TsuNAME vulnerability

Giovane C. M. Moura[1], Sebastian Castro[2],
John Heidemann[3], Wes Hardaker[3]

1: SIDN Labs, 2: InternetNZ, 3: USC/ISI

NS.SE Meetup

Virtual Meeting

2021-06-10

Introduction

- While working on a paper (2020), we obseved a strange behavior from Google Public DNS and `.nz`
- That later became TsuNAME
    - Resolver vulnerability that can be used for DDoS
- We carried out public, responsible disclosure
- We released source code
- Major parties fixed software
- More info on : `https://tsuname.io`

TL;DR slide

- TsuNAME is a vulnerability that can be used to DoS authoritative servers
- It requires three things:
    1. **Cyclic dependent** NS records
    2. **Vulnerable** resolvers
    3. User **queries** only to start/drive the process
- Problem: we've seen servers getting significant traffic for days
    - That's enough for going from 10qps to 5600qps (and more)
- To mitigate it:
    1. **Auth Ops**: detect cyclic records: use `CycleHunter`
        - BUT: difficult to prevent quick NS changes
    2. **Resolver Ops/Dev**: change resolvers
        - Google and Cisco fixed it
    3. (no way to prevent triggering queries)

What did we do?

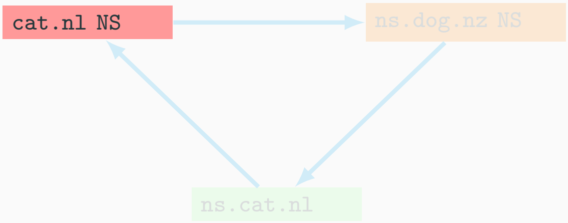
- We followed responsible disclosure guidelines

Date	Type	Group
2020-12-10	Private Disclosure	Google Notification
2020-12-10	Private Disclosure	SIDN DNSOPs
2021-02-05	Private Disclosure	OARC34
2021-02-22	Private Disclosure	APTLD
2021-02-22	Private Disclosure	NCSC-NL
2021-02-23	Private Disclosure	CENTR
2021-03-04	Private Disclosure	LACTLD
2021-02-18–2021-05-05	Private Disclosure	Private
2021-05-06	Public Disclosure	OARC35
2021-05-06	Public Disclosure	`https://tsuname.io`

Table 1: TsuNAME disclosure timeline

Cyclic Dependency is a loop; an error

- First described in Pappas2009 [1]

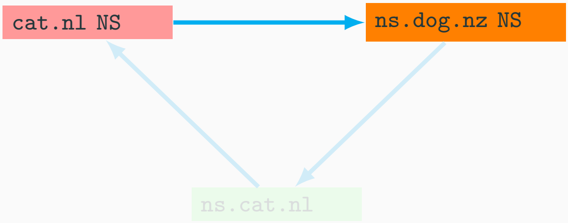


- Resolvers should return **SERVFAIL** , but some seem to loop a lot (huge amplification)

[1] Vasileios Pappas, Zhiguo Xu, Songwu Lu, Daniel Massey, Andreas Terzis, and Lixia Zhang. **Impact of configuration errors on DNS robustness**. SIGCOMM Comput. Commun. Rev., August 2004.

Cyclic Dependency is a loop; an error

- First described in Pappas2009 [1]

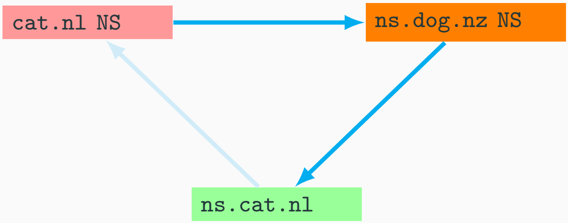


- Resolvers should return **SERVFAIL** , but some seem to loop a lot (huge amplification)

[1] Vasileios Pappas, Zhiguo Xu, Songwu Lu, Daniel Massey, Andreas Terzis, and Lixia Zhang. **Impact of configuration errors on DNS robustness**. SIGCOMM Comput. Commun. Rev., August 2004.

Cyclic Dependency is a loop; an error

- First described in Pappas2009 [1]

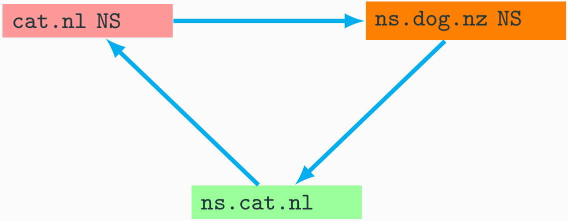


- Resolvers should return **SERVFAIL** , but some seem to loop a lot (huge amplification)

[1] Vasileios Pappas, Zhiguo Xu, Songwu Lu, Daniel Massey, Andreas Terzis, and Lixia Zhang. **Impact of configuration errors on DNS robustness**. SIGCOMM Comput. Commun. Rev., August 2004.

Cyclic Dependency is a loop; an error

- First described in Pappas2009 [1]

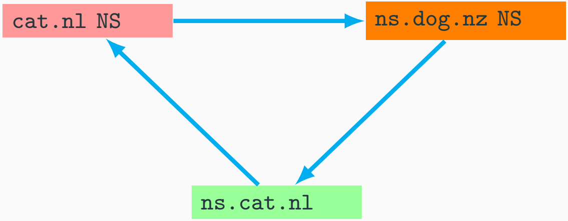


- Resolvers should return **SERVFAIL** , but some seem to loop a lot (huge amplification)

[1] Vasileios Pappas, Zhiguo Xu, Songwu Lu, Daniel Massey, Andreas Terzis, and Lixia Zhang. **Impact of configuration errors on DNS robustness**. SIGCOMM Comput. Commun. Rev., August 2004.

Cyclic Dependency is a loop; an error

- First described in Pappas2009 [1]



- Resolvers should return **SERVFAIL** , but some seem to loop a lot (huge amplification)

[1] Vasileios Pappas, Zhiguo Xu, Songwu Lu, Daniel Massey, Andreas Terzis, and Lixia Zhang. **Impact of configuration errors on DNS robustness**. SIGCOMM Comput. Commun. Rev., August 2004.

Cyclic Dependency is a loop; an error

- RFC1536 (1993)! mentioned the existence of such loops
  - We, however, show how it can be used for DDoS
- RFC1536 says that resolvers must “ bound the amount of work so a request can’t get into an infinite loop"
- We add that resolvers **must implement negative caching**, so subsequent queries don’t trigger extra queries

Cyclic Dependency (zone files)

`.nl` zone:

- `cat.nl NS ns1.dog.nz`

`.nz` zone

- `dog.nz NS ns1.cat.nl`
- as a TLD operator, you **cannot** know it just by analyzing your zone locally
- you have to query NS records (we have `CycleHunter` for that)

Cyclic Dependency (zone files)

`.nl` zone:

- `cat.nl NS ns1.dog.nz`

`.nz` zone

- `dog.nz NS ns1.cat.nl`
- as a TLD operator, you **cannot** know it just by analyzing your zone locally
- you have to query NS records (we have `CycleHunter` for that)

Cyclic Dependency (zone files)

`.nl` zone:

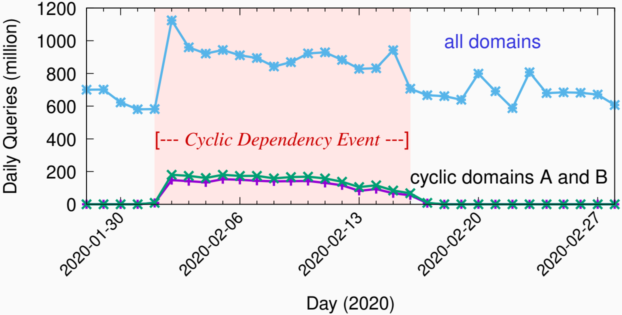
- `cat.nl NS ns1.dog.nz`

`.nz` zone

- `dog.nz NS ns1.cat.nl`
- as a TLD operator, you **cannot** know it just by analyzing your zone locally
- you have to query NS records (we have `CycleHunter` for that)

TsuNAME `.nz` event: traffic surged

- On 2020-02-01, two `.nz` domains (A and B) were misconfigured with cyclic dependency
- Total traffic **surged 50%**



Domains A and B: from 30k queries to 334M tops (x10^4)

Where these resolvers come from?

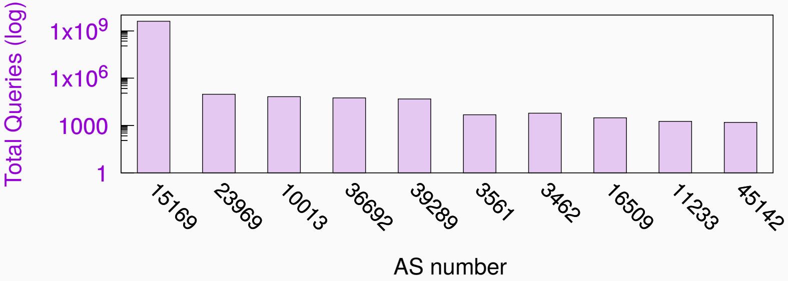


Figure 1: Queries for cyclic domains: 99% from Google (AS15169)

Where these resolvers come from?

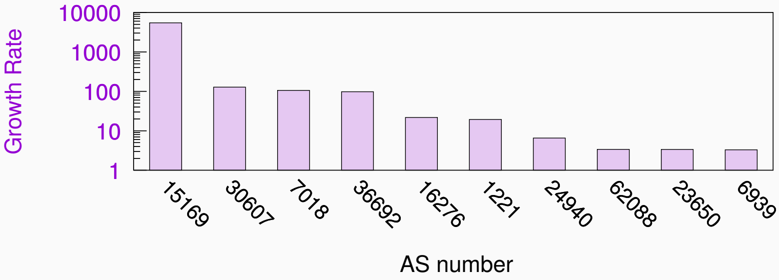


Figure 2: Traffic increase

- Traffic increase: queries during event / queries during “normal” period
- Both cover 16 days

AS list of `.nz` TsuNAME event

AS Number	AS name	Country
15169	Google	US
23969	TOT Public Company Limited	Thailand
10013	FreeBit	Japan
36692	Cisco OpenDNS	US
39289	MediaSeti	Russia
3561	CENTURYLINK-LEGACY-SAVVIS	US
3452	University of Alabama at Birmingham	US
16509	Amazon, Inc	US
11233	Gorge Networks	US
45142	Loxley Wireless	Thailand
200050	ITSVision	France
30844	Liquid Telecom	UK
15267	702 communications	US

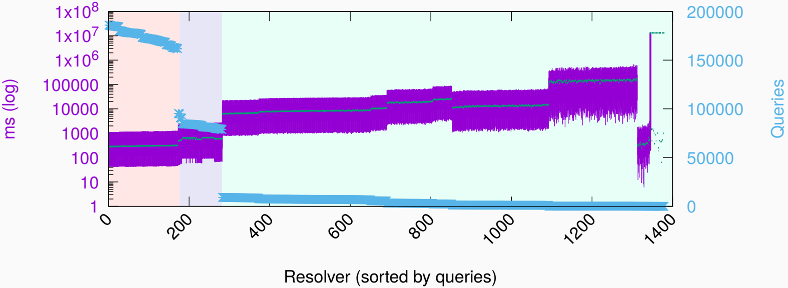
Table 2: List of top ASes per volume of queries during experiments.

What Google Resolvers asked ?

Query Name	Query Type	Queries(v4)	Queries(v6)
DomainA.nz	NS	13.0M	10.9M
DomainB.nz	NS	4.3M	3.0M
ns1.DomainA.nz	A	266.1M	281.3M
	AAAA	266.2M	281.4M
ns2.DomainA.nz	A	266.1M	281.2M
	AAAA	266.1M	281.4M
ns1.DomainB.nz	A	222.6M	237.9M
	AAAA	222.5M	237.7M
ns2.DomainB.nz	A	222.5M	237.7M
	AAAA	222.3M	237.5M

Table 3: Google queries during the TsuNAME event

How often Google sent queries to `.nz`?



Three groups of resolvers

- Heavy hitters: every 300ms
- Modetare hitters: every 600ms
- Rest: $>$ 1 s

The Real Threat

- `.nz` saw a 50% traffic surge due to 2 misconfigured domains
- **The threat:**
  - Adversary holds multple domains (register or already has)
  - then change their NS records (create cycles)
  - then query from a botnet (inject queries)

  That got us very **concerned**.
- How many anycast providers could withstand that?
- How many TLDs would remain up?
- That's why we are disclosing this here

The Real Threat

- `.nz` saw a 50% traffic surge due to 2 misconfigured domains
- **The threat:**
    - Adversary holds multple domains (register or already has)
    - then change their NS records (create cycles)
    - then query from a botnet (inject queries)

  That got us very **concerned**.
- How many anycast providers could withstand that?
- How many TLDs would remain up?
- That's why we are disclosing this here

The Real Threat

- `.nz` saw a 50% traffic surge due to 2 misconfigured domains
- **The threat:**
  - Adversary holds multple domains (register or already has)
  - then change their NS records (create cycles)
  - then query from a botnet (inject queries)

  That got us very **concerned**.
- How many anycast providers could withstand that?
- How many TLDs would remain up?
- That's why we are disclosing this here

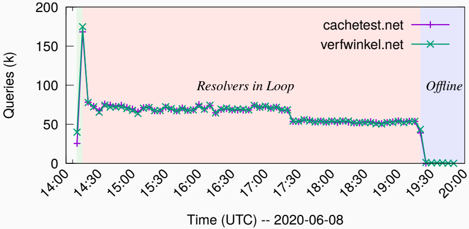
Was this an isolated event?

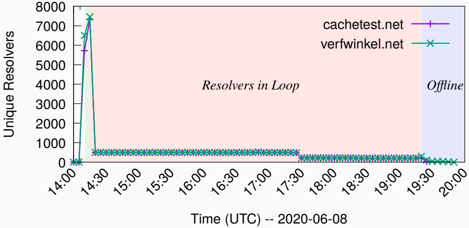
No: we managed to reproduce it multiple times

1. Lower bound with 1 query/resolver from Ripe Atlas
2. Influence of recurrent queries with Ripe Atlas
3. Domain without Atlas queries

Some resolvers will loop without user queries

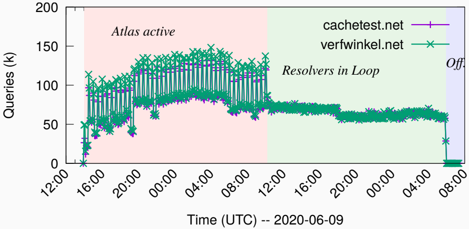
- 10k Ripe Atlas : 1 query to their local resolvers
- View from Auth Servers

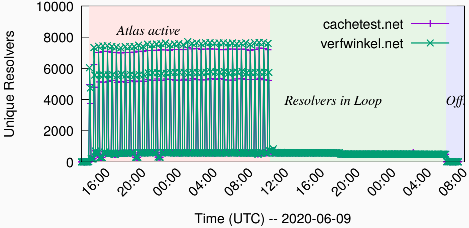




Recurrent Queries Amplify the Problem

- 10k Ripe Atlas : 1 query every 10min to local resolvers
- View from Auth Servers





What can we do prevent this?

- We don’t know how **big** a DDoS can get with this
  - We did not measure this: that’d be vandalism

1. Fix Resolvers: (**notification**)
   - We notified Google and Cisco OpenDNS; **they both fixed it**
   - Notified top 10 ASes, only 3 responded.
     - Two were running old DNS software: 2008 (MS) and 2015 (PowerDNS) versions
2. Auth OPs: **prevention**:
   - remove cyclic dependencies from zone files with `CycleHunter`, our open-source tool

What can we do prevent this?

- We don’t know how **big** a DDoS can get with this
    - We did not measure this: that’d be vandalism

1. Fix Resolvers: (**notification**)
    - We notified Google and Cisco OpenDNS; **they both fixed it**
    - Notified top 10 ASes, only 3 responded.
        - Two were running old DNS software: 2008 (MS) and 2015 (PowerDNS) versions
2. Auth OPs: **prevention**:
    - remove cyclic dependencies from zone files with `CycleHunter`, our open-source tool

What can we do prevent this?

- We don’t know how **big** a DDoS can get with this
  - We did not measure this: that’d be vandalism

1. Fix Resolvers: (**notification**)
   - We notified Google and Cisco OpenDNS; **they both fixed it**
   - Notified top 10 ASes, only 3 responded.
     - Two were running old DNS software: 2008 (MS) and 2015 (PowerDNS) versions
2. Auth OPs: **prevention**:
   - remove cyclic dependencies from zone files with `CycleHunter`, our open-source tool

CycleHunter

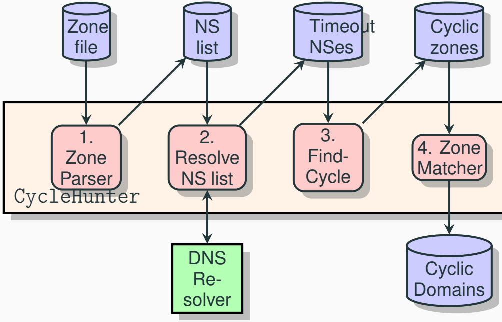


Figure 3: `CycleHunter` workflow

- We release it at: https://tsuname.io

Not many cyclic dependencies in the wild, ATM

zone	Size	NSSet	Cyclic	Affec.	Date
.com	151445463	2199652	21	1233	2020-12-05
.net	13444518	708837	6	17	2020-12-10
.org	10797217	540819	13	121	2020-12-10
.nl	6072961	79619	4	64	2020-12-03
.se	1655434	27540	0	0	2020-12-10
.nz	718254	35738	0	0	2021-01-11
.nu	274018	10519	0	0	2020-12-10
Root	1506	115	0	0	2020-12-04
Total	184409371	3602839	44	1435	

Table 4: `CycleHunter`: evaluated DNS Zones

- Human error plays a role

We found a parked `.nl` domain: it lasted for months

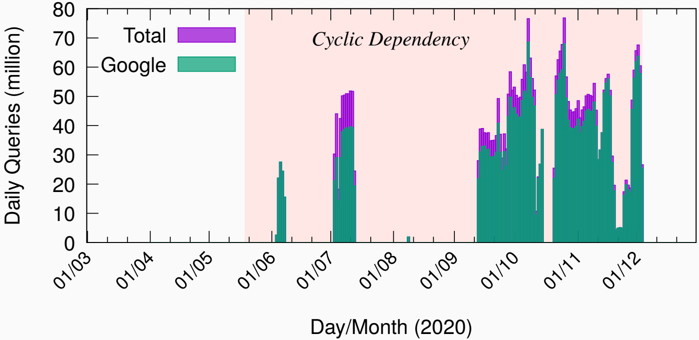


Figure 4: Timeseries of queries – it started on 2020-05-19

- From 300 daily queries to up to 75M (massive increase)
- This was an **accident**, a config. error
- We notified the registrar, who fixed it, queries return to 300/day

We found a parked `.nl` domain: it lasted for months

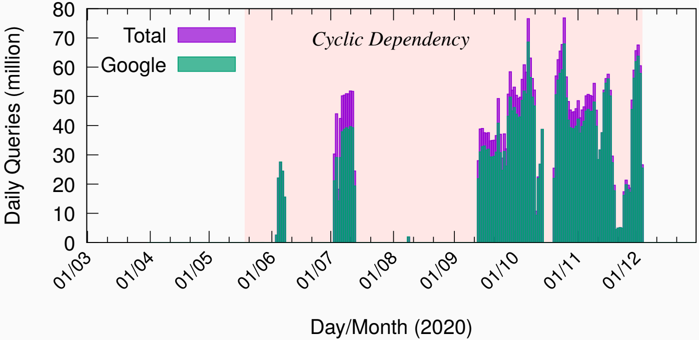


Figure 4: Timeseries of queries – it started on 2020-05-19

- From 300 daily queries to up to 75M (massive increase)
- This was an **accident**, a config. error
- We notified the registrar, who fixed it, queries return to 300/day

We evaluated other resolver software too

- No recurring cycles with these (they stop):
  - Unbound
  - BIND
  - PowerDNS
  - Public DNS: Quad1,Quad9
- But we don't know what other other ASes are running
- Whatever they are running, expect a long time to be fixed
- Looping old resolvers:
  - PowerDNS 3.6.2-2, from 2014 [1]
  - Windows 2008R2.

Shared materials on `https://tsuname.io`

- Technical Report
- Security Advisory
- `CycleHunter`

What have we learned since the private disclosure?

1. Longer cycles (triple) cause even more problems

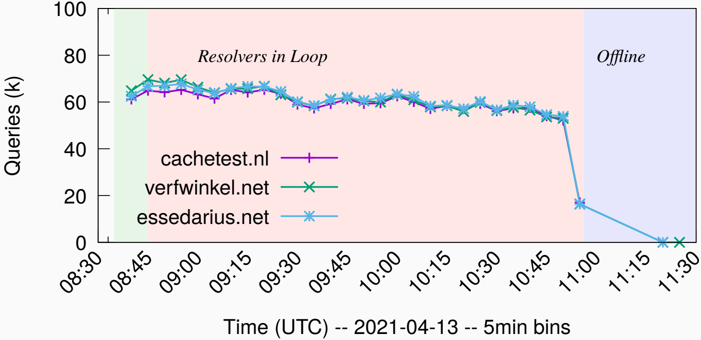


Figure 5: TripleDep measurement: Queries to authoritative servers (5min bins)

What have we learned since the private disclosure?

2. CNAME cycles are not as problematic

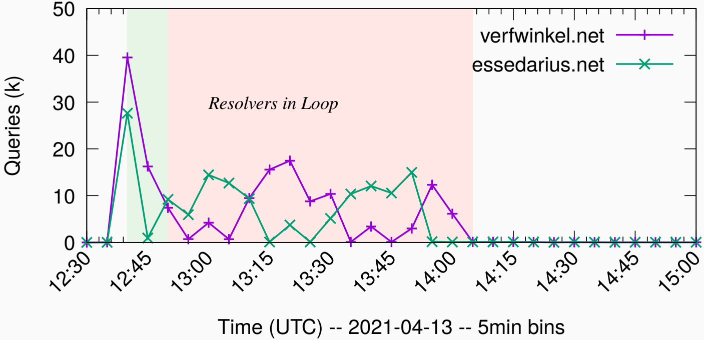


Figure 6: CNAME measurement: Querie to authoritative servers (5min bins)

What have we learned since the private disclosure?

3. Other ccTLDs have seen such events too

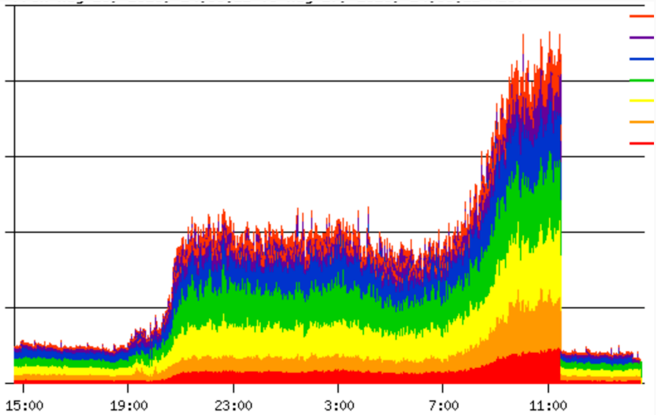


Figure 7: TsuNAME event at an Anonymous EU-based ccTLD operator.

What have we learned since the private disclosure?

What have we learned since the private disclosure?

5. **We identified the root causes of looping:**
    - Some resolvers will **loop** indefinitely (∞)
    - Others won’t loop, but they **won’t cache**: every new client query trigger new queries

The fix: **detect the loop, and cache it.**

What have we learned since the private disclosure?

6. We confirmed Google fixed its Public DNS

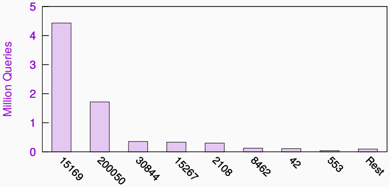


Figure 8: Measurement **BEFORE** Google fix

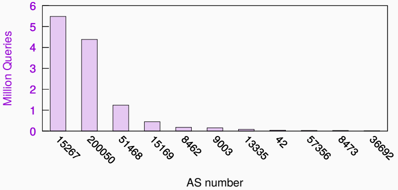


Figure 9: Measurement **AFTER** Google fix

Question: if I run `CycleHunter` once a day, will I be OK?

- **No**
- Changes may occur at any time:
  - `cat.nl NS ns1.dog.nz`
  - `ns1.dog.nz A 192.168.1.1`

  5 min later:
  - `cat.nl NS ns1.dog.nz`
  - `ns1.dog.nz NS ns1.dog.nl`
- This will find problems at point in time
- There is no continuous solution

Question: if I run `CycleHunter` once a day, will I be OK?

- **No**
- Changes may occur at any time:
  - `cat.nl NS ns1.dog.nz`
  - `ns1.dog.nz A 192.168.1.1`

  5 min later:
  - `cat.nl NS ns1.dog.nz`
  - `ns1.dog.nz NS ns1.dog.nl`
- This will find problems at point in time
- There is no continuous solution

Question: if I run `CycleHunter` once a day, will I be OK?

- **No**
- Changes may occur at any time:
  - `cat.nl NS ns1.dog.nz`
  - `ns1.dog.nz A 192.168.1.1`

  5 min later:
  - `cat.nl NS ns1.dog.nz`
  - `ns1.dog.nz NS ns1.dog.nl`
- This will find problems at point in time
- There is no continuous solution

Question: if I have cyclic dependencies, do I get DDoS'ed?

- **Maybe**
    - as in the `.nl` example, only having cyclic dependencies does not lead to DDoS per se
    - You'll need vulnerable resolvers to find you
    - We need someone to inject traffic
- An attacker can create these situations if they want

Question: if I have cyclic dependencies, do I get DDoS'ed?

- **Maybe**
    - as in the `.nl` example, only having cyclic dependencies does not lead to DDoS per se
    - You'll need vulnerable resolvers to find you
    - We need someone to inject traffic
- An attacker can create these situations if they want

Question: if I have cyclic dependencies, do I get DDoS'ed?

- **Maybe**
  - as in the `.nl` example, only having cyclic dependencies does not lead to DDoS per se
  - You'll need vulnerable resolvers to find you
  - We need someone to inject traffic
- An attacker can create these situations if they want

Question: I have RRL, so I'll be OK, right?

- No
    - RRL converts queries to TCP
    - Resolvers react to that by retrying heavily[2]
    - So they you have yet another amplification
- It may slow your attack, but it's not going to block it

[2] G. C. M. Moura, John Heidemann, Moritz Müller, Ricardo de O. Schmidt and Marco Davids. **When the Dike Breaks: Dissecting DNS Defenses During DDoS**. Proceedings of the 2018 ACM Internet Measurement Conference

Question: I have RRL, so I'll be OK, right?

- **No**
  - RRL converts queries to TCP
  - Resolvers react to that by retrying heavily[2]
  - So they you have yet another amplification
- It may slow your attack, but it's not going to block it

[2] G. C. M. Moura, John Heidemann, Moritz Müller, Ricardo de O. Schmidt and Marco Davids. **When the Dike Breaks: Dissecting DNS Defenses During DDoS**. Proceedings of the 2018 ACM Internet Measurement Conference

Question: I have RRL, so I'll be OK, right?

- **No**
  - RRL converts queries to TCP
  - Resolvers react to that by retrying heavily[2]
  - So they you have yet another amplification
- It may slow your attack, but it's not going to block it

[2] G. C. M. Moura, John Heidemann, Moritz Müller, Ricardo de O. Schmidt and Marco Davids. **When the Dike Breaks: Dissecting DNS Defenses During DDoS**. Proceedings of the 2018 ACM Internet Measurement Conference

Discussion

- If you're an **auth operato**r, check your zone
  - You can use `CycleHunter`
  - Don't forget about **collateral damage**
- if you're a **resolver op**/**dev,**
  - Detect cyclic dependencies and return SERVFAIL
  - Cache the SERVFAIL for future clients
  - Check your amplification factor

Slides and report :

- `https://tsuname.io/`

Discussion

- If you're an **auth operato**r, check your zone
    - You can use `CycleHunter`
    - Don't forget about **collateral damage**
- if you're a **resolver op/dev,**
    - Detect cyclic dependencies and return SERVFAIL
    - Cache the SERVFAIL for future clients
    - Check your amplification factor

Slides and report :

- `https://tsuname.io/`

References i

[1] PowerDNS.

Changelogs for all pre 4.0 releases.

https://doc.powerdns.com/recursor/changelog/pre-4.0.html, Jan. 2021.