedset. We summarized the scan results using the three seedsets of each search strategy and removed the duplicated addresses. HMap6 discovered the most number of addresses (i.e., 29.39M) and achieved the highest discovery rate (i.e., 0.23%), which is an 11.88% improvement on average over the state-of-the-art solutions.

Results of Discovery Performance. Fig. 3 shows the discovery rate and marginal benefit of HMap6 and the alternatives, using the three seedsets. We can see that the DHC algorithm (used by HMap6, 6Hit, 6Forest, and 6Tree) has an obvious advantage in discovery rate using a small budget (i.e., <50M). However, this advantage diminishes when the budget increases, as the subspaces generated by the AHC gradually cover all seeds. Meanwhile, 6Hit shows minor advantages compared to 6Tree and 6Forest on a small and medium scale budget, indicating that its adjustment of search direction improves the discovery rate performance. However, with the increase in budget, the advantages of the dynamic search strategy gradually disappear since the dynamic search strategy cannot provide more effective scan subspaces as the scan coverage increases. Under a large budget (e.g., >1B), the advantages of the AHC algorithm gradually appear as it can split more effective subspaces than the DHC algorithm. By modifying and integrating AHC and DHC, HMap6 achieves better performance on discovery rate than other alternatives in both small and large budget scans. This result further confirmed our previous analysis on the advantage of HMap6's subspace generation method.

It is worth noting that the marginal benefit decreases sharply with the increase in budget. For instance, the marginal benefit drops to <0.17 newly-found addresses per 100 probes when the budget increased from 1B to 7.81B for all the strategies, as shown in Fig. 3(a), indicating that the ability of seeds to guide the scan directions becomes weaker as the scan area grows. Compared with other alternatives, HMap6 slows the decline of marginal benefits when the budget is greater than 100M because it makes better use of seeds. For example, in the scan using the seed set S_{ICMP6}, HMap6 can generate 119.15K scan subspaces with a variable dimension of 4, which consumes a 7.81B probing budget. The alternative search strategy needs to scan some subspaces with a dimension of 5 to consume the 7.81B probing budget, and the discovery rate will decrease with the increase of scan subspace size.

To conclude, compared with the state-of-the-art solutions, HMap6 has achieved the maximum seed utilization and the best overall performance regarding discovery rate, discovery number, and marginal benefit in large-scale active IPv6 address scan.

C. Time Overhead Comparison

We evaluated each search strategy's training and probing time overhead using the three seedsets, as shown in Table VII. The training time refers to the time cost to generate the scan subspaces using the space partition algorithm, and the probing time indicates the time cost of scanning all the generated subspaces. Of note, we set 100 Kpps as the maximum probing speed in the probing phase.

We did not present the performance of 6Gan [5] because we could not run it successfully on our GPU-free testbed even with our smallest seedset (i.e., $S_{TCP\_ACK/80}$ with 0.43M seeds). Therefore, we conclude that the overly complex algorithm of 6Gan leads to too high training costs, and the deep learning-based approaches like 6Gan may not be suitable for large-scale scan tasks, at least on a regular computer. From Table VII, we can see that 6Gen requires the largest training time overhead, especially when the number of seeds is large, because the high time complexity of the AHC algorithm (i.e.,

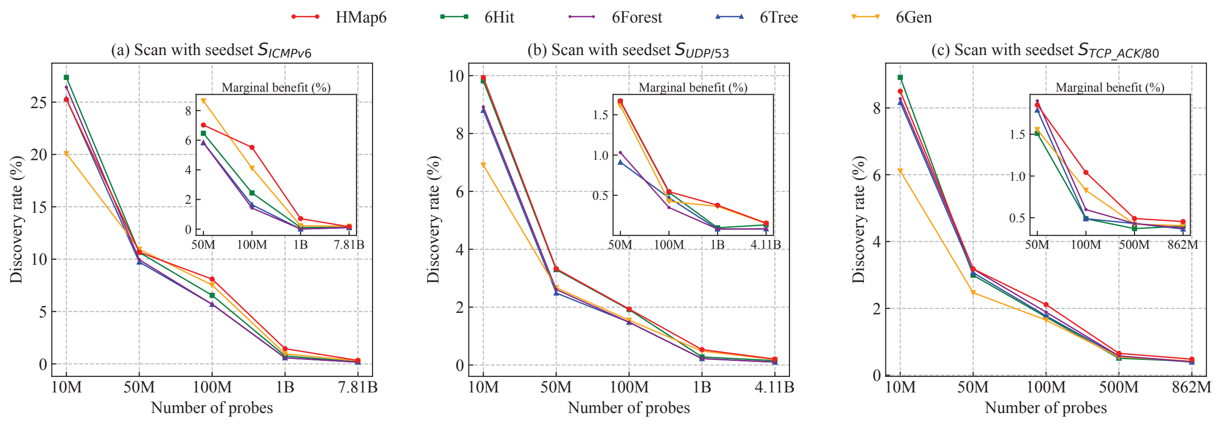


Fig. 3. The discovery rate and marginal benefits of each search strategy using various probing budgets.

TABLE VII
THE TIME COST OF EACH SEARCH STRATEGY.

Seedset (seed number, budget)	Search strategy	Training time cost	Probing time cost	Total time cost
S_{ICMPv6}	HMap6	0:0:43	22:50:10	22:50:53
	6Gen	56:24:36	22:48:15	79:12:51
	6Hit	0:0:23	36:10:11	36:10:34
(5.69M, 7.81B)	6Forest	0:0:30	22:49:15	22:49:45
	6Tree	0:0:23	22:49:13	22:49:36
$S_{UDP/53}$	HMap6	0:0:19	12:08:43	12:09:02
	6Gen	5:43:12	12:05:51	17:49:03
	6Hit	0:0:10	19:34:21	19:34:31
(2.58M, 4.11B)	6Forest	0:0:11	12:10:00	12:10:11
	6Tree	0:0:10	12:10:21	12:10:31
$S_{TCP\_ACK/80}$	HMap6	0:0:07	2:34:07	2:34:14
	6Gen	0:1:35	2:30:54	2:32:29
	6Hit	0:0:04	3:56:51	3:56:55
(0.43M, 0.86B)	6Forest	0:0:04	2:32:12	2:32:16
	6Tree	0:0:04	2:35:03	2:35:07

$O(n^3)$) leads to the fast growth of the time cost. HMap6 adopts the modified AHC and DHC algorithms, reducing the time complexity to $O(nlogn)$, which only takes seconds to complete the training of million-scale seeds.

Turning to the probing stage, the static search strategies (namely HMap6, 6Gen, 6Forest, and 6Tree) can almost achieve the maximum probing rate (i.e., 100 Kpps), since all target addresses are determined according to the seeds such that they can yield asynchronous stateless scans (i.e., sending and receiving packets in separate threads). As for the dynamic search strategy (6Hit), although it uses ZMapv6 [1] for address scanning, its search direction adjustment after each iteration severely disrupted the asynchronism of the scan, which greatly reduced the probing speed, especially when multiple iterations are performed.

To conclude, HMap6 found the most number of active addresses/prefixes using almost the least time in billion-scale scans.

V. Comparing the Staples of IPv6 Hitlists

A. *Head-to-Head Comparison*

To further illustrate the benefit of HMap6, we compare IPv6 Hitlists generated with HMap6 with Gasser's hitlists published on January 15th, 2022. From Table VIII, we can see that HMap6 creates larger hitlists that contain 31.79M unique /80 prefixes with active addresses, which are 12.64× more than the active /80 prefixes in Gasser's hitlists. Note that despite Gasser's large-scale hitlists (i.e., 109.79M unique addresses), the number of active /80 prefixes is only 2.33M due to the unbalanced distribution of addresses under prefixes and its passive collection technique. Regarding country/region coverage, our hitlists are similar to Gasser's (both covering 198 countries/regions), but our hitlists cover 515 more ASes.

We also classify the Interface Identifiers (IIDs) to compare the address patterns in the hitlists. Specifically, we first determine whether an address is an EUI-64 (Extended Unique Identifier) type by checking whether it has the specific nybbles (i.e., 'ff:fe') following the Organization Unique Identifier (OUI). Then we identify the IPv4-embedded type by pinging the IPv4 address extracted from the lowest-32 bits of the IPv6 address. Next, we check whether it is a low-byte pattern in which we define the IID (in nybble mode) with more than eight consecutive zeros as the low-byte pattern. If an IID has no discernible pattern or is unrecognized, the address is classified as "Randomized".

The low-byte pattern occupies the largest proportion of active addresses (we randomly sampled an address under each /80 prefix) in both hitlists. Our heuristic seed collection approach is based on this observation and iteratively collects the first address under the announced BGP prefixes. Consequently, HMap6 generates more low-byte pattern targets for scan, making the low-byte pattern addresses more prominent in our hitlists. The "randomized" pattern takes the second place of the most heavily represented addresses, which are usually generated by stateless address autoconfiguration (SLAAC) for devices.

B. *EUI-64 Address Distribution*

The EUI-64 addresses are also widely used as an auto-configuration mechanism to generate a unique global IPv6 address. It is of particular interest to analyze EUI-64 address distribution due to the tight relationship between EUI-64 addresses and security. Though the EUI-64 pattern can help

TABLE VIII
COMPARISON OF OUR HITLISTS WITH GASSER'S HITLISTS.

Hitlists	Method	# Active /80 prefixes	# Countries/regions	# ASes	Interface Identifiers (IIDs)			
					Randomized	Low-byte	IPv4-embedded	EUI-64
Gasser's hitlists	Passive & Active	2.33M	198	14190	0.67M 28.76%	1.33M 57.18%	0.08M 3.52%	0.24M 10.30%
Our hitlists	Active	31.79M	198	14705	8.36M 26.30%	23.04M 72.48%	0.15M 0.47%	0.14M 0.44%

TABLE IX
THE TOP 10 COUNTRIES, ASES, AND OUIS/MANUFACTURERS OF THE COLLECTED EUI-64 ADDRESSES.

Country	Ratio	AS name (ASN)	Ratio	OUI (manufacturer)	Ratio
China	70.41%	ChinaNet (4134)	21.87%	981333 (ZTE Corporation)	6.84%
Nepal	8.15 %	China Unicom (4837)	15.07%	001132 (Synology Incorporated)	2.50%
Germany	5.44 %	China Telecom (4812)	12.74%	803FBC (Unknown vendor)	2.49%
America	3.09 %	Classic Tech (55915)	8.15%	D4C1C8 (ZTE Corporation)	2.09%
Brazil	2.41%	China Mobile Communications (9808)	4.35%	525400 (Unknown vendor)	1.93%
Netherlands	1.96%	China Unicom (4808)	3.74%	689FF0 (ZTE Corporation)	1.83%
France	1.65%	China Telecom (140330)	3.66%	80EE25 (Skyworth Digital)	1.80%
Japan	1.26%	Choopa (20473)	2.01%	785F36 (Skyworth Digital)	1.63%
Russian	1.09%	Inexio (42652)	1.32%	28C01B (Skyworth Digital)	1.61%
Switzerland	0.88%	Free SAS (12322)	1.28%	309176 (Skyworth Digital)	1.29%

manage the network, this address pattern easily leaks the hardware MAC address of a device to the higher levels of the network stack and may pose a threat to the privacy and security of the device [9]. Recent works [22], [23] also show that if the home network gateway router, also referred to as customer premises equipment (CPE), uses an IPv6 addressing standard employing EUI-64, it is possible to track devices that use IPv6 at home using active measurements.

We analyzed the distribution of EUI-64 addresses that we proactively collected. Specifically, we analyzed the fraction of major countries, ASes, and OUIs/device manufacturers of the collected 381.68K (i.e., 243.50K from Gassser's hitlist and 138.18K from HMap6's scan results). As for address to manufacturer mapping, recall that the IID part of an EUI-64 IPv6 address is generated by inserting the 'ff:fe' hex string between the third and fourth bytes of a MAC address and setting the Universal/Local bit. So we extract the Organization Unique Identifier (OUI) part of the MAC address, i.e., the first three bytes, and use the official IEEE OUI database [24] for address-manufacturer mapping.

These active EUI-64 addresses cover 108 countries/regions, 2.63K ASes, and 11.81K OUIs/manufacturers. Table IX lists the top 10 countries, ASes, and OUIs/device manufacturers of the collected EUI-64 addresses. We can see that the EUI-64 addresses from China account for the largest proportion (i.e., 70.41%) and are distributed in multiple ASes within its cyber jurisdiction (e.g., ASN 4134, 4837, 4812, etc.). Moreover, most of the exposed devices in these ASes are made by Chinese manufacturers (e.g., ZTE Corporation and Skyworth Digital). This reflects the main market of the network device manufacturer in the geographical region. Meanwhile, [22] shows a high degree of device homogeneity at the AS-level, making the individual network less robust and resilient to cyber-attacks on specific vendors.

We expect that our analysis of EUI-64 IPv6 address distribution will attract more attention to the security and privacy issues of IPv6 and trigger future research on the impact of some common practice on these EUI-64 devices, e.g., IPv6 privacy extensions and prefix rotation.

VI. ETHICAL CONSIDERATIONS

When performing global IPv6 address probing, we ensure good Internet citizenship as suggested in [25]. We sent only *one* probe packet to each IP address in a scan. With the agreement of our network administrator, our probing rate is limited to 100K packets per second. The above setup would not cause any problem to the Internet.

VII. CONCLUSION

In this work, we developed HMap6, so far the most efficient scan method for Internet-wide IPv6 active address search and alias prefixes collection. HMap6 pushes the state-of-the-art with more comprehensive seed addresses, fast hierarchical clustering, and efficient alias prefix detection. Probing the Internet IPv6 with various budgets, HMap6 achieved much better performance on the discovery rate of IPv6 addresses than existing solutions. Overall, HMap6 found 4634 newly-discovered alias prefixes ranging from /28 to /120 and 12.64× more active /80 prefixes than the existing IPv6 hitlists.

ACKNOWLEDGMENT

This work is supported by the National Natura Science Foundation of China (No. 62072465), the Science and Technology Innovation Program of Hunan Province (No. 2022RC3061), the Open Foundation of Henan Key Laboratory of Cyberspace Situation Awareness (No. HNTS2022006). Tao Yang is also with Henan Key Laboratory of Cyberspace Situation Awareness, China. The corresponding author is Zhiping Cai.

References

[1] Z. Durumeric, E. Wustrow, J. A. Halderman, E. Wustrow, and J. A. Halderman, "ZMap: Fast Internet-wide Scanning and Its Security Applications," in *Proceedings of the USENIX Security Symposium (USENIX Security)*, 2013, pp. 605–620.

[2] R. D. Graham, "MASSCAN: Mass IP port scanner," 2022, https://github.com/robertdavidgraham/masscan.

[3] O. Gasser, Q. Scheitle, P. Foremski, Q. Lone, M. Korczynski, S. D. Strowes, L. Hendriks, and G. Carle, "Clusters in the Expanse: Understanding and Unbiasing IPv6 Hitlists," in *Proceedings of the ACM Internet Measurement Conference (IMC)*, 2018, pp. 364–378.

[4] B. Hou, Z. Cai, K. Wu, J. Su, and Y. Xiong, "6Hit: A Reinforcement Learning-based Approach to Target Generation for Internet-wide IPv6 Scanning," in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, 2021.

[5] T. Cui, G. Gou, G. Xiong, C. Liu, P. Fu, and Z. Li, "6GAN: IPv6 Multi-Pattern Target Generation via Generative Adversarial Nets with Reinforcement Learning," in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, 2021.

[6] A. Murdock, F. Li, P. Bramsen, Z. Durumeric, and V. Paxson, "Target Generation for Internet-wide IPv6 Scanning," in *Proceedings of the ACM Internet Measurement Conference (IMC)*, 2017, pp. 242–253.

[7] Z. Liu, Y. Xiong, X. Liu, W. Xie, and P. Zhu, "6Tree: Efficient Dynamic Discovery of Active Addresses in the IPv6 Address Space," *Computer Networks*, vol. 155, pp. 31–46, 2019.

[8] T. Yang, B. Hou, T. Zhou, and Z. Cai, "6Forest: An Ensemble Learning-based Approach to Target Generation for Internet-wide IPv6 Scanning," in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, 2022.

[9] E. C. Rye and R. Beverly, "Discovering the IPv6 Network Periphery," in *Proceedings of the Passive and Active Network Measurement (PAM)*, 2020, pp. 3–18.

[10] G. Huz, S. Bauer, K. Claffy, and R. Beverly, "Experience in using MTurk for Network Measurement," in *Proceedings of the ACM SIGCOMM Workshop on Crowdsourcing and Crowdsharing of Big (Internet) Data*, 2015.

[11] A. Yeow, "Bitnodes API," 2022, https://bitnodes.earn.com/.

[12] "RIPE NCC IPMap," 2022, https://ftp.ripe.net/ripe/ipmap/.

[13] M. Luckie, "Scamper: A Scalable and Extensible Packet Prober for Active Measurement of the Internet," in *Proceedings of the ACM Internet Measurement Conference (IMC)*, 2010.

[14] "RIPE NCC RIPEstat Data API," 2022, https://stat.ripe.net/docs/data_api/.

[15] Q. Hu and N. Brownlee, "How Interface ID Allocation Mechanisms are Performed in IPv6," in *Proceedings of the ACM International Conference on Emerging Networking Experiments and Technologies Student Workshop (CoNEXT Workshop)*, 2014, pp. 26–27.

[16] R. Beverly, R. Durairajan, D. Plonka, and J. P. Rohrer, "In the IP of the Beholder: Strategies for Active IPv6 Topology Discovery," in *Proceedings of the ACM Internet Measurement Conference (IMC)*, 2018, pp. 308–321.

[17] J. Black and P. Rogaway, "Ciphers with Arbitrary Finite Domains," in *Proceedings of the Topics in Cryptology — CT-RSA*, 2002, pp. 114–130.

[18] G. Huston, "The IPv6 Country Deployment Reports," 2022, http://labs.apnic.net/dists/v6dcc.html.

[19] "CAIDAs Prefix to ASN dataset," 2022, https://www.caida.org/data/routing/routeviews-prefix2as/.

[20] R. Beverly, "Yarrp'ing the Internet: Randomized High-speed Active Topology Discovery," in *Proceedings of the ACM Internet Measurement Conference (IMC)*, 2016, pp. 413–420.

[21] Y. Huang, M. Rabinovich, and R. Al-Dalky, "FlashRoute: Efficient Traceroute on a Massive Scale," in *Proceedings of the ACM Internet Measurement Conference (IMC)*, 2020, pp. 443–455.

[22] E. C. Rye, R. Beverly, and K. Claffy, "Follow the Scent: Defeating IPv6 Prefix Rotation Privacy," in *Proceedings of the ACM Internet Measurement Conference (IMC)*, 2021.

[23] S. J. Saidi, O. Gasser, and G. Smaragdakis, "One Bad Apple Can Spoil Your IPv6 Privacy," *ACM SIGCOMM Computer Communication Review*, vol. 52, no. 2, 2022.

[24] "IEEE Organizationally Unique Identifier (OUI) MAC Address Registry," 2022, http://standards-oui.ieee.org/oui/oui.txt.

[25] C. Partridge and M. Allman, "Ethical Considerations in Network Measurement Papers," *ACM Communications*, vol. 59, pp. 58–64, 2016.