Happy Eyeballs
A Good Servant or a Bad Master?

Radek Zajíc, radek@zajic.v.pytli.cz • RIPE NCC::Educa, 2020-06-08

about::myself

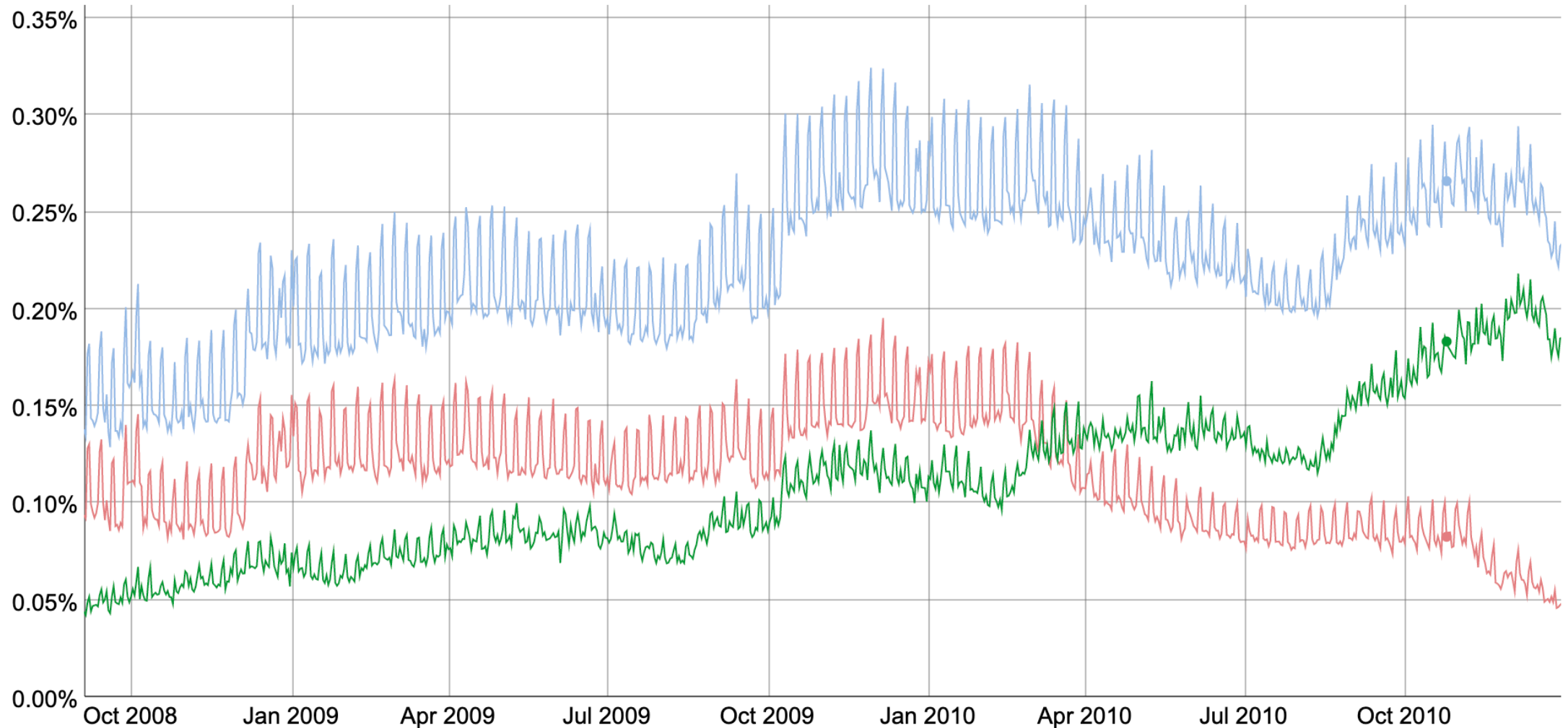Radek Zajíc

Blog: tech.showmax.com
Twitter: @ShowmaxDevs

Retrospective

IPv6 as we knew it back in 2008

Native: 0.18% 6to4/Teredo: 0.08% Total IPv6: 0.27% | 25. 10. 2010

IPv6 as we knew it back in 2008

6bone phased out 2006/6/6 (RFC 3701)

NAT-PT moved to Historic status in July 2007 (RFC 4966)

A6 DNS records were still a thing (for freaks)

Dual-Stack recommended in leaf networks (LANs)

6to4 and Teredo as the only IPv6 for many early adopters

Not a lot of content on IPv6 out there

IPv6 as we knew it back in 2008

Not a lot of content on IPv6 out there

`ipv6.google.com` launched 12 March 2008

`www.google.com IN AAAA` **would break users!**

Lorenzo Colitti, IPv6 at Google, RIPE 56, May 2008
https://meetings.ripe.net/ripe-56/presentations/Colitti-IPv6_at_Google.pdf
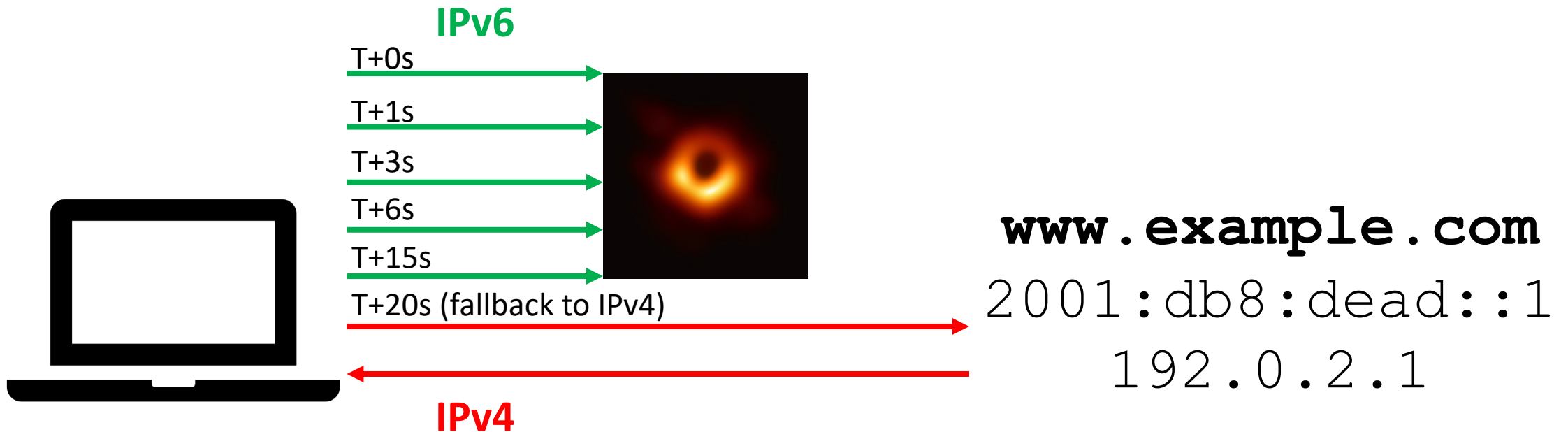
IPv6 as we know it today

Lots (but not all) of content on IPv6 out there

`www.google.com` IPv6-enabled on 6 June 2012

What has changed between 2008 and 2012?

IPv6 Brokenness

Connection brokenness in a nutshell

IPv6

T+0s

T+1s

T+3s

T+6s

T+15s

T+20s (fallback to IPv4)

IPv4

`www.example.com`
`2001:db8:dead::1`
`192.0.2.1`

Fallback can take from tens of seconds to tens of minutes, depending on a platform and number of AAAA records.

User's view: the service is unreachable

Decreasing use of automatic tunnels

Year	% of tunnelled traffic
2008	63,89%
2009	60,28%
2010	40,46%
2011	10,64%
2012	2,02%
2013	0,67%
2014	0,40%
2015	0,13%

Source: Google IPv6 statistics, https://www.google.com/intl/en/ipv6/statistics.html
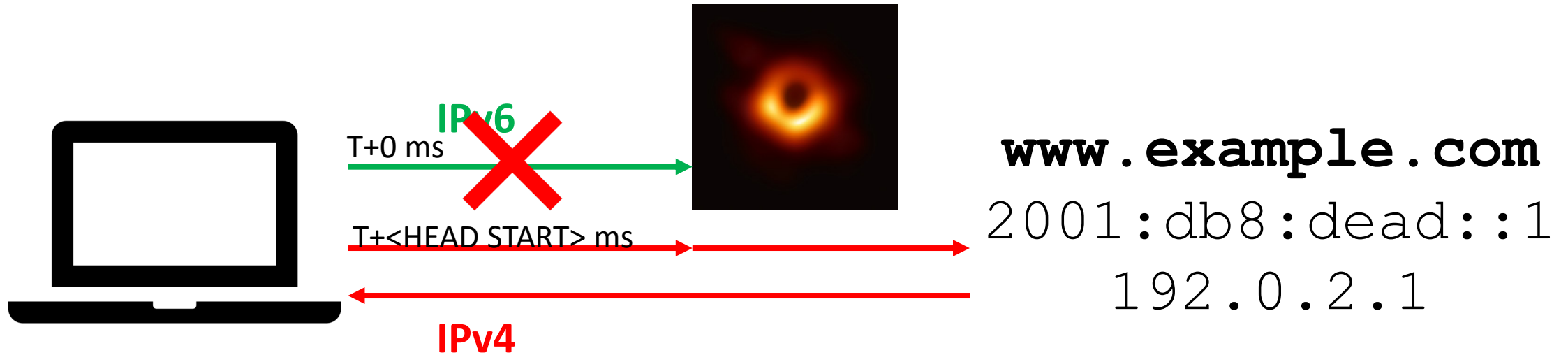
Some other types of brokenness

Routing issues (misconfigured/missing gateway, ACL, …)

Firewall, server misconfiguration (AAAA exists, server doesn't reply)
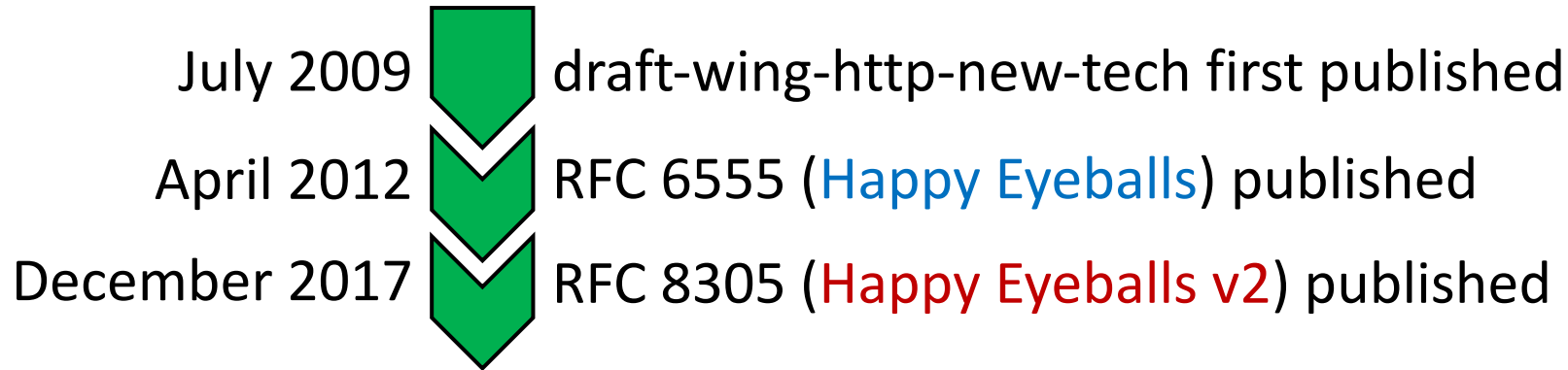
Path MTU issues (connection succeeds, transmission hangs)

Happy Eyeballs

Radek Zajíc, radek@zajic.v.pytli.cz, RIPE NCC::Educa

Happy Eyeballs (RFC 6555) in a nutshell

IPv6

T+0 ms

www.example.com

`2001:db8:dead::1`

T+<HEAD START> ms

`192.0.2.1`

IPv4

Client	Head start
Python `rfc6555`	0 ms
cURL	200 ms
Chrome, Firefox	300 ms
Safari (HE v2)	25 ms

Happy Eyeballs releases and support

July 2009 — draft-wing-http-new-tech first published

April 2012 — RFC 6555 (Happy Eyeballs) published

December 2017 — RFC 8305 (Happy Eyeballs v2) published

Google Chrome

Mozilla Firefox

cURL

Python `rfc6555`

Golang v1.12+

MacOS X, iOS

Happy Eyeballs

Environment	Issue	Resolution	Supported in
Any	PMTU issues	Not handled by Happy Eyeballs	N/A
Dual-Stack	Long TCP timeouts	Attempt to connect in parallel	v1 and v2, different timers
Dual-Stack	TCP connect blackholed or slow	Prefer **the other** IP protocol	v1 and v2
NAT64-only	Unreachable IPv4 address literals (1.1.1.1)	Perform DNS64 synthesis in the API, connect to the synthesized address	v2 only
NAT64-only	Dual-stack endpoint with unreachable IPv6	2s after failed connect attempt perform DNS64 synthesis using the IPv4 address, connect to the **synthesized** address	v2 only

Happy Eyeballs pitfalls

Most applications* support **neither v1 or v2**

Browsers** support **v1 only**

On IPv6-only access networks using DNS64/NAT64, the **benefits of HE v1 are erased**

HE v2 only supported on Cupertino platforms

* Unless it's Chrome, Firefox, Safari, Golang-based app or a proprietary app where the devs implemented HE
** Except Safari on MacOS X/iOS, which uses system-wide API that implements HE v2

Brokenness in 2020

Brokenness in 2020

Teredo and 6to4 are a niche now

Every now and then new **access networks** enable IPv6

Every now and then new **content providers** enable IPv6

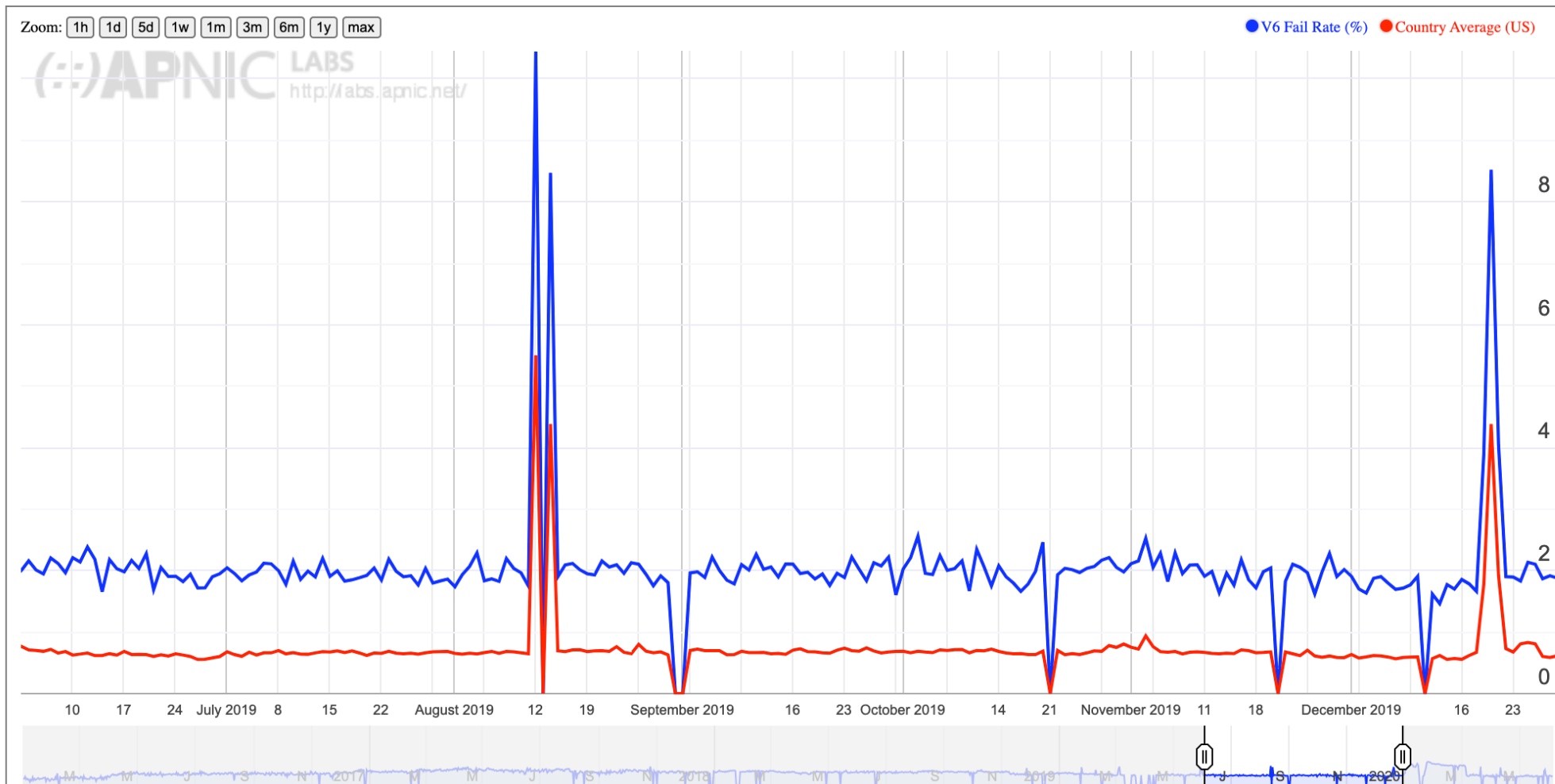Mistakes are natural and do happen

Happy Eyeballs v2 is not omnipresent

Users on NAT64-enabled network see the brokenness

But what are the brokenness levels, really?
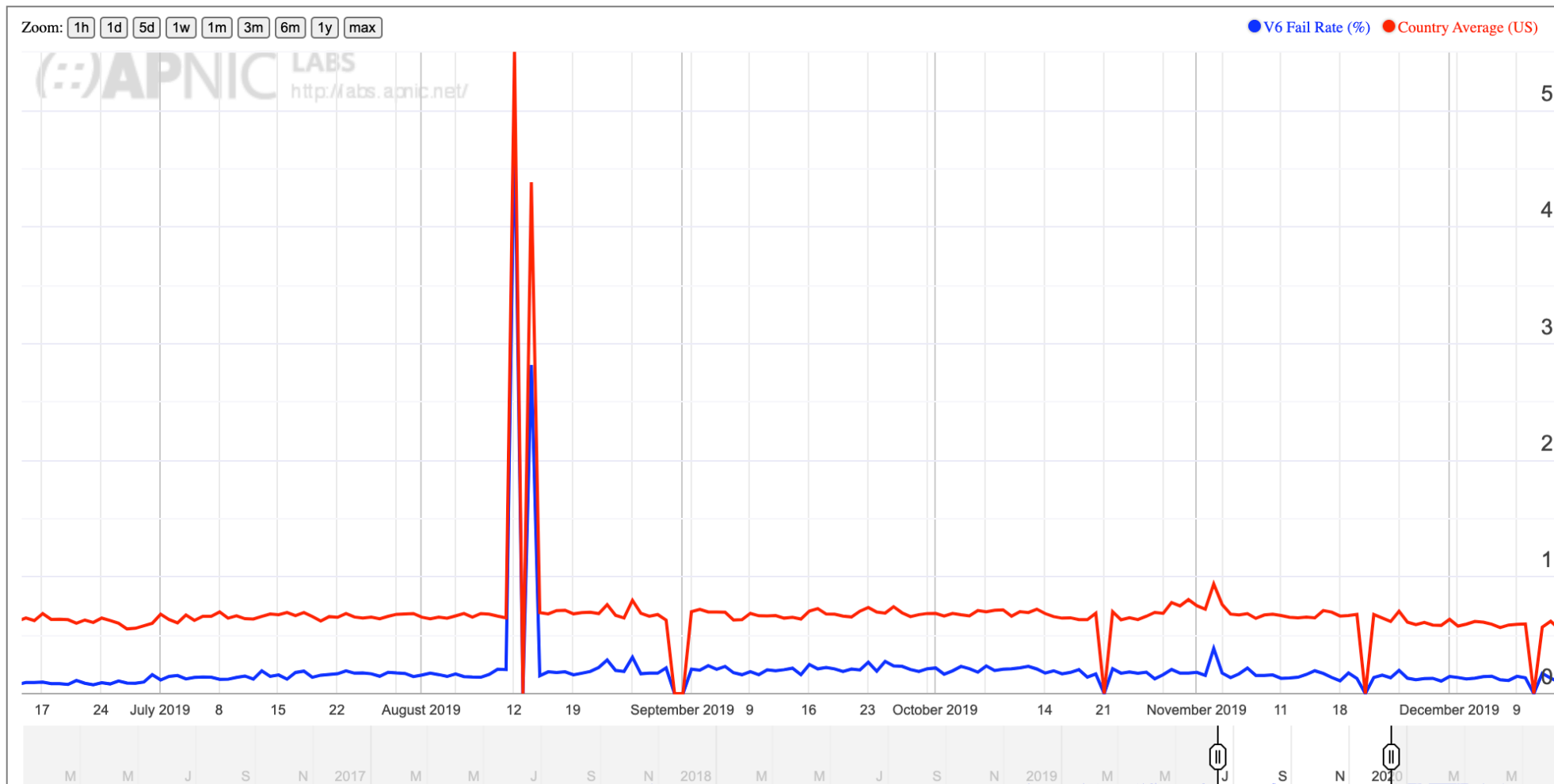
Brokenness in examples

V6 Connection Failure Rate for AS10796: TWC-10796-MIDWEST

Country average:
0.6 %

TWC:
2 %

https://stats.labs.apnic.net/v6perf/AS10796?a=10796&c=US&x=0&s=&g=0&t=1&s=1&f=1&w=30&d=now

Brokenness in examples

V6 Connection Failure Rate for AS21928: T-MOBILE-AS21928

Country average:

0.6 %

TM US:

0,1 %

https://stats.labs.apnic.net/v6perf/AS21928?a=21928&c=US&x=0&s=0&g=0&t=1&s=1&f=1&w=30&d=now

Brokenness in examples

Code	Region	Mean RTT Diff	Pair Samples	V6 Failure Rate ▼	V6 Samples
FJ	Fiji, Melanesia, Oceania	29.98	61	38.46%	117
CL	Chile, South America, Americas	-1.21	2,065	22.29%	3,051
FO	Faeroe Islands, Northern Europe, Europe	-10.72	174	18.75%	272
SD	Sudan, Northern Africa, Africa	-13.34	10	16.67%	12
CR	Costa Rica, Central America, Americas	-54.55	217	12.33%	300
UA	Ukraine, Eastern Europe, Europe	-2.13	6,032	8.36%	7,612
SG	Singapore, South-Eastern Asia, Asia	8.79	67,140	8.35%	91,207
GT	Guatemala, Central America, Americas	-42.52	30,715	8.02%	43,065
TZ	United Republic of Tanzania, Eastern Africa, Africa	26.27	31	7.62%	105
CN	China, Eastern Asia, Asia	98.97	600,491	7.60%	1,002,608
IQ	Iraq, Western Asia, Asia	27.63	45	6.67%	45
BD	Bangladesh, Southern Asia, Asia	24.69	1,143	6.58%	1,201
SA	Saudi Arabia, Western Asia, Asia	-11.61	158,081	5.84%	226,471

https://stats.labs.apnic.net/v6perf

Brokenness in examples

This major internet routing blunder took A WEEK to fix. Why so long? It was IPv6 – and no one really noticed

When you meant to type /127 but entered /12 instead

So Airtel AS9498 announced the entire IPv6 block 2400::/12 for a week and no-one notices until Tom Strickx finds out and they confirm it was a typo of /127.

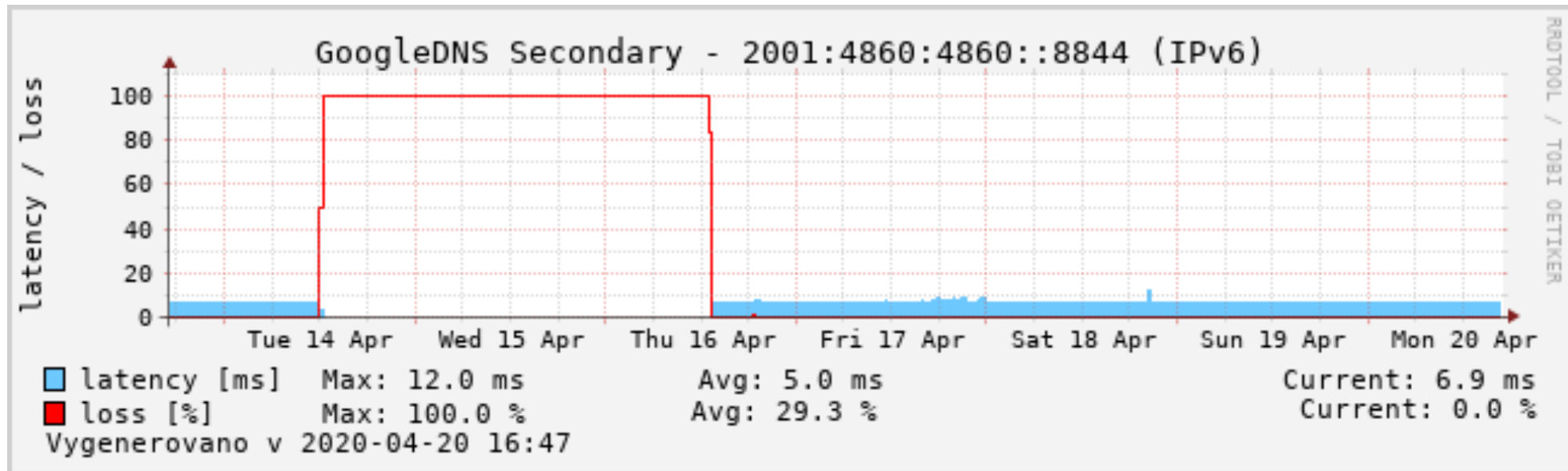https://www.theregister.com/2019/07/02/ipv6_routing_error/

Brokenness in examples

Conclusions

Should we abandon Happy Eyeballs?

Rather not. It's a good servant of ours, *masking* the brokenness.

Should we depend on it to *solve* the brokenness problem?

No! It's not everywhere. And v1 fails in DNS64/NAT64 networks.

Should we deploy more DNS64/NAT64 enabled end-user's networks?

If we ever want to turn IPv4 off, then **yes** is the answer.

Is IPv6 a second class citizen?

Sadly, yes. Monitoring is heavily under-estimated and neglected.

Do not wait for users to be the first to identify your misconfigurations.

Q & A
Thank you

@zajDee