# SIIT-DC: IPv4 Service Continuity for IPv6 Data Centres

Tore Anderson

Redpill Linpro AS

UKNOF36, London, January 2017


# Incremental IPv6 deployment in DC

- IPv4-only
- IPv4-only + IPv6 via NAT/proxy/etc
- Dual-stacked public frontend, IPv4 BE
- Full dual-stack
- Dual-stacked public frontend, IPv6 BE
- IPv6-only + IPv4 via NAT/proxy/etc
- IPv6-only

# IPv4 sucks

- Not enough addresses
- Default route through stateful NAPT44 boxes
- Overlap with customer use of RFC1918
- Renumbering/resizing server LANs
- No IPv6 for customers who want / require it (such as the entire Norw€gian pub£ic $ector)

# Dual stack sucks MORE

- Needs IPv4 - see previous slide for why it sucks
- Dual stack = Dual WORK and Dual COMPLEXITY
  - 2x ACLs / firewall rules
  - 2x monitoring targets
  - 2x places where errors can occur (esp. human errs)
  - 2x protocols the server and apps guys must learn
  - *N*x possible application commication patterns
- IPv4 becomes like a spreading cancer which it's almost impossible to safely remove later

# What's realistic today?

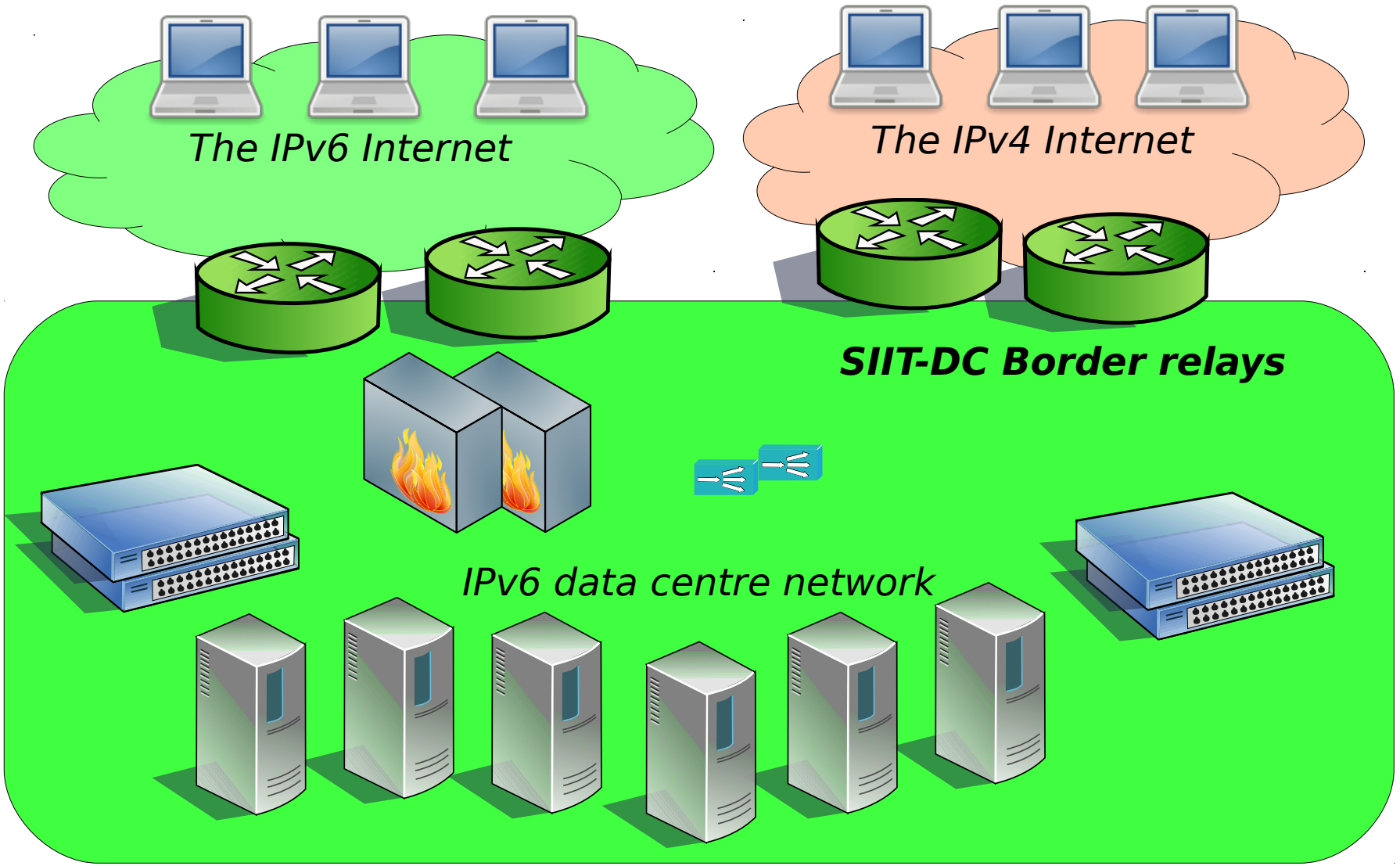
- IPv4-only
- IPv4-only + IPv6 via NAT/proxy/etc
- Dual-stacked public frontend, IPv4 BE
- Full dual-stack
- Dual-stacked public frontend, IPv6 BE
- IPv6-only + IPv4 via NAT/proxy/etc
- ~~IPv6-only~~

***more than 80% of end-users world-wide do not have IPv6!***
***(source: https://www.google.com/intl/en/ipv6/statistics.html)***

# So let's take a shortcut...

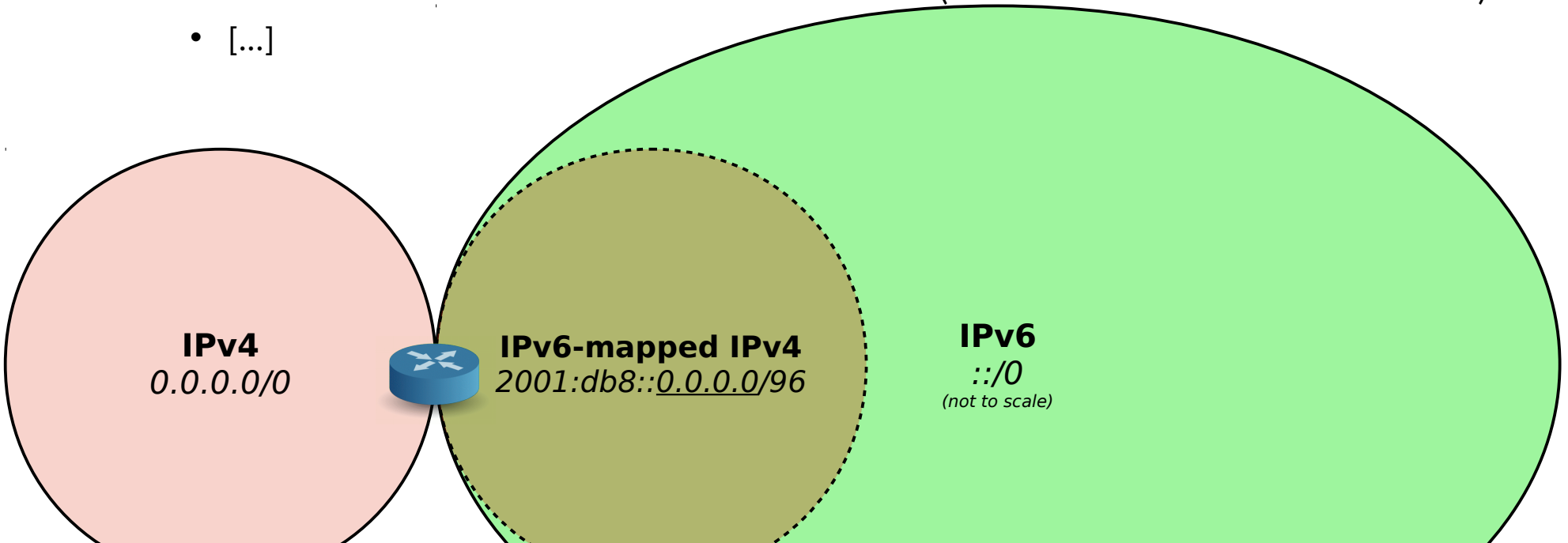
- IPv4-only
- IPv4-only + IPv6 via NAT/proxy/etc
- Dual-stacked public frontend, IPv4 BE
- Full dual-stack
- Dual-stacked public frontend, IPv6 BE
- IPv6-only + IPv4 via NAT/proxy/etc
- IPv6-only

# SIIT-DC: Stateless IP/ICMP Translation for IPv6 Data Centre environments (RFC 7755)

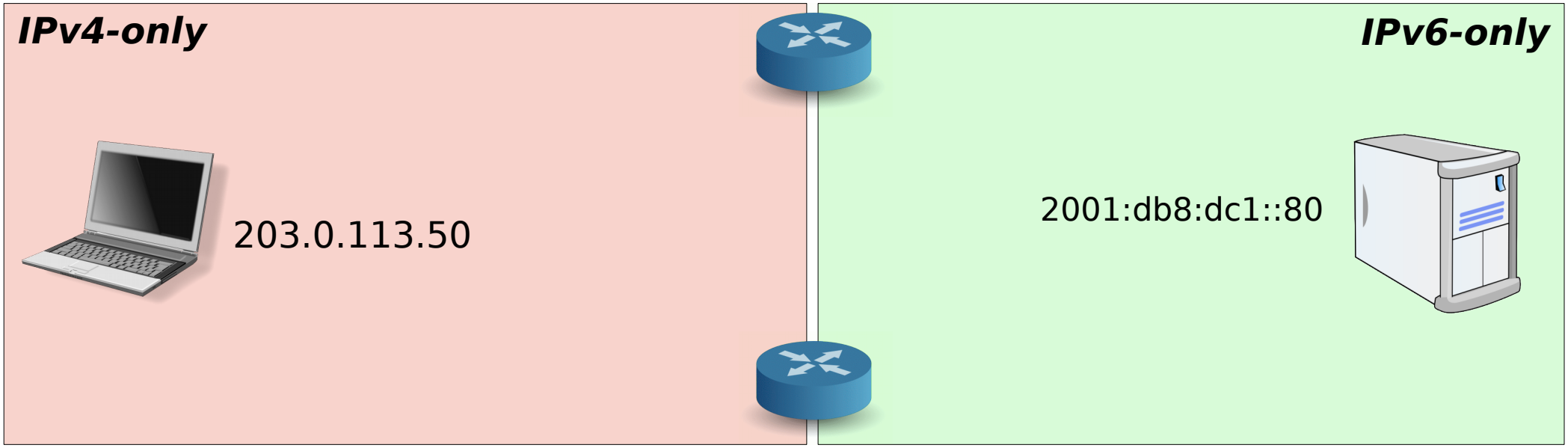


# SIIT-DC operation in a nutshell

- The IPv4 internet is mapped into an IPv6 translation prefix
  - IPv4 0.0.0.0/0 -> IPv6 2001:db8::**0.0.0.0**/96 [same as 2001:db8::/96]
  - *For example*: 203.0.113.10 -> 2001:db8::**203.0.113.10** [same as 2001:db8::**cb00:710a**]
- A table of explicit 1:1 IPv4:IPv6 mappings determine which IPv6 addresses are reachable through which IPv4 addresses
  - A pool of public IPv4 addresses is required - use your last /22, for example
    - 185.47.43.0 -> 2001:db8:dc1::80 (“web server in data centre 1”)
    - 185.47.43.1 -> 2001:db8:dc2::25 (“smtp server in data centre 2”)
    - 185.47.43.2 -> 2001:db8:dc1::389 (“ldap server in data centre 1”)
    - 185.47.43.3 -> 2001:200:dff:fff1:216:3eff:feb1:44d7 (kame.net - somewhere on the Internet)
    - […]

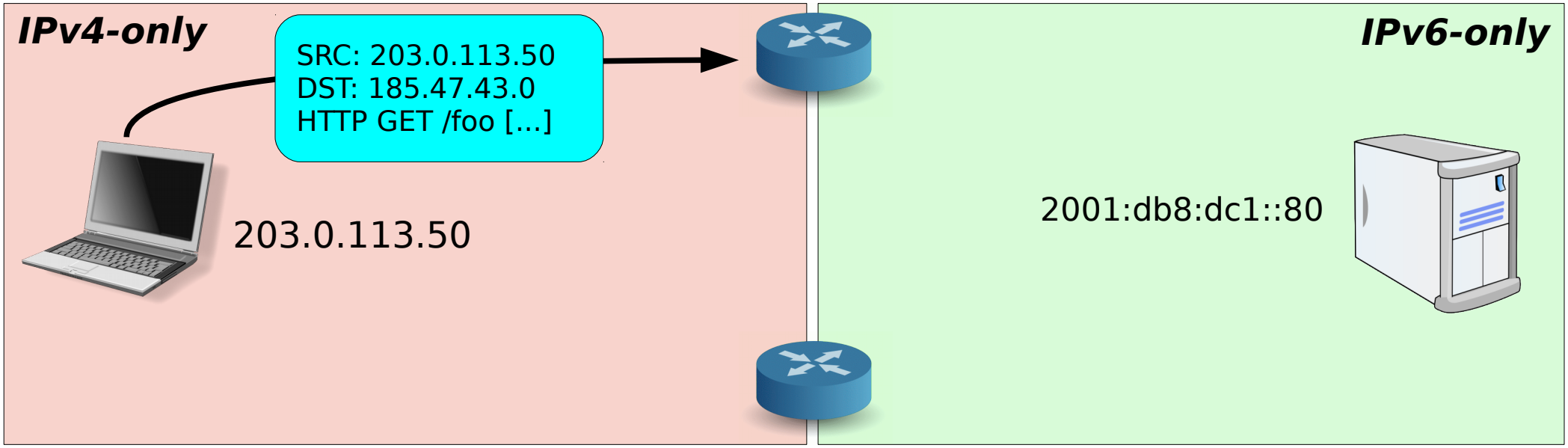


# SIIT-DC packet flow



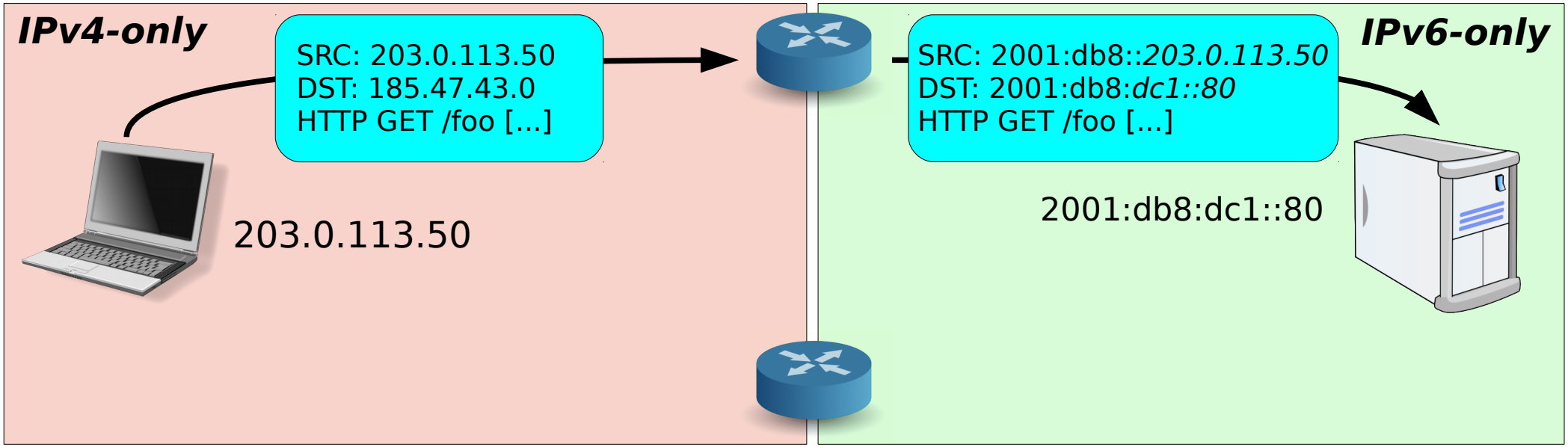

- A completely normal IPv4-only client wants to connect to a web site hosted on an IPv6-only server
- A redundant pair of SIIT-DC Border Relays provides the glue between IPv4 and IPv6

# The IPv4 client connecting



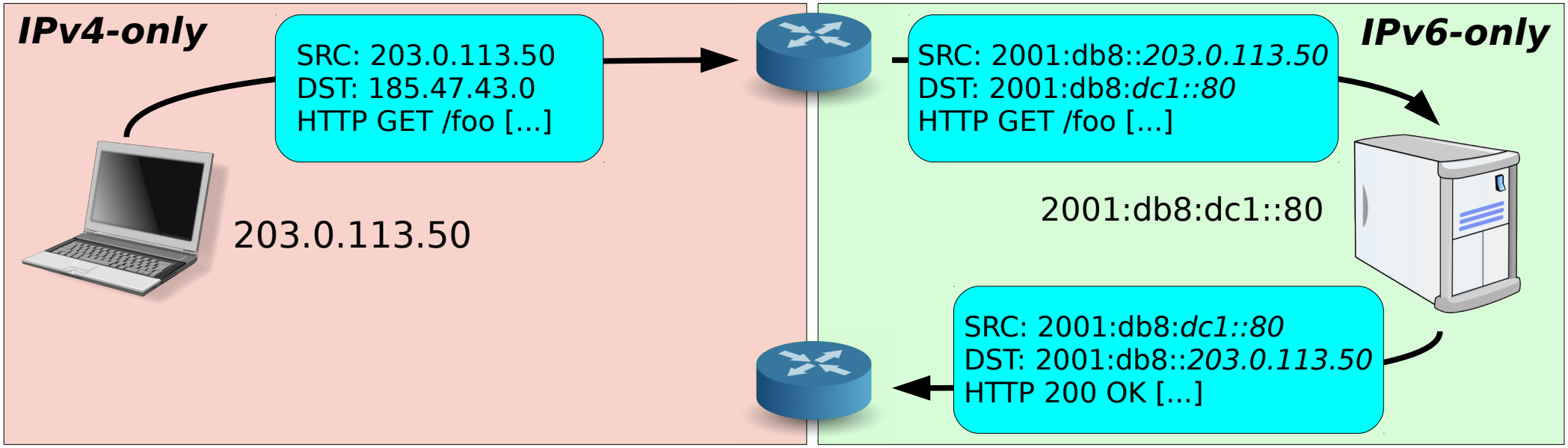

- The IPv4 service address is published as a regular A record for the service in DNS
- It's routed to the provider's SIIT-DC border relay using standard IPv4 routing techniques
- IPv4 clients connect to it in a normal way

# IPv4->IPv6 translation



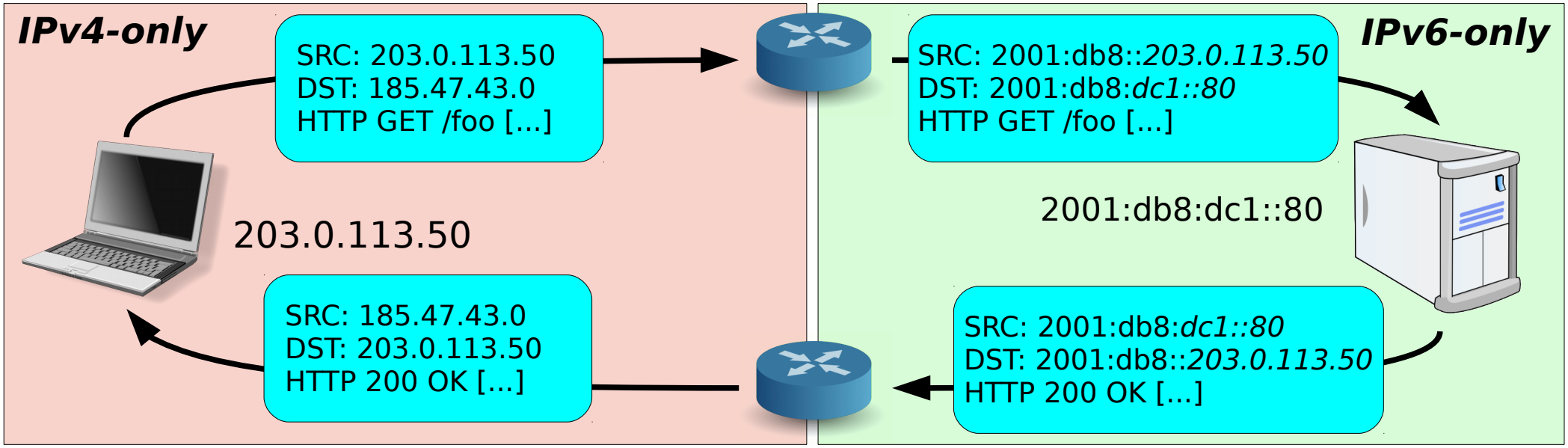

- The pre-defined /96 prefix is prepended to the IPv4 packet's SRC field by the SIIT-DC BR
- The DST address is swapped according to configured 1:1 IPv4:IPv6 mapping by the SIIT-DC BR
- Layer 4 payload is copied verbatim
- The packet is then routed to the server as a completely ordinary IPv6 packet

# IPv6 server processing




- The server responds to the packet just as it would with any other IPv6 packet
- The original IPv4 source address isn't lost
- The /96 prefix (equivalent to the IPv4 default route) is routed to closest SIIT-DC BR

# IPv6->IPv4 translation




- The /96 prefix is stripped from the IPv6 packet's DST field
- 1:1 IPv4:IPv6 mapping is used to swap SRC field
- Again, layer 4 payload is untouched
- The resulting IPv4 packet is returned to the client which processes it normally

# SIIT-DC features / highlights

- No special software needed on endpoints (IPv4 client thinks he's talking to a IPv4 server; IPv6 server thinks he's talking to an IPv6 client)
- IPv4 SRC address not lost (think if server/app wants to do geo-loc, logging, etc.)
- It's STATELESS! Anycast, ECMP, no session tables or connection tracking.
- Server admins, monitoring, ACLs - IPv6 only
- Super easy to eventually decommission

# It's really, *really* simple to set up (use the coffee break!)

- A complete Cisco ASR/CSR example config to the right
- Other implementations exist
  - Brocade ADX, F5 BIG-IP LTM, Linux/TAYGA, Linux/Jool, Linux/fd.io/VPP
  - On server: Linux/clatd/TAYGA
- Incremental deployment
  - It doesn't require an IPv6-***only*** network, just an IPv6 one (dual-stack networks like the Internet included)

```
!
interface GigabitEthernet1
 ip address 192.168.1.2 255.255.255.252
 nat64 enable
 nat64 settings mtu minimum 1500
 ipv6 address 2001:db8::2/64
!
ip route 0.0.0.0 0.0.0.0 192.168.1.2
ip route 185.47.43.0 255.255.255.0 Null0
!
ipv6 route ::/0 2001:db8::1
!
nat64 prefix stateful 2001:db8:46::/96
nat64 v6v4 static 2001:db8:dc1::80 185.47.43.0
nat64 v6v4 static 2001:db8:dc2::25 185.47.43.1
nat64 v6v4 static 2001:db8:dc1::389 185.47.43.2
nat64 v6v4 static 2001:200:dff:fff1:216:3eff:feb1:44d7 185.47.43.3
nat64 settings fragmentation header disable
nat64 settings flow-entries disable
!
```

[Never mind the *«stateful»*, it's really stateless because of *«flow-entries disable»*. The 10 lines that are directly related to SIIT-DC are highlighted in bold, the rest are just standard IPv4/IPv6 network connectivity. Note that 185.47.43.0/24 and 2001:db8:46::/96 must be routed to the ASR/CSR box somehow.]