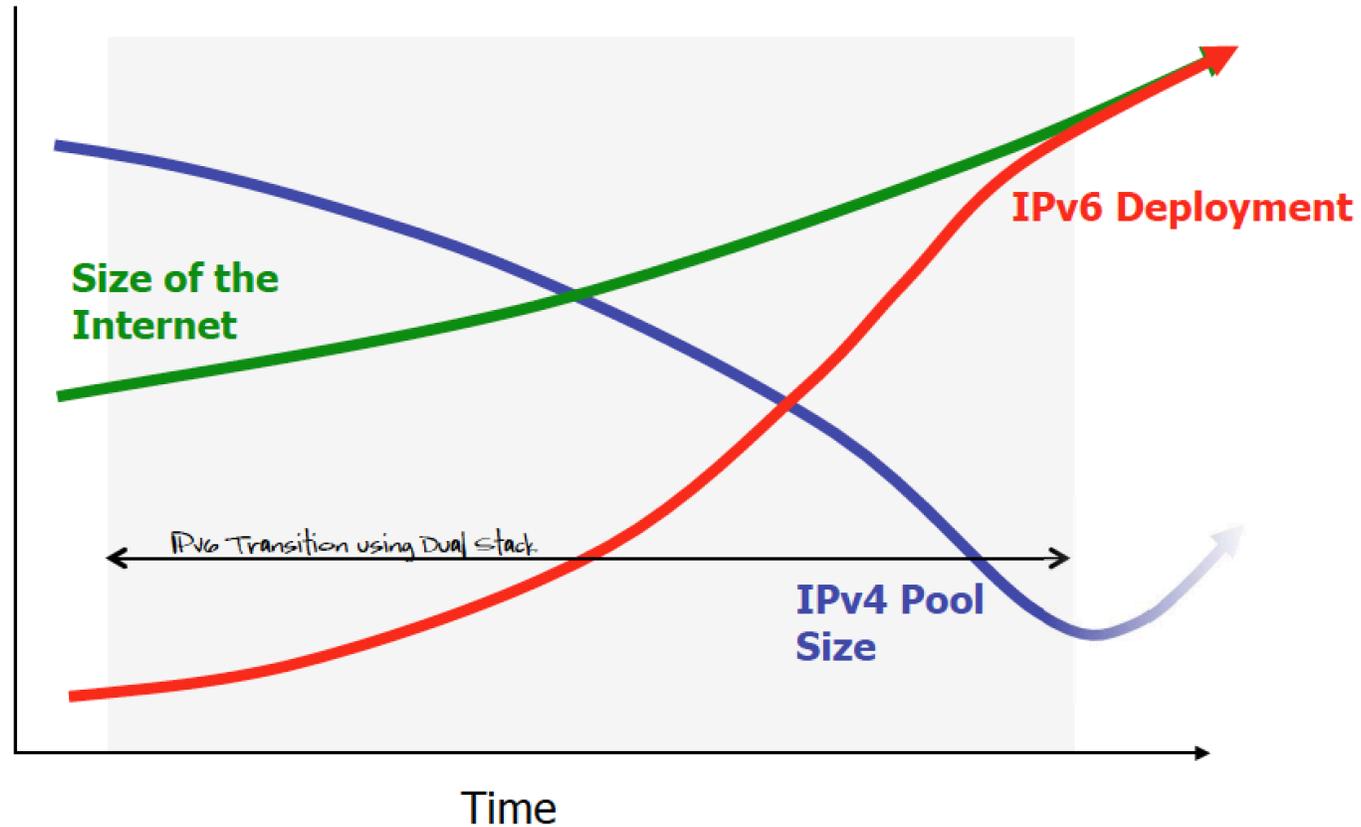


IPv6: Are we really
ready to turn off
IPv4?

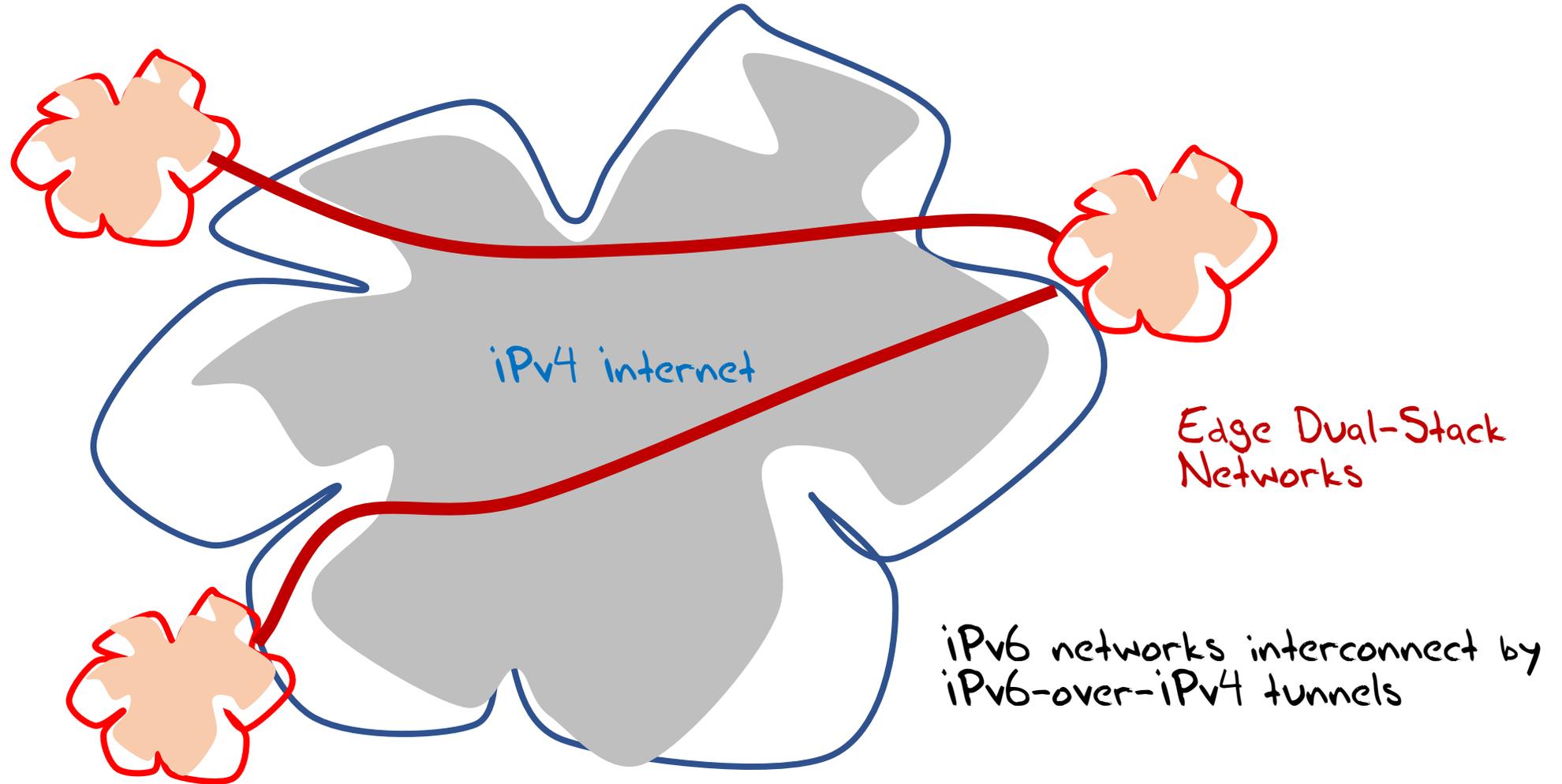
In-situ transition...

We had this plan ...



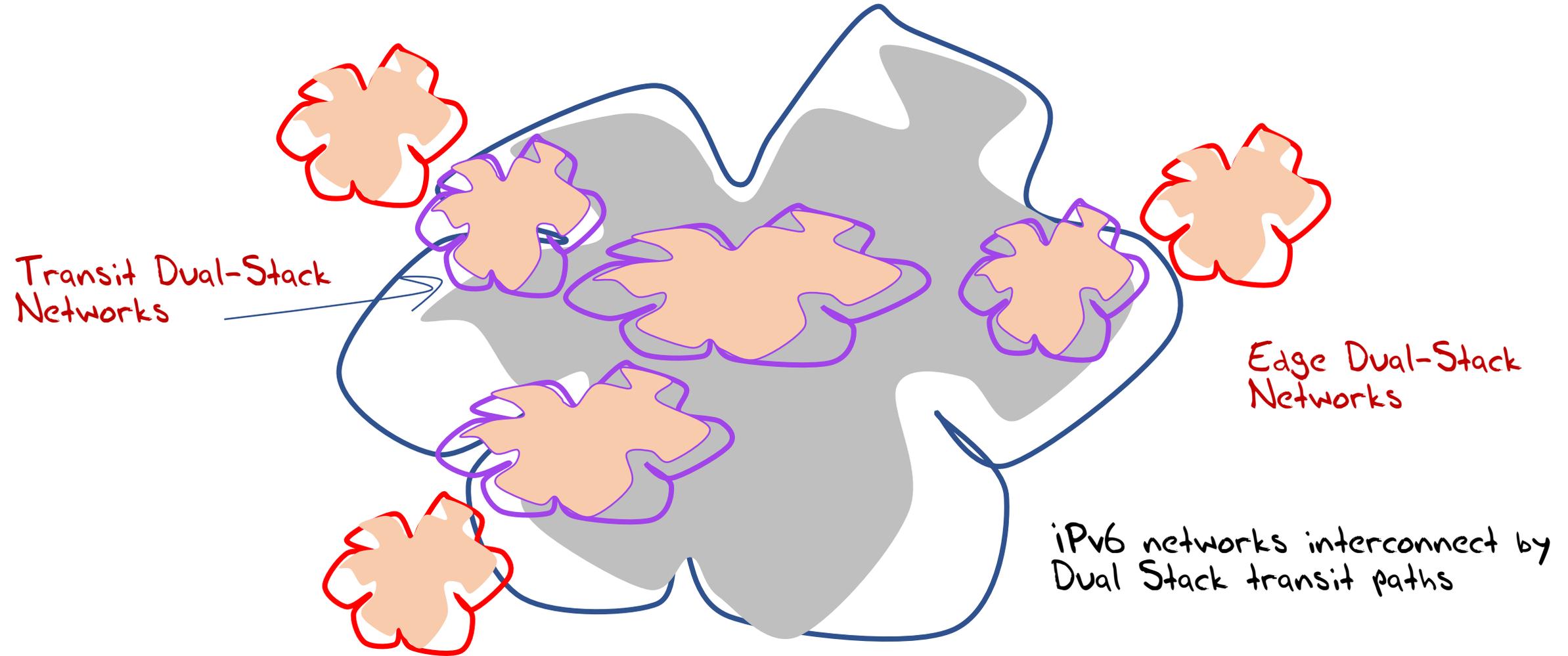
In-situ transition...

Phase 1 - Early Deployment



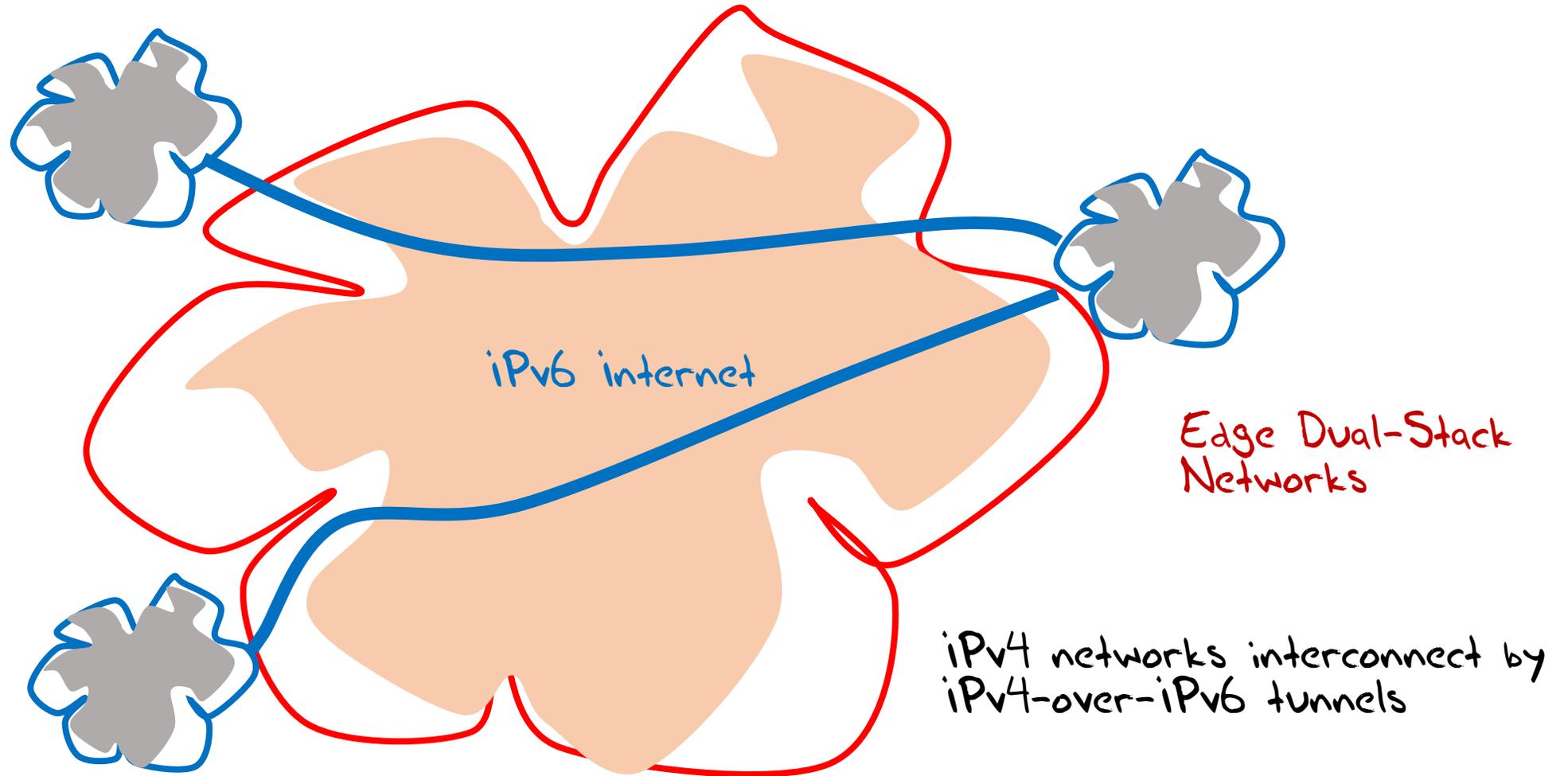
In-situ transition...

Phase 2 - Dual Stack Deployment



In-situ transition...

Phase 3 - IPv4 Sunset

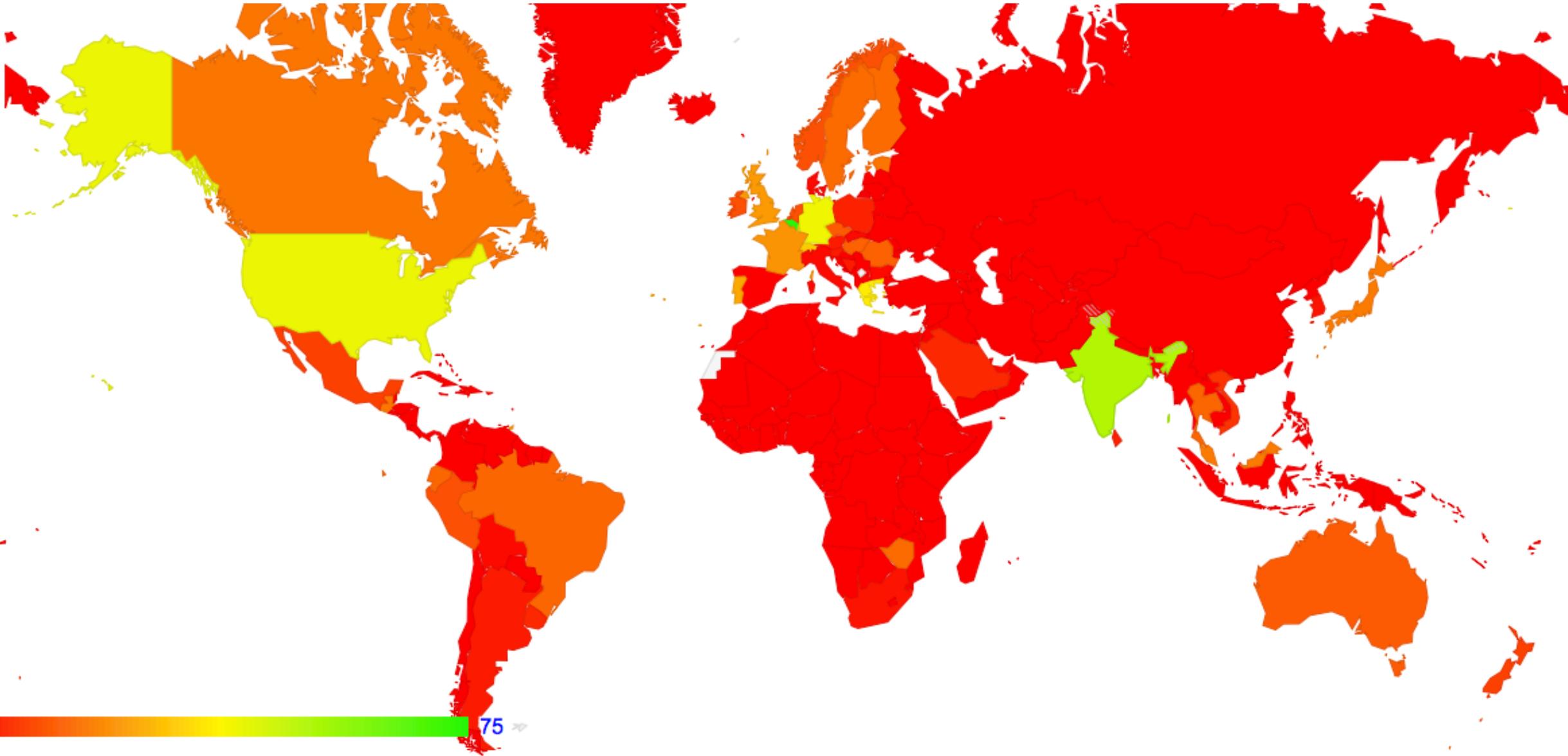


We are currently in Phase 2 of this transition

Some 15% to 20% of internet users have IPv6 capability

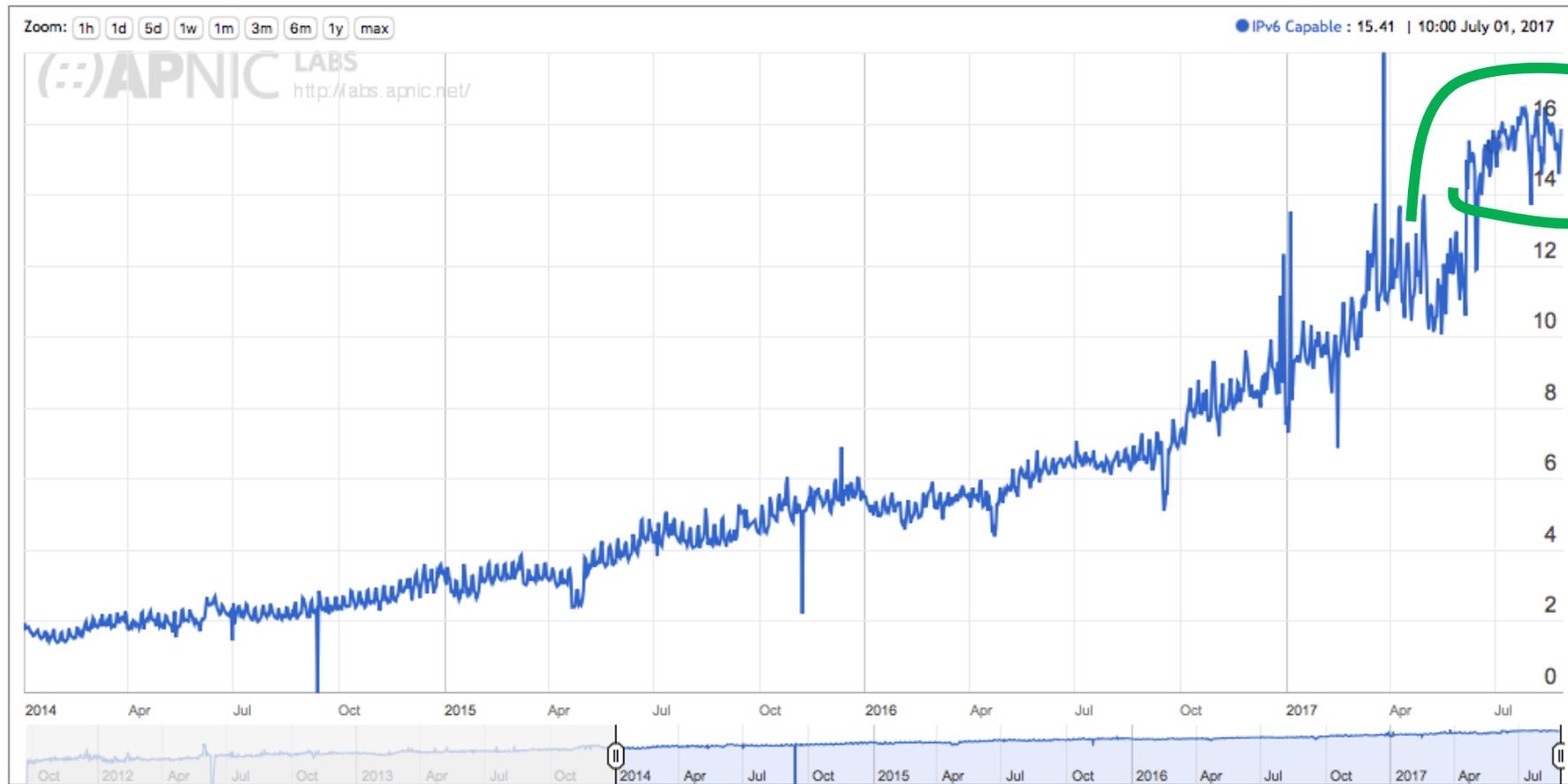
Most new IP deployments rely on NATS and IPv4 (and may, or may not, also have IPv6)

IPv4-only Legacy networks are being slowly migrated to dual stack



The Map of IPv6 penetration - August 2017

Use of IPv6 for World (XA)



0

The Map of IPv6 penetration - August 2017

We are currently in Phase 2 of this transition

Some 15% of internet users have :-

Most new IP deployments :-

IPv4-only :-

The current situation is that the universal "glue" of the internet remains IPv4, while IPv6 is still an optional feature!

- being (gradually) migrated to dual stack

Today

We appear to be in the middle of the transition!

Dual Stack networks cannot drop support for IPv4 as long as significant services and user populations do not support IPv6 - and we can't tell when that may change

Nobody is really in a position to deploy a robust at-scale ipv6-only network service today, even if they wanted to!

And we are not even sure if we can!

Today

We appear to be in the middle of the transition!

Dual Stack networks cannot drop support for IPv4 as long as significant services and user populations do not support IPv6 - and we can't tell when that may change

Nobody is really in a position to deploy a robust at-scale IPv6-only network service today, even if they wanted to!

And we are not even sure if we can!

The Issue

We cannot run Dual-Stack services indefinitely

At some point we need to support networks that only have IPv6

is that viable?

In other words...

What do we rely on today in IPv4 that does not appear to have a clear working counterpart in IPv6?

if the answer is "nothing" then we are done!

But if there are issues here, then we should be working on it!

IPv6: What changed?

Type of Service is changed to Traffic Class

Flow Label Added

Options and Protocol fields replaced by Extension Headers

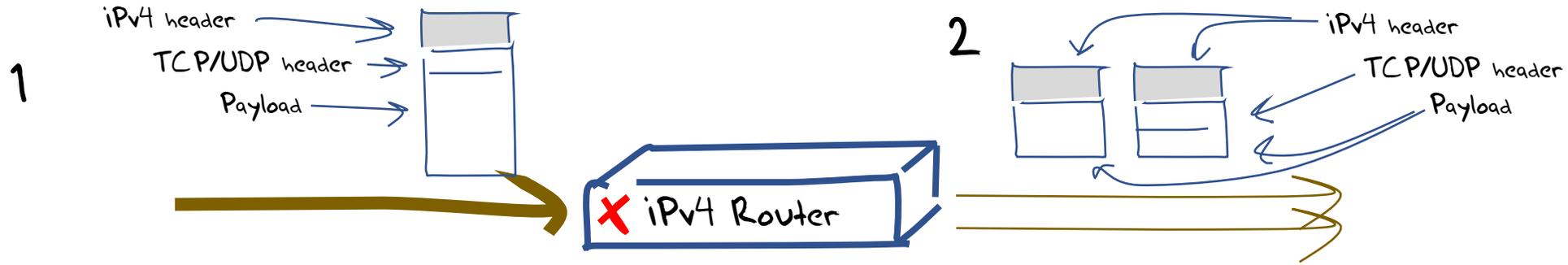
32 bit Fragmentation Control were pushed into an Extension Header

Checksum becomes a media layer function

The substantive change with IPv6 is the handling of fragmentation and the use of Extension Headers

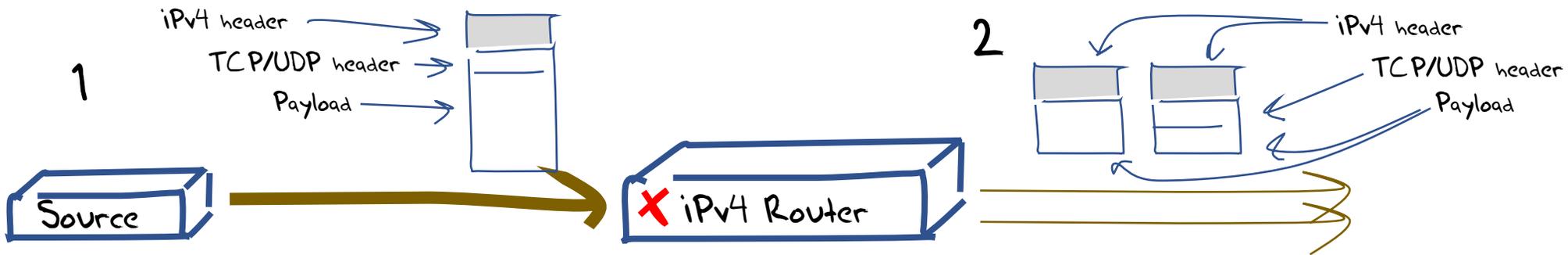
IPv6: What changed?

IPv4 "Forward Fragmentation"

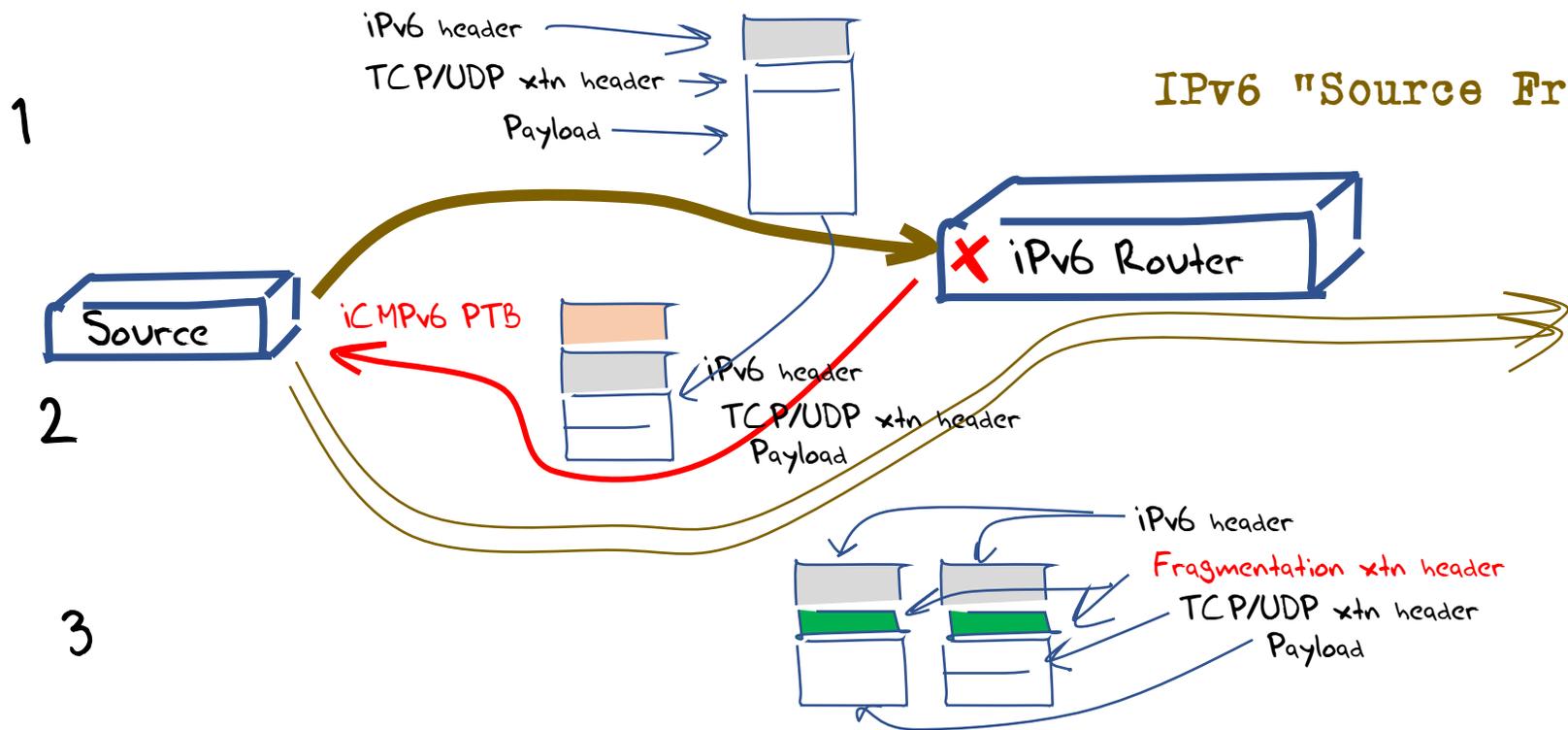


IPv6: What changed?

IPv4 "Forward Fragmentation"



IPv6 "Source Fragmentation"



New Dependencies

For IP fragmentation to work in IPv6 then:

- all ICMPv6 messages have to be passed **backwards** from the interior of the network to the sender
- IPv6 packets containing a IPv6 Fragmentation Extension header should **not** be dropped

ICMPv6

Only the sending host now has control of fragmentation - this is a new twist

A received ICMPv6 message needs to alter the sender's state to that destination:

For TCP, if the ICMP payload contains the TCP header, then you can pass this to the TCP control block. TCP can alter the session MSS and resend the dropped data, or you can just alter the local per-destination MSS and hope that TCP will be prompted to resend

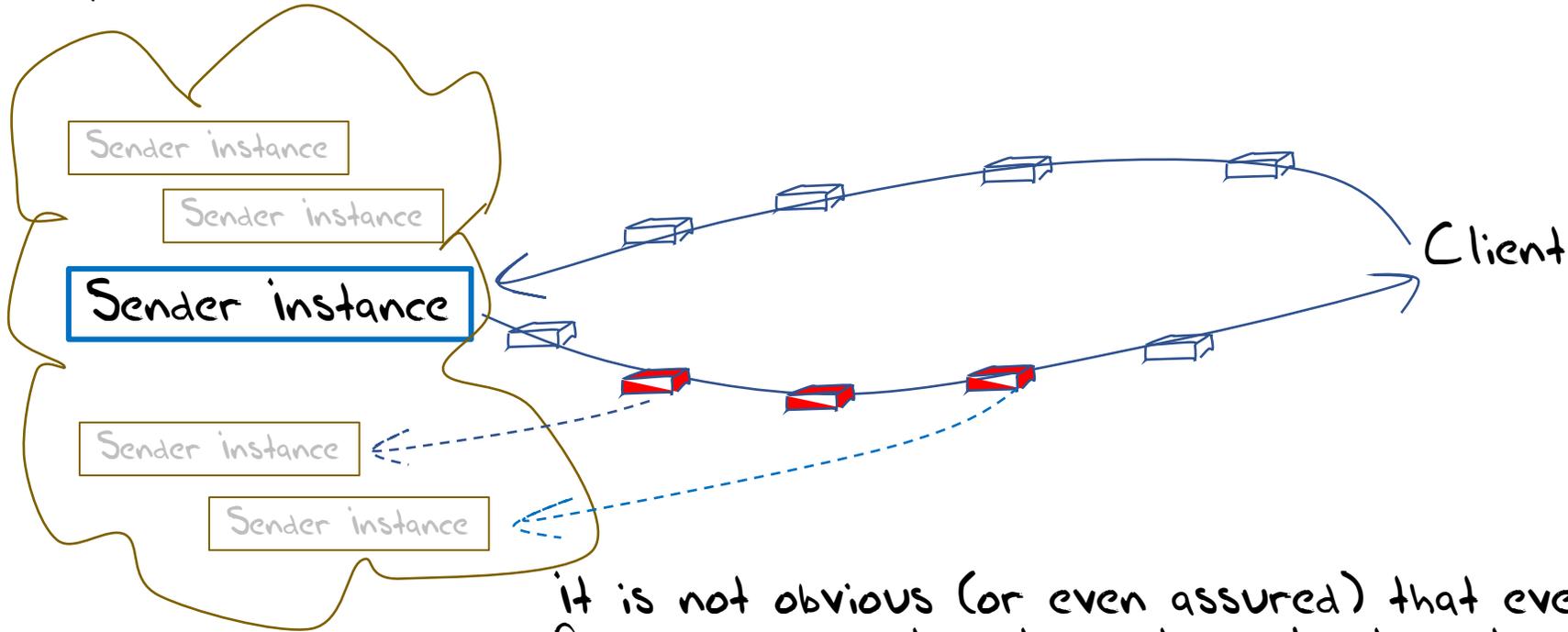
For UDP - um, err, um well

Maybe you should store the revised path MTU in a host forwarding table cache for a while

if you ever need to send another UDP packet to this host you can use this cache entry to guide your fragmentation behaviour

ICMPv6 and Anycast

Anycast Constellation

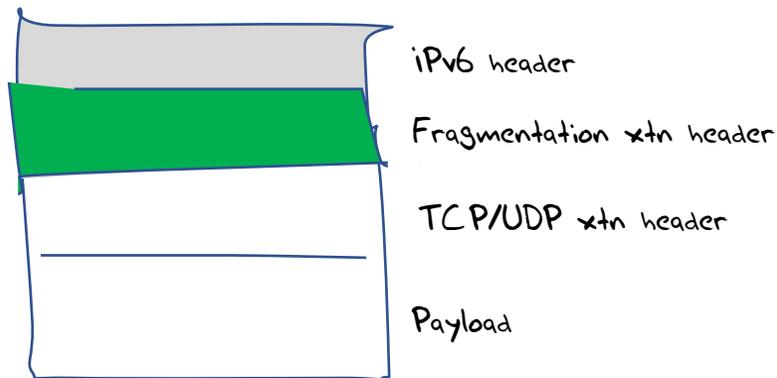


it is not obvious (or even assured) that every router on the path from an anycast instance to a client host will necessarily be part of the same anycast instance "cloud"

The implication is that in anycast, the reverse ICMPv6 PTB messages will not necessarily head back to the original sender!

IPv6 Fragmentation Extension Header Handling

The extension header sits between the IPv6 packet header and the upper level protocol header for the leading fragged packet, and sits between the header and the trailing payload frags for the trailing packets



Practically, this means that transport-protocol aware packet processors/switches need to decode the extension header chain, if its present, which can consume additional cycles to process/switch a packet - and the additional time is not predictable. For trailing frags there is no transport header!

Or the unit can simply discard all ipv6 packets that contain extension headers!

Which is what a lot of transport protocol sensitive IPv6 deployed switching equipment actually does (e.g. load balancers!)

IPv6 Fragmentation Extension Header Handling

There is a lot of "drop" behaviour in the ipv6 internet for Fragmentation Extension headers

RFC 7872 - recorded EH packet drop rates of 30% - 40%

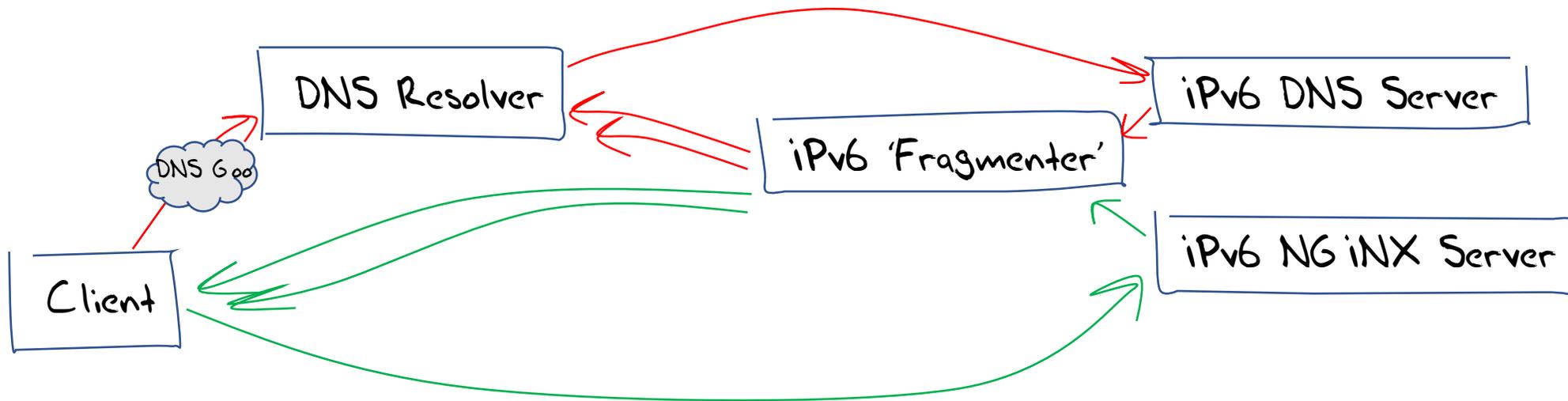
This experiment sent fragmented packets towards well-known servers and observed whether the server received and reconstructed the fragmented packet

But sending fragmented queries to servers is not all that common - the reverse situation of big responses is more common

So what about sending fragmented packets BACK from servers - what's the drop rate of the reverse case?

IPv6 Fragmentation Extension Header Handling

We used an ad-based measurement system, using a custom packet fragmentation wrangler as a front end to a DNS and Web server to test IPv6 fragmentation behaviour



IPv6 Fragmentation Extension Header Handling

We used an ad-based measurement system, using a custom packet fragmentation wrangler as a front end to a DNS and Web server to test IPv6 fragmentation behavior.

We use a technique of "glueless" delegation and fragmentation of the NS query response to allow us to detect if the DNS resolver received the fragmented response



We track TCP ACKs at the server to see if the client received the fragmented TCP response

IPv6 Fragmentation Extension Header Handling

Our experiments across some 40M individual sample points:

37% of end users who used IPv6-capable DNS resolvers could not receive a fragmented IPv6 response

20% of IPv6-capable end users could not receive a fragmented IPv6 packet

IPv6 Fragmentation is very unreliable

Why don't we see this unreliability in today's IPv6 networks affecting user transactions?

Because IPv4 papers over the problem!

In a Dual-Stack environment there is always the option to flip to use IPv4 if you are stuck with IPv6.

The DNS does this, and Happy Eyeballs does this

So there is no user-visible problem in a dual stack environment

This means that there is no urgent imperative to correct these underlying problems in deployed IPv6 networks

There is little in the way of practical incentives to fix this today!

Living without IPv6 Fragmentation

if we apparently don't want to fix this, can we live with it?

We are living with it in a Dual Stack world, because ipv4 just makes it all better!

But what happens when there is no ipv4 left?

We have to avoid IPv6 Fragmentation!

TCP can work as long as IPv6 sessions use conservative MSS sizes

UDP can work as long as UDP packet sizes are capped so as to avoid fragmentation

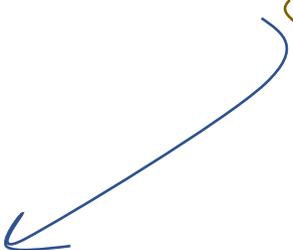
Who needs to use large UDP packets anyway?

DNSSEC!

Where are we?

in terms of protocol support and reliability, it seems that we are mostly ready for an IPv6-only environment, with the one exception of IPv6 packet fragmentation handling.

The consequence is that today's environment cannot support an IPv6-only environment for the DNS, and DNSSEC in particular



Change the deployed IPv6 network and change deployed vendor equipment to correctly manage fragmentation, and stop using anycast!



Change host configurations and change protocol behaviours to avoid any reliance at all on correct handling of packet fragmentation

An IPv6-only Internet?

The issue of the unreliability of IPv6 fragmentation is a significant issue.

These mitigation approaches represent significant effort and cost

Effort and cost that is unnecessary for as long as IPv4 can paper over the problem!

So we are taking the easy option, and collectively we are doing nothing at all!

Maybe if we close our eyes long enough all this will just go away!

Thanks!