Network Security and Measurement

- Scanning the Internet -

Prof. Dr. Thomas Schmidt

http://inet.haw-hamburg.de | t.schmidt@haw-hamburg.de

Agenda

Internet-wide scanning

Applications of high-speed scanning

Reducing the scanning footprint

How to scan IPv6?

Observing IPv6 scanners

Discovery at Large

INTERNET-WIDE SCANNING

Measurement objectives

Which IP address is online?

Which IP address runs which service?

Which type of host or service is behind an IP or port?

You don’t have access to flow data.

You want to answer these questions for (almost) all IP addresses.

Network Mapper: NMAP

NMAP was the first integrated tool for Internet scanning – released in September 1997 by Gordon Lyon (Fyodor)

Host discovery

- Originally using network ranges (lists)
- Random IP generation

Operating system discovery

- Originally fingerprinting the TCP/IP stack
- Response matching in OS database

Service discovery

- Determine open ports from protocol reply
- Determine closed ports from ICMP reply

Fingerprinting

OS:

- Analyze protocol options and implementation details of IP/ICMP/TCP/UDP
- Predict the uptime from TCP timestamps

TCP service:

- Complete the connect handshake
- Many services send a banner

UDP service:

- UDP does not respond by itself
- Send protocol-specific payloads and match responses

Fingerprinting

Fingerprinting is a complex process of correlating various properties observed from the system

OS:

- Analyse protocol options and implementation details of IP/ICMP/TCP/UDP
- Predict the uptime from TCP timestamps

TCP service:

- Complete the connect handshake
- Many services send a banner

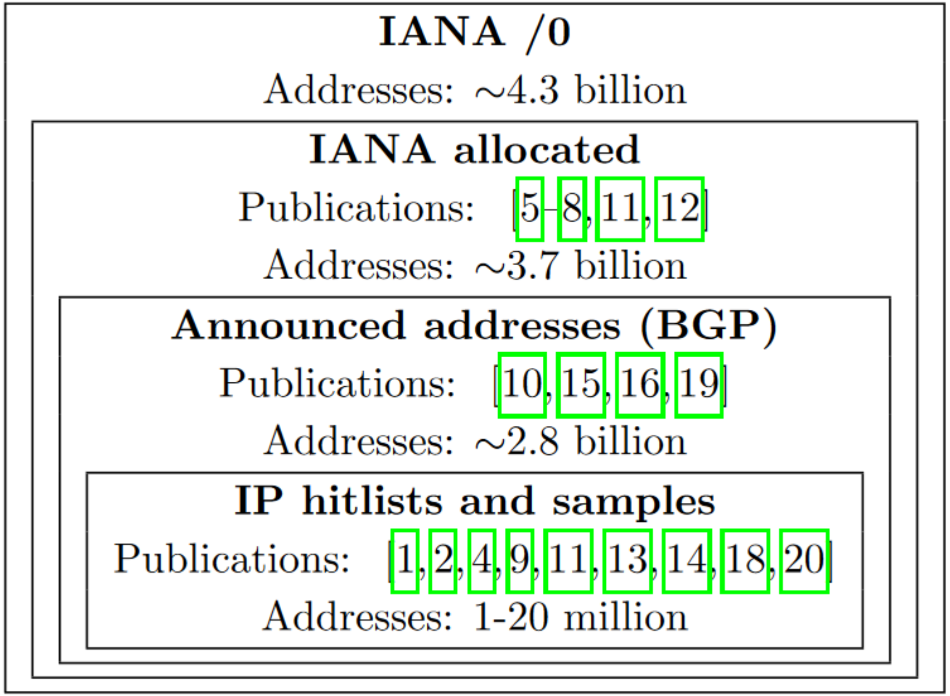
UDP service:

- UDP does not respond by itself
- Send protocol-specific payloads and match responses

This is All Rather Complex

How do we boost this to Internet scale?

Common scanning strategies



IANA /0

Addresses: ~4.3 billion

IANA allocated

Publications: [5–8, 11, 12]

Addresses: ~3.7 billion

Announced addresses (BGP)

Publications: [10, 15, 16, 19]

Addresses: ~2.8 billion

IP hitlists and samples

Publications: [1, 2, 4, 9, 11, 13, 14, 18, 20]

Addresses: 1-20 million

IP hitlists are lists of IP addresses that most likely offer the scanned services.

Challenges

Target probing

How to avoid overload of target networks?

Packet transmission

How to send packets as fast as possible?

Packet reception

How to identify valid responses?

Challenges

Target probing

Packet transmission

Packet reception

How to avoid overload of target networks?

How to send packets as fast as possible?

How to identify valid responses?

We discuss how ZMap overcomes these challenges compared to common approaches such as nmap.

Target probing

Sending probes to targets in numerical order may easily overload destination networks

Sending probes in random order prevents this problem

How do you know which addresses you already contacted?

Target probing: An inexpensive approach

How do we randomly scan addresses without excessive states?

Core idea

1. Scan hosts according to random permutation
2. Iterate over multiplicative group of integers modulo p

Brief math excursion: Multiplicative cyclic groups

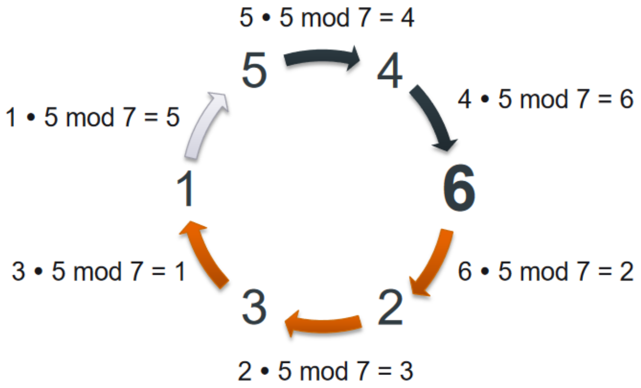
If this is a primitive root, we can iterate over all elements subsequently.

a * r mod p

Group is cyclic if p is prime. For IPv4: 2^32+15 is the smallest prime larger 2^32.

Target probing: An inexpensive approach, details

a * r mod p



Simplified example [USENIX Security 2013]

Details to generate a fresh random permutation for each scan

1. Generate a primitive
2. Choose a random starting address

Negligible state overhead to store

1. Primitive root
2. Current address
3. Starting address

Common packet transmissions

Sending packets via common socket interface introduces overhead

- Buffer creation and table updates
- Routing table lookup
- ARP cache lookup
- Potential network filters check packets
- TCP handshakes

Fast packet transmissions

Scan packets are different from typical application layer packets.

Send packets directly at the Ethernet layer and enable

Caching of Ethernet header
(except checksum header is constant)

Reduced TCP state management

Validating responses

Problems

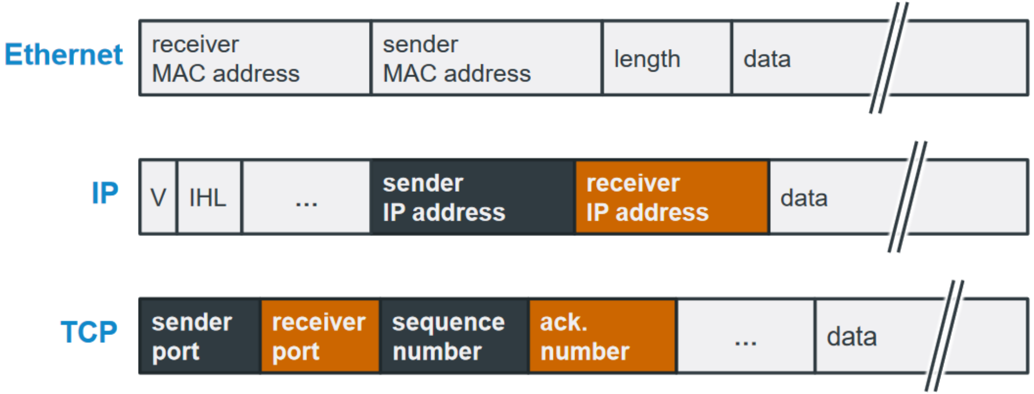
Measurement probe may see unsolicited data (other scan background traffic …)

Per-target states are expensive

Solution

Encode secrets into mutable fields of probe packets that will have recognizable effect on responses

Validating responses



Ethernet: receiver MAC address | sender MAC address | length | data

IP: V | IHL | ... | sender IP address | receiver IP address | data

TCP: sender port | receiver port | sequence number | ack. number | ... | data

Solution

Encode secrets into mutable fields of probe packets that will have recognizable effect on responses

These ideas have been implemented in ZMap

Simple network scanners

Reduce state by scanning in batches

- Time lost due to blocking
- Results lost due to timeouts

Track individual hosts and retransmit

- Most hosts will not respond

Avoid flooding through timing

- Time lost waiting

Utilize existing OS network stack

- Not optimized for immense number of connections

ZMap

Eliminate local per-connection state

- Fully asynchronous components
- No blocking except for network

Shotgun Scanning Approach

- Always send n probes per host

Scan widely dispersed targets

- Send as fast as network allows

Probe-optimized Network Stack

- Bypass inefficiencies by generating Ethernet frame

Performance of ZMap

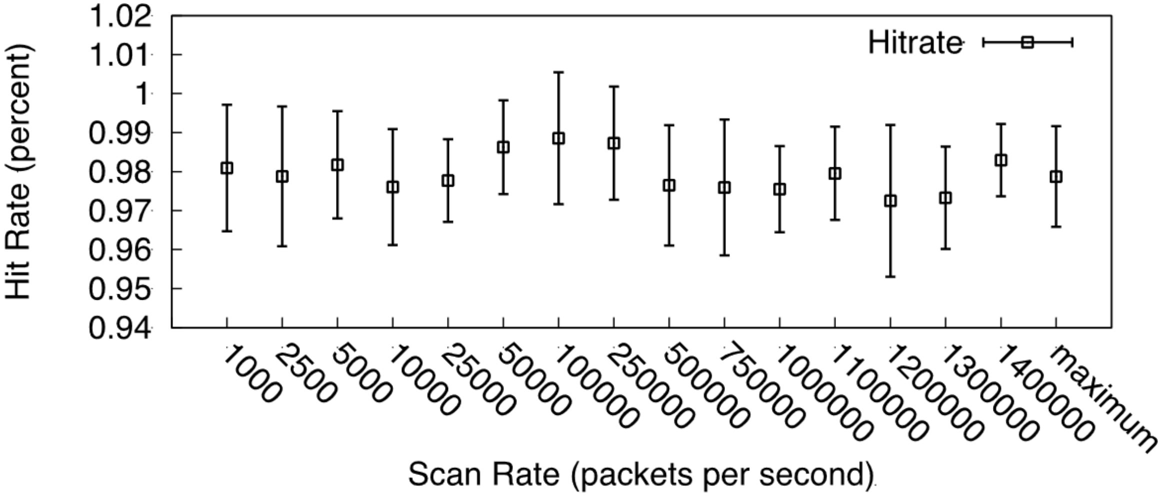
Complete scan of v4 address space takes 44 minutes with a gigabit Ethernet connection

Experiment hardware: Xeon E3-1230 3.2 GHz, 4GB RAM

Scan rate: How fast is too fast?

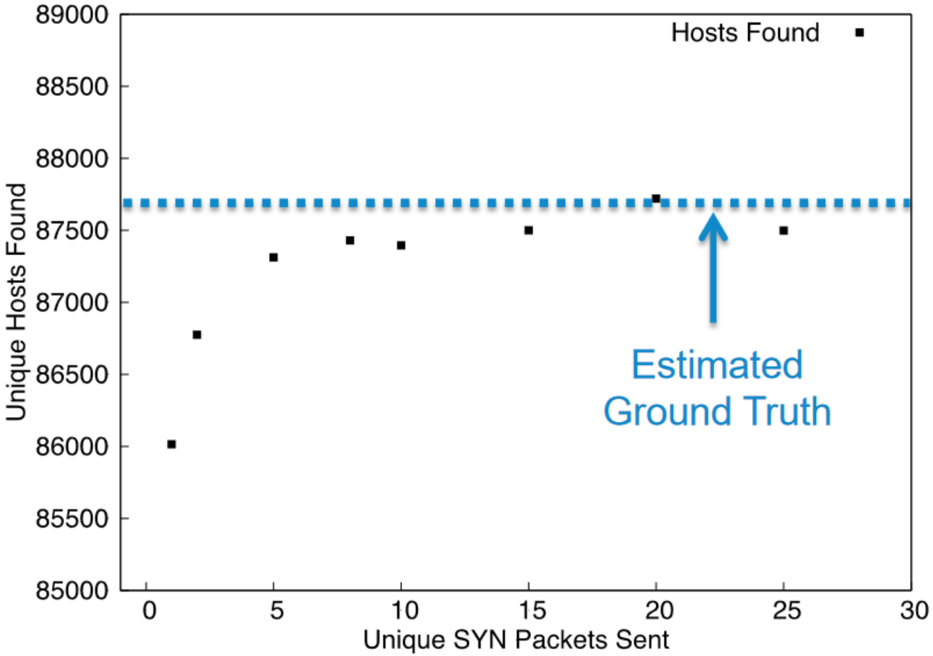
No correlation between hit-rate and scan-rate

Slower scanning does not reveal additional hosts

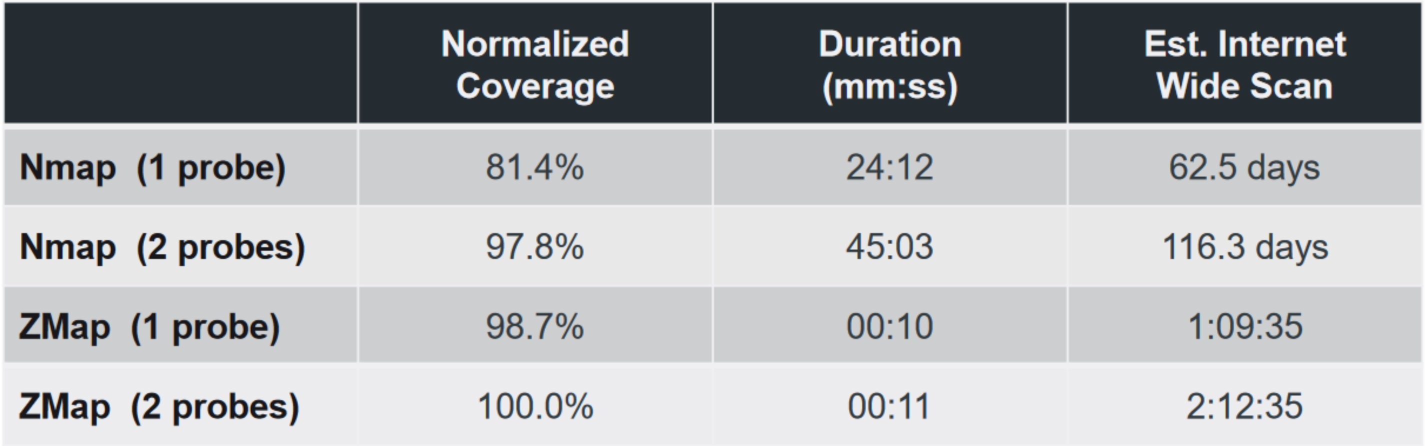


Coverage: Is one SYN enough?

Plateau approximates the real number of listening hosts.



Comparison with Nmap

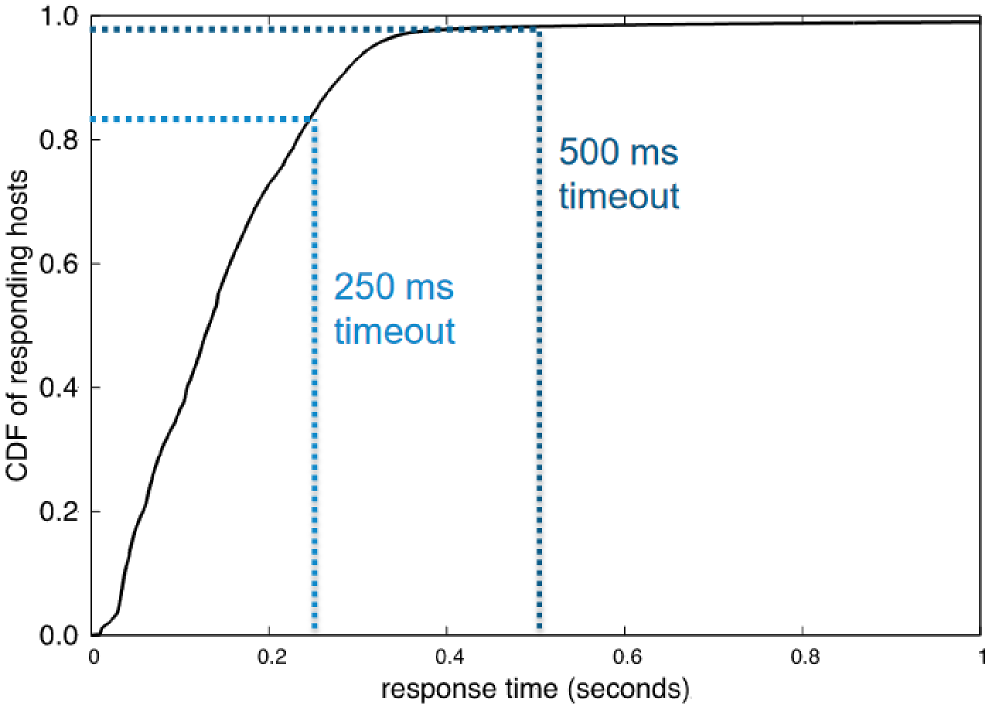


	Normalized Coverage	Duration (mm:ss)	Est. Internet Wide Scan
Nmap (1 probe)	81.4%	24:12	62.5 days
Nmap (2 probes)	97.8%	45:03	116.3 days
ZMap (1 probe)	98.7%	00:10	1:09:35
ZMap (2 probes)	100.0%	00:11	2:12:35

Averages for scanning 1 million random hosts

Why does ZMap find more hosts?

Statelessness leads to both higher performance and increased coverage.



APPLICATIONS OF HIGH-SPEED SCANNING

Enumerating vulnerable UPnP hosts

150 lines of code to perform UPnP handshake

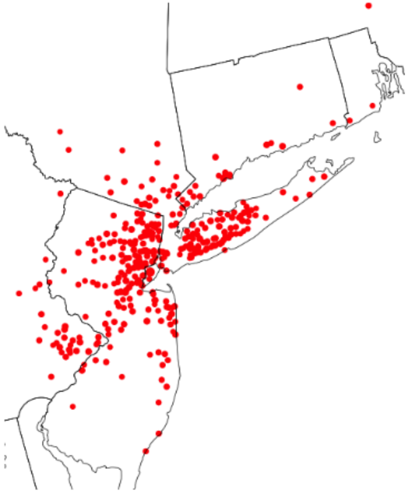
Took <2 hours to scan complete v4 addresses

HD Moore disclosed vulnerabilities in several common UPnP frameworks in January 2013
Exposure possible with a single UDP packet!

Durumeric et al. found that 3.34 M of 15.7 M devices were still vulnerable.

Think about the misuse of ZMap

Monitoring service availability

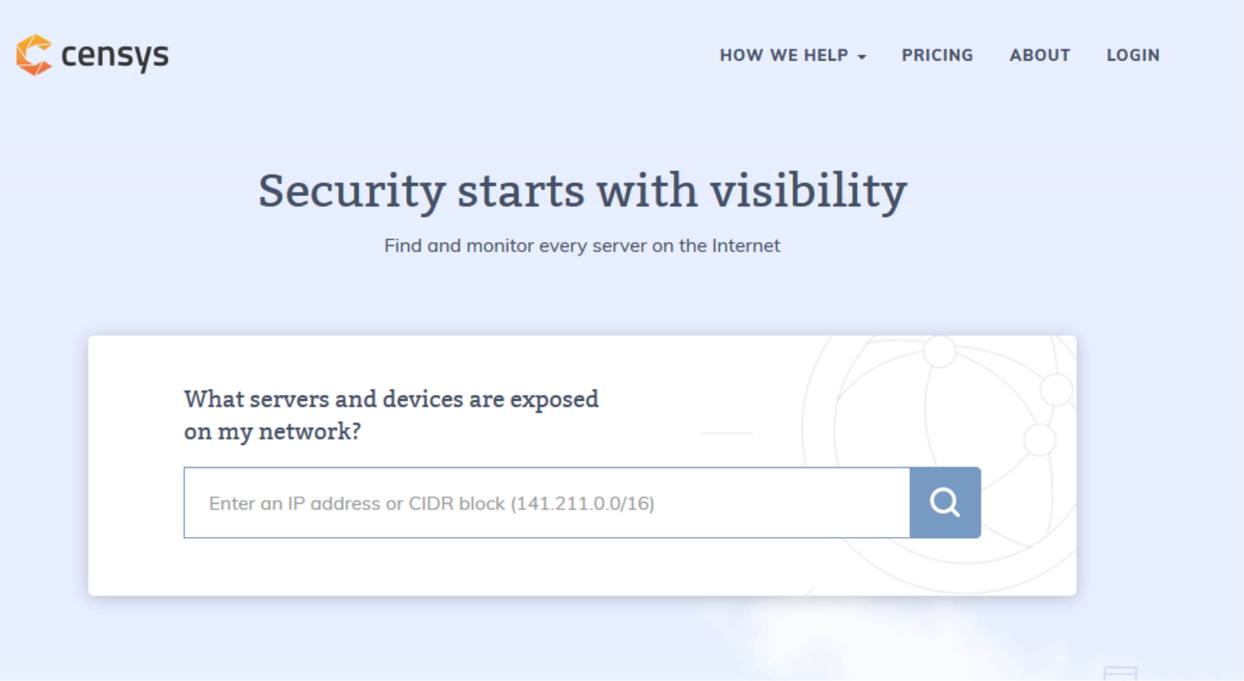


Snapshot of HTTPS outages caused by Hurricane Sandy

Specific protocol module help to identify the deployment of service

Simple ICMP echo request scans can help to track Internet outages

censys.io: Search engine that uses ZMap



Security starts with visibility

Find and monitor every server on the Internet

What servers and devices are exposed on my network?

Enter an IP address or CIDR block (141.211.0.0/16)

Literature

Zakir Durumeric, Eric Wustrow, and J. Alex Halderman: ZMap: Fast Internet-wide Scanning and Its Security Applications. In Proceedings of USENIX Security 2019, USENIX, USA, 605-620.

ZMap: Fast Internet-Wide Scanning and its Security Applications

Zakir Durumeric
University of Michigan
zakir@umich.edu

Eric Wustrow
University of Michigan
ewust@umich.edu

J. Alex Halderman
University of Michigan
jhalderm@umich.edu

Abstract

Internet-wide network scanning has numerous security applications, including exposing new vulnerabilities and tracking the adoption of defensive mechanisms, but probing the entire public address space with existing tools is both difficult and slow. We introduce ZMap, a modular, open-source network scanner specifically architected to perform Internet-wide scans and capable of surveying the entire IPv4 address space in under 45 minutes from user space on a single machine, approaching the theoretical maximum speed of gigabit Ethernet. We present the scanner architecture, experimentally characterize its performance and accuracy, and explore the security implications of high speed Internet-scale network surveys, both offensive and defensive. We also discuss best practices for good Internet citizenship when performing Internet-wide surveys, informed by our own experiences conducting a long-term research survey over the past year.

1 Introduction and Roadmap

Internet-scale network surveys collect data by probing large subsets of the public IP address space. While such scanning behavior is often associated with botnets and worms, it also has proved to be a valuable methodology for security research. Recent studies have demonstrated that Internet-wide scanning can help reveal new kinds of vulnerabilities, monitor deployment of mitigations, and shed light on previously opaque distributed ecosystems [10, 12, 14, 15, 25, 27]. Unfortunately, this methodology has been more accessible to attackers than to legitimate researchers, who cannot employ stolen network access or spread self-replicating code. Comprehensively scanning the public address space with off-the-shelf tools like Nmap [23] requires weeks of time or many machines.

In this paper, we introduce ZMap, a modular and open-source network scanner specifically designed for performing comprehensive Internet-wide research scans. A single mid-range machine running ZMap is capable of scanning for a given open port across the entire public IPv4 address space in under 45 minutes—over 97% of the theoretical maximum speed of gigabit Ethernet—without requiring specialized hardware [11] or kernel modules [8, 28]. ZMap's modular architecture can support many types of single-packet probes, including TCP SYN scans, ICMP echo request scans, and application-specific UDP scans, and it can interface easily with user-provided code to perform follow-up actions on discovered hosts, such as completing a protocol handshake.

Compared to Nmap—an excellent general-purpose network mapping tool, which was utilized in recent Internet-wide survey research [10, 14]—ZMap achieves much higher performance for Internet-scale scans. Experimentally, we find that ZMap is capable of scanning the IPv4 public address space over 1300 times faster than the most aggressive Nmap default settings, with equivalent accuracy. These performance gains are due to architectural choices that are specifically optimized for this application:

Optimized probing While Nmap adapts its transmission rate to avoid saturating the source or target networks, we assume that the source network is well provisioned (unable to be saturated by the source host), and that the targets are randomly ordered and widely dispersed (so no distant network or path is likely to be saturated by the scan). Consequently, we attempt to send probes as quickly as the source's NIC can support, skipping the TCP/IP stack and generating Ethernet frames directly. We show that ZMap can send probes at gigabit line speed from commodity hardware and entirely in user space.

No per-connection state While Nmap maintains state for each connection to track which hosts have been scanned and to handle timeouts and retransmissions, ZMap forgoes any per-connection state. Since it is intended to target random samples of the address space, ZMap can avoid storing the addresses it has already scanned or needs to scan and instead selects addresses according to a random permutation generated by a cyclic

USENIX Association

22nd USENIX Security Symposium 605

Making it even leaner

REDUCING THE FOOTPRINT OF INTERNET-WIDE SCANS

Problems of Internet-wide scans

Scan packets are overhead

Abuse reports

Threats of legal action

Impact on research results by

Load on intrusion detection systems

IP Blacklisting

Rate limiting by routers

IP hitlists vs announced addresses (BGP)

Announced addresses (BGP)

High scan overhead

Results: stable over time

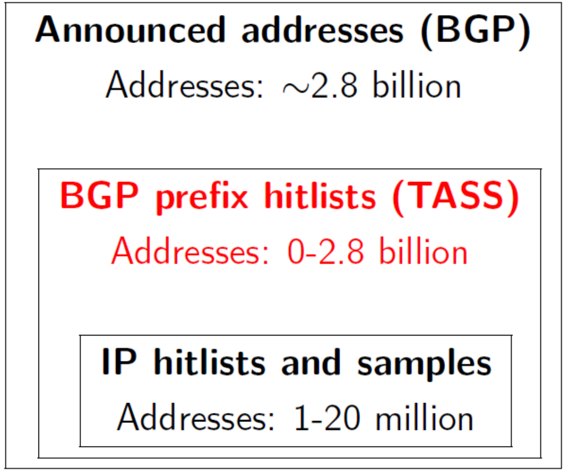
IP hitlists

Low scan overhead

Results: unstable over time (dynamic IPs)

Can we do better?

Idea: Topology Aware Scanning Strategy (TASS)



Announced addresses (BGP)

Addresses: ~2.8 billion

BGP prefix hitlists (TASS)

Addresses: 0-2.8 billion

IP hitlists and samples

Addresses: 1-20 million

Hypothesis

Hosts with dynamic IP addresses do not often change their announced BGP network prefix.

TASS approach

1. Perform a full IPv4 scan once

2. Get, sort, and select prefixes by their host density until desired host coverage has been reached

3. Scan only the selected prefixes for a given time period

May reduce scan traffic by 35-90 % and miss only 1-10 % service responses

Step 1: Perform a full IPv4 scan once

Use data from existing scan projects, e.g., censys.io

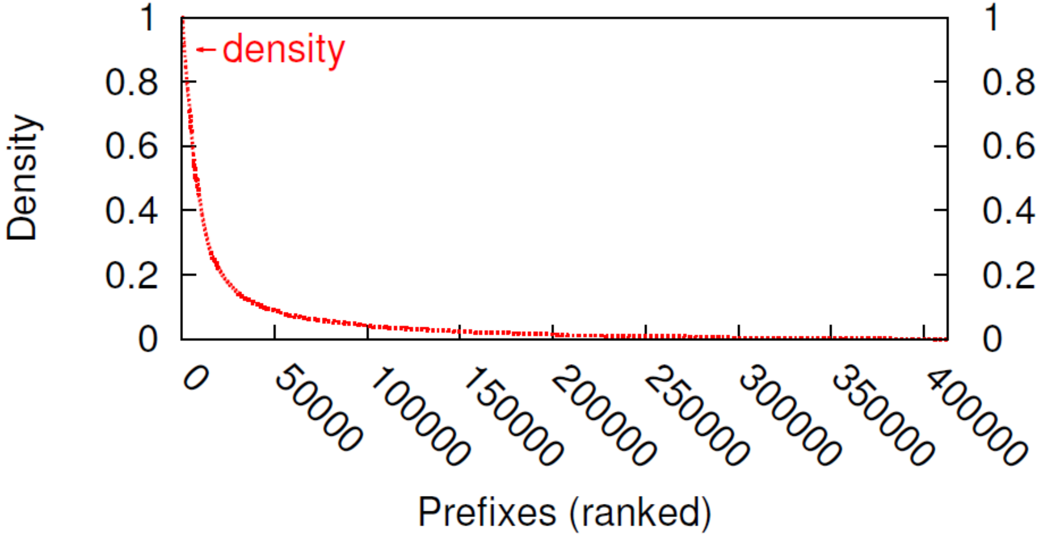
Following results show IPv4 scan data from Censys.io: HTTP(S), FTP, CWMP (CPE WAN Management Protocol), 09/2015 to 03/2016

Step 2: Get and Sort prefixes (HTTPS)

Prefixes obtained by CAIDA Routeviews Prefix-to-AS database + some own optimizations

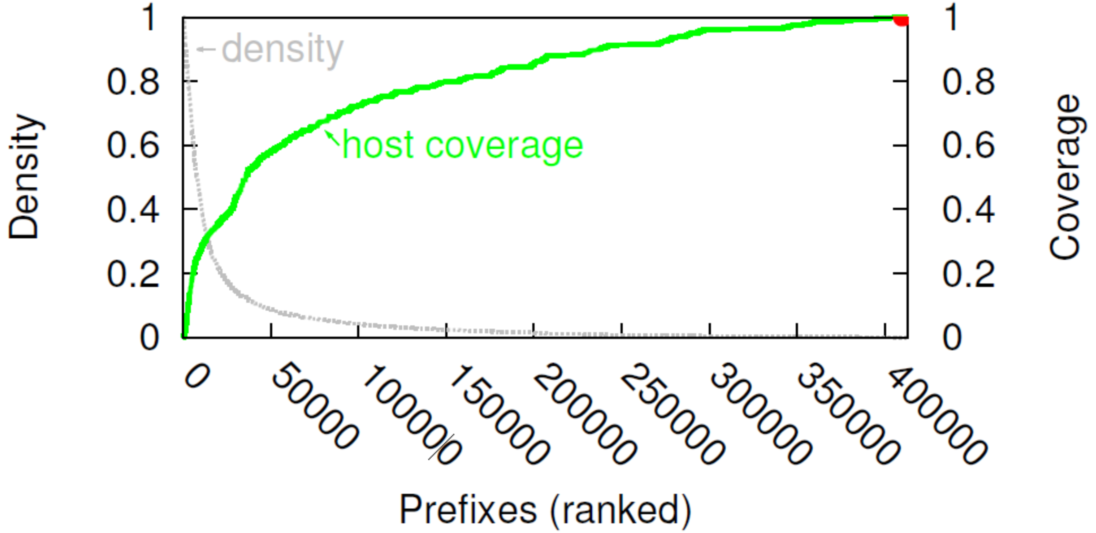
Host **density** = **#hosts** divided by **#IP addresses** contained by the prefix

Prefixes sorted by their density



Step 2: Select prefixes (HTTPS)

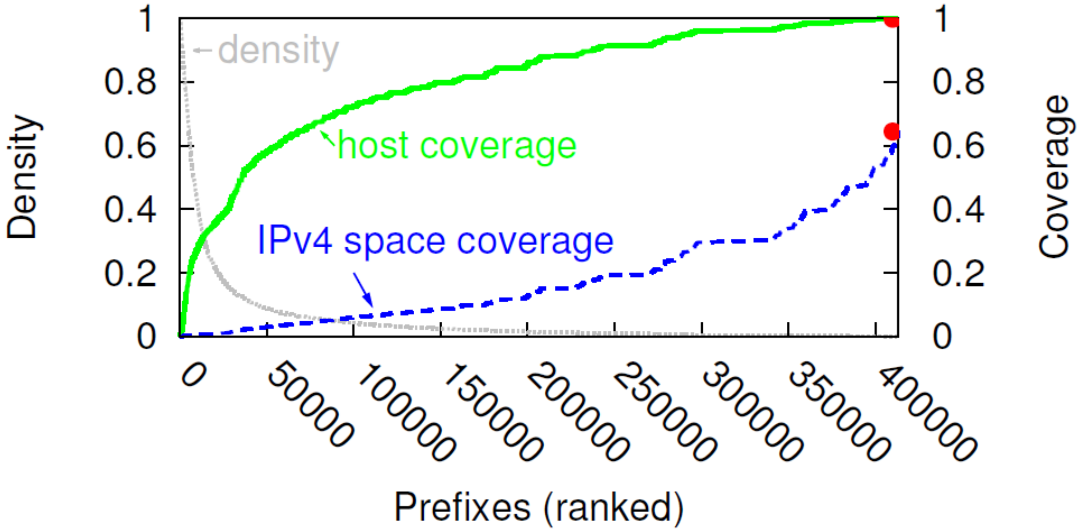
100 % of the HTTPS host are distributed over 410,000 prefixes.



Step 2: Select prefixes (HTTPS)

Select all prefixes with density > 0

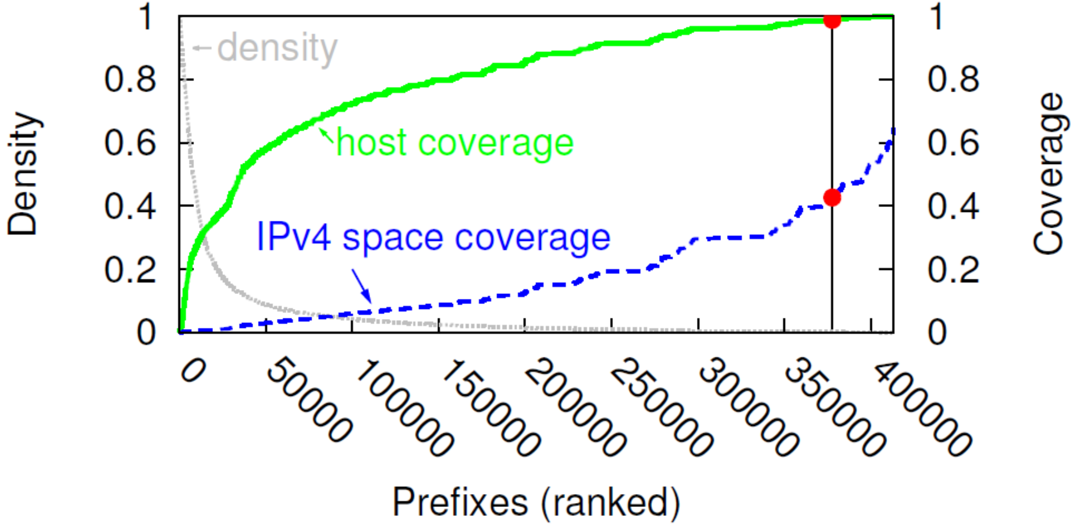
Scanning 100 % of the HTTPS host results in a IPv4 address space coverage of 64,5 %.



Step 2: Select prefixes (HTTPS)

Scanning **99%** of all HTTPS hosts results in an address space coverage of only **42,7%**

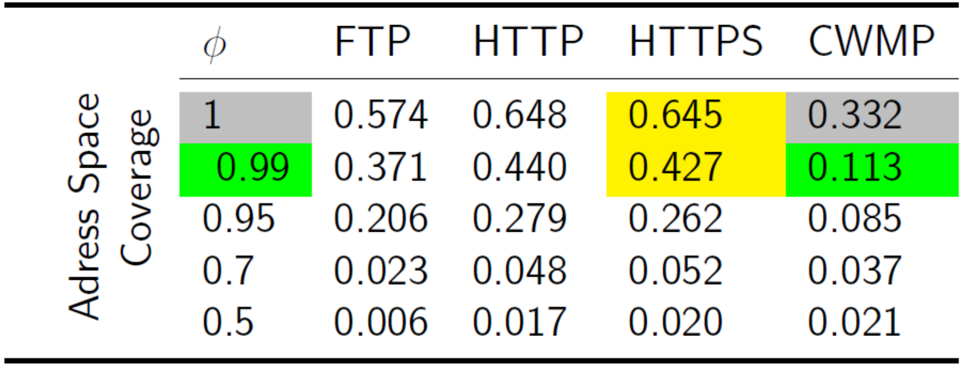
Skipping some prefixes with the lowest density



Host Coverage vs. IPv4 Space Coverage

Little tweaks on the host coverage have an important impact on the needed address space coverage

Host / address space coverage ratio depends on the protocol.



	ϕ	FTP	HTTP	HTTPS	CWMP
Adress Space Coverage	1	0.574	0.648	0.645	0.332
	0.99	0.371	0.440	0.427	0.113
	0.95	0.206	0.279	0.262	0.085
	0.7	0.023	0.048	0.052	0.037
	0.5	0.006	0.017	0.020	0.021

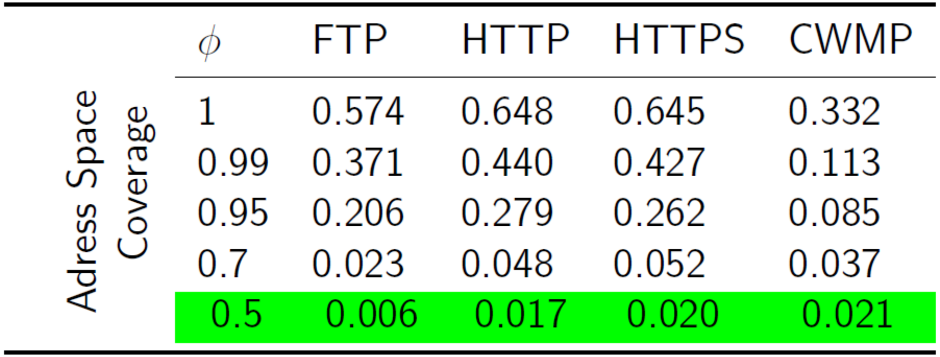
Host coverage

IPv4 space coverage

Host Coverage vs. IPv4 Space Coverage

We are able to scan every second host by scanning just 2% of the announced IPv4 address space!

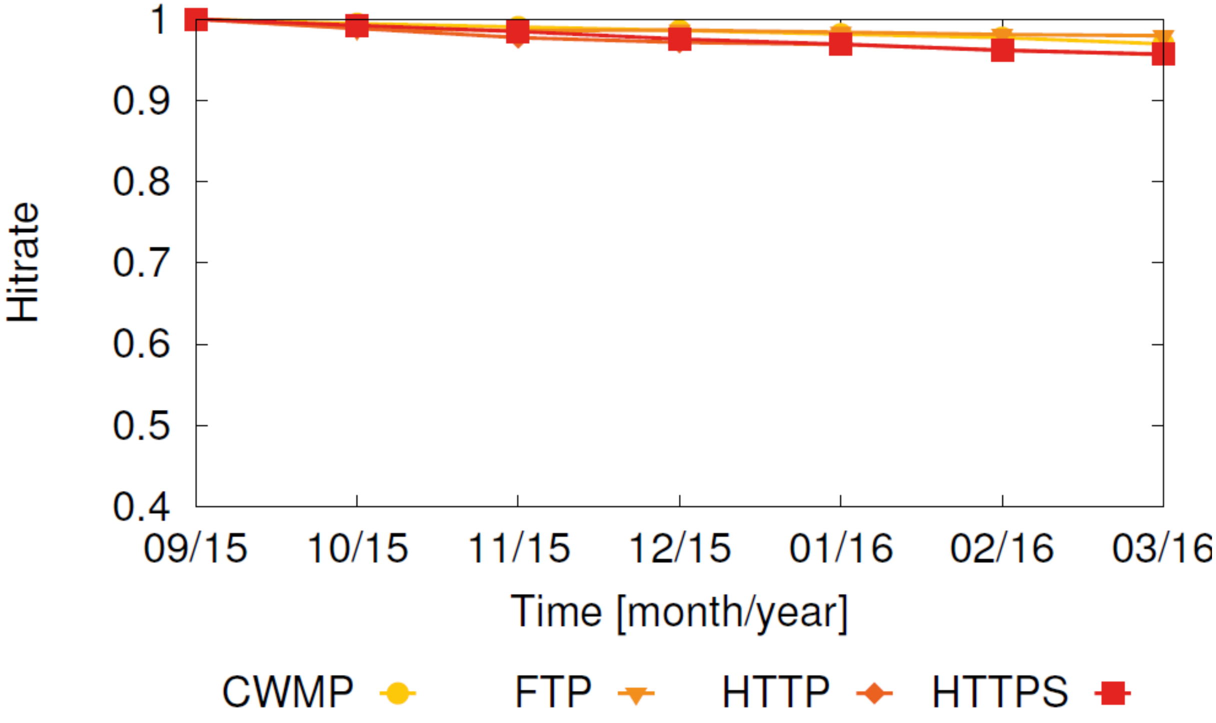
This results in a scan traffic reduction of 98 % compared to an IPv4 full scan.



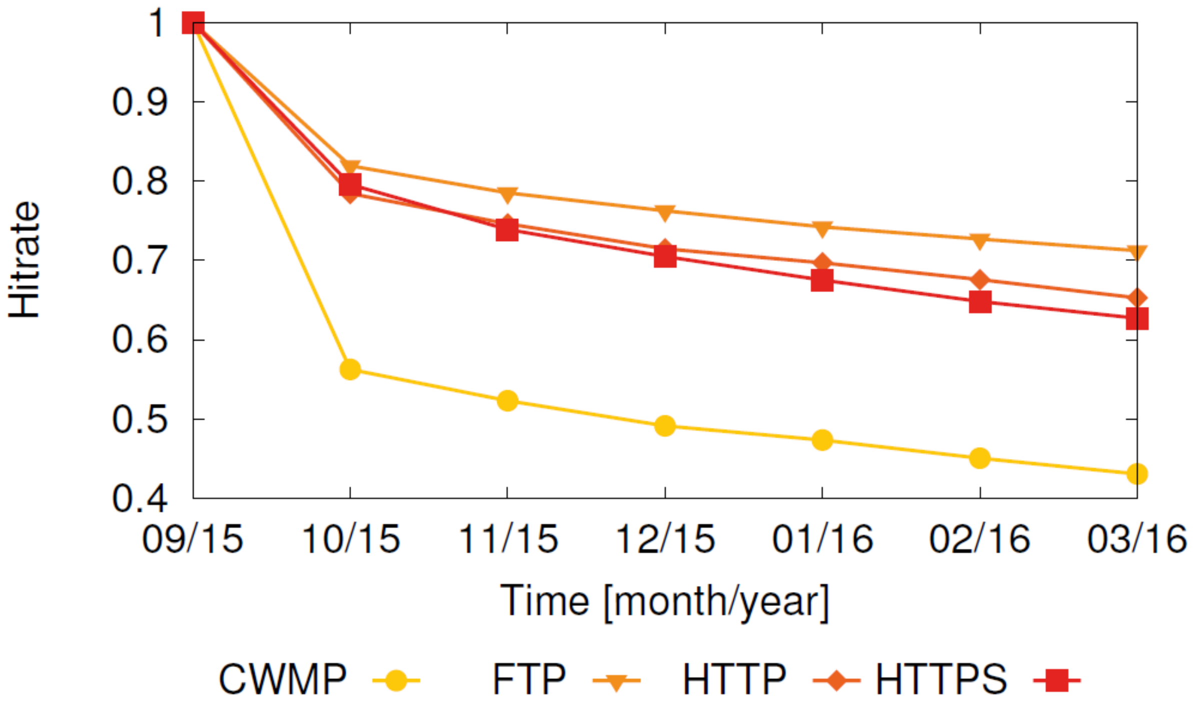
Adress Space Coverage	ϕ	FTP	HTTP	HTTPS	CWMP
	1	0.574	0.648	0.645	0.332
	0.99	0.371	0.440	0.427	0.113
	0.95	0.206	0.279	0.262	0.085
	0.7	0.023	0.048	0.052	0.037
	0.5	0.006	0.017	0.020	0.021

TASS compared to a IPv4 full scan (density = 1)

After six months, TASS finds only 4% less hosts than a IPv4 full scan



After six months, IP hitlists finds 30-55% less hosts than an IPv4 full scan.



Literature

Johannes Klick, Stephan Lau, Matthias Wählisch, and Volker Roth. 2016. Towards Better Internet Citizenship: Reducing the Footprint of Internet-wide Scans by Topology Aware Prefix Selection. In *Proceedings of the 2016 Internet Measurement Conference* (IMC '16). ACM, New York, NY, USA, 421-427. DOI: https://doi.org/10.1145/2987443.2987457

Towards Better Internet Citizenship: Reducing the Footprint of Internet-wide Scans by Topology Aware Prefix Selection

Johannes Klick
Freie Universität Berlin
johannes.klick@fu-berlin.de

Stephan Lau
Freie Universität Berlin
stephan.lau@fu-berlin.de

Matthias Wählisch
Freie Universität Berlin
m.waehlisch@fu-berlin.de

Volker Roth
Freie Universität Berlin
volker.roth@fu-berlin.de

ABSTRACT

Internet service discovery is an emerging topic to study the deployment of protocols. Towards this end, our community periodically scans the entire advertised IPv4 address space. In this paper, we question this principle. Being good Internet citizens means that we should limit scan traffic to what is necessary. We conducted a study of scan data, which shows that several prefixes do not accommodate any host of interest and the network topology is fairly stable. We argue that this allows us to collect representative data by scanning less. In our paper, we explore the idea to scan all prefixes once and then identify prefixes of interest for future scanning.

Based on our analysis of the *censys.io* data set (4.1 TB data encompassing 28 full IPv4 scans within 6 months) we found that we can reduce scan traffic between 25-90% and miss only 1-10% of the hosts, depending on desired trade-offs and protocols.

1. INTRODUCTION

Fast Internet-wide scanning is growing in popularity among researchers. At the time of writing, researchers regularly scan the Internet for vulnerable SSL certificates [6, 12], SSH public keys [10], and for the banners of plain text protocols such as SMTP, HTTP, FTP, and Telnet [5]. The majority of researchers scan at least 2.8 billion addresses advertised in the IPv4 address space [5–8, 10–12, 15, 16, 19]. Hitrates, the fraction of probed addresses from which a response is received, are very often under two percent [7]. This means that most scan traffic is overhead. Most of these scans are done periodically for trend analyses, which exacerbates the amount of unnecessary scan traffic. For example, the ongoing Internet-wide research project *censys.io* [5, 7] probes the IANA allocated address space for 19 protocols on a continuous basis. This results in 72.2 billion generated IP-packets per week. which causes several hostile responses ranging from threatening legal actions to conducted denial-of-service attacks [7]. Whereas scanning the IPv4 address space is feasible this is not any more the case for IPv6. When IPv6 becomes more popular, we need scanning strategies that limit scans to parts of the address space that are in use.

IMC 2016, November 14 - 16, 2016, Santa Monica, CA, USA

ISBN 978-1-4503-4526-2/16/11 . . . $15.00

DOI: http://dx.doi.org/10.1145/2987443.2987457

Many measurement scenarios require only partial scans instead of exploring the full IP address space. However, we currently lack a systematic understanding of the deployment of Internet services with respect to IP address ranges.

In this paper, we want to start the discussion how we can reduce scan traffic systematically. We present the *Topology Aware Scanning Strategy (TASS)*, a new IP prefix based and topology aware scanning strategy for periodic scanning. *TASS* enables researchers to collect responses from 90-99% of the available hosts for six months by scanning only 10-75% of the announced IPv4 address space in each scan cycle (protocol dependent). *TASS* is seeded with the results of a full advertised IPv4 address scan for a given protocol and time period. The prefixes for all responses will be selected for periodic scans of the given protocol.

Periodic scanning of only selected prefixes reduces scan traffic significantly while hitting most of the hosts of interest. For instance, our analysis reveals that responsive prefixes obtained from a full FTP scan cover

The Bigger Network

HOW TO SCAN IPV6

2^32 IPv4 addresses scanned in 44 minutes

1,7*10^-10 seconds per address

2^32 IPv4 addresses scanned in 44 minutes
1,7*10^-10 seconds per address

2^128 IPv6 addresses scanned in ??

We want to scan the IP address space

Easy. Really?

We will not be able to scan every IPv6 address!

2^128 IPv6 addresses scanned in ??

Approaches to find active IPv6 addresses

DNS (DB) techniques

Structural properties

Combined Hitlists

Crowd-sourcing

DNS techniques based on reverse IPv4 DNS

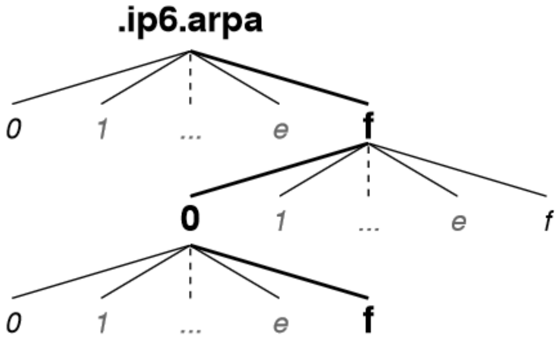
Limited to finding Dual Stack Hosts

Derive v4 addresses from passive BGP measurements

Query reverse DNS entry for all these addresses

Query AAAA (IPv6) record for responses

DNS techniques based on reverse IPv6 DNS



Leverage non-existent domain name record (NXDOMAIN)

There are no entries under this DNS subtree

Enumerate the reverse IPv6 DNS tree and ignore complete subtrees if NXDOMAIN replied

Challenges: Scaling, non-standard compliant servers …

Structural properties

Apply machine learning on IPv6 input data set to identify address plans

Find dense regions in the v6 address space and generate neighboring addresses, based on input addresses

Calculate Hamming distance on granularity of nybbles (= 4 bit of hex character in IPv6 addresses)

Combined Hitlists

Passive

Flow data of large networks

Active

Alexa Top 1M

Rapid7 IPv4 rDNS

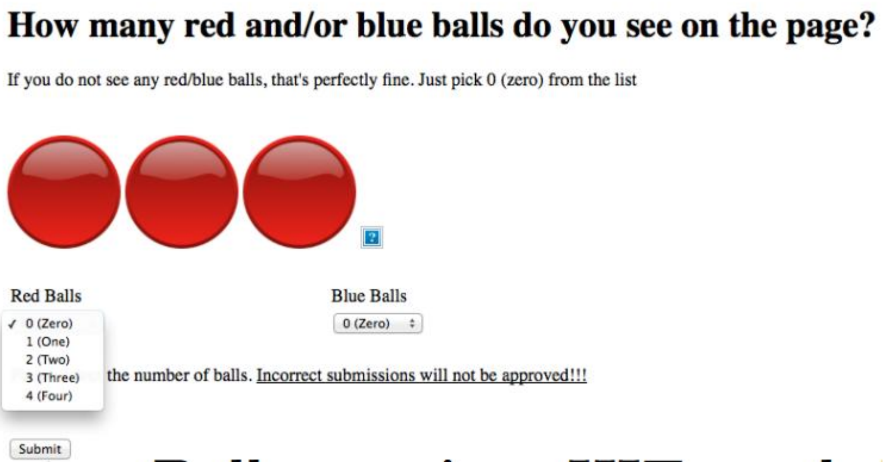
Rapid7 DNS ANY

DNS zone files

CAIDA IPv6 router DNS names

Traceroute

Crowdsourcing



How many red and/or blue balls do you see on the page?

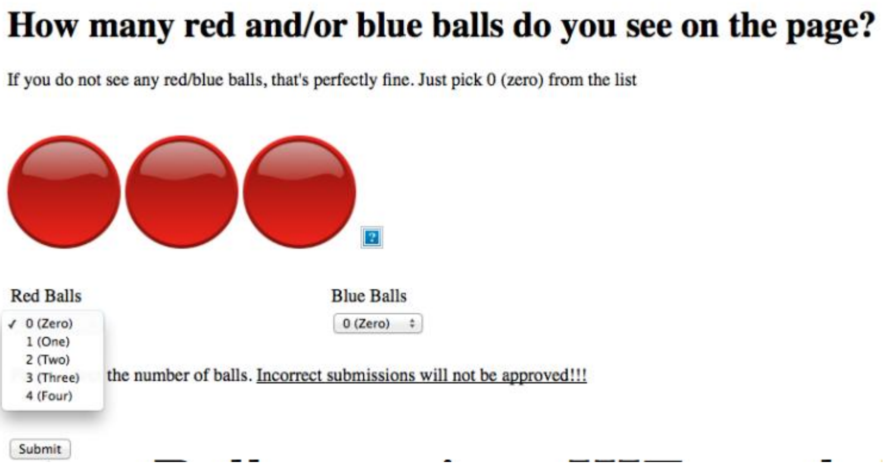
If you do not see any red/blue balls, that's perfectly fine. Just pick 0 (zero) from the list

Red Balls

Blue Balls: 0 (Zero)

the number of balls. Incorrect submissions will not be approved!!!

Crowdsourcing



How many red and/or blue balls do you see on the page?

If you do not see any red/blue balls, that's perfectly fine. Just pick 0 (zero) from the list

Red Balls

✓ 0 (Zero)
1 (One)
2 (Two)
3 (Three)
4 (Four)

Blue Balls

0 (Zero)

the number of balls. Incorrect submissions will not be approved!!!

Blue balls are only served by an IPv6-enabled server

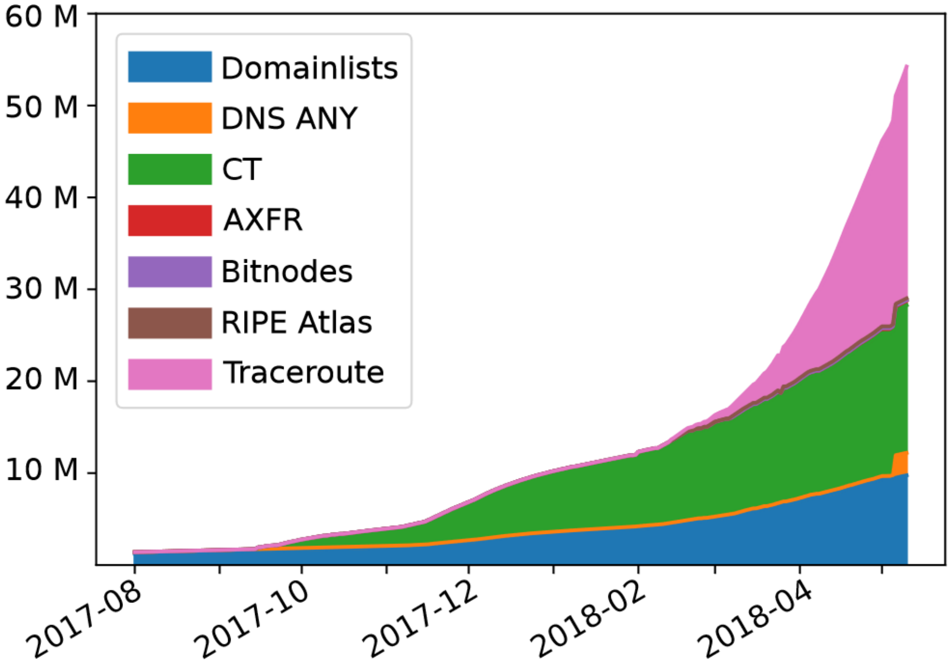
Inspect server logs to measure host addresses

Looking at the entire IPv6 node space

How biased are sources of IPv6 addresses?

Cumulative increase of v6 addresses

Strong increase of traceroute due to home routers



Understanding traceroute grow in more detail

`…::ff:fe:…`

Indicates SLAAC addresses

Roughly, split 48 bit MAC address into two 24 bit blocks, separated by ff:fe

(Privacy extensions exist …)

Understanding traceroute grow in more detail

…::ff:fe:…

90% were SLAAC addresses

47% ZTE

47% AVM

1% Huawei

+ long tail of 240 other vendors

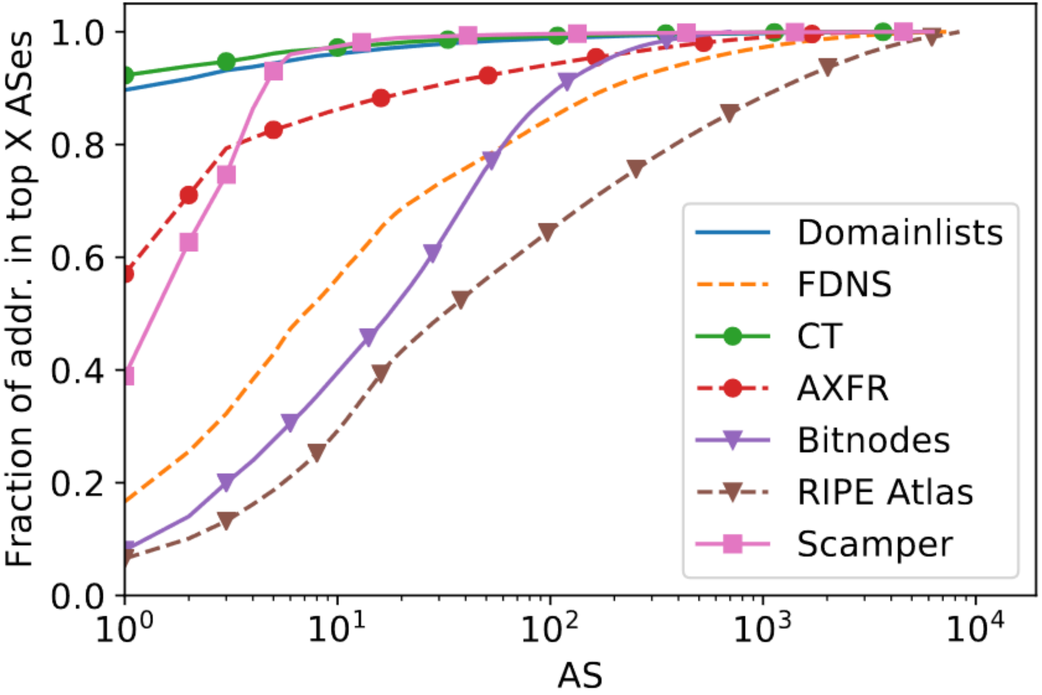
Indicates SLAAC addresses

Roughly, split 48 bit MAC address into two 24 bit blocks, separated by ff:fe

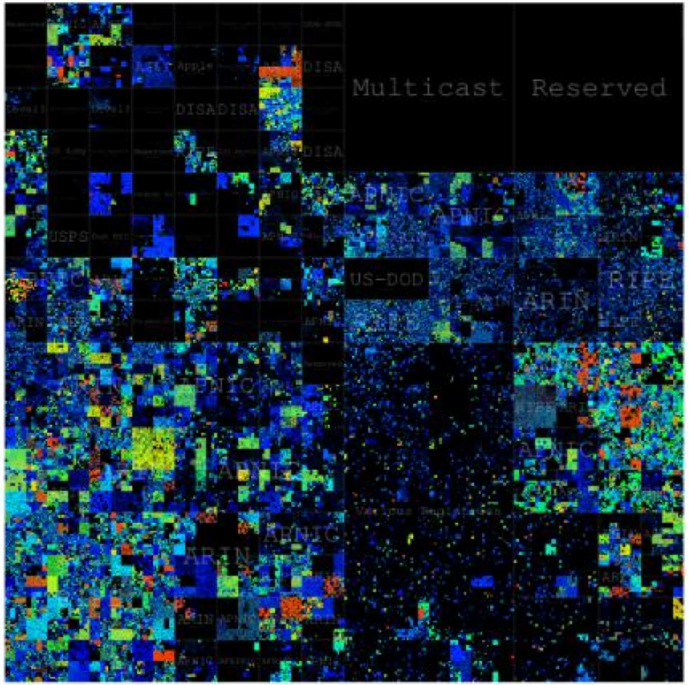
(Privacy extensions exist …)

Do the sources cover many ASes?

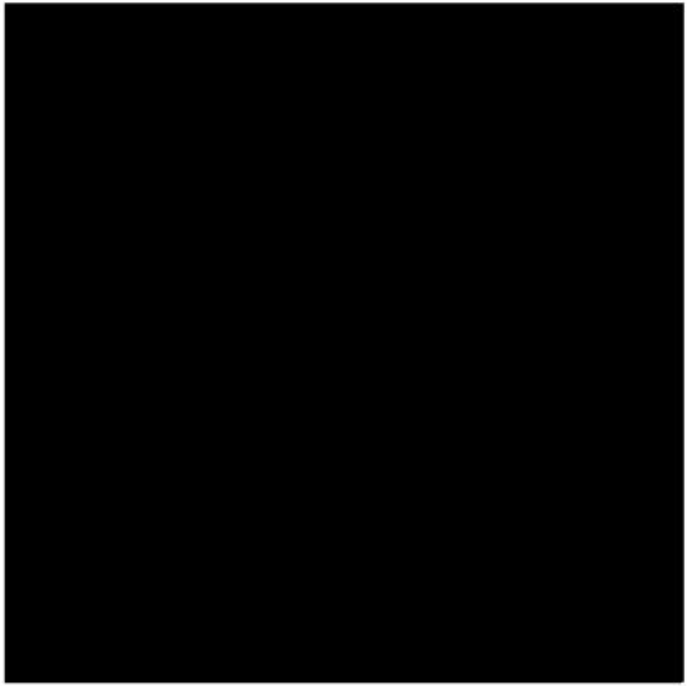
Unbalanced (CT, domain lists)
vs. balanced (RIPE Atlas)



Visualizing IP address space



IPv4



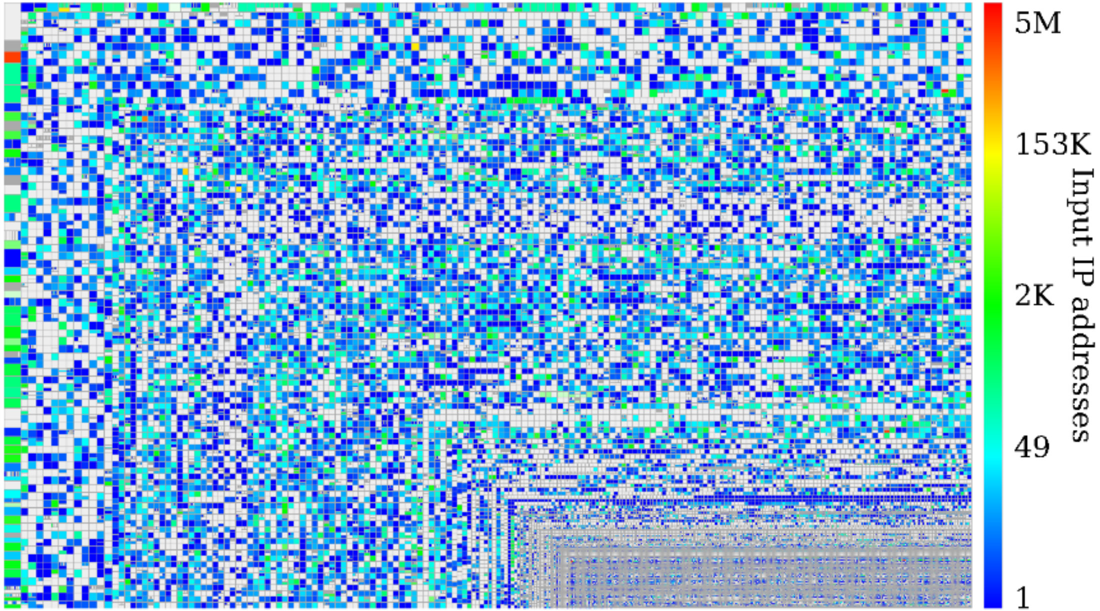
IPv6

zesplot: Visualizing v6 announced address space

IPv6 prefixes represented as a rectangle

Order prefixes by {prefix-size, ASN}

Start by filling vertical row, then horizontal row, then vertical row etc.



Some prefixes contain unusually large numbers of addresses. Why?

Challenge: Aliased network prefixes

Complete prefix is assigned to a host

Host listens on all possible addresses

Consequence

Artificial inflation of hitlists

Some hosts will over-represent the hitlist

Alias detection: Fixed prefix length

Assumption

It is unlikely that a randomly selected IPv6 address replies

Approach

Construct medium-sized prefixes (e.g., /96)

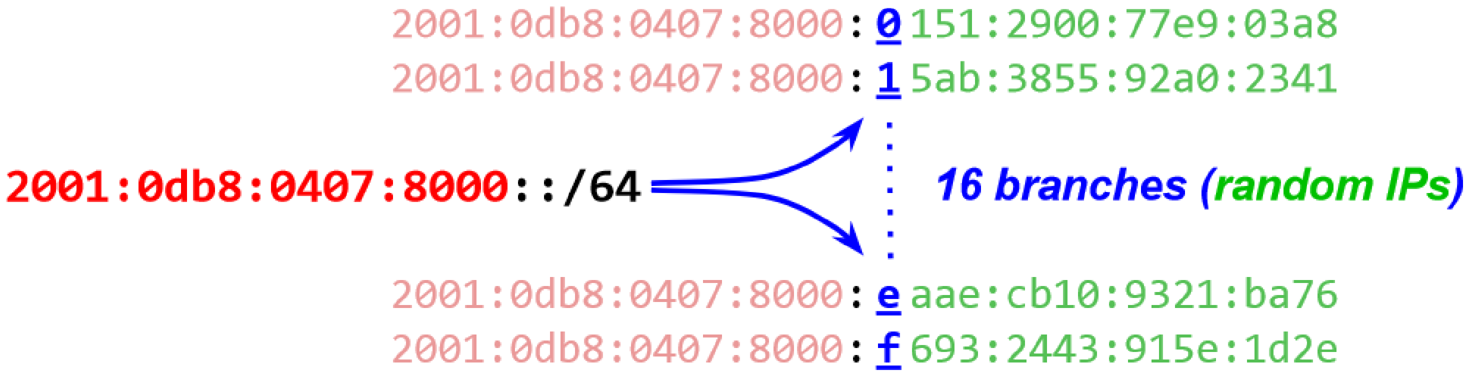
Send probes to n randomly selected addresses in the prefixes

If you receive n replies, likely because of aliased prefix

Alias detection: Dynamic prefix length

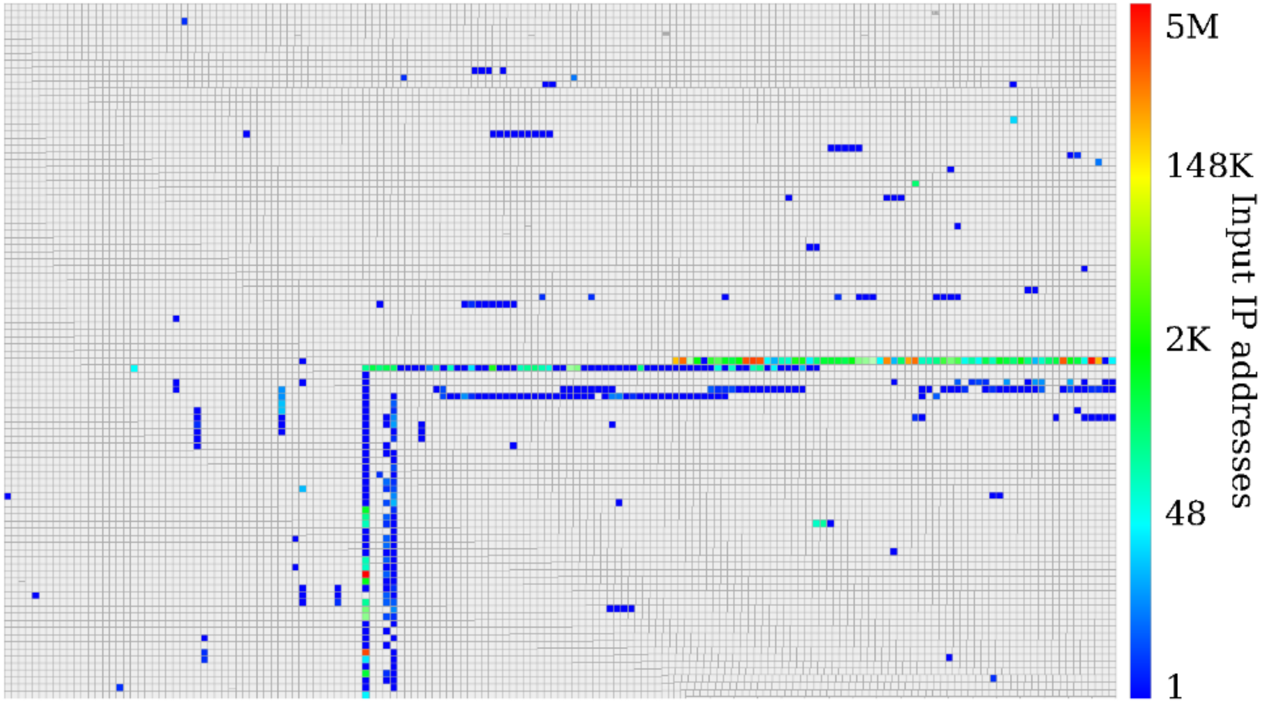
Detection at different prefix lengths

Generate pseudo-random address for each 4-bit sub-prefix



```
                    2001:0db8:0407:8000:0151:2900:77e9:03a8
                    2001:0db8:0407:8000:15ab:3855:92a0:2341
                                        :
2001:0db8:0407:8000::/64                :   16 branches (random IPs)
                                        :
                    2001:0db8:0407:8000:eaae:cb10:9321:ba76
                    2001:0db8:0407:8000:f693:2443:915e:1d2e
```

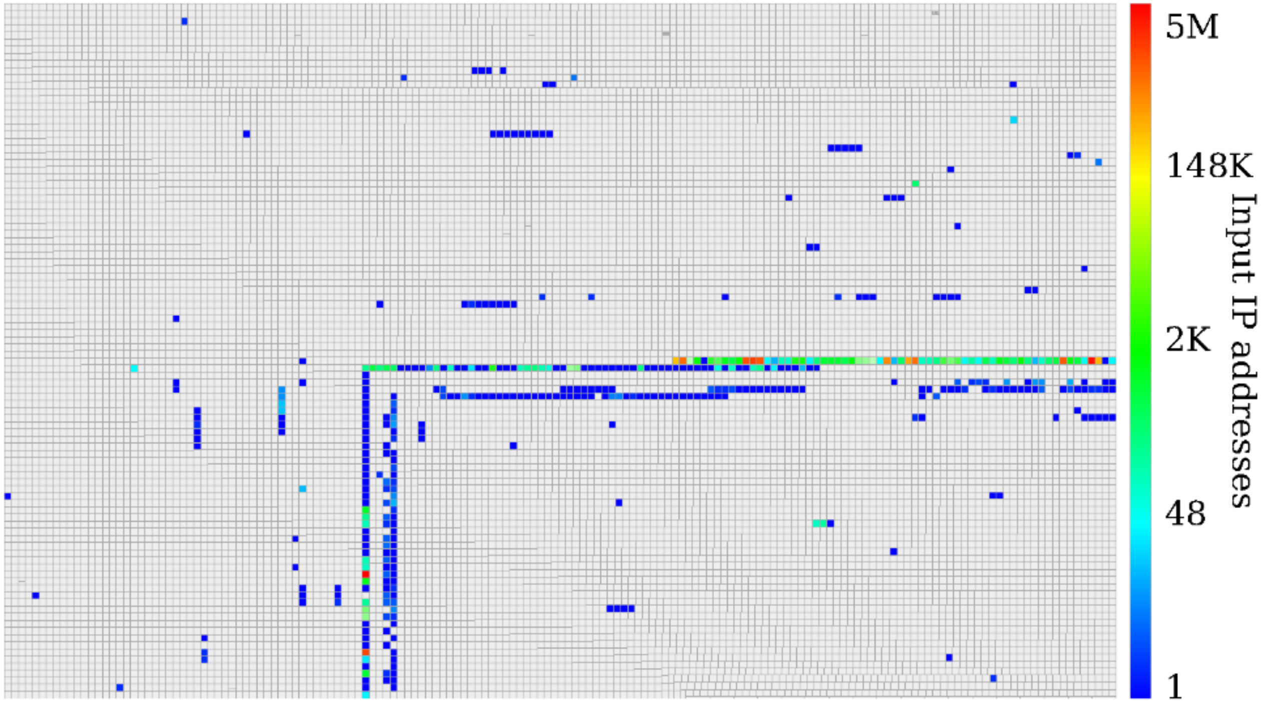
Detected aliased prefixes

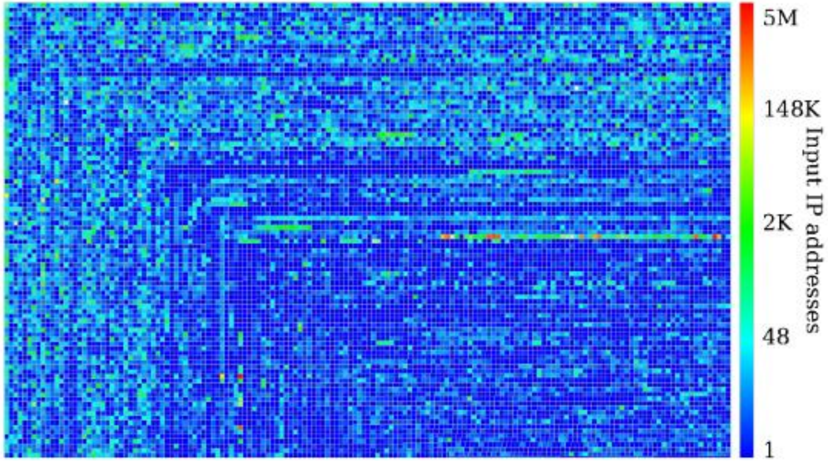


Detected aliased prefixes

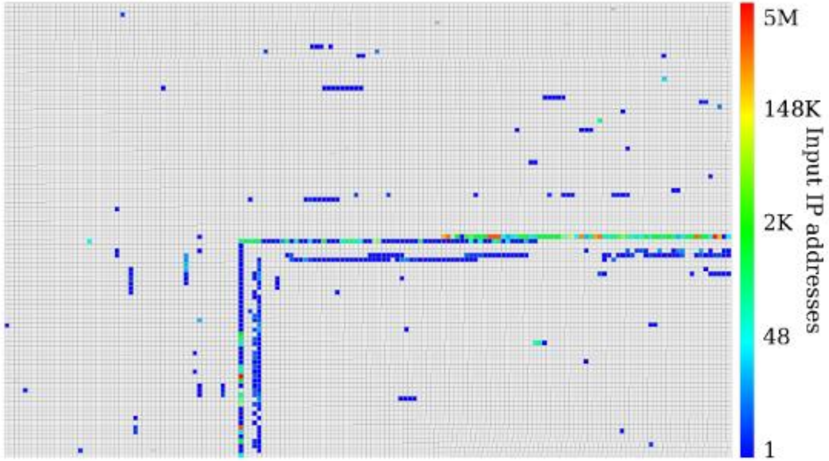
All /48 prefixes

Majority belongs to Amazon and Incapsula (both cloud providers)





All prefixes covered by hitlist



Aliased prefixes

Can we identify common addressing schemes in hitlists?

Techniques to learn new addresses

Entropy/IP

- Generate new addresses by leveraging entropy of seed addresses
  - Similar approach to grouping addresses based on their structure as shown earlier

6Gen

- Generate new addresses in dense address regions
  - If we see addresses
    - `2001:0db8:0407:8000::4`
    - `2001:0db8:0407:8000::5`
    - `2001:0db8:0407:8000::8`
- Likely other valid addresses
  - `2001:0db8:0407:8000::6`
  - `2001:0db8:0407:8000::7`

Entropy clustering

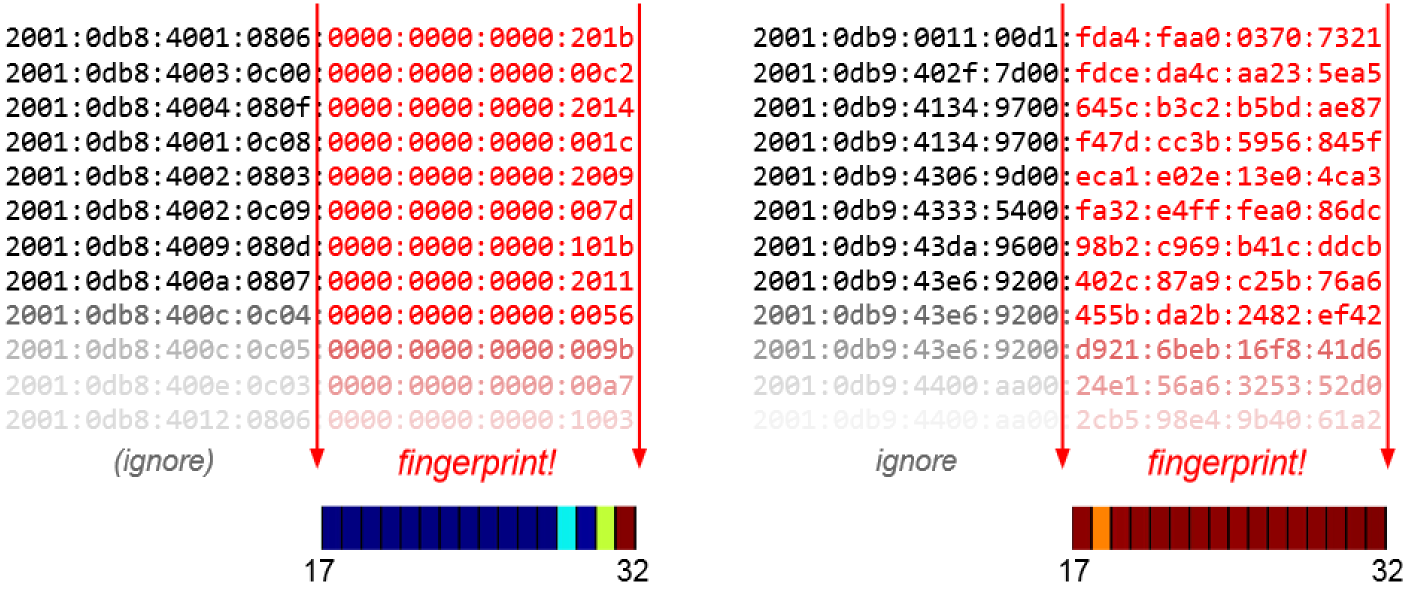
Take a set of responsive IPv6 addresses from a particular network (e.g., /32 prefix, a prefix from BGP dumps, or an AS)

Calculate the normalized Shannon entropy for each IPv6 nybble (4 bits = one hex char) for all addresses in the set; repeat for each network

Use these fingerprints as input for k-means clustering to predict more responsive addresses

Plot median fingerprints and cluster popularity

Entropy clustering

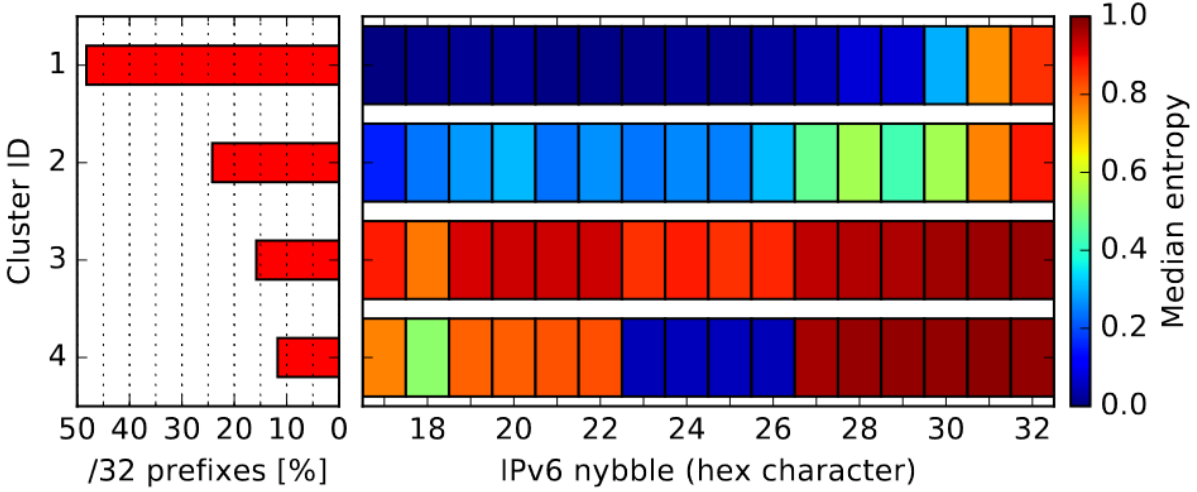


```
2001:0db8:4001:0806:0000:0000:0000:201b
2001:0db8:4003:0c00:0000:0000:0000:00c2
2001:0db8:4004:080f:0000:0000:0000:2014
2001:0db8:4001:0c08:0000:0000:0000:001c
2001:0db8:4002:0803:0000:0000:0000:2009
2001:0db8:4002:0c09:0000:0000:0000:007d
2001:0db8:4009:080d:0000:0000:0000:101b
2001:0db8:400a:0807:0000:0000:0000:2011
2001:0db8:400c:0c04:0000:0000:0000:0056
2001:0db8:400c:0c05:0000:0000:0000:009b
2001:0db8:400e:0c03:0000:0000:0000:00a7
2001:0db8:4012:0806:0000:0000:0000:1003
```

```
2001:0db9:0011:00d1:fda4:faa0:0370:7321
2001:0db9:402f:7d00:fdce:da4c:aa23:5ea5
2001:0db9:4134:9700:645c:b3c2:b5bd:ae87
2001:0db9:4134:9700:f47d:cc3b:5956:845f
2001:0db9:4306:9d00:eca1:e02e:13e0:4ca3
2001:0db9:4333:5400:fa32:e4ff:fea0:86dc
2001:0db9:43da:9600:98b2:c969:b41c:ddcb
2001:0db9:43e6:9200:402c:87a9:c25b:76a6
2001:0db9:43e6:9200:455b:da2b:2482:ef42
2001:0db9:43e6:9200:d921:6beb:16f8:41d6
2001:0db9:4400:aa00:24e1:56a6:3253:52d0
2001:0db9:4400:aa00:2cb5:98e4:9b40:61a2
```

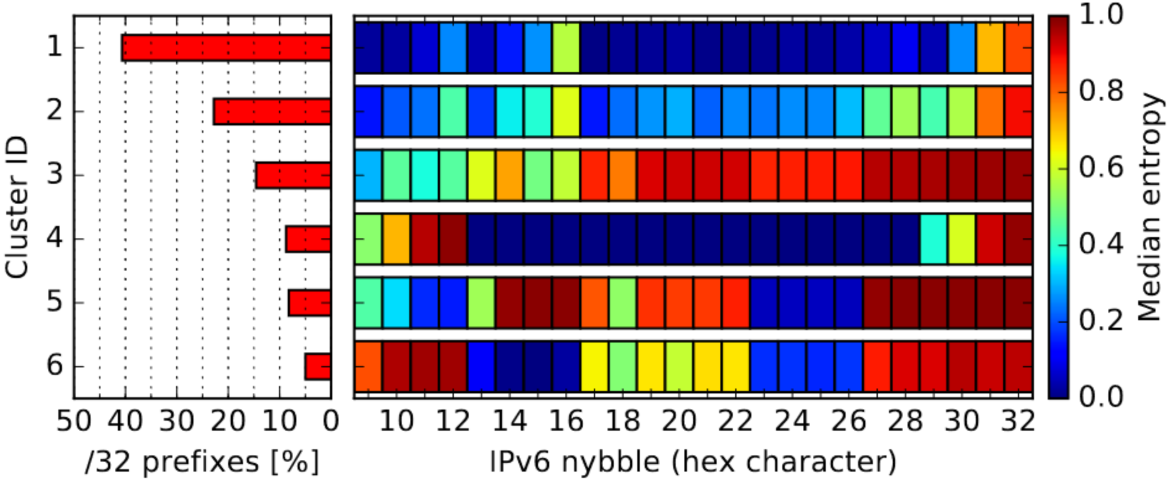
Entropy clustering of /32 prefixes (consider only interface identifiers)

Fingerprint is only based on nybbles 17-32



Entropy clustering of /32 prefixes (Full address)

Just a handful of schemes deployed in the Internet

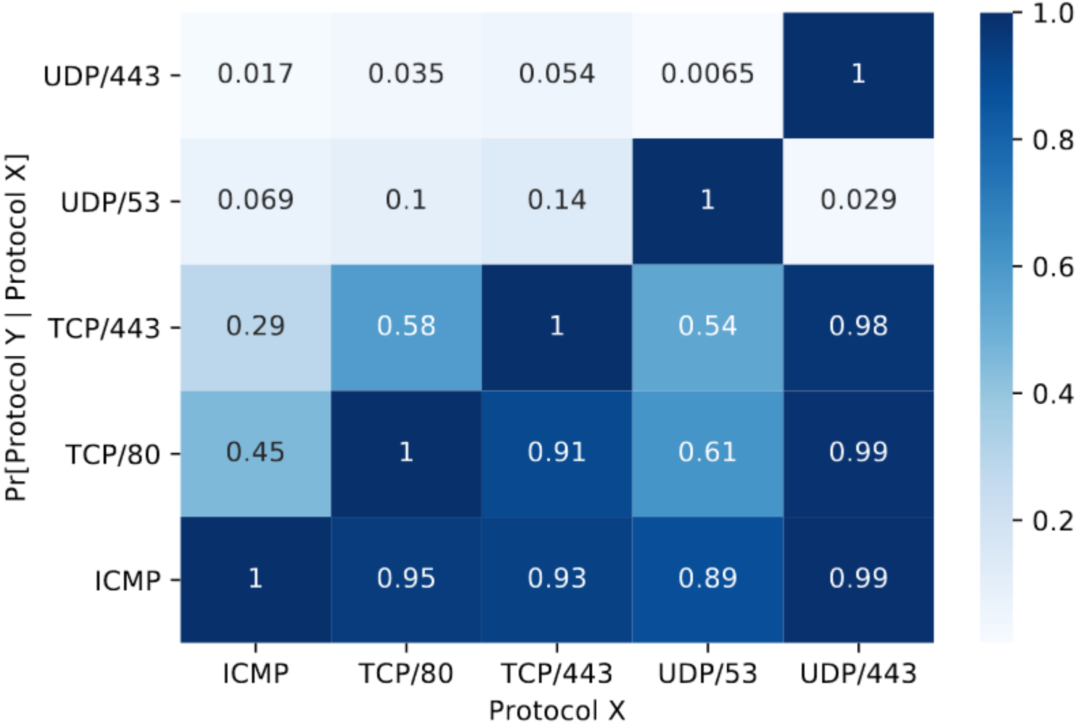


How does cross-protocol responsiveness look like?

Generate v6 targets and probe daily

If address responds on protocol X, how likely is it to respond on protocol Y?

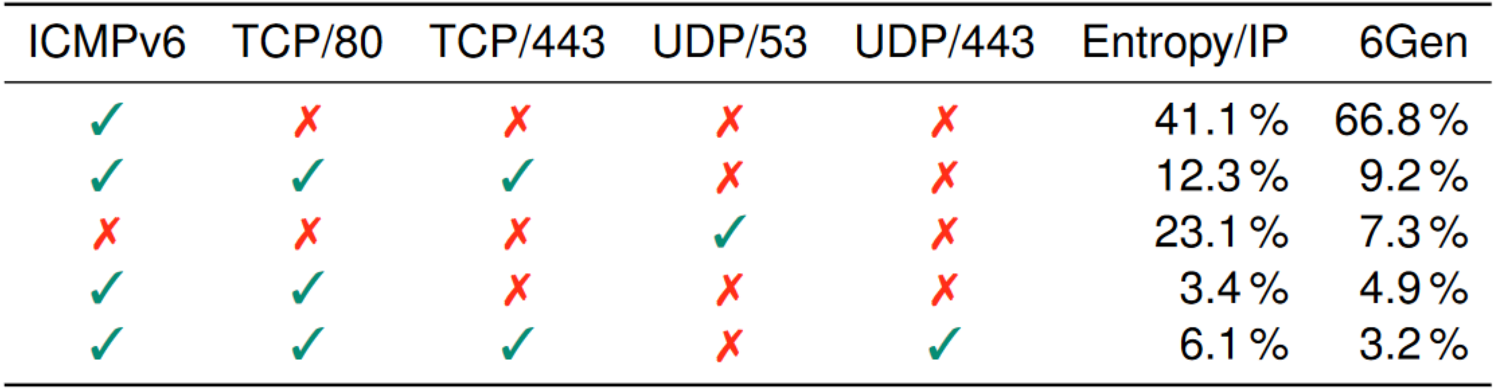
Helps to identify relevant addresses for specific measurements



Pr[Protocol Y | Protocol X]	ICMP	TCP/80	TCP/443	UDP/53	UDP/443
UDP/443	0.017	0.035	0.054	0.0065	1
UDP/53	0.069	0.1	0.14	1	0.029
TCP/443	0.29	0.58	1	0.54	0.98
TCP/80	0.45	1	0.91	0.61	0.99
ICMP	1	0.95	0.93	0.89	0.99

Is there a benefit of using more than one address learning tool?

Comparing Entropy/IP and 6Gen and responsiveness



ICMPv6	TCP/80	TCP/443	UDP/53	UDP/443	Entropy/IP	6Gen
✓	✗	✗	✗	✗	41.1 %	66.8 %
✓	✓	✓	✗	✗	12.3 %	9.2 %
✗	✗	✗	✓	✗	23.1 %	7.3 %
✓	✓	✗	✗	✗	3.4 %	4.9 %
✓	✓	✓	✗	✓	6.1 %	3.2 %

Discussions

Time-to-measurements

IPv6 server are more responsive compared to home devices and clients

When using hitlists as input, client devices need to be measured in minutes

Hitlist tailoring

Prevent bias by removing aliased prefixes

Tailor down to ASes, protocols etc. depends on study

Unresponsive addresses

Can be used to understand addressing schemes inside a prefix

Literature

Oliver Gasser, Quirin Scheitle, Pawel Foremski, Qasim Lone, Maciej Korczyński, Stephen D. Strowes, Luuk Hendriks, and Georg Carle. Clusters in the Expanse: Understanding and Unbiasing IPv6 Hitlists. In *Proceedings of the Internet Measurement Conference 2018* (IMC '18). ACM, 364-378, 2018. DOI: https://doi.org/10.1145/3278532.3278564

Clusters in the Expanse: Understanding and Unbiasing IPv6 Hitlists

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

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

Pawel Foremski
IITiS PAN
pjf@iitis.pl

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

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

Stephen D. Strowes
RIPE NCC
sdstrowes@gmail.com

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

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

ABSTRACT

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

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

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

CCS CONCEPTS

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

KEYWORDS

IPv6, Hitlist, Clustering, Aliasing, Entropy

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

IMC '18, October 31-November 2, 2018, Boston, MA, USA
© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-5619-0/18/10…$15.00
https://doi.org/10.1145/3278532.3278564

1 INTRODUCTION

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

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

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

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

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

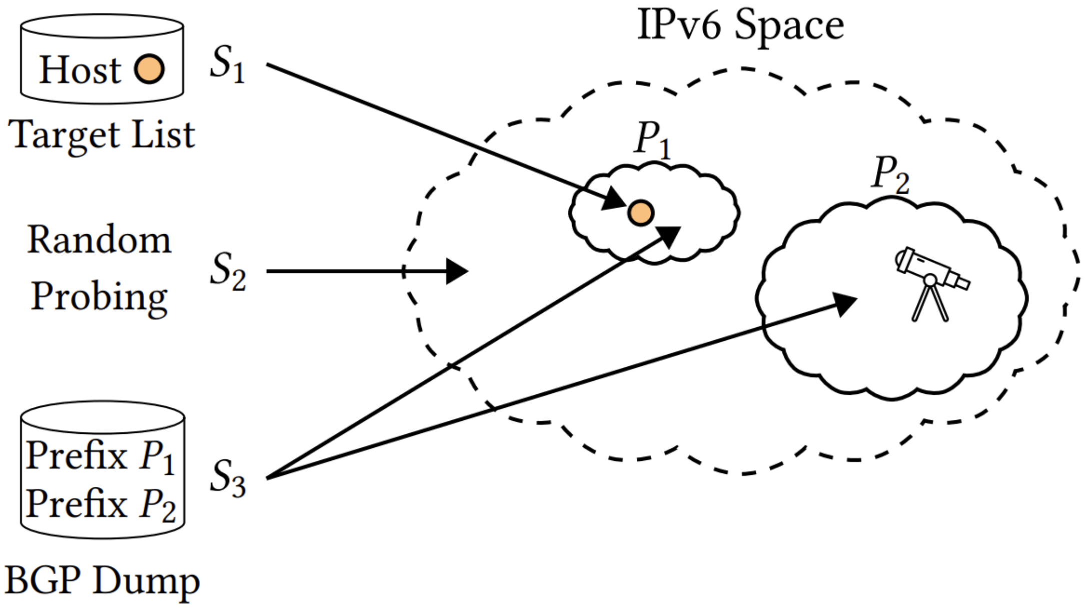
Longitudinal Stability Probing: To find reliably responsive addresses, we conduct longitudinal scans for our hitlist across several protocols. As expected, we find only a fraction of discovered

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

Measuring in the wild

OBSERVING IPV6 SCANNERS

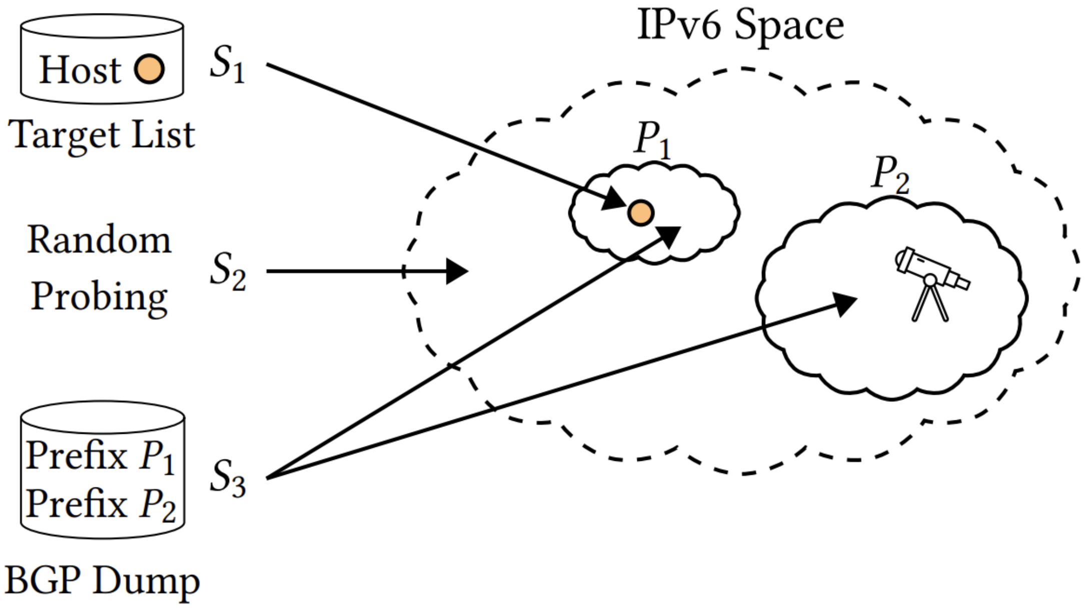
Three approaches to probe IPv6 address space?



Three approaches to probe IPv6 address space

If we understand IPv6 scanners, we can deploy observation points with more precise focus.

This may reduce costs and increase accuracy.



What is this study about?

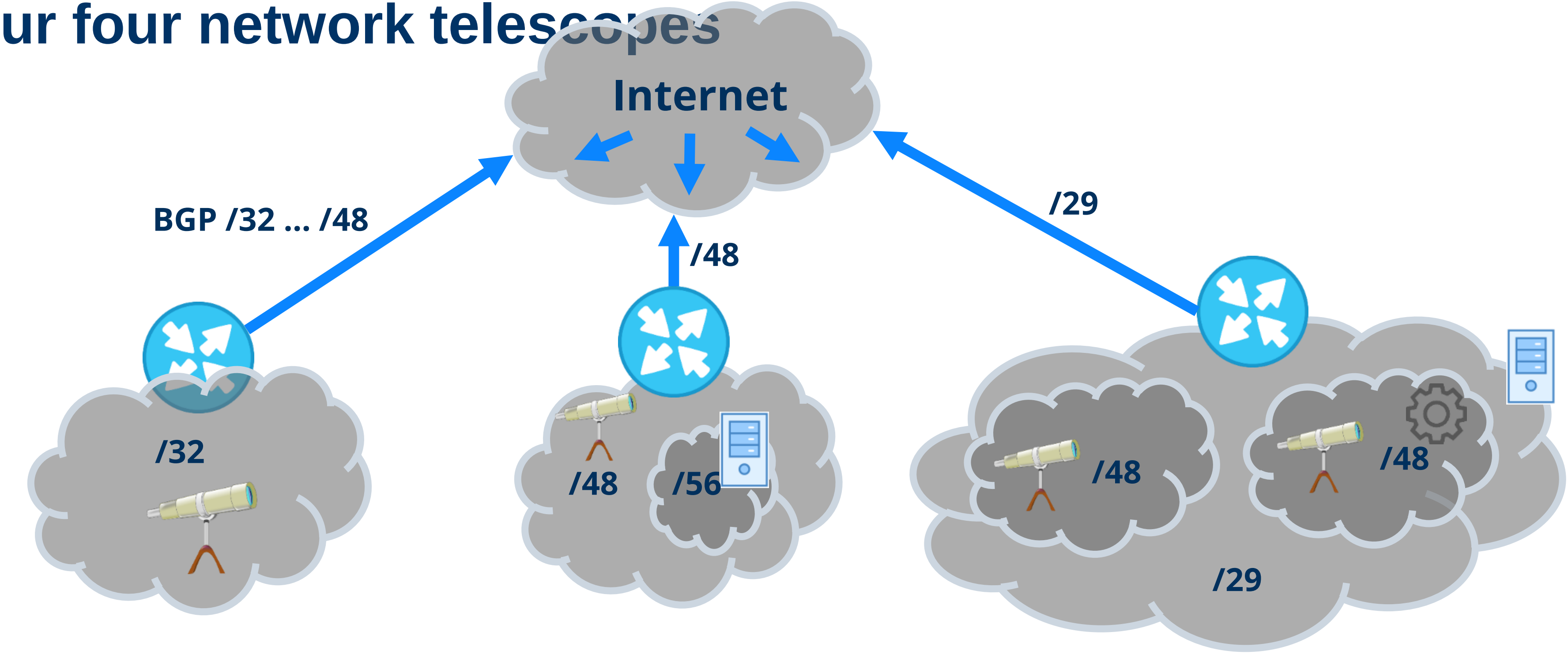
Better understanding of IPv6 scanners.

How should we design IPv6 network telescopes to capture IPv6 scanners?

Which limitations do specific network telescopes have?

Which bias is introduced from the perspective of a telescope?

Our four network telescopes



T1: Passive, BGP controlled

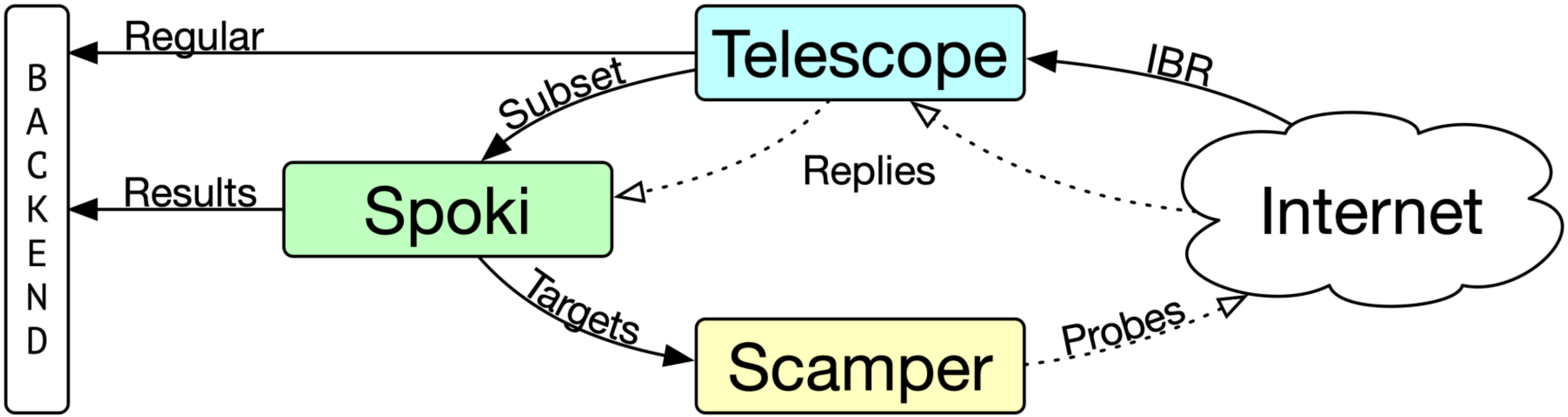
T2: Partially productive

T3: Silent

T4: Reactive (TCP)

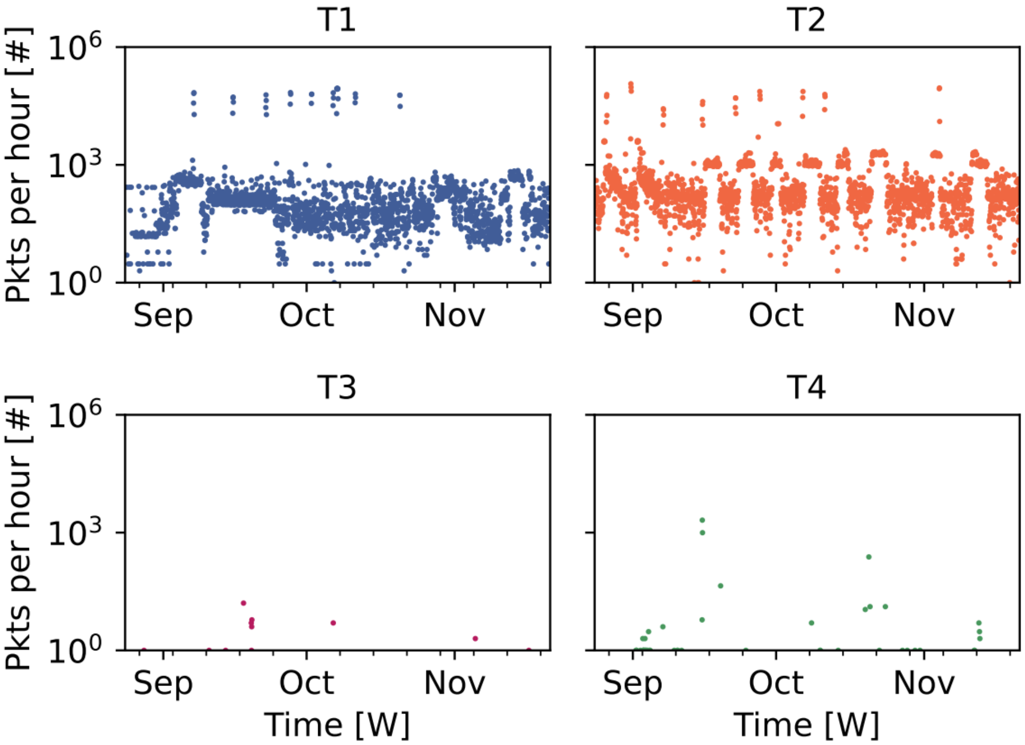
Spoki: Reactive telescope to continue dialog with attacker

- Replies to (stateless two-phase) scanning to explore attack surface
- Asynchronously accepts and matches (2^{nd} phase) connections



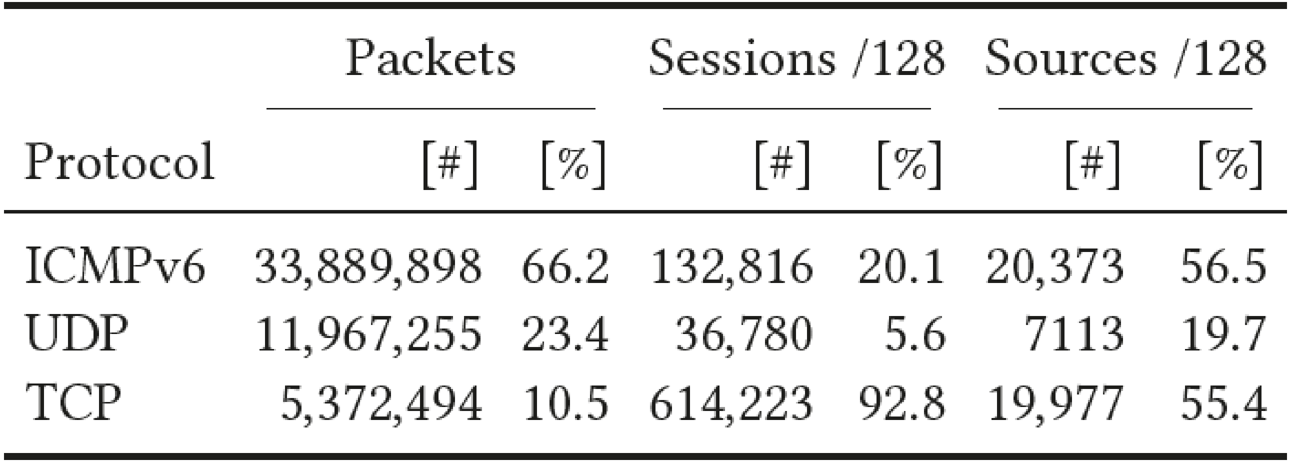
Raphael Hiesgen, Marcin Nawrocki, Alistair King, Alberto Dainotti, Thomas C. Schmidt, Matthias Wählisch,
Spoki: Unveiling a New Wave of Scanners through a Reactive Network Telescope,
In: *Proc. of 31st USENIX Security Symposium,* pp. 431-448, USENIX Association : Berkeley, CA, USA, August 2022.

Unsolicited traffic across the telescopes during initial observation period of 12 weeks



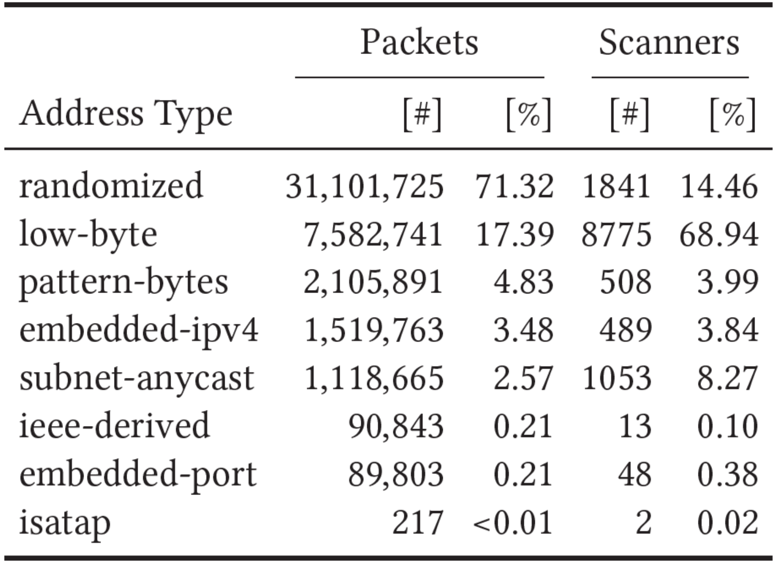
How popular are protocols?

Packets vs. sources vs. sessions

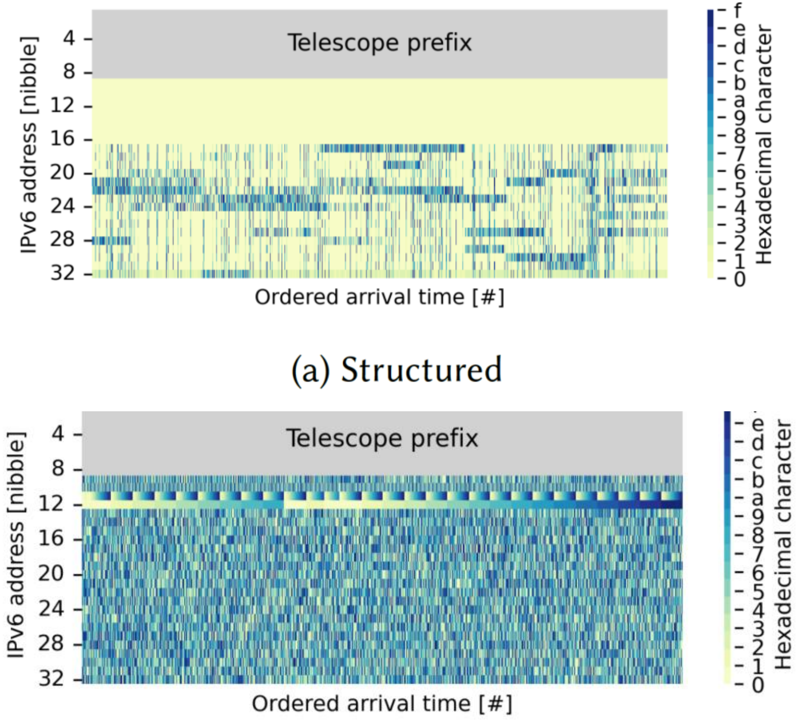


	Packets		Sessions /128		Sources /128	
Protocol	[#]	[%]	[#]	[%]	[#]	[%]
ICMPv6	33,889,898	66.2	132,816	20.1	20,373	56.5
UDP	11,967,255	23.4	36,780	5.6	7113	19.7
TCP	5,372,494	10.5	614,223	92.8	19,977	55.4

Which type of addresses do scanners target?



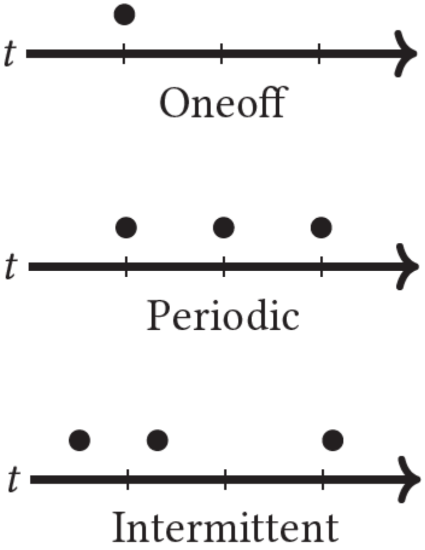
	Packets		Scanners	
Address Type	[#]	[%]	[#]	[%]
randomized	31,101,725	71.32	1841	14.46
low-byte	7,582,741	17.39	8775	68.94
pattern-bytes	2,105,891	4.83	508	3.99
embedded-ipv4	1,519,763	3.48	489	3.84
subnet-anycast	1,118,665	2.57	1053	8.27
ieee-derived	90,843	0.21	13	0.10
embedded-port	89,803	0.21	48	0.38
isatap	217	<0.01	2	0.02



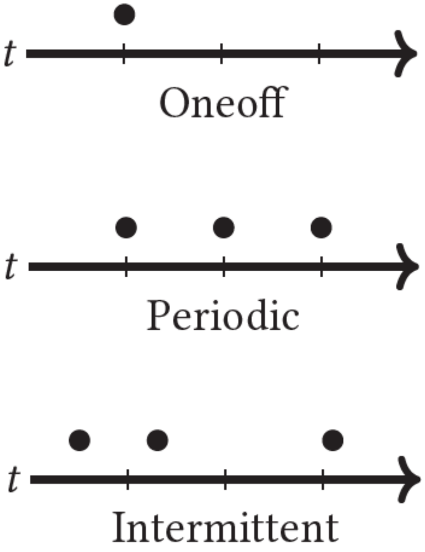
(a) Structured

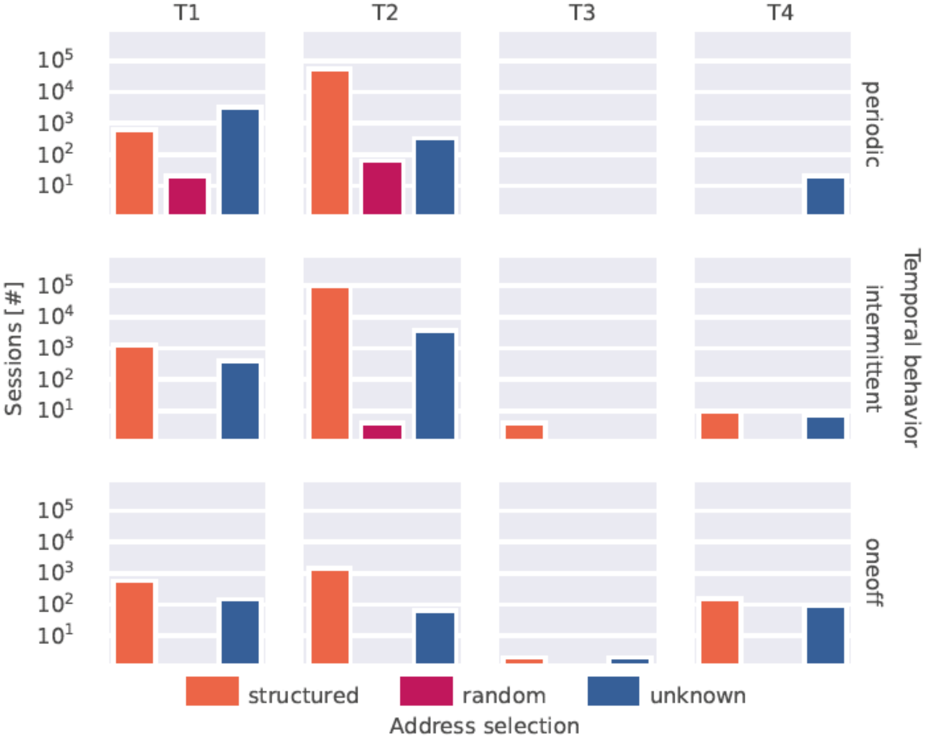
(b) Random

Classifying temporal behavior of scanners



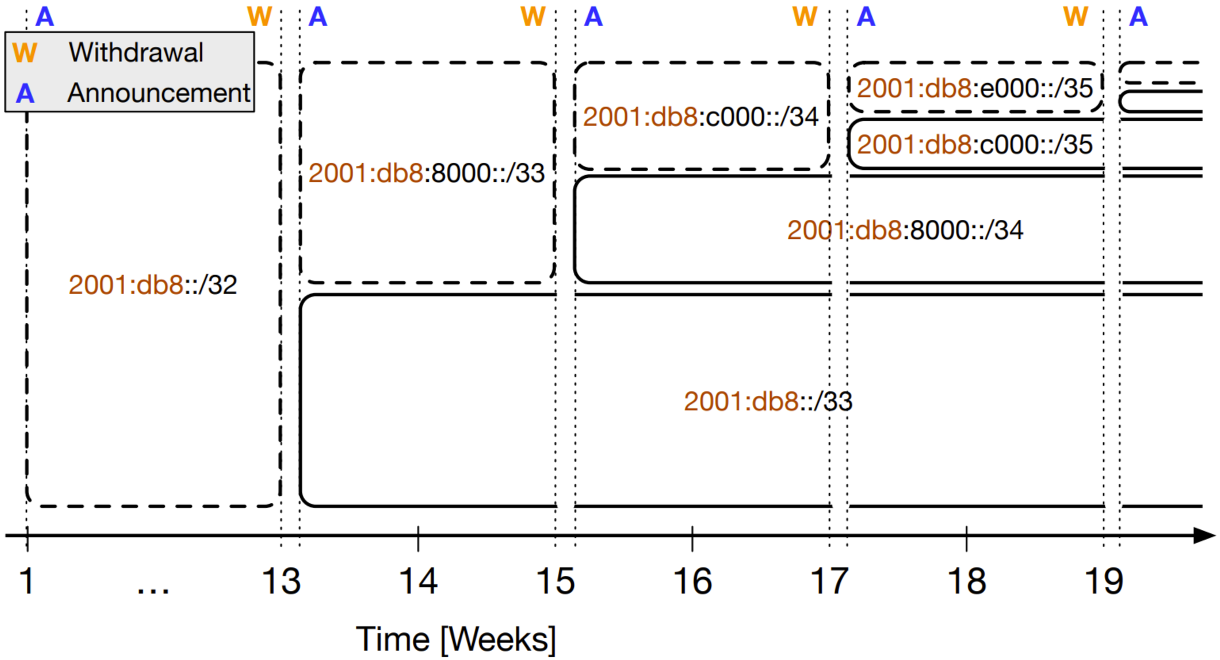
Classifying temporal behavior of scanners



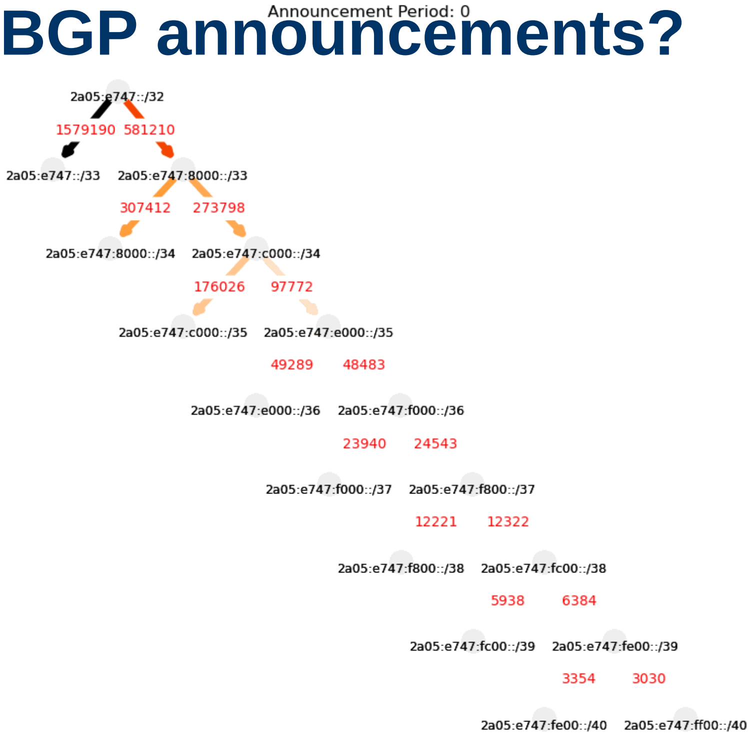


Method to create BGP signals

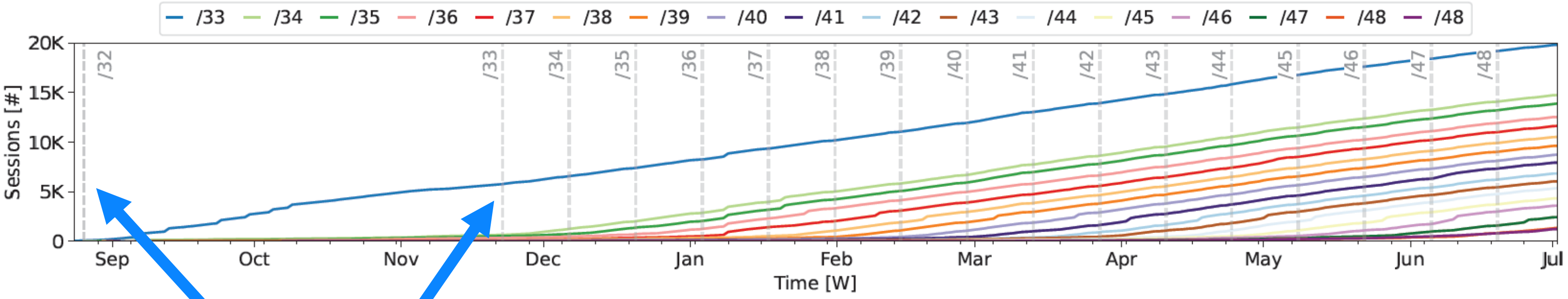
Controlled, passive measurements



How do scanners react on our BGP announcements?



How do scanners react on our BGP announcements?



Start of BGP announcement

As soon as we announce a more specific prefix, scanners start probing this more specific prefix.

Conclusion

How to build an attractive telescope? Network visibility largely depends on announcing the telescope prefix individually in BGP.

Are observations in telescopes unbiased? No. Scanners contact telescopes following external triggers, which in turn means that triggers attract only those scanners that react to them.

Are IPv6 telescopes suitable to monitor DDoS? No. Telescopes commonly monitor DDoS by capturing the backscatter from randomly spoofed attack traffic.