Testing Resolver Behaviours With Broken Authoritative Servers

How crazy is .internal -> NS/. anyway?

Wes Hardaker <hardaker@google.com>

Outline

- Background: DNS names that do not exist
- Experimental Testing
- Results
- Conclusions

Background

- The DNS fails in many ways
  - Operators fail to deploy zones properly
    - Broken infrastructure
  - Or the name doesn’t exist globally

- Internal name spaces have (likely) always existed
  - The famous three: .corp, .mail, .home
  - SSAC113: recommends using “**.internal**”

- What happens when a name doesn’t exist?
  - For example: something.internal

.internal background

- SSAC113: reserve the TLD “internal” for private namespace
  - Similar to private addresses spaces like 10.0.0.0/8, etc
  - This process is horribly simplified (involves IANA, IAB, etc)

- Board resolution 2024.07.29.06:

  Resolved (2024.07.29.06), the Board reserves .INTERNAL from delegation in the DNS root zone permanently to provide for its use in private-use applications. The Board recommends that efforts be undertaken to raise awareness of its reservation for this purpose through the organization's technical outreach.

Full resolution pieces

- Board resolution 2024.07.29.06:

  *“Resolved (2024.07.29.06), the Board reserves .INTERNAL **from delegation** in the DNS root zone permanently to provide for its use in private-use applications. The Board recommends that efforts be undertaken to raise awareness of its reservation for this purpose through the organization's technical outreach.”*

- From Rationale for Resolution 2024.07.29.06:

  “**Reserving a string from delegation permanently** is in the public interest for the reasons outlined in this resolution and rationale.”

  …

  “**Reserving a string from delegation permanently** is neither a defined policy process with ICANN's supporting organizations nor an ICANN administrative function.”

Resolving *child.parent*

The cases:

1. *parent* doesn't exist at all
2. *parent* exists, but *child* doesn't
3. *parent* NS record is broken
    a. broken NS: internal with no glue
    b. broken NS: external that doesn't exist
    c. broken NS: pointing to "."

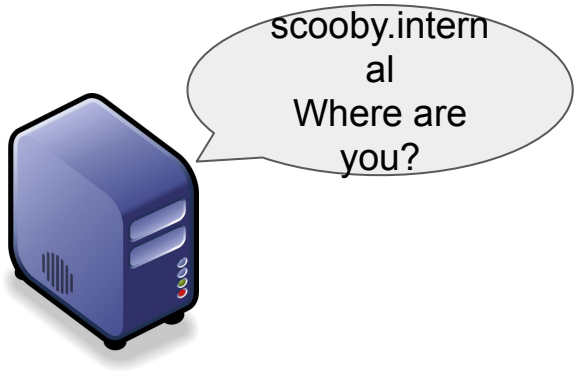
Note: This is not *.internal* specific!

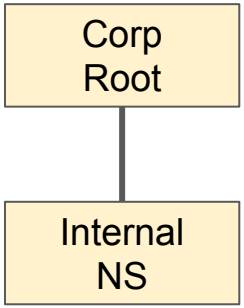
Resolving *child.parent*

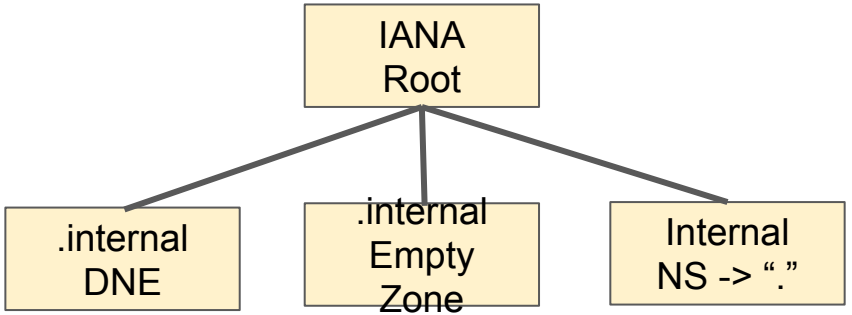
The cases:

1. *parent* doesn't exist at all
2. *parent* exists, but *child* doesn't
3. *parent* NS record is broken
    a. broken NS: internal with no glue
    b. broken NS: external that doesn't exist
    c. broken NS: pointing to "."

What does a validating resolver do when querying “.internal”?



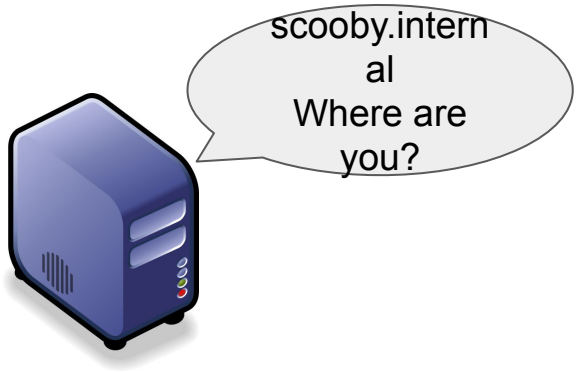


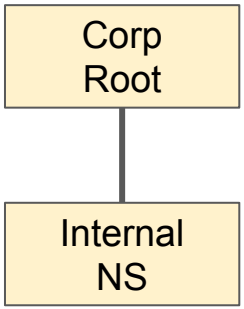


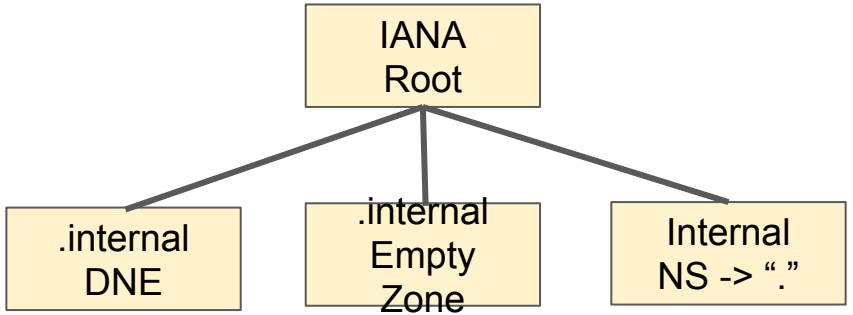
Today!

Response	A Record
Signed?	Maybe?
TTL	Record TTL

What does a validating resolver do when querying “.internal”?



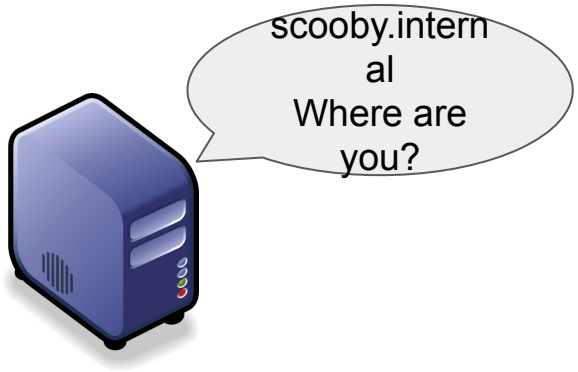


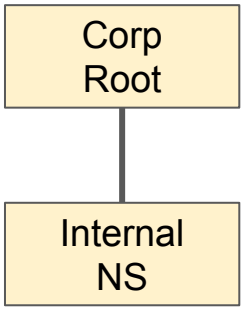


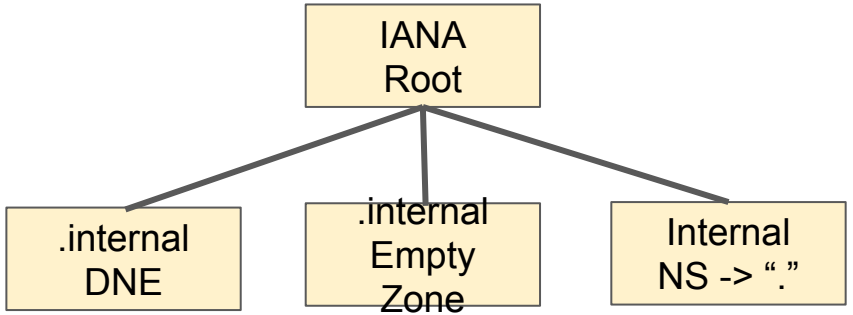
Today!

Response	A Record	NSEC(NXDomain)
Signed?	Maybe?	Yes
TTL	Record TTL	NSEC TTL

What does a validating resolver do when querying “.internal”?



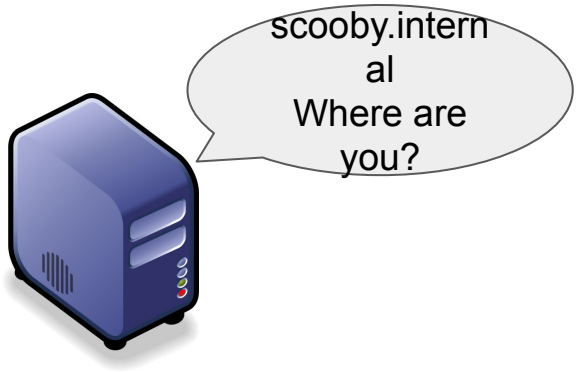


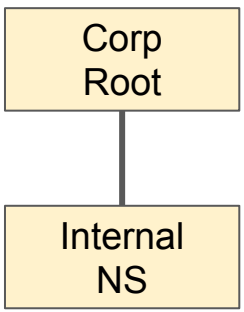


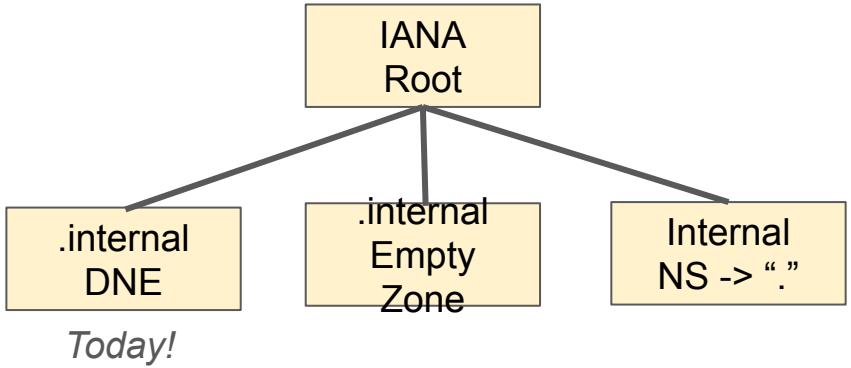
Today!

Response	A Record	NSEC(NXDomain)	NXDOMAIN
Signed?	Maybe?	Yes	No/Maybe
TTL	Record TTL	NSEC TTL	max(,Empty Zone SOA Neg Cache TTL, NS parent TTL, NS child TTL)

What does a validating resolver do when querying “.internal”?



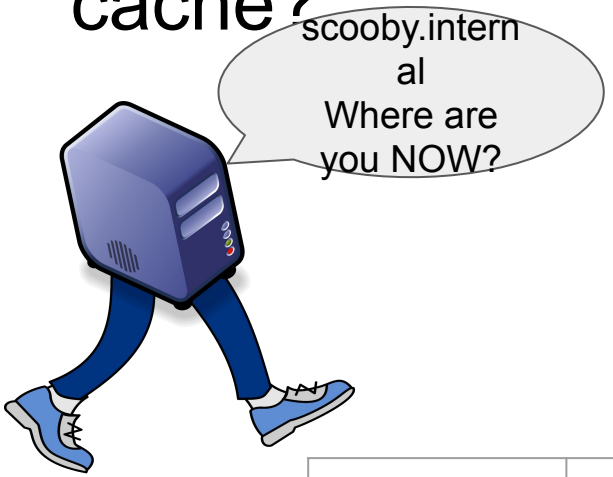


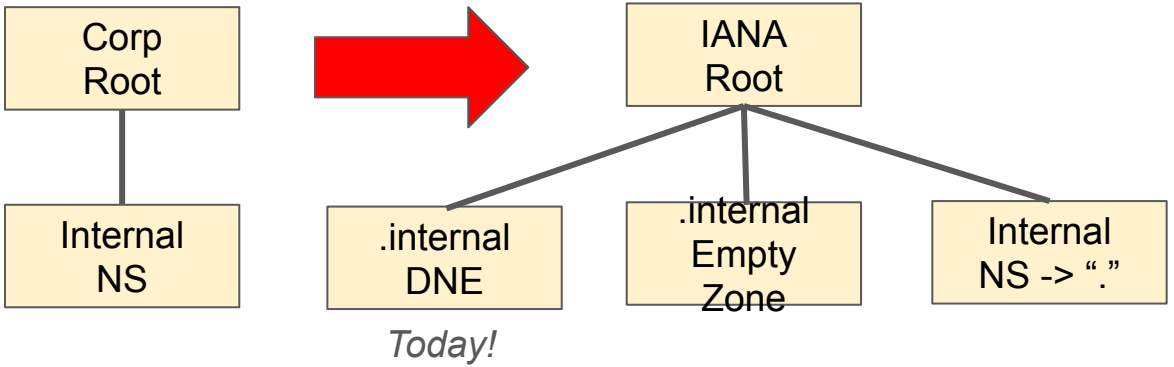


Today!

Response	A Record	NSEC(NXDomain)	NXDOMAIN	SERVFAIL
Signed?	Maybe?	Yes	No/Maybe	No
TTL	Record TTL	NSEC TTL	max(Empty Zone SOA Neg Cache TTL, NS parent TTL, NS child TTL)	SHORT

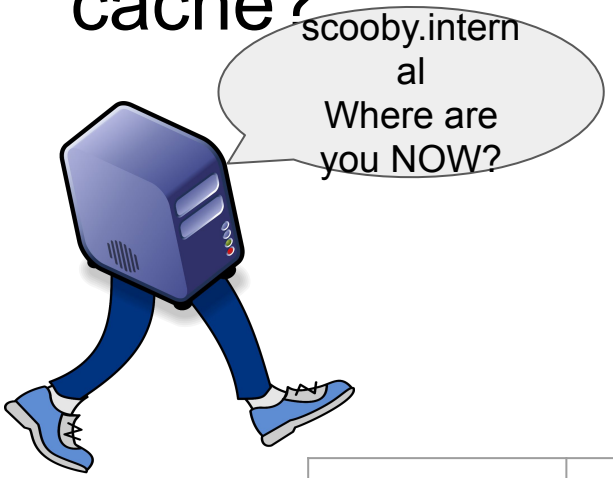
What if the resolvers move outside CORP with a cache?

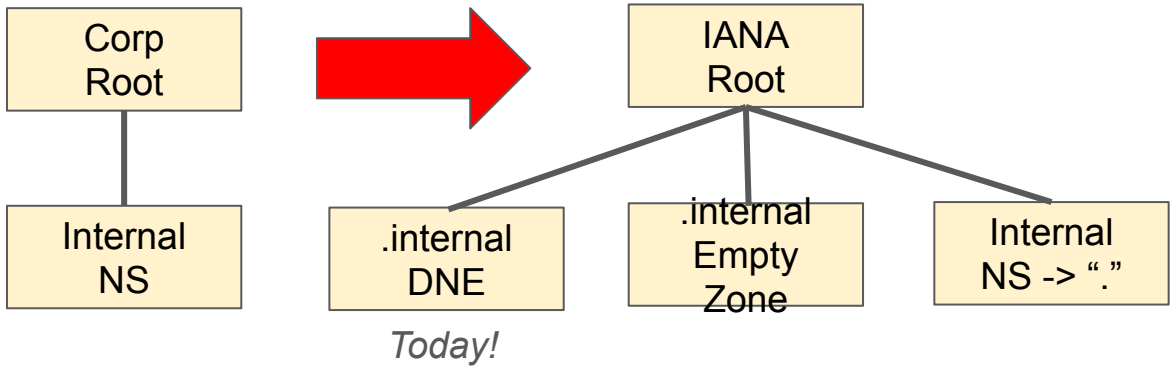




Response	A Record
Signed?	Maybe?
TTL	Record TTL

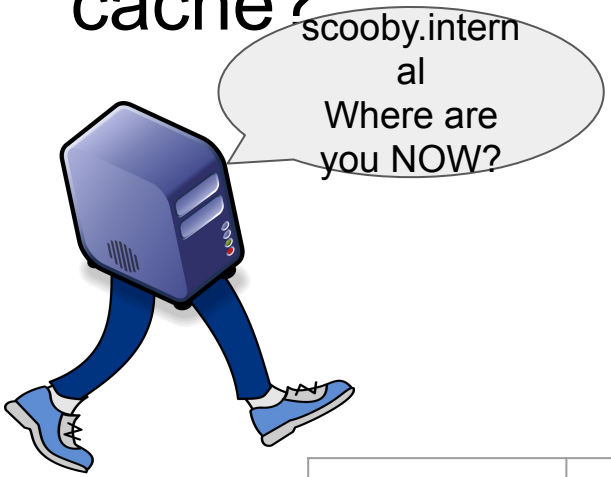
What if the resolvers move outside CORP with a cache?

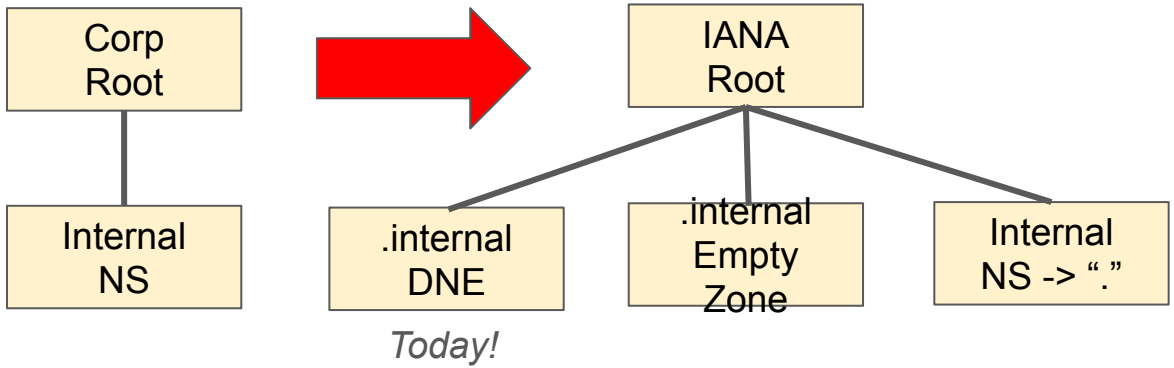




Response	A Record	→	Cached Record	Cached Record	Cached Record
Signed?	Maybe?	→	Maybe	Maybe	Maybe
TTL	Record TTL	→	Cached TTL	Cached TTL	Cached TTL

What if the resolvers move outside CORP with a cache?

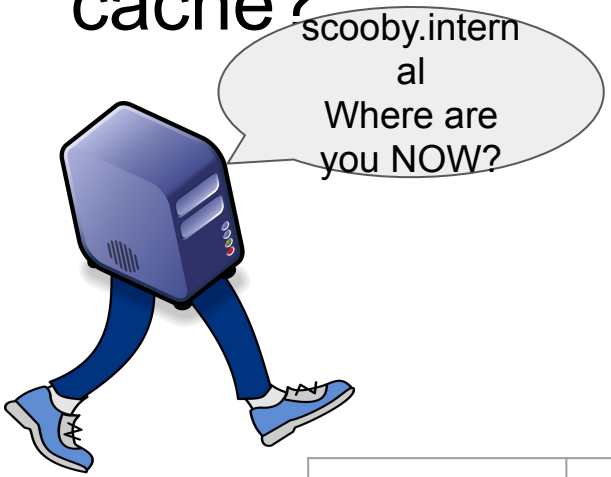


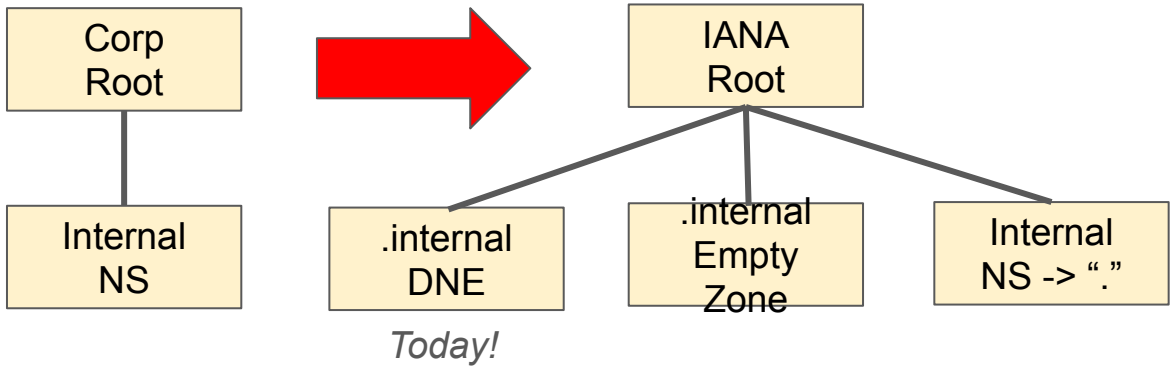


Response	A Record	⇨	Cached Record	Cached Record	Cached Record
Signed?	Maybe?	⇨	Maybe	Maybe	Maybe
TTL	Record TTL	⇨	Cached TTL	Cached TTL	Cached TTL

Note: these may not be reachable records

What if the resolvers move outside CORP with a cache?

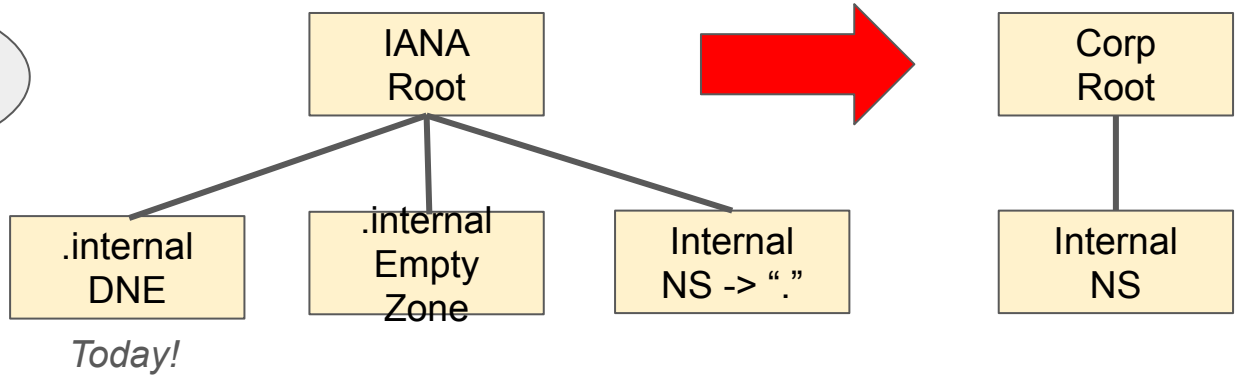




Response	A Record	⇨	Cached Record	Cached Record	Cached Record
Signed?	Maybe?	⇨	Maybe	Maybe	Maybe
TTL	Record TTL	⇨	Cached TTL	Cached TTL	Cached TTL

The corporation controls the cache length

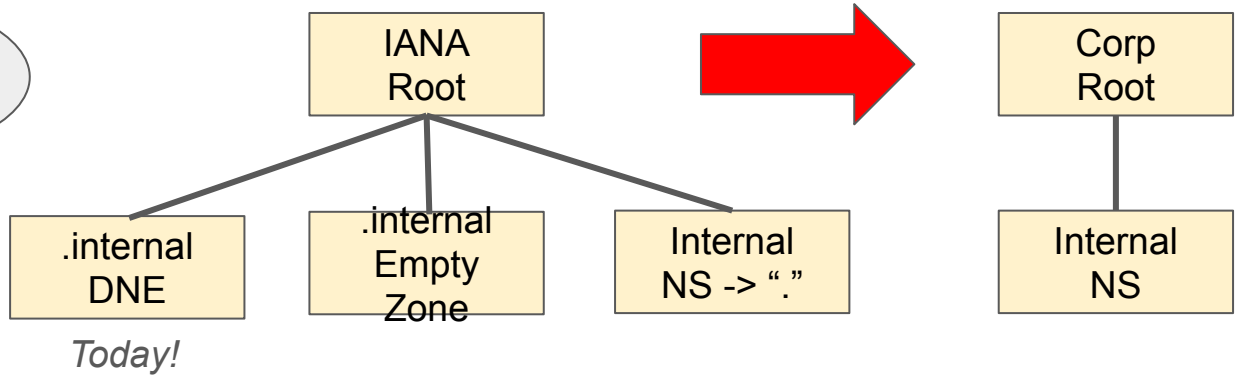
What if the resolvers move from global DNS to CORP?



Response	NSEC(NXDOMAIN)
Signed?	Yes
TTL	NSEC TTL

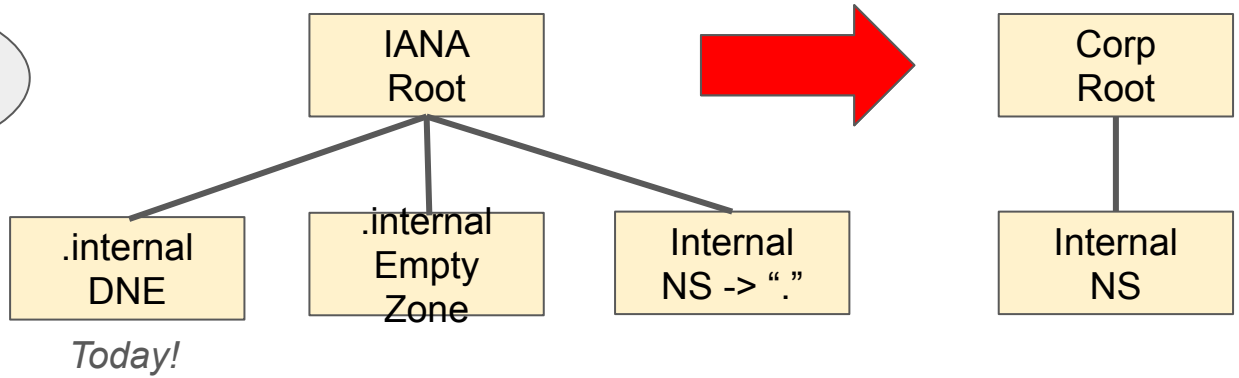
.internal doesn’t exist!
(remember that please)

What if the resolvers move from global DNS to CORP?



Response	NSEC(NXDOMAIN)			→	NSEC(NXDOMAIN)
Signed?	Yes			→	Yes
TTL	NSEC TTL			→	**1 day!**

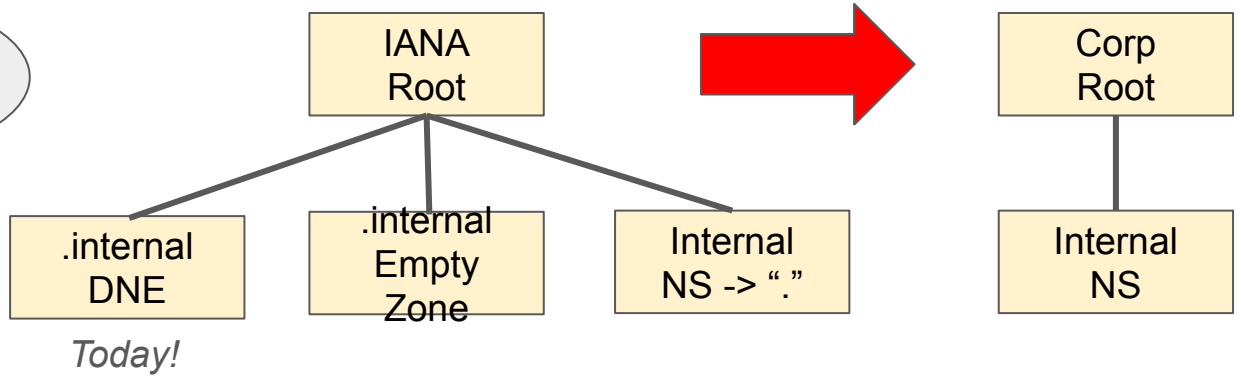
What if the resolvers move from global DNS to CORP?



Response		NXDOMAIN		→	NXDOMAIN
Signed?		No / Maybe		→	
TTL		Controlled		→	Short?

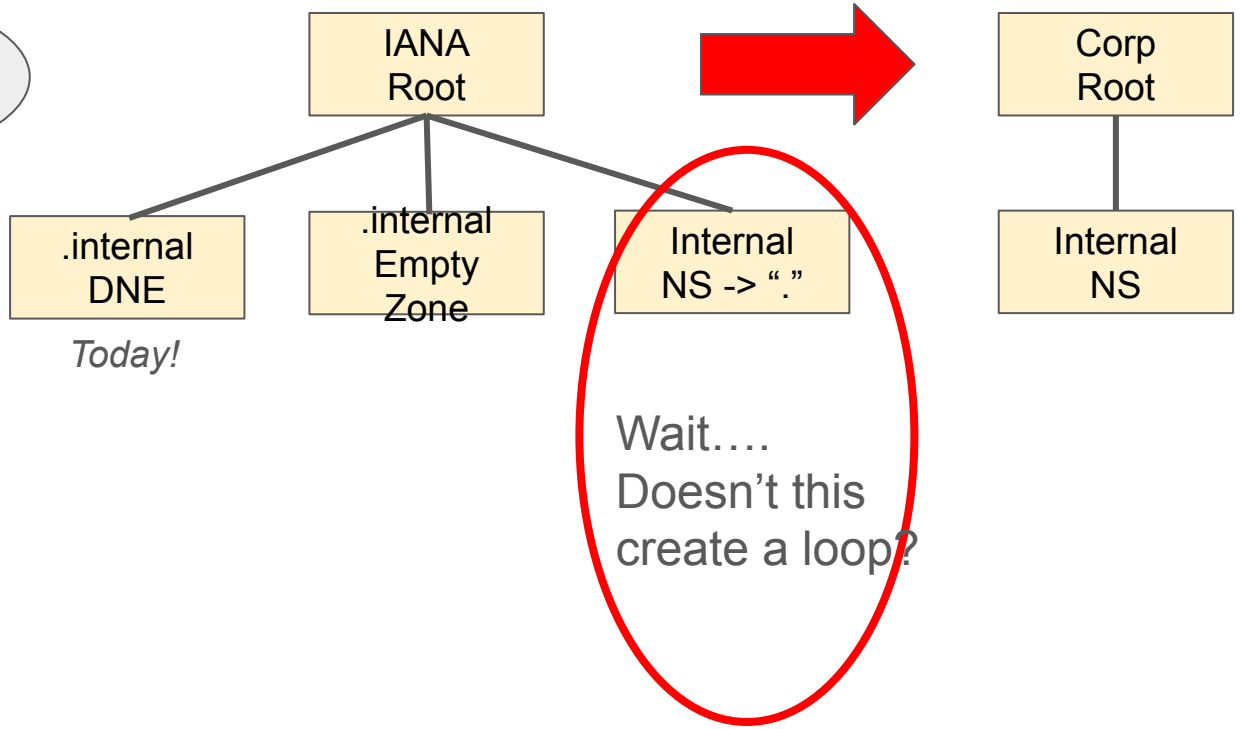
Problem: There is a negative answer TTL, and the zone NS TTL
For root TLDs, the NS TTLs are 2 days

What if the resolvers move from global DNS to CORP?



Response			SERVFAIL	→	SERVFAIL
Signed?			No	→	
TTL			Short	→	Short

What if the resolvers move from global DNS to CORP?



Experimental Testing

Let’s create test scenarios

- 5000 RIPE Atlas nodes
- Dedicated domains under two TLDs
  - sub-bbb.frostedaxe.com
  - sub-bbb.frostedaxe.games
- 1 authoritative name server
  - Recording all queries under test in PCAPs

Testing resolvers in various scenarios

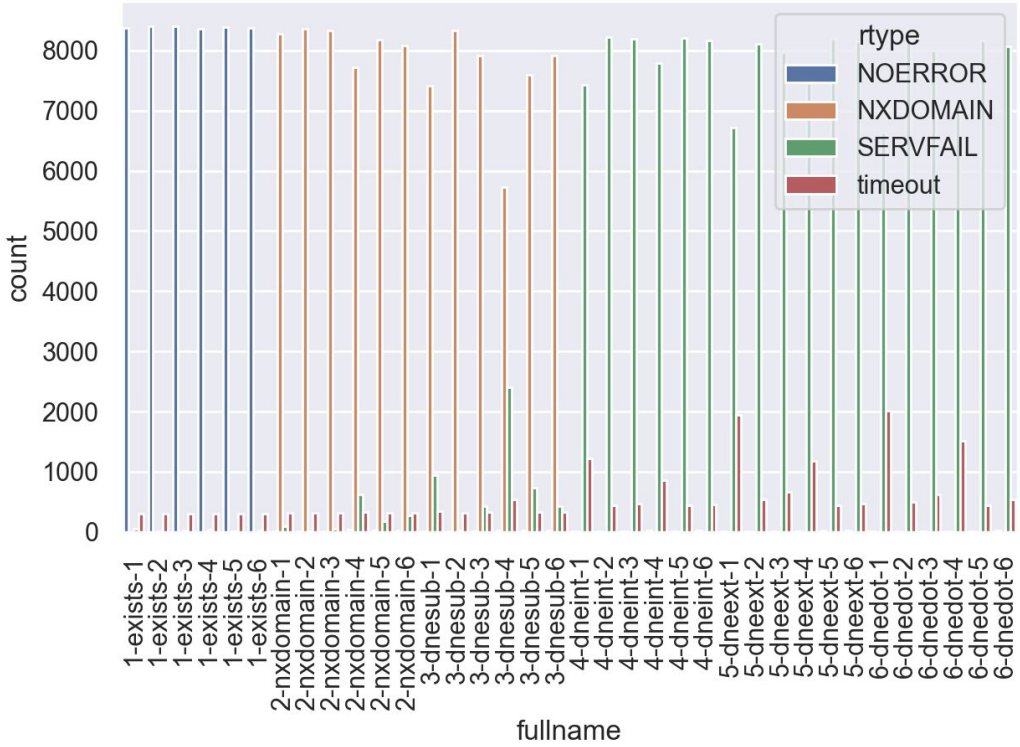
1. Querying an existing domain name
2. Querying a non-existing domain name
3. Querying a domain name under a non-existing sub-domain
4. Querying a domain name under a broken internal bailiwick
5. Querying a domain name under a broken external baliwick
6. Querying a domain name under a NS record with a “.” target

Analysis from three vantage points

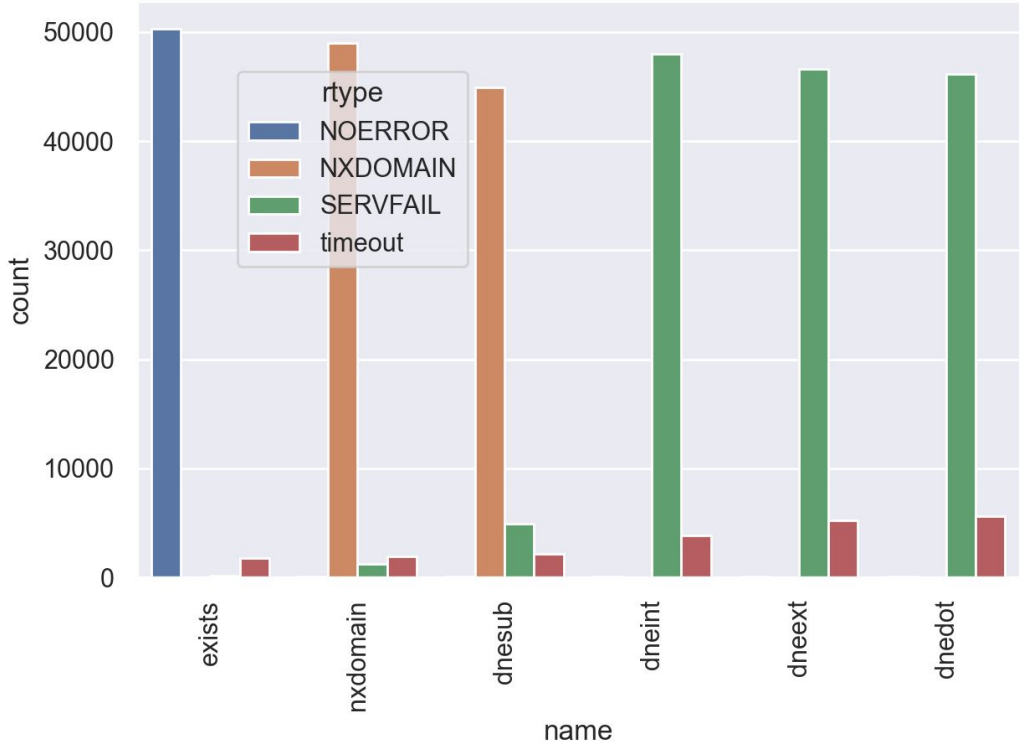
1. The RIPE Atlas data itself
2. The PCAPs at the authoritative server
3. Traffic received at B-Root

RIPE Atlas Data

Probe Answers Per Test



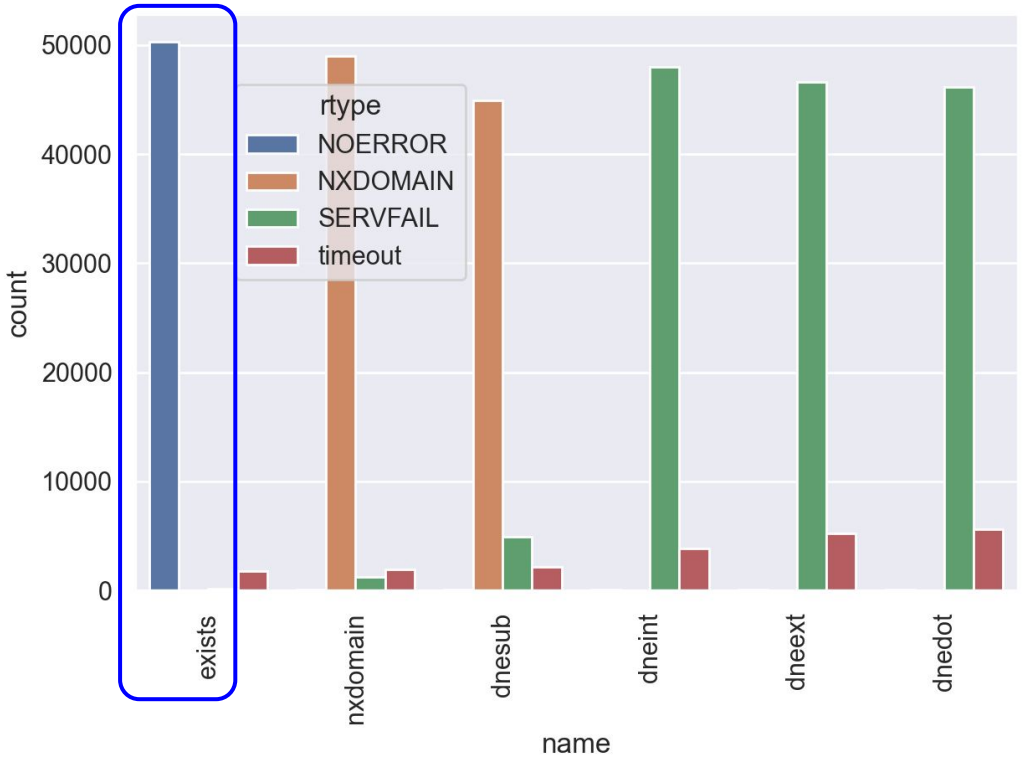
Probe answers aggregated per test type



Probe answers aggregated per test type

Exists

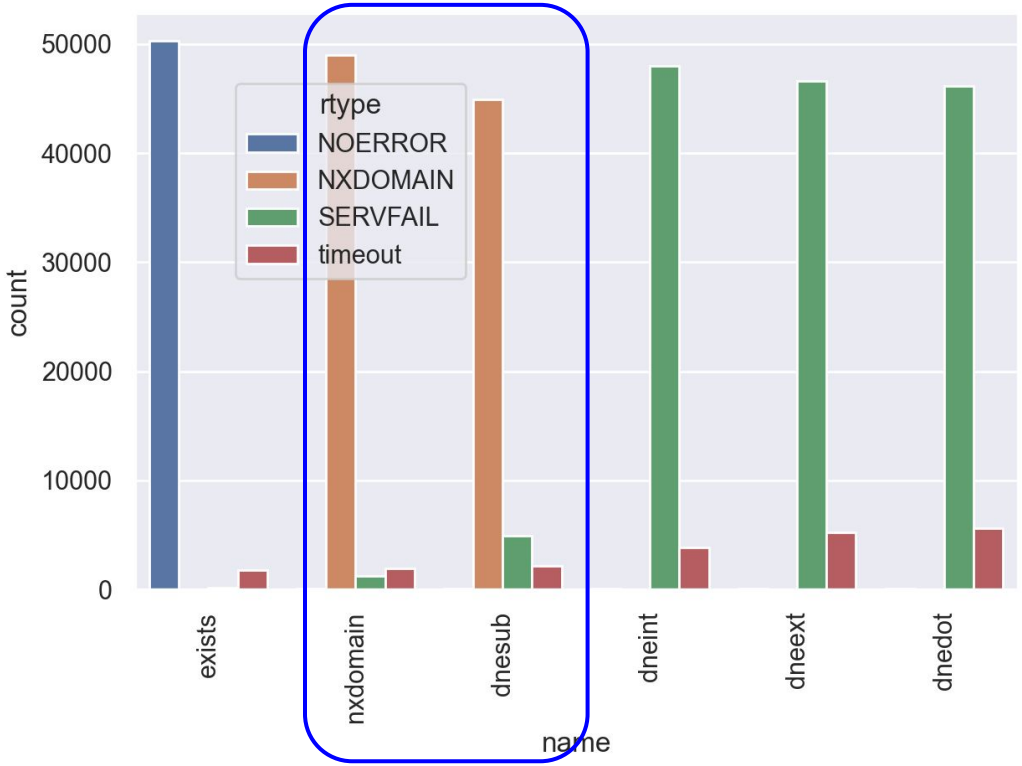
mostly NOERROR (answered)



Probe answers aggregated per test type

Does not exist

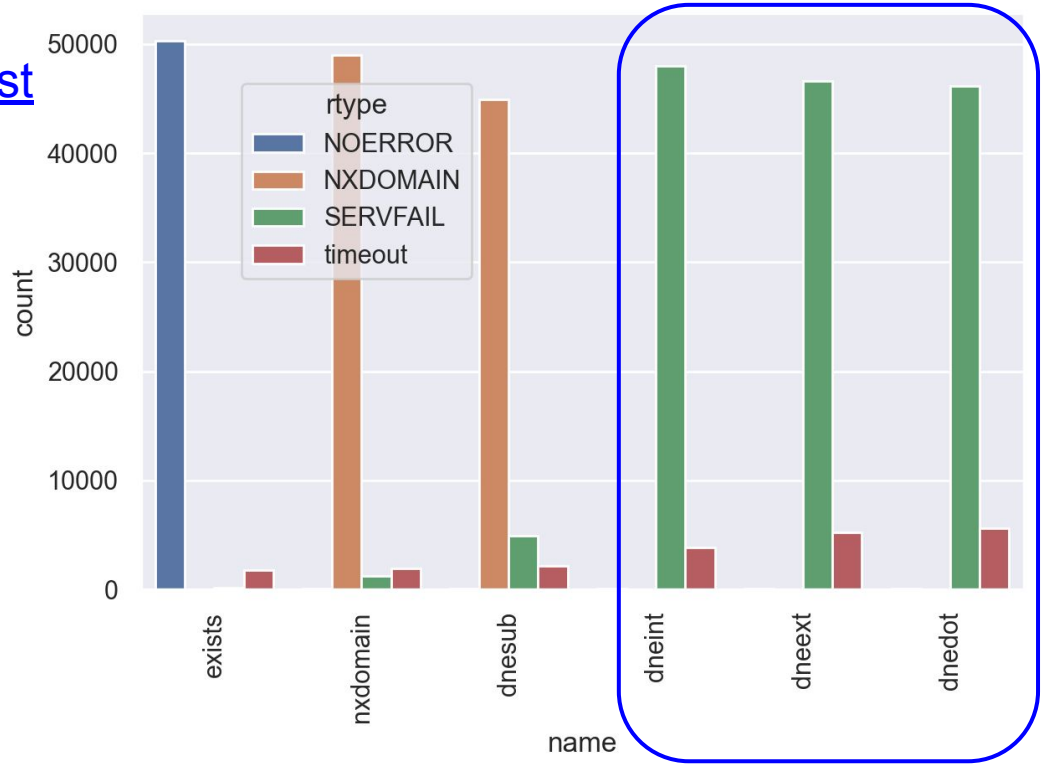
mostly NXDOMAIN



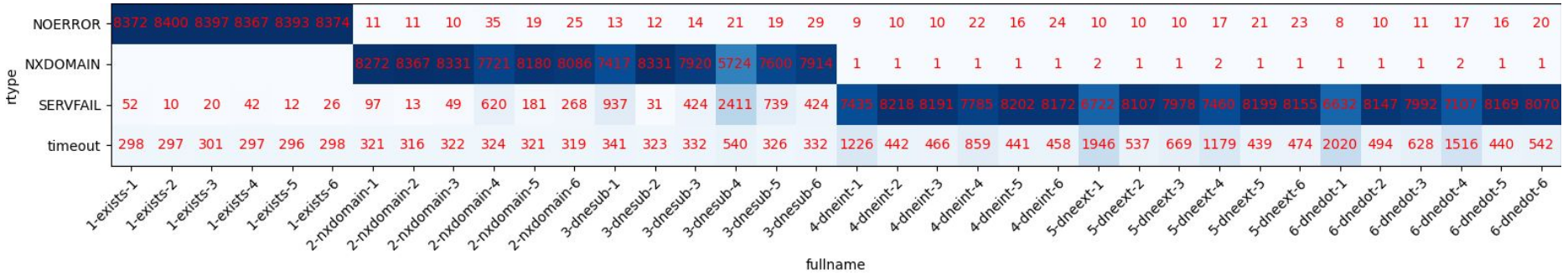
Probe answers aggregated per test type

NS does not exist

mostly SERVFAIL



Probe answers in a heat map

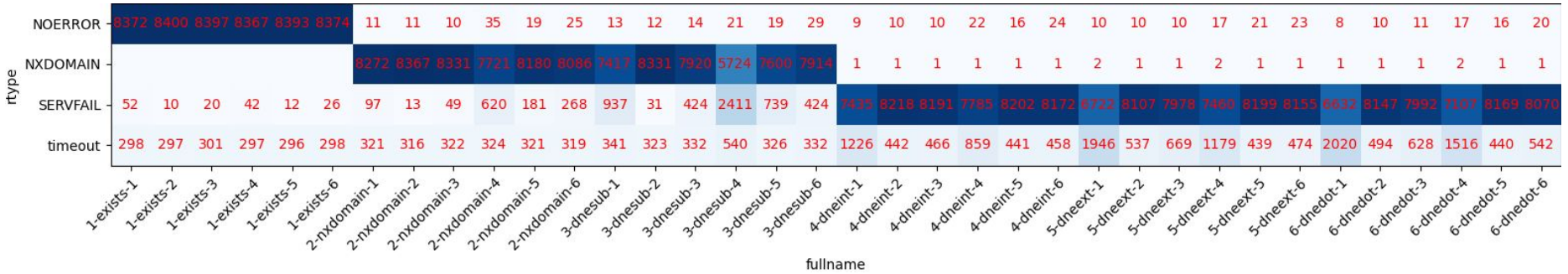


Probe answers in a heat map

Exists

Doesn’t Exist

Broken

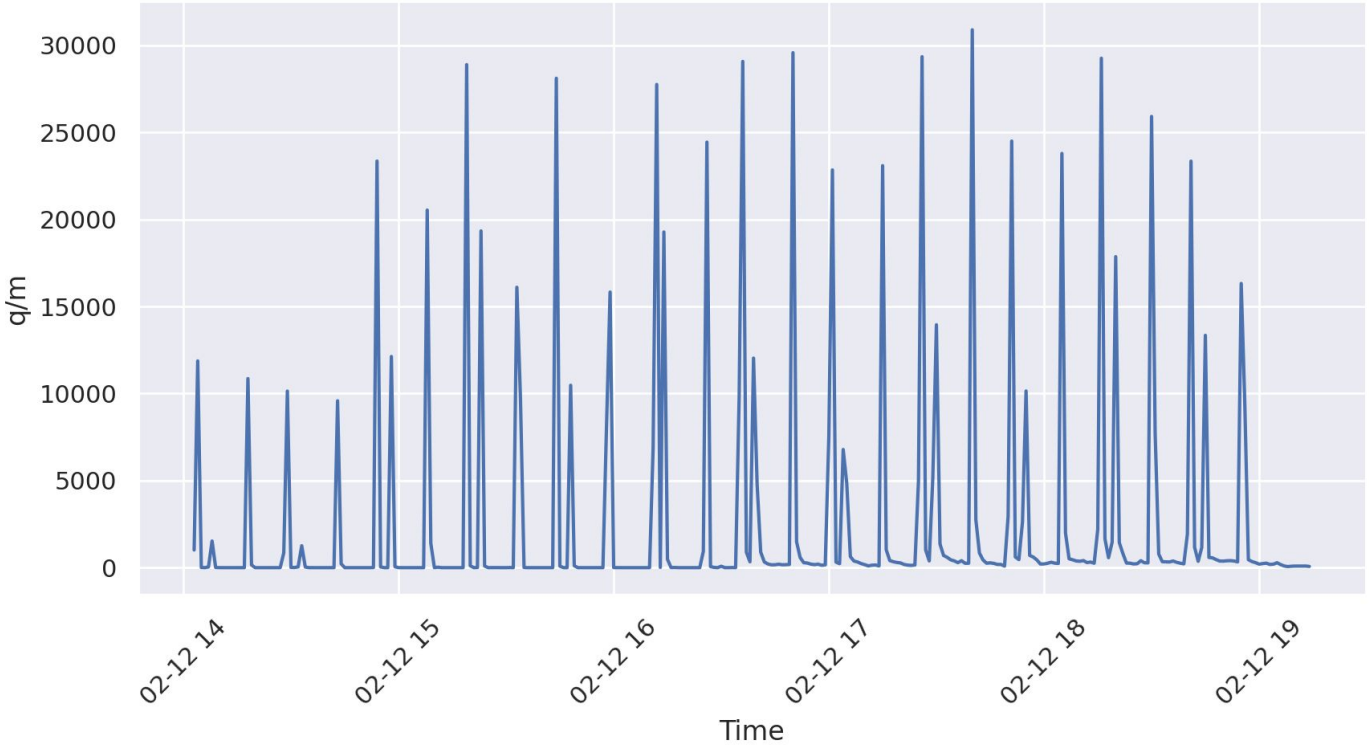


RIPE Atlas Conclusions

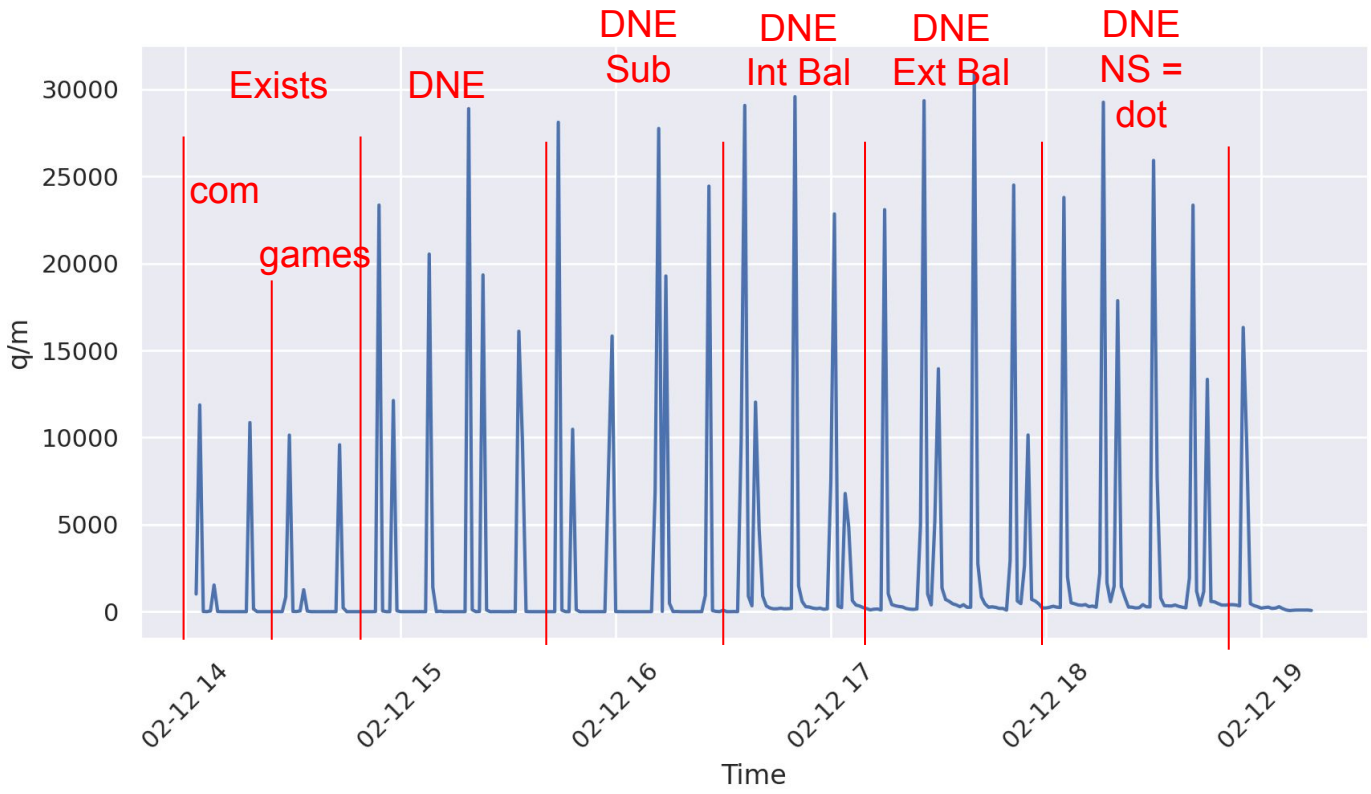
- Many results expected:
  - Exists == NOERROR
  - DNE == NXDOMAIN
  - Bad NS == SERVFAIL
- Some weirdness
- NS -> “.”: No worse than the other failure cases

PCAP Analysis

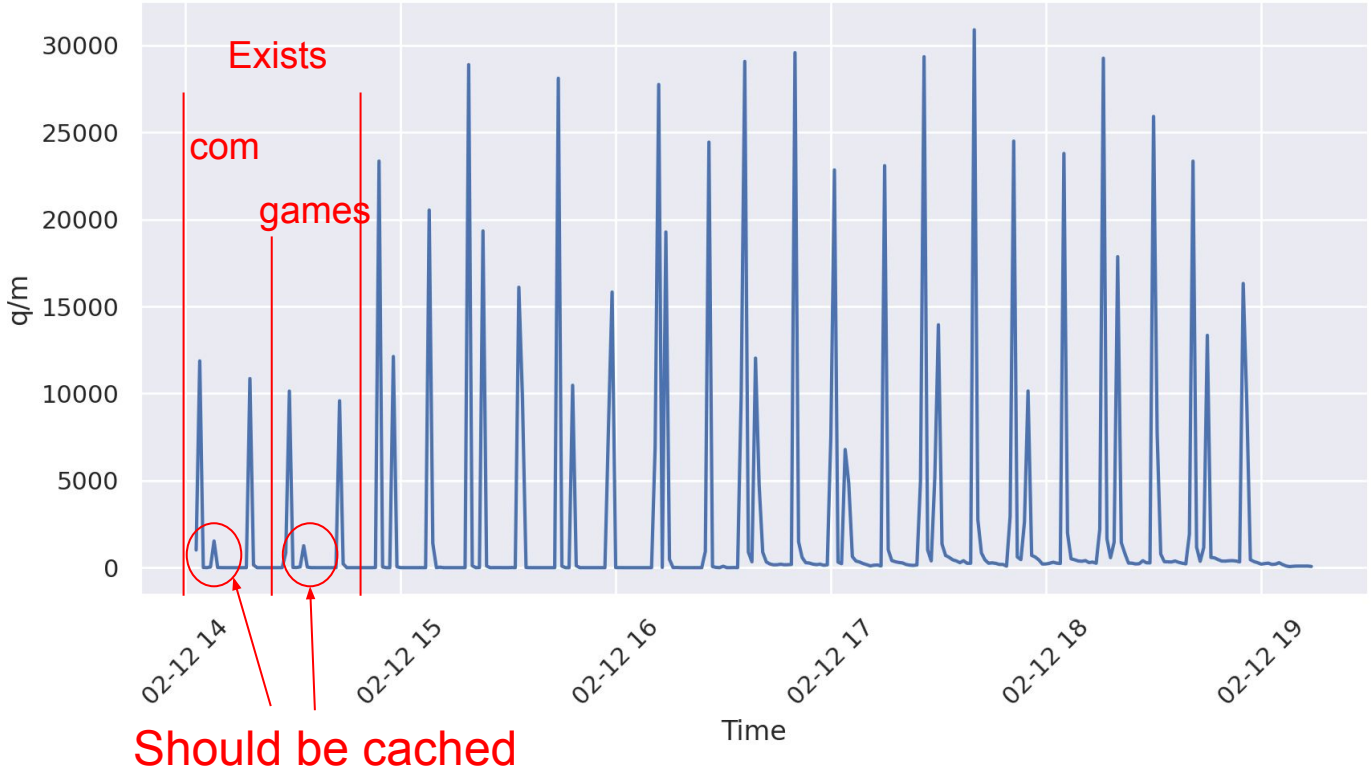
Total Traffic



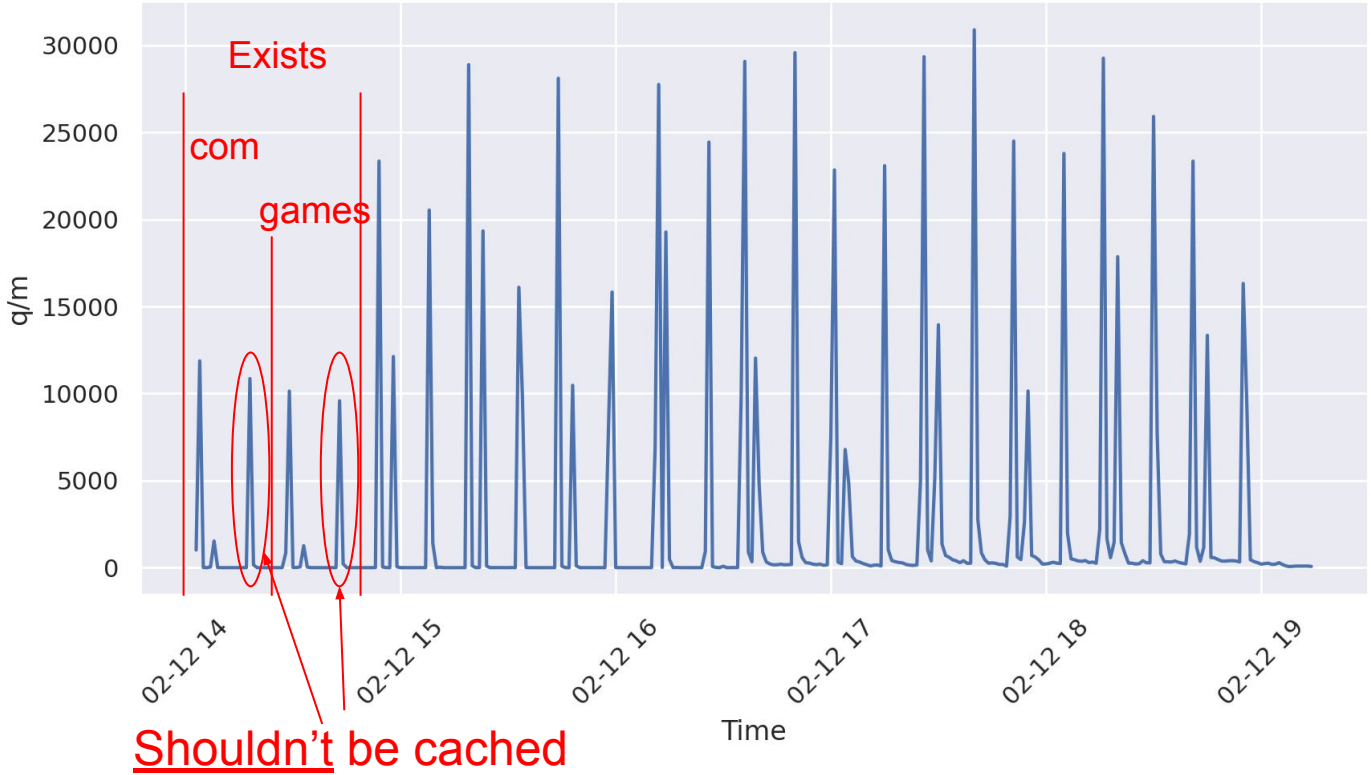
Total Traffic



Total Traffic

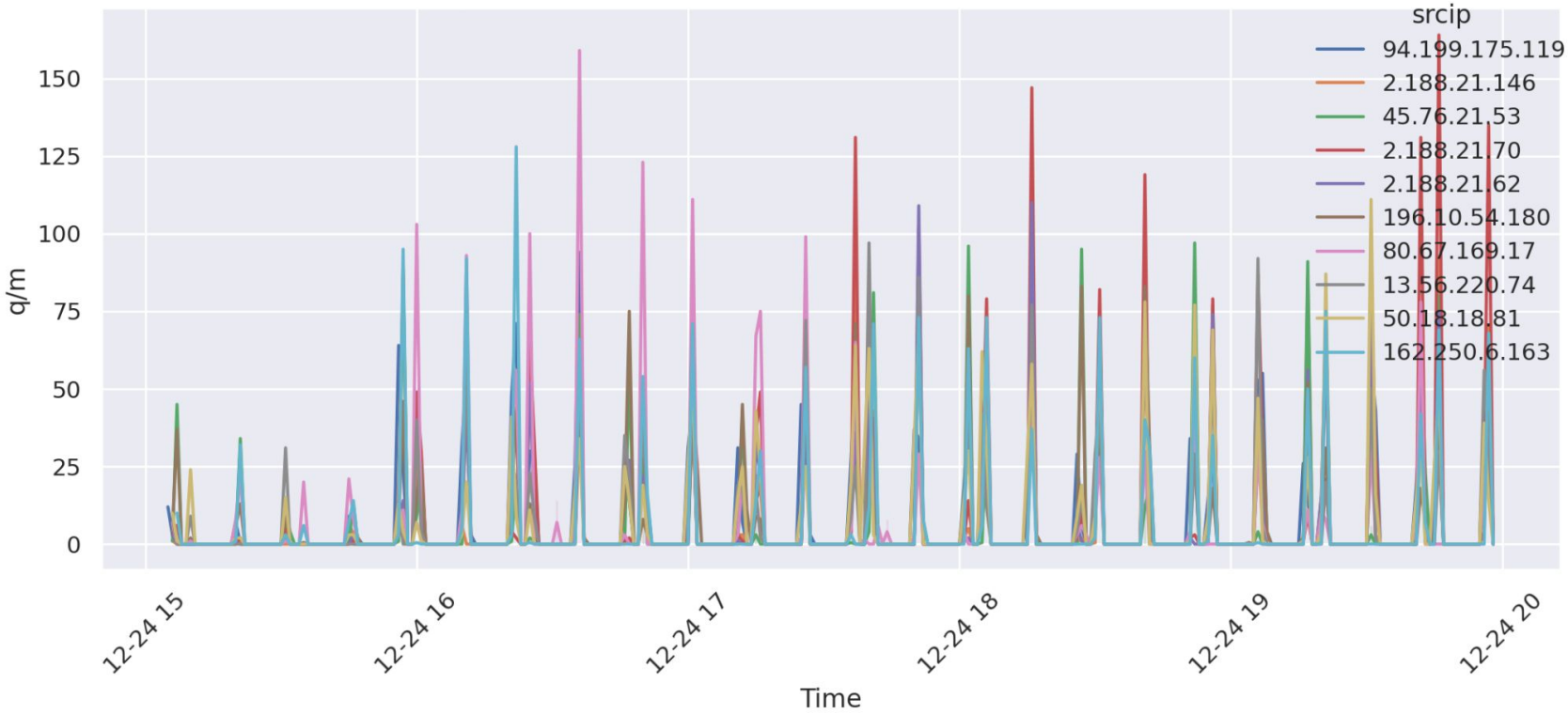


Total Traffic

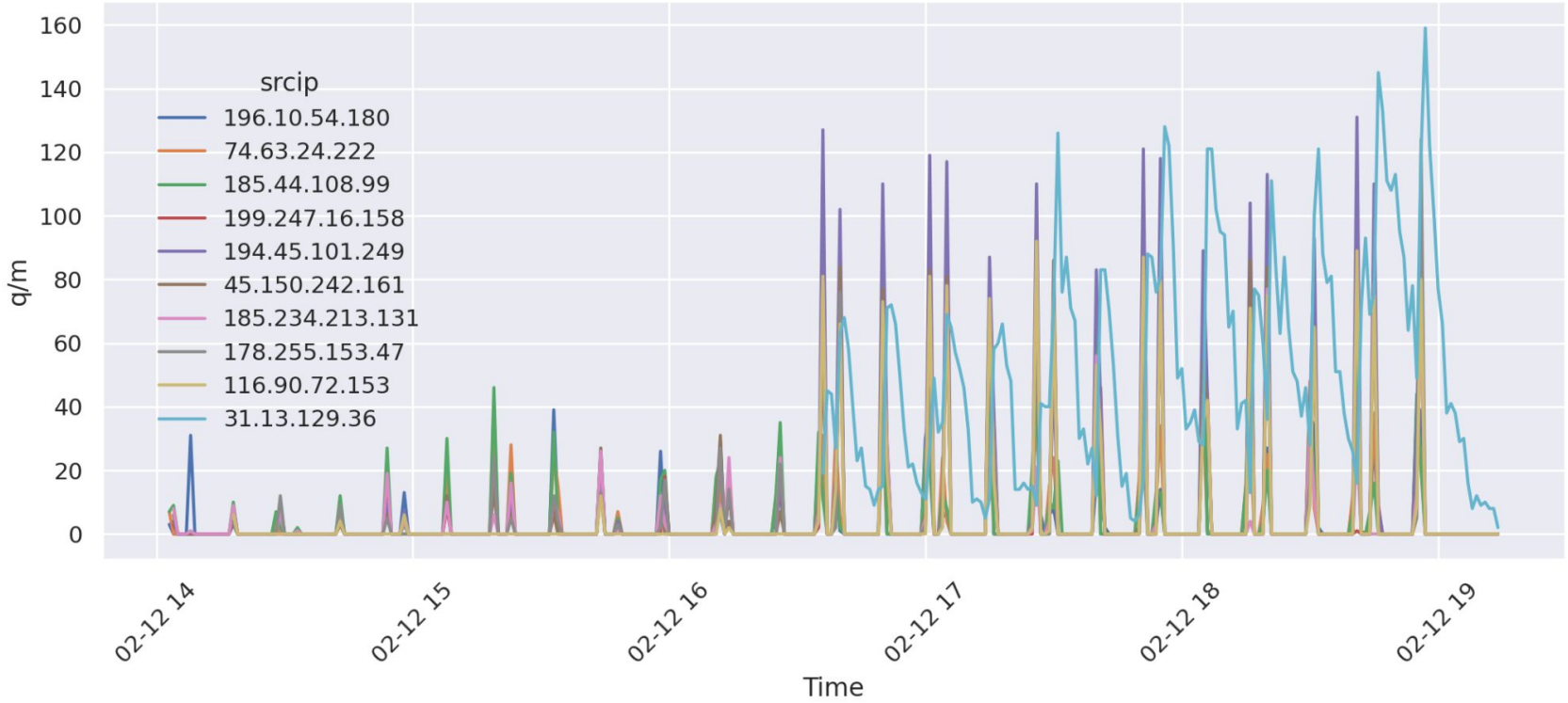


Shouldn’t be cached

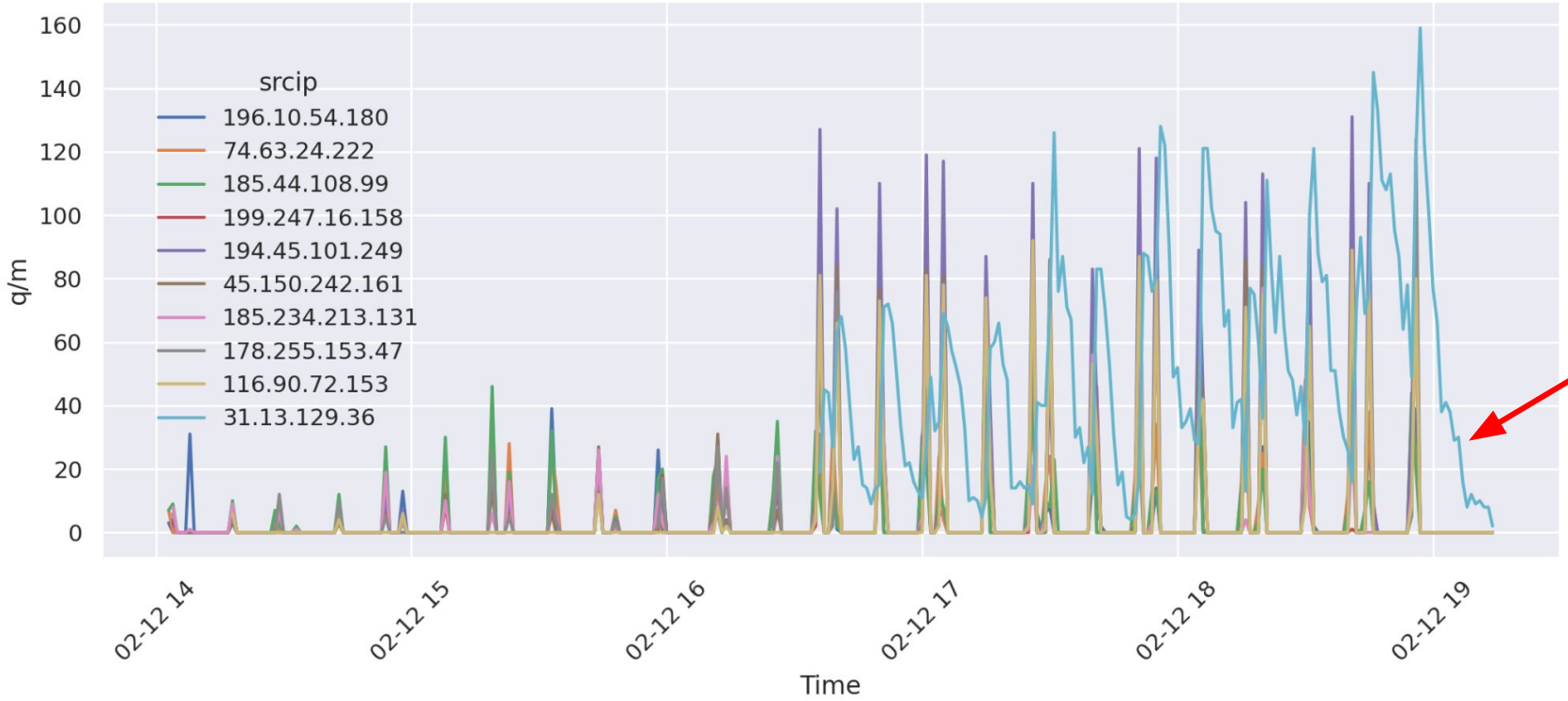
Top 10 Source IPs – My Second Run



Top 10 Source IPs – Final Run

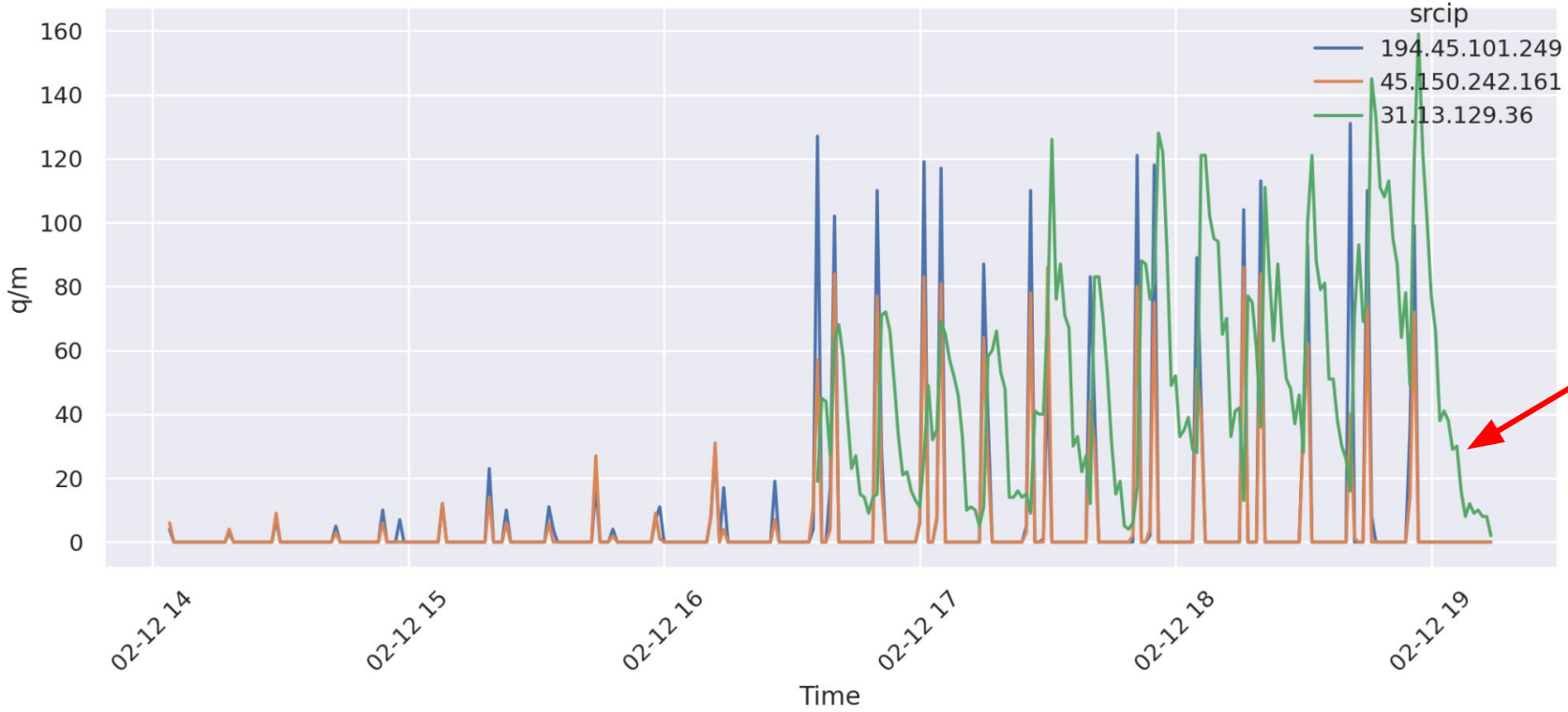


Top 10 Source IPs – Final Run



Ruh
Roh

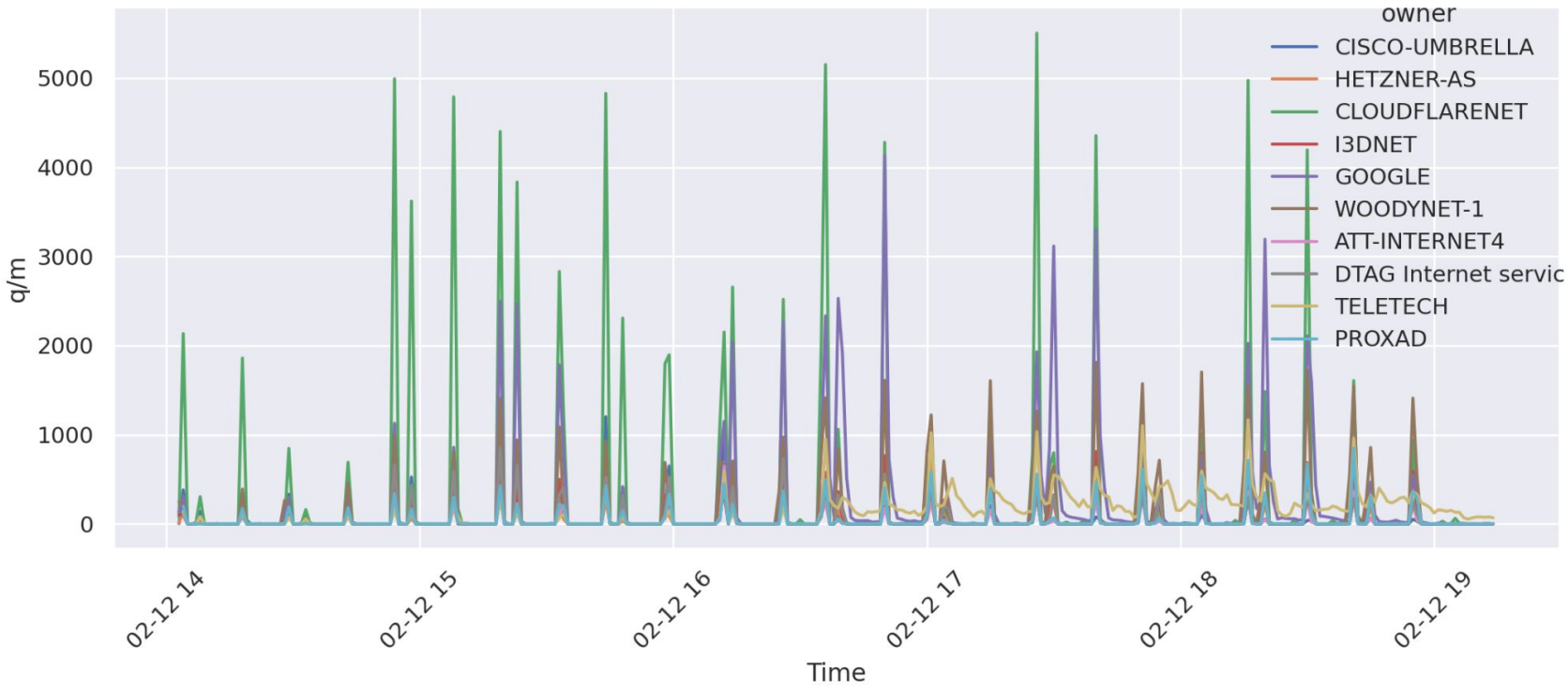
Top 3 Source IPs



Where is it?

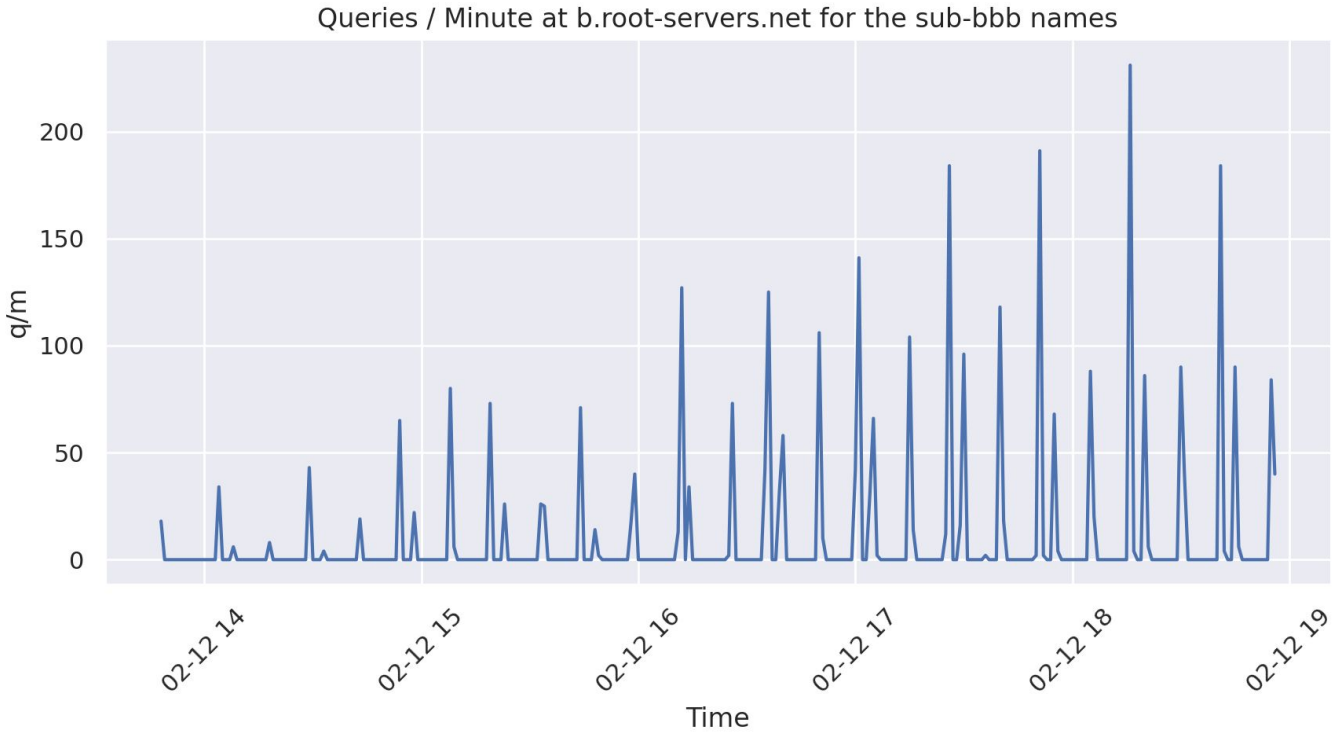
```
$ ip2asn 31.13.129.36

Address: 31.13.129.36

Numeric ip: 520978724

          ASN: 197765

        Owner: ITPARK_DC

      Country: RU

     ip_range: 31.13.128.0 - 31.13.135.255
```

Top 10 ASes

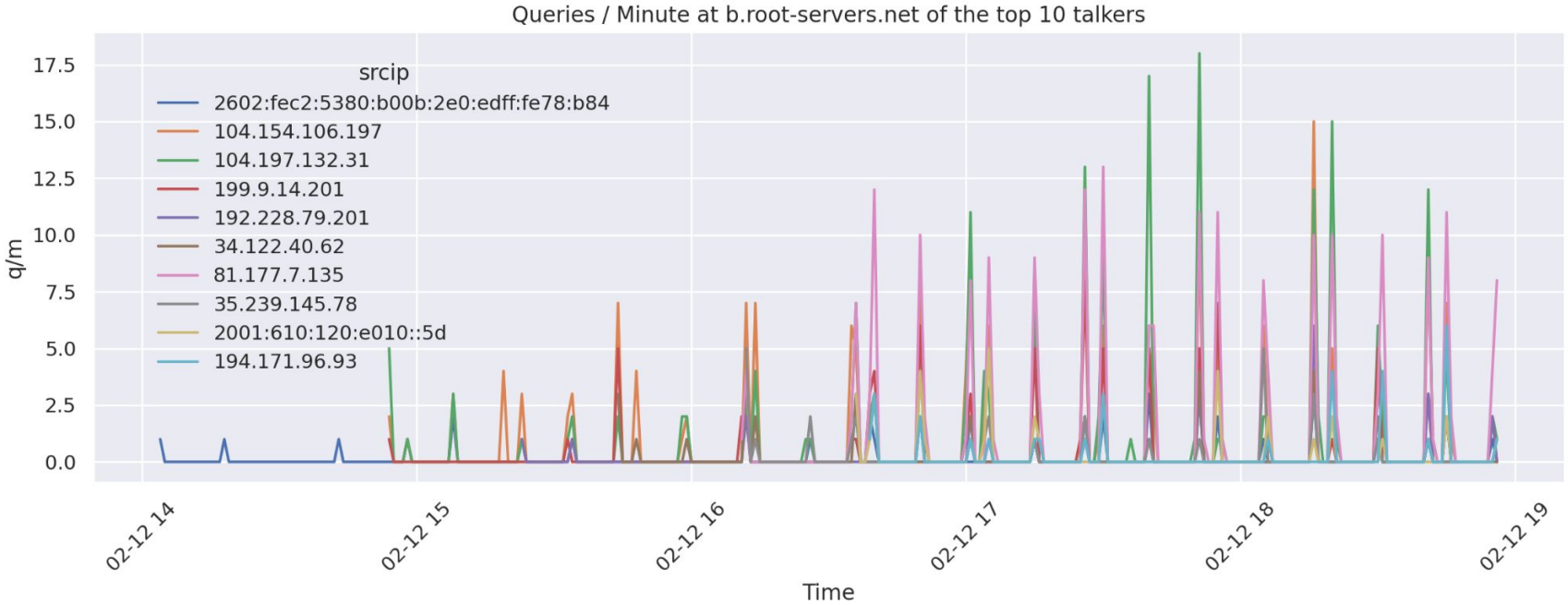

owner
CISCO-UMBRELLA
HETZNER-AS
CLOUDFLARENET
I3DNET
GOOGLE
WOODYNET-1
ATT-INTERNET4
DTAG Internet servic
TELETECH
PROXAD
q/m
5000
4000
3000
2000
1000
0
02-12 14
02-12 15
02-12 16
02-12 17
02-12 18
02-12 19
Time

B-Root Data

Checking requests seen at b.root-servers.net



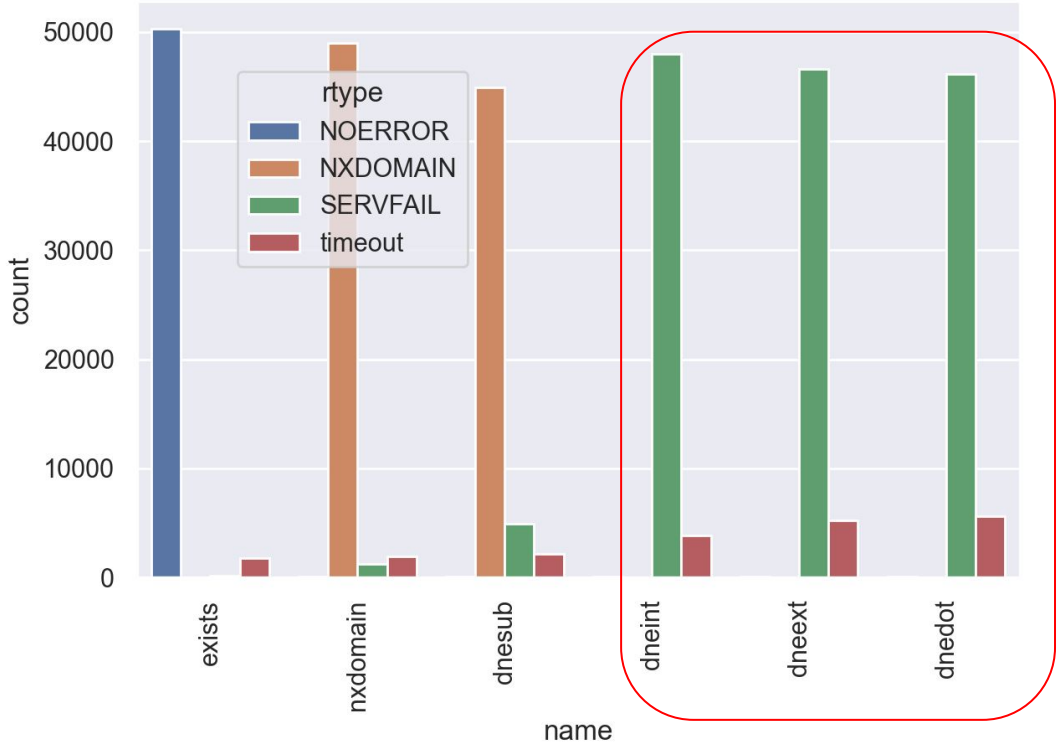
Checking requests seen at b.root-servers.net



Conclusions
aka… did we learn anything?

Conclusion 1: All error conditions cause more traffic

Conclusion 2: Broken auth DNS is the worst

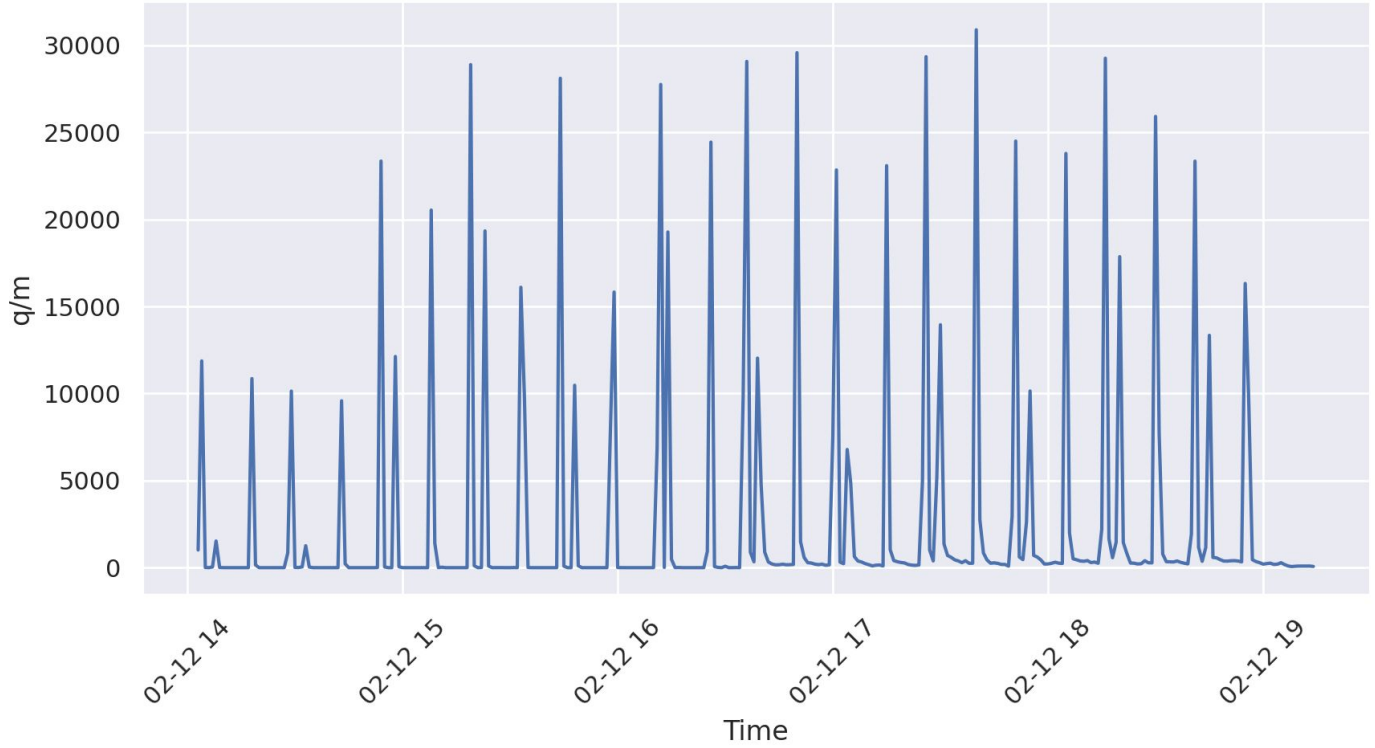


SERVFAILs aren't cached

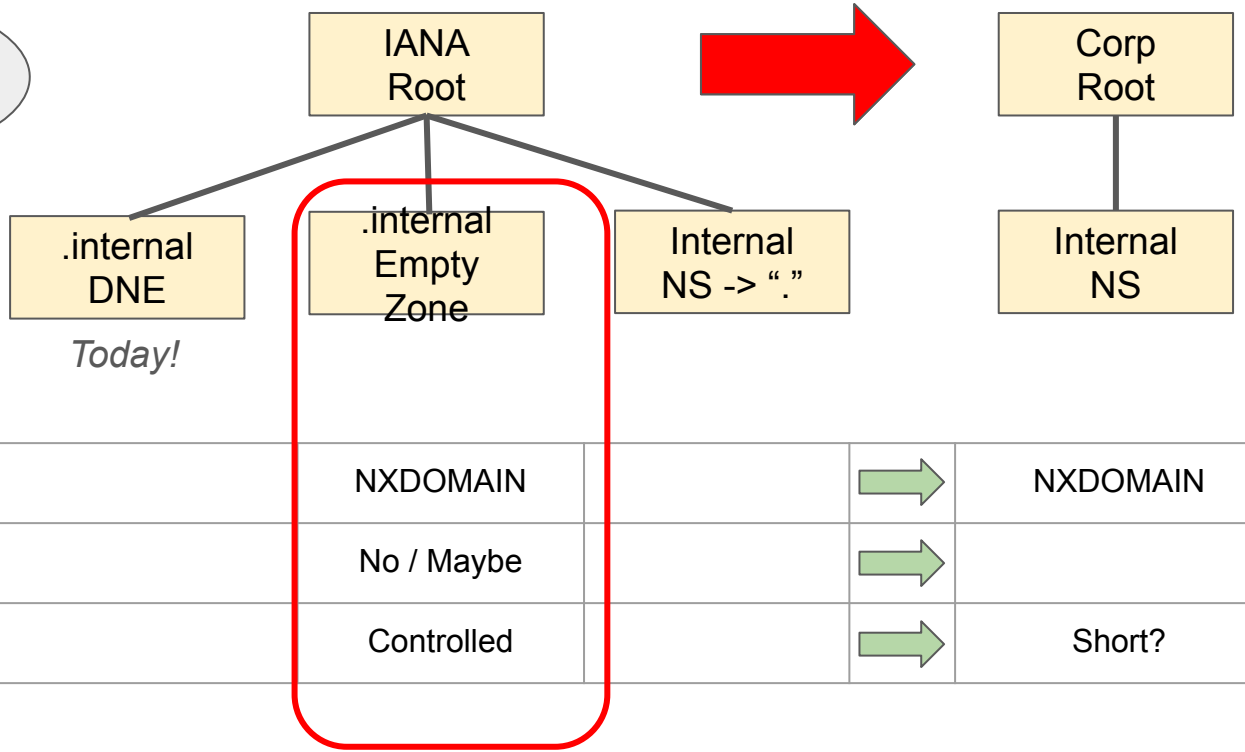
(future work)

Conclusion 3: NS => . is no worse than other errors

Conclusion 4: not much difference between .com / .gam[illegible]

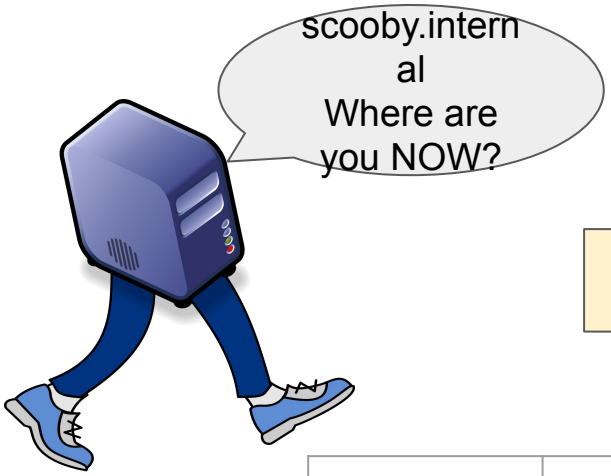


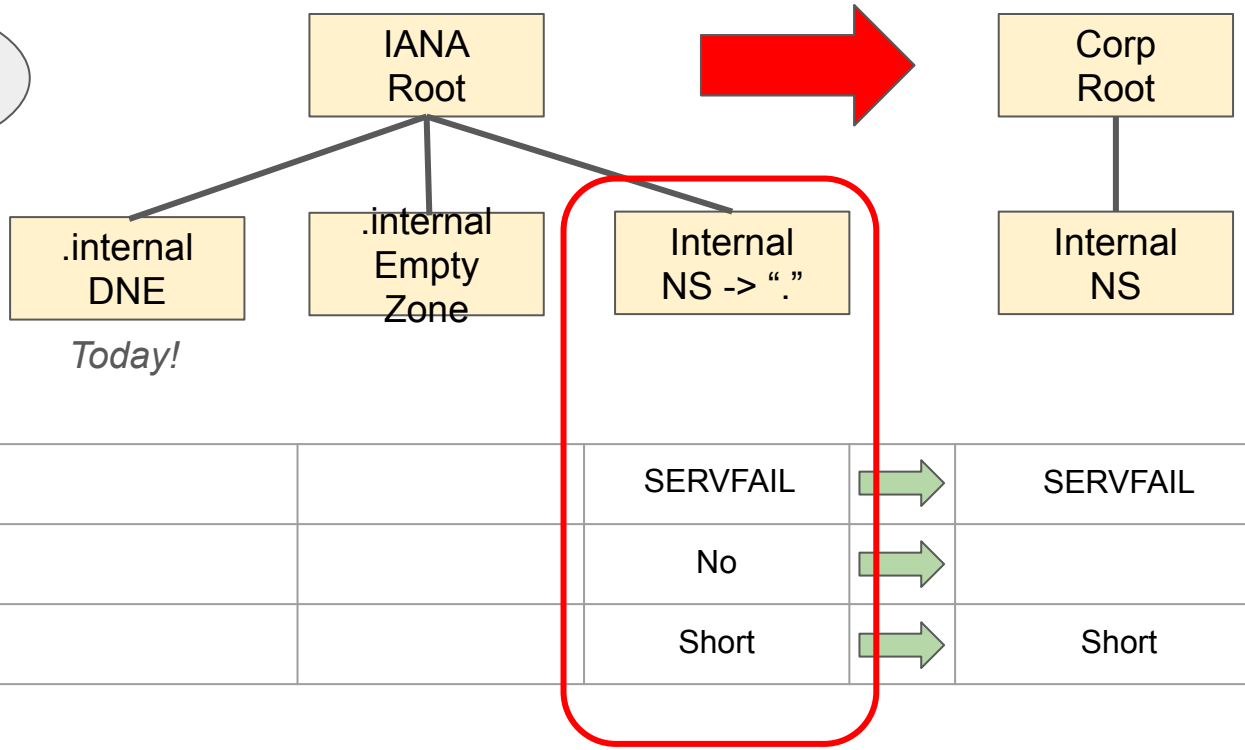
Conclusion 5: An empty one is best when you control NS



Response		NXDOMAIN		→	NXDOMAIN
Signed?		No / Maybe		→	
TTL		Controlled		→	Short?

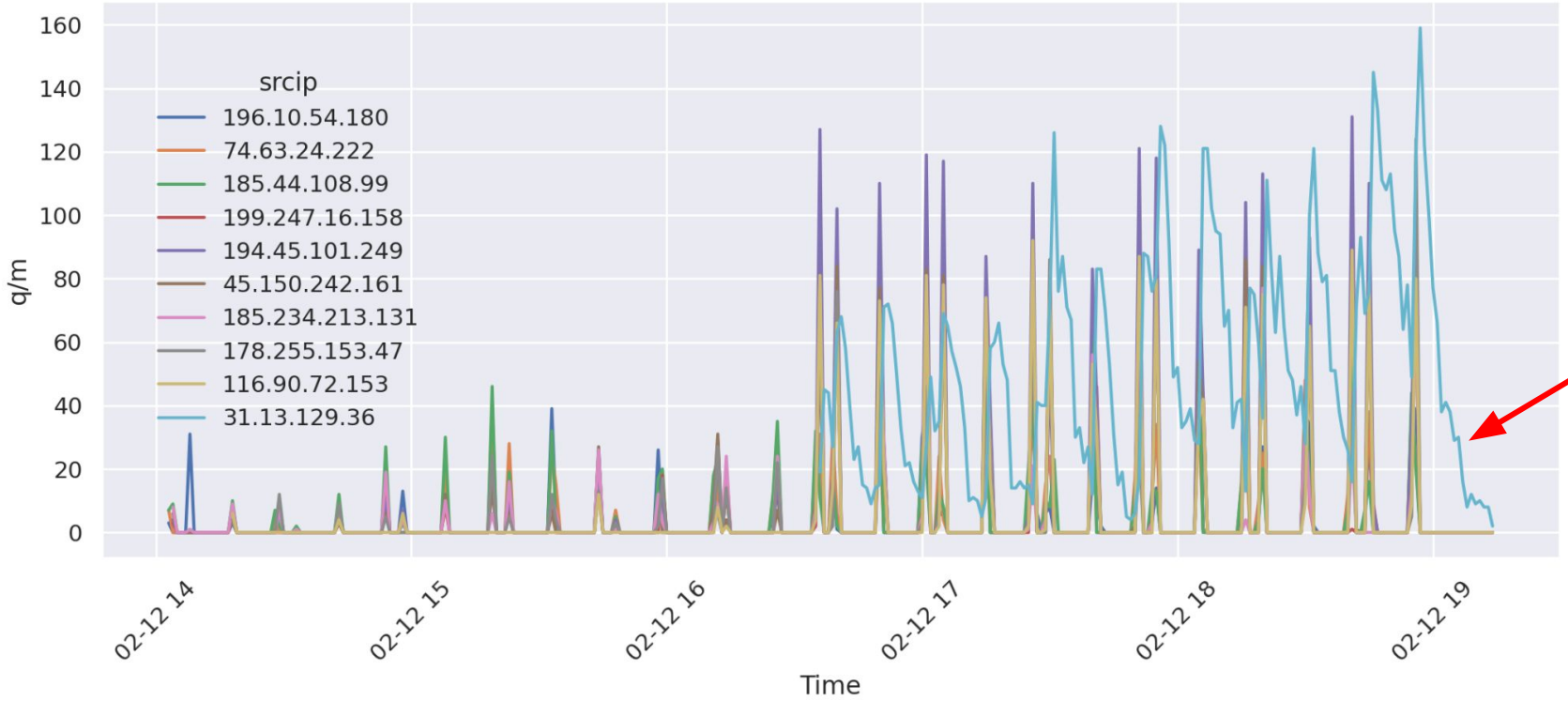
Conclusion 6: NS = “.” might be best otherwise?



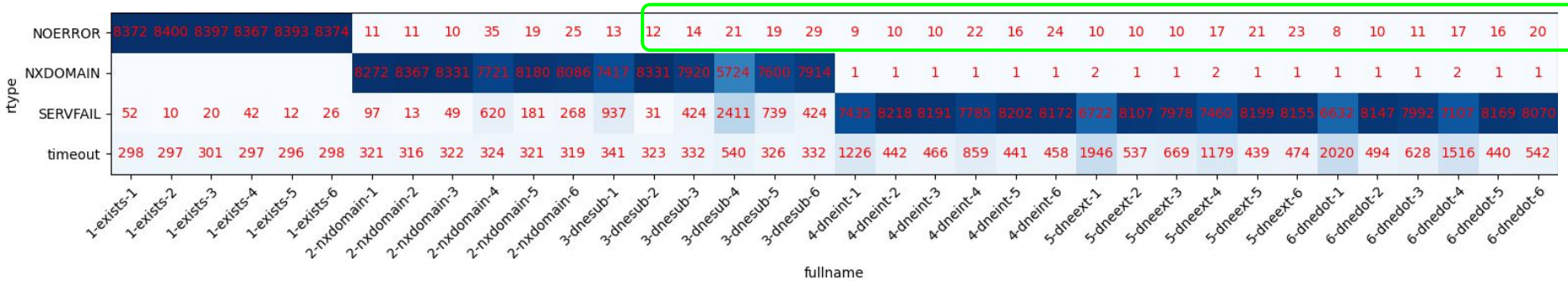


Response			SERVFAIL	→	SERVFAIL
Signed?			No	→	
TTL			Short	→	Short

Conclusion 7: Strange things always exist



Conclusion 8: Stranger things always exist



Conclusion 9: Even stranger things are afoot at the circle K

QName	QType	Count
ns1.sub-bbb.frostedaxe.games.	A6	12
ns1.sub-bbb.frostedaxe.com.	A6	10
exists.sub-bbb.frostedaxe.games.	CNAME	4
ns1.sub-bbb.frostedaxe.games.	MX	4
ns1.sub-bbb.frostedaxe.games.	TXT	2

Questions? *(Conclusion 7-9 excluded)*