MAnycast^2 – Using Anycast to Measure Anycast

Raffaele Sommese
University of Twente
r.sommese@utwente.nl

Leandro Bertholdo
University of Twente
l.m.bertholdo@utwente.nl

Gautam Akiwate
UC San Diego
gakiwate@cs.ucsd.edu

Mattijs Jonker
University of Twente
m.jonker@utwente.nl

Roland van Rijswijk-Deij
University of Twente
r.m.vanrijswijk@utwente.nl

Alberto Dainotti
CAIDA/UC San Diego
alberto@caida.org

KC Claffy
CAIDA/UC San Diego
kc@caida.org

Anna Sperotto
University of Twente
a.sperotto@utwente.nl

ABSTRACT

Anycast addressing – assigning the same IP address to multiple, distributed devices – has become a fundamental approach to improving the resilience and performance of Internet services, but its conventional deployment model makes it impossible to infer from the address itself that it is anycast. Existing methods to detect anycast IPv4 prefixes present accuracy challenges stemming from routing and latency dynamics, and efficiency and scalability challenges related to measurement load. We review these challenges and introduce a new technique we call "MAnycast^2" that can help overcome them. Our technique uses a distributed measurement platform of anycast vantage points as *sources* to probe potential anycast *destinations*. This approach eliminates any sensitivity to latency dynamics, and greatly improves efficiency and scalability. We discuss alternatives to overcome remaining challenges relating to routing dynamics, suggesting a path toward establishing the capability to complete, in under 3 hours, a full census of which IPv4 prefixes in the ISI hitlist are anycast.

CCS CONCEPTS

• **Networks** → **Naming and addressing**; *Public Internet*.

ACM Reference Format:
Raffaele Sommese, Leandro Bertholdo, Gautam Akiwate, Mattijs Jonker, Roland van Rijswijk-Deij, Alberto Dainotti, KC Claffy, and Anna Sperotto. 2020. MAnycast^2 – Using Anycast to Measure Anycast. In *ACM Internet Measurement Conference (IMC '20), October 27–29, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3419394.3423646

1 INTRODUCTION

Originally developed to ease service discovery in internetworks [23], on the Internet anycast is the operational routing practice of originating a particular block of address space from multiple, distributed, topological locations, leaving to the BGP interdomain routing system to route packets according to its route selection scheme [17]. An early anycast success was to enable the mitigation of a limitation of the original DNS specification, which allowed only thirteen root nameservers. Starting in the early 2000s, root server operators began to distribute instances of these authoritative name servers around the world to improve resolution time, resiliency, and robustness to DDoS attacks. Public recursive DNS resolvers such as those operated by Cloudflare, Google, Quad9, and OpenDNS also use anycast. Anycast has also gained widespread adoption by cloud providers, content delivery networks (CDNs), and hundreds of other Internet services [4, 7, 9, 14, 15].

As anycast became a common way to improve resiliency of Internet services, and resiliency of critical communications infrastructure became a public policy issue, researchers have pursued methods for third-party inference of anycast deployment, *i.e.,* identifying which addresses are anycasted and from where [7–9]. For example, identifying address prefixes that are anycast would enable more accurate assessment of resilience properties.

Although IPv6 introduced functionality to use a special format for anycast addresses [18], the IPv4 approach, which also works in IPv6, is to assign multiple hosts the same unicast address, leaving the fact that it is anycast opaque to the routing system and the end users. This opacity creates a measurement challenge. In this paper, we propose a new measurement and inference technique to efficiently detect anycast prefixes, and we analyze its practical challenges. Our technique—which we call *MAnycast^2*—is inspired by a study from De Vries et al. [11], which probes targets across the Internet from a mesh of anycast nodes in order to learn their *anycast catchment, i.e.,* which anycast node receives response traffic from a given probed target. Similar to that work, we use anycast vantage points (VPs) as *sources*, but in our case the goal is to infer whether a set of target *destination* IP addresses are themselves anycast. More precisely, contrary to the Verfploeter original goal, our is to learn characteristics of the forward path by inspecting the reverse one. Unlike previous approaches to infer whether a given prefix is anycast [9], our approach eliminates any sensitivity to latency dynamics, and improves efficiency and scalability: in less than 3 hours, we inferred the anycast state of 6.1 million IPv4 addresses from the ISI IPv4 Address Hitlist [13, 27], mapping a significant responsive portion of the IPv4 prefix space.

IMC '20, October 27–29, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-8138-3/20/10…$15.00
https://doi.org/10.1145/3419394.3423646

MAnycast^2 does not enable identification of anycast catchments, *i.e.,* which anycast instance receives packets from which source IP addresses. However, one could pipeline our technique with traditional, but less efficient latency-based inference methods. Specifically, one could use our lightweight protocol-agnostic technique to probe a large set of IP addresses, identifying a much smaller set of inferred anycast prefixes to which they could then apply a catchment-inference measurement method. Thus, our technique can reduce the time required for a comprehensive anycast detection census of the IPv4 space from weeks to days. However, we identified open challenges that require consideration to perform an accurate anycast census using our technique.

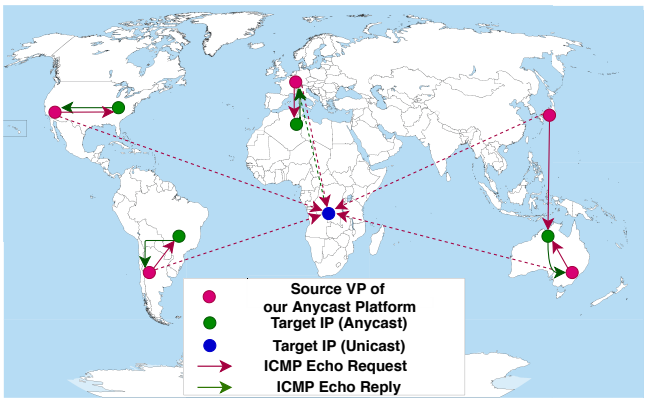

Figure 1: MAnycast^2 overview; anycast VPs are red, target IPs green, arrows are probe and response, respectively. Responses arrive at a single VP if target IP is unicast (dotted line), or at multiple VPs if target IP is anycast (continue line)

2 RELATED WORK AND BACKGROUND

As anycast deployment of DNS services ramped up, some operators used non-standard conventions to identify which name server instance responded to a particular DNS query. One such convention re-purposed an older BIND implementation feature, the CHAOS-class DNS record, to return a unique string per anycast server instance. The IETF later proposed more standard approaches [3, 29], including a Best Current Practice (BCP) to use unique autonomous system (AS) numbers for each anycast instance [21].

Starting in 2013, researchers began to develop methods for third-party detection and enumeration of anycast instances. Xun et al. [30] inferred the use of anycast for DNS top-level domain (TLD) servers, enumerating instances using the `CHAOS` query, as well as regular DNS queries, and using traceroute to resolve ambiguities in resulting responses. In 2015, Cicalese et al. [8] introduced iGreedy (§2.1), a method based on the Great-Circle Distance (GCD), which later they used to perform a census of anycast deployment on the Internet [9]—we provide additional background on GCD in Appendix §A and discuss iGreedy further in §2.1. Bian et al. [6] proposed a passive approach to detect anycast prefixes that did not rely on any active measurements, but rather used public BGP data from route collectors. They built a machine-learning classifier using features extracted from this BGP data, such as the number of upstream ASes, distances between ASes, and AS path lengths. They trained the classifier using near-ground-truth data from Cicalese et al. [9]. They achieved an accuracy of 90% in detecting ~4K anycast prefixes. They also discovered a performance impact, specifically an average increase in RTT, for anycast prefixes affected by remote peering. Other studies have also explored performance aspects of anycast deployments [19].

Our approach builds on work by De Vries et al. [11], who deployed active measurements from Cloudflare's anycast CDN. Using an established technique that used pings to map the catchment of anycast nodes [12], their goal was to identify spoofed traffic reaching their network. They observed that for each probed /24 prefix, ping responses return consistently to the same anycast instance, regardless of which instance sent the probes. We expand upon this intuition by using anycast instances as *sources, i.e.,* VPs, to discover other anycast prefixes across the Internet.

2.1 iGreedy

In this paper, we use GCD [7, 9] as the reference method for latency-based anycast detection. GCD relies on RTT measurements to estimate, based on triangulation, if a set of responses could be originated by the same geographical source. `iGreedy 1.0`, developed by Cicalese and Rossi, implemented the GCD approach using PlanetLab servers and RIPE Atlas probes as unicast VPs. iGreedy was used in [9] with 50 to 200 PlanetLab servers (now discontinued) for anycast detection, and 350 to 500 RIPE Atlas probes for the follow-up enumeration and geolocation. iGreedy, and GCD-based approaches in general, are time and resource-consuming measurements: a complete census using iGreedy on the ISI IP hitlist [13] would require measurement of ~6.1 million IP addresses from ~200 measurement nodes. To put this requirement into perspective, classifing 50k IPs (~1% of the IP hitlist) with this method would cost ~20M RIPE Atlas credits. RIPE's limit on the number of IPs a user can measure concurrently means that it would require two days using 200 RIPE Atlas probes. Using only 10 probes (suggested by the `iGreedy` documentation for a simple anecdotal usage) and performing only detection would lower this cost to 1M credits for 50k IP addresses, which still amounts to ~122M credits to probe the entire set of 6.1 million IP addresses in the hitlist. For reference, an Atlas probe earns 21600 credits per day [24]. Given the credit costs and time requirements, it is not possible to sustain a long-term census of anycast using this methodology on the RIPE Atlas platform. More importantly, in terms of measuring responsibly, one probably should not try: the Atlas team confirmed to us that one-time measurements, like the iGreedy ones, put a significant strain on the Atlas platform. An alternative would be using other platforms (e.g. Ark, MLab) or cloud providers. However diversity in connectivity of VPs also plays a role in validity of results.

Given the important role that anycast plays in resilience of modern Internet services, we pose the question: *how can we responsibly and efficiently perform a regular census of anycast deployment?* We

propose a new method that offers a promising path toward pursuing this challenge, and identify how to approach the refinements required to accommodate the complexity of the Internet's topology.

3 METHODOLOGY

The intuition behind MAnycast^2 is that we can use an anycast service to detect other anycast deployments. In this section we explain our approach in detail.

3.1 Anycast Inference Technique

Our methodology leverages anycast instances on an existing anycast fabric as VPs. We use these VPs in concert to identify if a target IP address is anycast. Since both source and destination IP addresses are potentially anycast, we will use the terms *VP instance* and *target IP instance* to distinguish these two sets of nodes, and *closest* to refer to the anycast instance closest to an IP address in terms of BGP routing. Our technique relies on the hypothesis of stable routing of anycast deployments, which has been analyzed in [28].

As the VPs are anycast, a response to a probe sent from any VP to a target IP address will be received by the VP instance closest to the responding target IP. If the target IP is anycast, then the probe will reach the closest *target IP instance* of the target's anycast deployment, and the response to this probe will reach the *VP instance* closest to the responding target IP instance. Thus, assuming stable routing, a target IP is likely unicast if all responses from the target IP are received at a single VP instance. Conversely, a target IP is likely anycast if responses are received at multiple VP instances. Fig. 1 illustrates these two scenarios.

If we compare our approach to the traditional anycast detection technique discussed in §2.1, we note that we do not rely on RTT measurements (with their consequent limitations), but instead leverage how the routing system uses topological proximity to route traffic to the nearest instance of an anycast service in both directions (from VP to target, and from target back to VP).

3.2 Anycast Measurement Framework

We use the *Tangled* [5] framework to implement our anycast measurement infrastructure. *Tangled* has ten anycast instances (VPs), receiving transit from a combination of Internet Service Providers (ISPs), commercial data-centers, academic networks, and Internet Exchange Points (IXPs) to preserve diversity of connections. *Tangled* has its own Autonomous System (AS) and IP prefix to support experiments. In Table 5, we report a summary of *Tangled* VPs. For probing we use Verfploeter [12], a global probing system running on the *Tangled* testbed that was developed to monitor anycast *catchment* distributions. Verfploeter probes the target IPs with ICMP Echo Requests and uses the ICMP Reply from each target IP to map clients to an anycast instance. For a given list of target IPs, each VP sequentially probes all the targets. Once one VP finishes probing, the next VP repeats the sequence of probing.

Our implementation infers whether an IP address is unicast or anycast with one ping from each of ten VPs, although potentially, our approach could work with UDP or TCP scanning. Since our methodology relies on ICMP Echo Replies not being filtered, we use the ISI IPv4 hitlist [13] to target IP addresses likely to respond to pings. Using this list, we probe addresses in ~6.1M IPv4 /24 prefixes.

Classification	# VPs	Distinct /24s	Distinct ASNs
Unicast	1	3451133	55209
Anycast*	2	10393	1058
Anycast*	3	719	162
Anycast	4	1378	86
Anycast	5	2467	83
Anycast	6	567	39
Anycast	7	13	9
Anycast	10	3	1
Total Anycast	*	15540	1234

Table 1: Classification and Breakdown of /24s by number of *Tangled* VPs that receive responses

Under recommended best practices, /24 is the most-specific prefix length that globally propagates via BGP on the Internet [20]. For this reason, once we classify a target IP address as anycast or unicast, we extend this classification to its covering /24.

4 PRELIMINARY RESULTS

We next describe our preliminary results when applying our method to infer which prefixes in a large set of IPv4 addresses are anycast (§4.1). We report the results of a validation we perform against popular anycast platforms such as root servers, public recursive resolvers (§4.2), and other services (§4.3). Finally, we compare our results against iGreedy (§4.4) and the passive approach in [6] (§4.5).

4.1 IPv4 anycast measurement

Table 1 classifies /24 prefixes, and their originating ASNs, by the number of *Tangled*'s anycast VPs that received ICMP Echo Replies in response to our probes. On May 5, 2020, we tested all 6,125,756 target /24 prefixes from the ISI IPv4 hitlist, choosing 1 IP for each /24 prefix. Of these, 3.47 million (56.5%) responded to at least one probe. For 3.45 million /24 prefixes (99.55% of the responding prefixes), only one VP received ICMP Echo Replies, so we classified these prefixes as unicast. For another 15,540 (0.45%) /24 prefixes, between 2 and 10 VPs received ICMP Echo Replies. We considered them candidate anycast prefixes. In our 10-node testbed, we completed measurements for the entire ISI IPv4 hitlist [13] in ~2.5 hours, requiring only 10 pings per target IP (one from each node).

4.2 Ground Truth Validation

Our first approach to validation used DNS root servers and public DNS resolvers. We correctly classifed all root servers as anycast except C-Root. C-Root's misclassification was a false negative (*i.e.,* we failed to detect an anycast IP). We also correctly classified the anycast prefixes serving three of the four major public DNS resolvers: CloudFlare, OpenDNS, and Quad9. However, we incorrectly classified as unicast the prefix for the Google Public DNS Resolver. We discuss the cases we misclassified in §5.

4.3 Validation from AS Operators

Table 2 shows the Top-10 anycast providers in the Internet by number of /24 prefixes that we classified as anycast. For prefixes that we classified as anycast, we verified if the iGreedy technique came to the same conclusion. We also asked for ground truth validation

ASN	Org Name	#Prefixes		
		MAnycast^2	iGreedy	Operator
16509	Amazon	4870	520	524
13335	Cloudflare	3127	3127	Confirmed
32475	SingleHop	747	0	
54113	Fastly	285	213	
12041	Afilias	215	214	
16625,21342,24319 26008,32787,35994 20940,34164,34164	Akamai	212	89	90
701,703,2828 14153,15133	Verizon	154	153	
42	PCH	134	134	134
15169,19527 36040,36384	Google	133	29	
12008,19905,19911	NeuStar	110	110	

Table 2: Top 10 Anycast ASes detected by the MAnycast^2 measurement, validated with iGreedy and operators

of our inferences to four well-known operators of anycast services: Amazon, Cloudflare, Afilias, and Packet Clearing House/PCH.

For some operators, like SingleHop, and, to some degree, Akamai and Google, we found conflicting results between iGreedy and our methodology. For Amazon, our methodology detected 10× as many anycast prefixes as iGreedy. Looking at reverse DNS data for these prefixes, it appears that 3,555 prefixes we detected as anycast were related to EC2 instances, unlikely to be anycast given that Amazon does not offer anycast service on EC2 IP addresses. We conclude that AWS's routing policies (§5) might mislead our methodology. In private communication with us, Amazon reported 524 /24 blocks as anycast. We discovered 520 of these; using the list of public IP ranges of AWS [1], we discovered that they belonged to AWS Global Accelerator, an anycast platform that AWS operates to support its customers [26]. The remaining 4 /24 prefixes also belonged to Global Accelerator and are supposed to be anycast, but both our method and iGreedy incorrectly inferred them as unicast.

Cloudflare has one of the largest anycast deployments. We detected 3,127 /24 anycast prefixes, consistent with iGreedy's inferences. Cloudflare operators confirmed our overall findings, with the caveat that they are revising their address assignment plan, preventing an exact prefix-by-prefix confirmation.

PCH confirmed our inference of 134 /24 prefixes as anycast. GoDaddy, not in the Top-10 list, confirmed that we correctly classify its two IP ranges as anycast. Microsoft, also not in the Top-10 list, confirmed that we correctly detected 51 of their 75 /24 anycast prefixes, and that we misclassified 7 /24 unicast prefixes as anycast.

4.4 Comparison against iGreedy

To compare our results with those obtained by iGreedy (Table 3), we ran iGreedy against a set of target addresses. We used 200 random RIPE Atlas probes, as geographically diverse unicast measurement nodes. We manually verified that there were at least 4 probes per continent, to limit the bias of the heavy representation of Europe in RIPE Atlas probe deployment. Given the performance bottleneck of a large-scale measurement with iGreedy (§ 2.1), we sampled ∼2% of prefixes we identified as unicast and tested them with iGreedy. We then ran iGreedy against all prefixes in our anycast candidate set. In total, we ran iGreedy on 82,270 /24 prefixes.

# VPs	Class.	# /24	iGreedy Classification			% Diff. (resp.)
			Uni	Any	Unresp.	
1	Uni	66730	66658	72	0	0.1%
2	Any	10393	8072	887	1434	90.1%
3	Any	719	93	603	23	13.3%
4	Any	1378	3	1375	0	0.2%
5	Any	2467	0	2467	0	0%
6	Any	567	0	567	0	0%
7	Any	13	0	13	0	0%
10	Any	3	0	3	0	0%

Table 3: Comparison against iGreedy: We show MAnycast^2 classification, iGreedy classification and difference between their results

We observed slight differences, as low as 0.1% for the sample unicast prefixes, indicating low false negative rates. If we received answers on 4 distinct VPs, the percentage difference was 0.2%, while when we received answers on 5 or more distinct VPs, our results agreed with iGreedy. The disparity was extremely high when we received responses at only two distinct VPs (90%), and although it significantly dropped, it was still higher than 10% for cases where we received responses at three distinct VPs. We suspect that the disparity when there were few VPs derives from routing dynamics, which we discuss further in §5.

4.5 Preliminary comparison against a passive approach

Another technique for detecting anycast deployment is the passive approach proposed by Bian et al. [6], which relies on applying machine-learning classification to features extracted from BGP data (§2). Unfortunately, up-to-date ground-truth data to train this classifier is not available—the last iGreedy Census was performed in April 2017—preventing us from performing a *fair* comparison against MAnycast^2. However, to attempt a preliminary comparison, we ran a trained classifier provided by the authors of [6] (*i.e.,* trained with the iGreedy Census data from April 2017) and applied it to current BGP data (from May 05, 2020). Of the 5,915 prefixes marked by MAnycast^2 and iGreedy as anycast, the passive approach classified only 3,899 of these prefixes as anycast. Because we did not update this 3-year-old model, we cannot tell if the limited overlap with MAnycast^2/iGreedy is due to the model needing retraining, anycast deployments that were not visible in the signals available to the passive approach, or both. In any case, we expect the key BGP properties used by the classifier to not change significantly over 3 years, allowing us to gather some first-hand insight by manually inspecting some cases of potential misclassification.

With respect to false positives (*i.e.,* unicast classified as anycast), we found cases where the passive approach, in contrast with MAnycast^2 and iGreedy, classified large prefixes (*i.e.,* /8, /10) as anycast. This yielded many /24 blocks marked as anycast. The reason behind this misclassification is related to the nature of the ASes that own such prefixes, which are large backbone Internet providers. In this case, the high number of providers interconnected to these networks misleads the passive approach to classify them as anycast. MAnycast^2 classified all these cases as unicast in agreement with

iGreedy. Moreover, as studied in [6], remote peering is another cause of misclassification.

With respect to false negatives, we identified some interesting cases that mislead the passive approach. Consider Neustar a large anycast operator that the passive approach misclassified. Neustar used an ASN for their anycast network that was directly connected to another Neustar ASN as upstream provider. Accordingly, the classifier saw only one upstream provider and marked these prefixes as unicast. A similar phenomenon happened for other operators, such as Akamai. This cause of misclassification was also identified in [6] as a common cause for false positives. We found other cases of false-negative misclassification where only one classification feature (specifically a long observed AS path) was indicative of unicast behavior. We believe this might be an artifact of using a classifier trained with older ground truth.

Finally, we examined the anycast deployment size inferred by iGreedy for all the /24 blocks where it agreed with MAnycast^2. The distribution of prefixes misclassified (FN) by the passive approach was largely skewed toward small deployments. These results are preliminary. We plan to conduct a new anycast census by pipelining MAnycast^2 and iGreedy, which will allow us to retrain the classifier developed by Bian et al. [6].

5 OPEN CHALLENGES

Our experiments with MAnycast^2 show that the approach is promising, with a low (0.1%) false negative rate (anycast addresses mistakenly classified as unicast) (§4) and, if responses are received at more than 4 VPs, a low or zero false positive rate. As we discussed in §4.2, however, our MAnycast^2 approach misclassifies two prominent anycast services (C-Root and Google Public DNS), and provides ambiguous results when only 2-3 VPs receive responses. In this section we consider open challenges that underlie these wrong inferences, and how future work may overcome these.

5.1 Conditions for Success

To understand the open challenges, we first need to consider the necessary conditions for the MAnycast^2 approach to successfully detect prefixes as anycast. The most important factor in this context is connectivity between the vantage points and the anycast service we are trying to detect. As demonstrated in [22], the number and type of upstream providers of an anycast network have a significant impact on the interaction of the anycast network with the Internet. Anycast networks with many upstream providers have more options to manipulate their BGP announcements to improve the catchments of their constituent sites. In the same way, connectivity impacts MAnycast^2 measurements.

This raises the question: *What are the minimum conditions, in terms of connectivity, for our methodology to detect an anycast deployment?* From a theoretical point of view, the simplest answer to this question is that there should be at least two VPs that prefer different PoPs, which themselves prefer different VPs. This will result in traffic routed back to two different VPs in our measurement, thus, in the detection of that network as anycast. Ensuring this property of an anycast network is not always possible. When this minimal connection is not satisfied, the MAnycast^2 methodology may fail to detect an anycast service, because connectivity between the anycast service and MAnycast^2 VPs may result in all traffic from the anycast nodes ending up at a single MAnycast^2 VP.

In the following paragraphs, we examine three aspects of connectivity structure and routing dynamics that can lead MAnycast^2 to misclassify prefixes.

Impact of Routing Policies. We discovered two cases where routing policies led to failures to detect anycast prefixes.

C-Root: Single Preferred Route – C-Root, which MAnycast^2 misclassified as unicast, is managed by Cogent Communications via AS2149. In private communication with a Cogent operator, we discovered that Cogent considered one of the Tangled providers, Vultr, as a *preferred-route*, but only received routes from one of the three Tangled sites that used Vultr as provider. For this reason, Cogent delivered traffic only to the London VPs. We confirmed that all traffic reached the London VP with traceroutes from one of Cogent's public looking glasses [10].

Google Public DNS: Direct Peering Preferred – For Google Public DNS resolver (AS15169), we also observed a preferred receiving VP, this time located in South America. A reasonable explanation for this behavior is that, at São Paulo IXP, Google had a direct connection to our anycast testbed. Previous studies have found that Google prefers to route packets entirely through their global network whenever possible [2, 31]. Similar to the Cogent case, from whichever node we probed Google Public DNS, a single VP received all the answers.

To explore the impact of routing decisions, we performed the following test. From our testbed location in Japan, we probed Google Public DNS with two separate packets: the first using the anycast IP of the testbed as the source address, and the second using the unicast (management) IP address of the same host. We received the response to the first packet at our VP in Sao Paulo, with an RTT of ~120ms. The response to the unicast probe arrived at our VP in Japan, with an RTT of ~2ms.

These examples establish an open challenge for our methodology: accommodating preferred routing strategies from large network operators, especially in combination with local connectivity characteristics of testbed VPs. Enriching the testbed's connectivity, both in terms of path diversity and number of VPs, will increase the chance of observing multiple paths, thus increasing the probability of success of our our methodology.

Routing Flaps and Load Balancing. Another routing phenomenon, which can mislead our method in the other direction, i.e., misclassifying unicast prefixes as anycast, are routing flaps and load balancing (traffic engineering). Based on our analysis in §4.4, we believe this is mostly likely to happen when we receive responses at only two (or occasionally three) VPs. A key factor seems to be the time that elapses between probing a target IP address from distinct VPs. Currently, our implementation probes the entire hitlist from one VP, then moves to probing from another VP. This cadence can leave ~13 minutes between pings from different VPs to the same IP. Further investigation will improve our understanding of how this gap allows routing flaps to mislead our inferences. One way to compensate for this risk is to probe a single target IP from all VPs before moving to the next target IP. Load balancing is harder to identify and filter, but generally, using more VPs can prevent corner cases where we receive packets at only 2 VPs. We did not identify

	Probing Overhead #IPs		#ICMP	iGreedy as Ground-Truth			
	200 Atlas VPs	10 Tang. VPs	Sent	TP	TN	FP	FN
iGreedy	6.1 M	-	1220M	100%	100%	0%	0%
MA^2	-	6.1M	61M	39.1%	99.9%	60.9%	0.1%
Combo	11K	6.1M	63M	98.8%	99.9%	0.2%	0.1%

Table 4: Comparison between iGreedy, MAnycast^2 and a combined approach (iGreedy for <=3VPs MAnycast^2 results) in terms of overhead, footprint and classification rate

any specific VPs pairs causing this problem. Repeated measurements performed at different times/days could discriminate some of these corner cases. For example, we repeated our Internet-wide measurement on May 25, 2020, and were able to resolve 90% of the incorrect classifications discussed in §4.3.

Regional and Topological Blindspots. Our method's accuracy appears to vary by region, perhaps due to variation in density of connectivity relative to different VPs in our testbed. *Tangled* has relatively few nodes (10 in total), which may prevent detection of regional anycast services. Latency-based approaches face similar challenges in detecting small anycast deployments. In general, regional anycast services are challenging to detect and require a widely distributed geographical infrastructure with many nodes.

5.2 Validation experiment with PEERING

In the interests of repeatability and reproducibility, and to understand whether the open challenges we identify are independent of the particular setup of our *Tangled* testbed, we performed an experiment with MAnycast^2 on the PEERING testbed [25].

We ran a measurement on September 11, 2020 using 7 PEERING nodes. Although some nodes had multiple upstream providers that could give us additional information about which route a response took, since the current version of Verfploeter does not record Layer 2 information, we considered each PEERING node as a single VP. The results show an overlap of 90% with the MAnycast^2 measurement performed from *Tangled* in May 2020 for answers received on 4 or more VPs. We report additional results in Appendix §C. The MAnycast^2 measurement performed through PEERING detected fewer anycast prefixes, confirming that the number and the connectivity of VPs impacts the anycast detection capability. The experiment on PEERING also detected Google Public DNS resolver as unicast. Further inspection showed that all responses from Google Public DNS reached the Amsterdam node of PEERING, which directly interconnected to Google via the AMS-IX route servers. A follow-up experiment, withdrawing the announcement from the Amsterdam node, showed that we were able to correctly detect Google Public DNS resolver as anycast. This additional example again illustrates the impact of routing policies and connectivity on the MAnycast^2 measurement.

5.3 Considerations on Applicability

If MAnycast^2 receives answers at 4 or more VPs, we can safely assume that an address is anycast, but our preliminary experiments warn against this conclusion for the case of 2 or 3 VPs. However, our methodology shows strong results when classifying unicast. A possible use of our methodology is therefore to filter out, efficiently and at scale, unicast addresses so that one can apply the heavier-weight latency-based method on a smaller remaining set of prefixes for which we are uncertain (2 or 3 VPs). Table 4 reports for this combined approach the overhead in terms of measured IPs with the different platforms, the traffic footprint in terms of ICMP Echo requests generated, and the classification rates compared to iGreedy and pure MAnycast^2. The combined approach provides classification results close to iGreedy with a substantially reduced measurement overhead. We believe our methodology can therefore significantly contribute to scaling anycast detection. A further improvement to MAnycast^2 could be, when deployed on VPs with multiple peers, to consider each incoming upstream connection as a separate VPs. In this way, probes from different incoming routes can identify an anycast target even if they are received at the same VP. Finally, multiple peers will offer the opportunity of manipulating routes (*e.g.,* prepending, selective announcements, etc.).

6 CONCLUSION

We introduced MAnycast^2, a new measurement methodology based on the idea of using anycast IPs as VPs to launch active measurements to candidate anycast destinations, in order to infer whether a given /24 prefix is anycast. We compared preliminary results obtained with our methodology with results of a state-of-the-art latency-based methodology, and validated results against publicly available ground-truth and confirmation from operators. This validation process allowed us to identify false positives and false negatives that suggested open challenges in broader application of this method. Our minimal false-negative rate suggests the substantial value of our methodology in an IPv4-side census of anycast, because it allows a first-pass quick detection to eliminate most unicast IPs, leaving a far smaller list of anycast prefixes that a latency-based methodology could then further confirm. Future improvements to our methodology will focus on reducing the false-negative classification rate by carefully considering differences in levels of connectivity at different vantage points of our measurement framework. We will also consider RTT data obtained when ping responses arrive at the originating VP, which may enable geolocation and enumeration of anycast deployments.

ACKNOWLEDGEMENTS

We thank our shepherd Ethan Katz-Bassett and the anonymous reviewers for their insightful suggestions and feedback. We also thank Wouter de Vries, Luuk Hendriks and Moritz Müller for software support. We thank Johan ter Beest, Kyle Schomp, Duane Wessels, Matt Calder, Marwan Fayed, Kabindra Shrestha, Bill Woodcock, and Geoffrey M. Voelker for their valuable time, insights, and feedback. A special thank to Rui Bian and Hao Shuai for providing updated data based on their paper on passive anycast detection. This work uses measurements from RIPE Atlas (https://atlas.ripe.net), an open measurement platform operated by RIPE NCC. This work was supported in part by: the NWO-DHS MADDVIPR project (628.001.031/FA8750-19-2-0004); National Science Foundation grants CNS-1764055, CNS-1903612, CNS-1705024,and CNS-190151; DARPA Coop. Agg. HR00112020014; and the EU H2020 CONCORDIA project (830927).

REFERENCES

[1] Amazon. 2020. AWS IP address ranges. (Apr 2020). https://docs.aws.amazon.com/general/latest/gr/aws-ip-ranges.html

[2] T. Arnold, M. Calder, I. Cunha, A. Gupta, H. V. Madhyastha, M. Schapira, and E. Katz-Bassett. 2019. Beating BGP is Harder than We Thought. In *Proceedings of the 18th ACM Workshop on Hot Topics in Networks (HotNets '19)*. Association for Computing Machinery, New York, NY, USA, 9–16. https://doi.org/10.1145/3365609.3365865

[3] R. Austein. 2007. DNS Name Server Identifier (NSID) Option. RFC 5001. (Aug. 2007). https://doi.org/10.17487/RFC5001

[4] H. Ballani and P. Francis. 2005. Towards a Global IP Anycast Service. In *Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM '05)*. Association for Computing Machinery, New York, NY, USA, 301–312. https://doi.org/10.1145/1080091.1080127

[5] L. Bertholdo, J. M. Ceron, W. B. de Vries, R. de O. Schmitt, L. Zambenedetti Granville, R. van Rijswijk-Deij, and A. Pras. 2020. Tangled: A Cooperative Anycast Testbed. (2020). arXiv:cs.NI/2008.12881

[6] R. Bian, S. Hao, H. Wang, A. Dhamdere, A. Dainotti, and C. Cotton. 2019. Towards Passive Analysis of Anycast in Global Routing: Unintended Impact of Remote Peering. *SIGCOMM Comput. Commun. Rev.* 49, 3 (Nov. 2019), 18–25. https://doi.org/10.1145/3371927.3371930

[7] D. Cicalese, J. Augé, D. Joumblatt, T. Friedman, and D. Rossi. 2015. Characterizing IPv4 Anycast Adoption and Deployment. In *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies (CoNEXT '15)*. Association for Computing Machinery, New York, NY, USA, Article 16, 13 pages. https://doi.org/10.1145/2716281.2836101

[8] D. Cicalese, D. Joumblatt, D. Rossi, M. Buob, J. Augé, and T. Friedman. 2015. A fistful of pings: Accurate and lightweight anycast enumeration and geolocation. In *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2776–2784. https://doi.org/10.1109/INFOCOM.2015.7218670

[9] D. Cicalese and D. Rossi. 2018. A Longitudinal Study of IP Anycast. *SIGCOMM Comput. Commun. Rev.* 48, 1 (April 2018), 10–18. https://doi.org/10.1145/3211852.3211855

[10] Cogent. 2020. Cogent Looking Glass. (2020). https://www.cogentco.com/lookingglass.php [Online; accessed 01-June-2020].

[11] W. B. de Vries, S. Aljammundefinedz, and R. van Rijswijk-Deij. 2020. Global-Scale Anycast Network Management with Verfploeter. In *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*. IEEE Press, 1–9. https://doi.org/10.1109/NOMS47738.2020.9110449

[12] W. B. de Vries, R. de O. Schmidt, W. Hardaker, J. Heidemann, P-T de Boer, and A. Pras. 2017. Broad and Load-Aware Anycast Mapping with Verfploeter. In *Proceedings of the 2017 Internet Measurement Conference (IMC '17)*. Association for Computing Machinery, New York, NY, USA, 477–488. https://doi.org/10.1145/3131365.3131371

[13] X. Fan and J. Heidemann. 2010. Selecting Representative IP Addresses for Internet Topology Studies. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement (IMC '10)*. Association for Computing Machinery, New York, NY, USA, 411–423. https://doi.org/10.1145/1879141.1879195

[14] M. J. Freedman, K. Lakshminarayanan, and D. Mazières. 2006. OASIS: Anycast for Any Service. In *Proceedings of the 3rd Conference on Networked Systems Design & Implementation - Volume 3 (NSDI'06)*. USENIX Association, USA, 10.

[15] P Gilmore. 2013. Serving at the Edge: Good for Performance, Good for mitigating DDoS. (Apr 2013). https://blogs.akamai.com/2013/04/serving-at-the-edge-good-for-performance-good-for-mitigating-ddos-part-ii.html

[16] B. Huffaker, M. Fomenkov, D. Plummer, D. Moore, and K. Claffy. 2002. Distance Metrics in the Internet. In *IEEE International Telecommunications Symposium (ITS)*. IEEE, Brazil, 200–202.

[17] K. Lindqvist J. Abley. 2006. Operation of Anycast Services. RFC 4786. (Dec. 2006). https://doi.org/10.17487/RFC4786

[18] D. Johnson and S. Deering. 1999. Reserved IPv6 Subnet Anycast Addresses. RFC 2526. (March 1999). https://doi.org/10.17487/RFC2526

[19] Z. Li, D. Levin, N. Spring, and B. Bhattacharjee. 2018. Internet Anycast: Performance, Problems, & Potential. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication (SIGCOMM '18)*. Association for Computing Machinery, New York, NY, USA, 59–73. https://doi.org/10.1145/3230543.3230547

[20] A. Lutu, M. Bagnulo, and O. Maennel. 2013. The BGP Visibility Scanner. In *2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. 115–120.

[21] D. R. McPherson, R. Donnelly, and F. Scalzo. 2011. Unique Origin Autonomous System Numbers (ASNs) per Node for Globally Anycasted Services. RFC 6382. (Oct. 2011). https://doi.org/10.17487/RFC6382

[22] S. McQuistin, S. P. Uppu, and M. Flores. 2019. Taming Anycast in the Wild Internet. In *Proceedings of the Internet Measurement Conference (IMC '19)*. Association for Computing Machinery, New York, NY, USA, 165–178. https://doi.org/10.1145/3355369.3355573

[23] C. Partridge, T. Mendez, and W. Milliken. 1993. Host anycasting service. RFC 1546. (Nov. 1993). https://doi.org/10.17487/RFC1546

[24] RIPE. 2020. RIPE Atlas - The Credit System. (2020). https://atlas.ripe.net/docs/credits/

[25] B. Schlinker, T. Arnold, I. Cunha, and E. Katz-Bassett. 2019. PEERING: Virtualizing BGP at the Edge for Research. In *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies (CoNEXT '19)*. Association for Computing Machinery, New York, NY, USA, 51–67. https://doi.org/10.1145/3359989.3365414

[26] R. Shaun. 2018. AWS Global Accelerator for Availability and Performance. (Nov 2018). https://aws.amazon.com/it/blogs/aws/new-aws-global-accelerator-for-availability-and-performance/

[27] USC/ISI. 2020. USC/ISI ANT Datasets. (2020). https://ant.isi.edu/datasets/ip%5Fhitlists/ [Online; accessed 05-May-2020].

[28] L. Wei and J. Heidemann. 2017. Does anycast hang up on you?. In *2017 Network Traffic Measurement and Analysis Conference (TMA)*. IEEE, 1–9. https://doi.org/10.23919/TMA.2017.8002905

[29] S. Woolf and D. Conrad. 2007. Requirements for a Mechanism Identifying a Name Server Instance . RFC 4892. (June 2007). https://doi.org/10.17487/RFC4892

[30] F. Xun, J. Heidemann, and R. Govindan. 2013. Evaluating anycast in the domain name system. *2013 Proceedings IEEE INFOCOM* (2013), 1681–1689. https://doi.org/10.1109/INFCOM.2013.6566965

[31] K. Yap, M. Motiwala, J. Rahe, S. Padgett, M. Holliman, G. Baldus, M. Hines, T. Kim, A. Narayanan, A. Jain, V. Lin, C. Rice, B. Rogan, A. Singh, B. Tanaka, M. Verma, P. Sood, M. Tariq, M. Tierney, D. Trumic, V. Valancius, C. Ying, M. Kallahalla, B. Koley, and A. Vahdat. 2017. Taking the Edge off with Espresso: Scale, Reliability and Programmability for Global Internet Peering. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication (SIGCOMM '17)*. Association for Computing Machinery, New York, NY, USA, 432–445. https://doi.org/10.1145/3098822.3098854

A GREAT CIRCLE DISTANCE METHODS

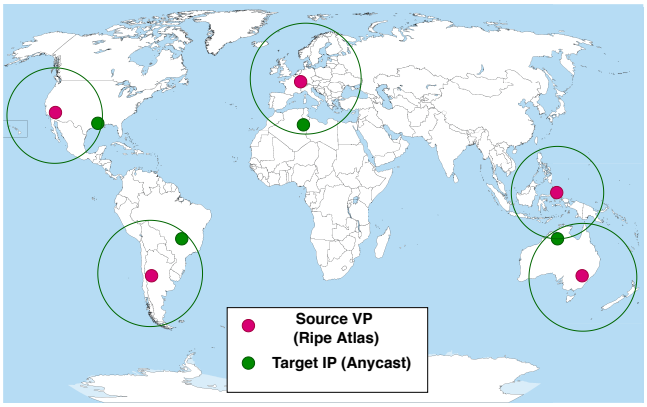

Figure 2: Great Circle Distance Technique

The traditional technique to detect anycast prefixes—the Great-Circle Distance (GCD) technique—relies on the execution of round trip time (RTT) measurements from geographically distributed unicast VPs. Based on VP locations, one can infer a circular region of possible geolocation for a target IP whose diameter is constrained by the observed RTT and speed of communication. Fig. 2 illustrates the approach: if a target IP is unicast, then these circles will all intersect and the intersection is the approximate location of the target IP. However, if a target IP is anycast, then the RTT from each node is likely smaller compared to the anycast case, as the ping response is returned by the instance closest to the target IP (in terms of routing). In the anycast scenario, not all circles may intersect and the various intersections of circles will approximate the different locations of the anycast instances. One can use this

VP ID	Location	Transit Provider	IXP	Peers
au-syd	Sydney (AU)	Vultr (20473)	–	1
br-gru	Sao Paulo (BR)	Ampath(20080) ANSP(1251)	spo.IX.br	1892
br-poa	Porto Alegre (BR)	Leovin(262605) Nexfibra(264575)	poa.IX.br	218
dk-cop	Copenhagen(DK)	DK-Hostmaster (39839)	–	1
uk-lnd	London(UK)	Vultr (20473)	Linx	1
fr-par	Paris(FR)	Vultr (20473)	France-IX	1
jp-hnd	Tokyo(JP)	Wide (2500)	–	1
nl-ens	Enschede(NL)	UTwente (1133)	–	1
us-mia	Miami(US)	Ampath (20080)	–	1
us-was	Washington(US)	Los Nettos (226)	–	1

Table 5: Tangled VPs location and connectivity

technique from a geographically diverse set of unicast nodes to not only infer if an IP is anycast, but also to estimate the geographic footprint of the corresponding anycast fabric. The GCD technique relies on accurate latency measurements, which require multiple measurements from multiple nodes. GCD can be sensitive to latency dynamics or path characteristics [16]. This sensitivity can lead to false negatives—a failure to detect an anycast prefix—if deployed across underprovisioned infrastructure.

B TANGLED ANYCAST DEPLOYMENT

Table 5 reports a summary of *Tangled* VPs, including locations of upstream transit providers (with ASN), IXP connectivity, and the number of peers to which the node established an interconnection.

VP ID	Location	Transit Provider	IXP	Peers
wisc01*	Madison (US)	University of Wisconsin (3128)	–	1
gatech01*	Atlanta (US)	Georgia Institute of Technology (2637)	–	1
amsterdam01*	Amsterdam (NL)	Bit BV (12859) Netwerkvereniging Coloclue (8283)	AMS-IX	861
uw01	Seattle (US)	Pacific Northwest Gigapop (101)	–	1
grnet01	Athens(GR)	GRNet (5408)	–	1
ufmg01*	Belo Horizonte (BR)	RNP (1916)	mg.IX.br	94
seattle01	Seattle (US)	RGNet (3130)	SIX	334

Table 6: PEERING VPs location and connectivity

Classification	# VPs	Distinct /24	Distinct ASN
Unicast	1	3390077	54343
Anycast*	2	15780	905
Anycast*	3	1243	111
Anycast	4	2092	36
Anycast	5	1061	10
Anycast	7	1	1

Table 7: Classification and Breakdown of /24s by number of PEERING VPs that receive responses

Distinct /24 # VPs PEERING	# VPs Tangled Unresp.	1 (U)	2 (A*)	3 (A*)	≥4(A)	Total
1 (Unicast)	404426 (11.9%)	2977255 (87.8%)	8044 (0.2%)	199 (0%)	153 (0%)	3390077 (100%)
iGreedy Anycast	–	53	266	123	152	–
2 (Anycast*)	1392 (8.8%)	12922 (81.9%)	837 (5.3%)	331 (2.1%)	298 (1.9%)	15780 (100%)
iGreedy Anycast	10	89	332	308	297	–
3 (Anycast*)	10 (0.8%)	16 (1.3%)	61 (4.9%)	108 (8.7%)	1048 (84.3%)	1243 (100%)
iGreedy Anycast	7	9	53	106	1048	–
≥4 (Anycast)	28 (0.9%)	9 (0.3%)	223 (7.1%)	43 (1.4%)	2851 (90.4%)	3154 (100%)
iGreedy Anycast	27	8	218	43	2851	–

Table 8: Comparison between PEERING and Tangled results by numbers of VPs that receive responses on the two platforms. For each intersection, we also report the numbers of anycast prefixes according to iGreedy. The ≥4 intersection shows the high overlap in detection of anycast instances by the two platforms

C PEERING MEASUREMENT RESULTS

In this section, we report additional results of measurements performed with the PEERING platform. Table 6 shows a summary of PEERING VPs used. Table 7 reports the results of the PEERING measurement performed on Sep 11, 2020. Table 8 reports a comparison with the *Tangled* measurement performed on May 05, 2020. It is important to bear in mind the possible temporal bias in these results because the two measurements occurred 4 months apart. As shown in §5.2, in 90% of the cases, when 4 or more PEERING VPs received responses, 4 or more Tangled VPs also did. When MAnycast^2 in PEERING classified the prefixes as unicast, it agreed with Tangled for 87.8% of the cases, and disagreed for just 0.3%. Of this 0.3%, most of the cases were when MAnycast^2 in Tangled received answers on 2 VPs. The iGreedy data confirmed that only 266 /24 prefixes of these 8044 were anycast. As stated in §5, this misclassification by MAnycast^2 in Tangled was probably caused by route flaps and load balancing. For prefixes with answers received on 3 or more VPs on Tangled and classified by MAnycast^2 in PEERING as unicast, iGreedy showed that 275 of 352 /24 prefixes were anycast. When MAnycast^2 in PEERING classified prefixes as anycast in disagreement with Tangled, for answers received on 2 VPs on PEERING, only 89 of 12922 /24s were anycast according to iGreedy. For answers received on 3 or more VPs, 17 of 25 /24s were anycast. These results confirm that MAnycast^2 provides good results when answers arrive on 3-4 or more VPs, and that repeated measurement, with different sets of VPs, can reveal more anycast prefixes and filter out misclassified unicast prefixes due to route flaps and load balancing.

For comparison, we repeated the measurement with PEERING with fewer VPs – the four marked with * in Table 6. For prefixes classified in the previous measurement as anycast (with answers on 4 or more VPs), we discovered in the new measurement that 4% of these were misclassified as unicast, 26% were received at only 2 VPs, 49.7% at only 3 VPs, and only 0.3% (10 prefixes) at 4 VPs. These results confirm the importance of using a high number of VPs, to lower the measurement overhead required by iGreedy and to avoid missing anycast prefixes.