IPv6 Stuff

Nathan Ward

6to4

Automatic IPv6 over IPv4

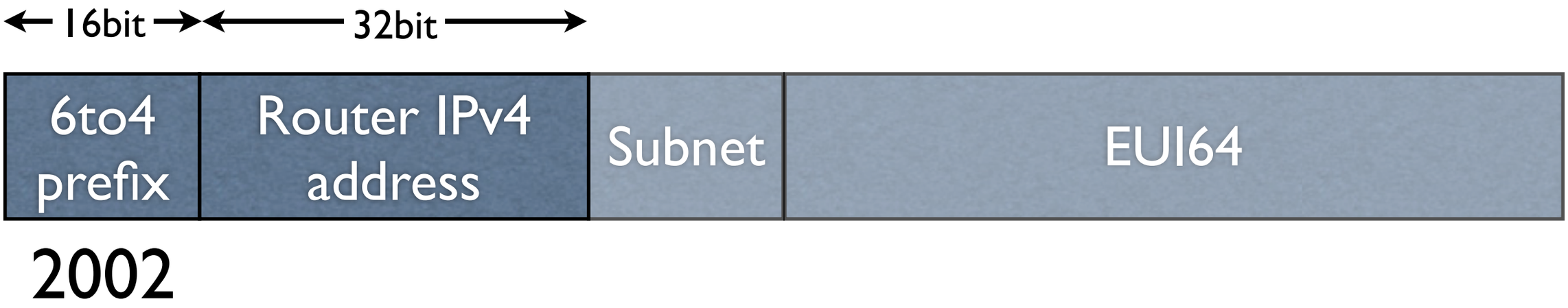
6to4 notes

- RFC3056 and RFC3068
- Requires a single public IPv4 address, and gives a 48-bit prefix (65536 /64s - 1208925819614629174706176)
- Site border router (ie. NAT box, firewall, etc.) needs to support 6to4 - most IPv6 capable hardware does.
- Multipoint-to-multipoint, between border routers of different networks.
- Allows sites to run native IPv6 internally - no specific 6to4 support required on hosts - just normal IPv6.

6to4 notes

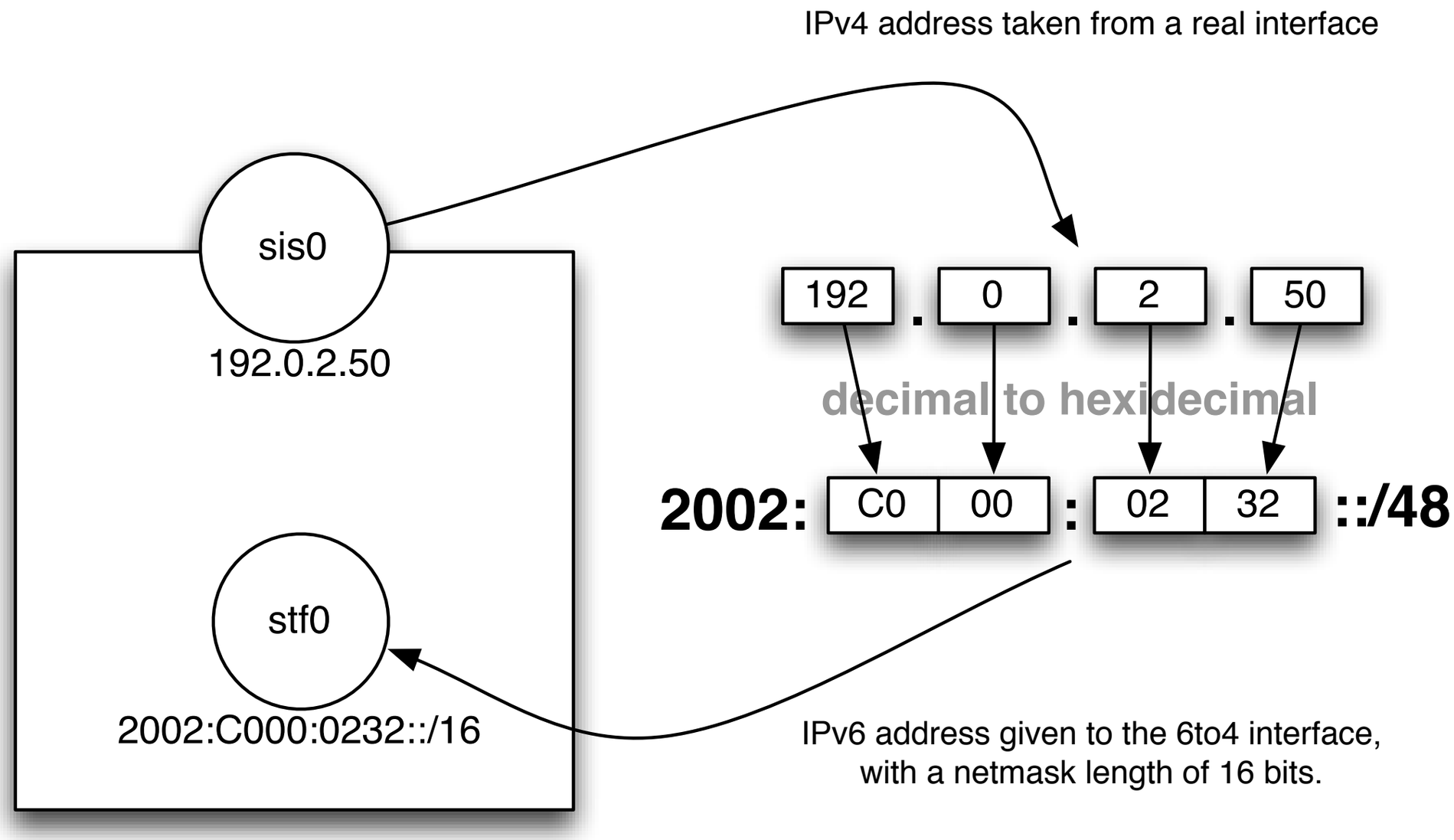
- 6to4 addresses in the form 2002:AABB:CCDD:: where AA.BB.CC.DD is the hex form of the public IPv4 address.
- Only ‘nodes’ exist, no servers to configure - just set it up on your border, and you’re away.
- ‘6bone’ connected 6to4 nodes can relay traffic for non-6bone connected 6to4 nodes, and their networks.
- Makes use of anycast for efficient routing - RFC3068 defines an public IPv4 anycast prefix for relays:
  192.88.99.0/24 (use .1)

6to4 address format



- Subnetting can be done however you like, the above subnet/EUI64 is a common example.

6to4 addressing



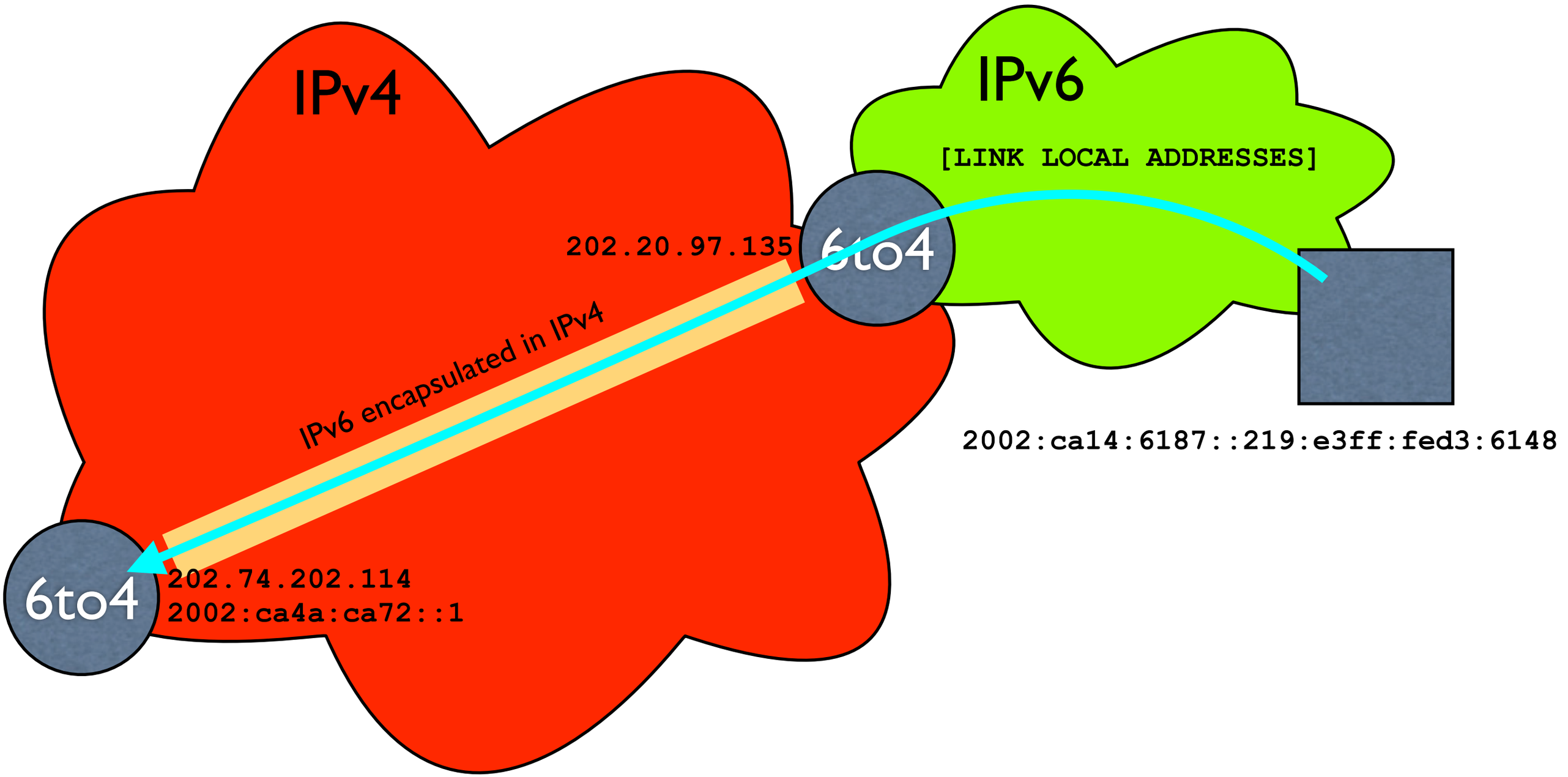
A node's 6to4 address is derived from an IPv4 interface

6to4 packet

Super simple - just wrap an IPv6 packet in an IPv4 header

IP protocol 0x29 (ipv6)

6to4 -> 6to4

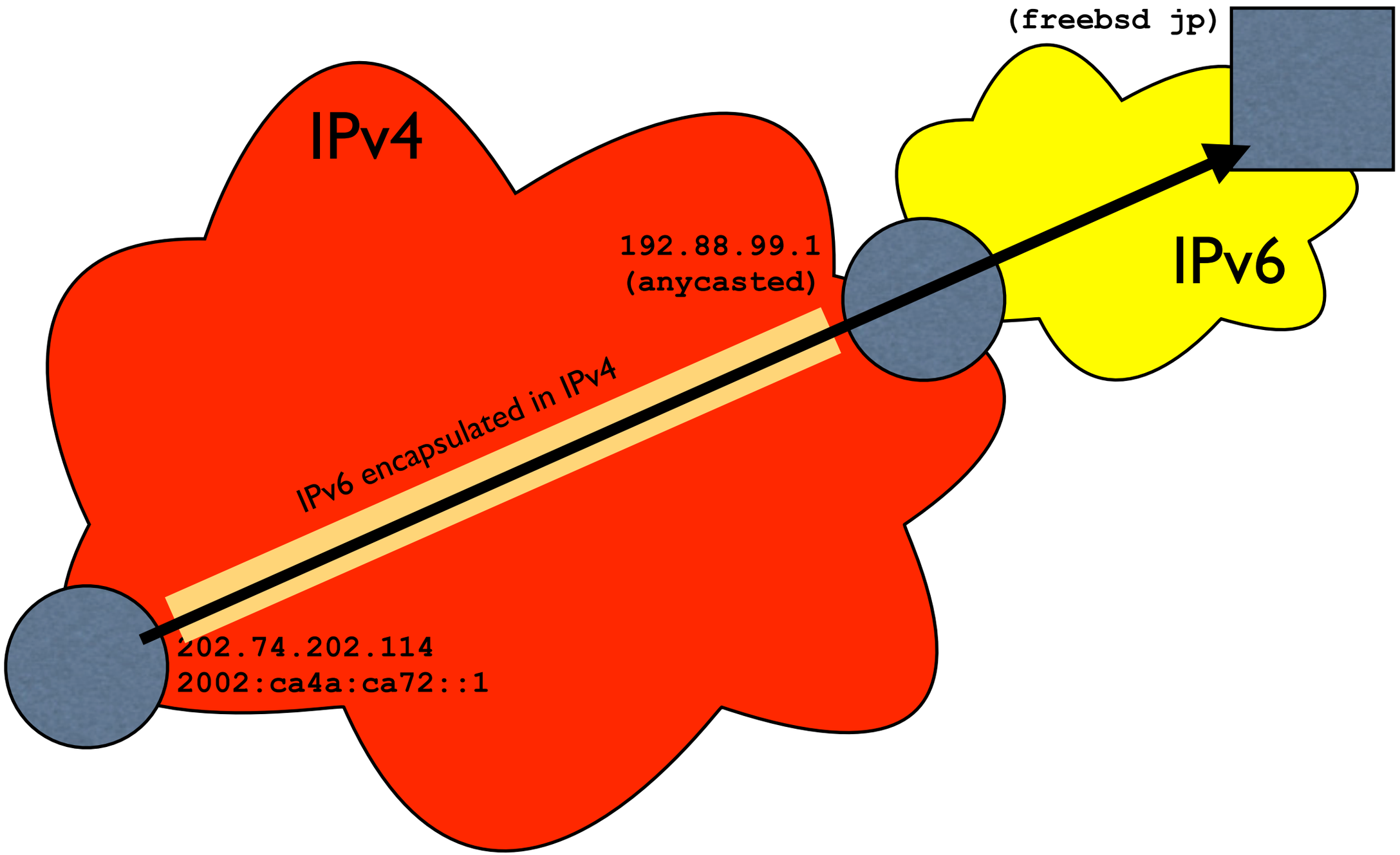

IPv4
IPv6
[LINK LOCAL ADDRESSES]
202.20.97.135
6to4
IPv6 encapsulated in IPv4
2002:ca14:6187::219:e3ff:fed3:6148
6to4
202.74.202.114
2002:ca4a:ca72::1

6to4 -> 6to4

```
Ethernet II, Src: Axxceler_0b:02:22 (00:c0:69:0b:02:22), Dst: 3com_61:7b:2c (00:10:5a:61:7b:2c)
Internet Protocol, Src: 202.20.97.135 (202.20.97.135), Dst: 202.74.202.114 (202.74.202.114)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 76
    Identification: 0x12c6 (4806)
    Flags: 0x00
    Fragment offset: 0
    Time to live: 24
    Protocol: IPv6 (0x29)
    Header checksum: 0xcf6a [correct]
    Source: 202.20.97.135 (202.20.97.135)
    Destination: 202.74.202.114 (202.74.202.114)
Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 16
    Next header: ICMPv6 (0x3a)
    Hop limit: 63
    Source address: 2002:ca14:6187::219:e3ff:fed3:6148 (2002:ca14:6187::219:e3ff:fed3:6148)
    Destination address: 2002:ca4a:ca72::1 (2002:ca4a:ca72::1)
Internet Control Message Protocol v6
```

(magical hex->dotted-decimal conversion)

6to4 -> native


2001:2f0:104:1:2e0:18ff:fea8:16f5
(freebsd jp)
IPv4
192.88.99.1
(anycasted)
IPv6
IPv6 encapsulated in IPv4
202.74.202.114
2002:ca4a:ca72::1

6to4 -> native

```
Ethernet II, Src: 3com_61:7b:2c (00:10:5a:61:7b:2c), Dst: Axxceler_0b:02:22 (00:c0:69:0b:02:22)
Internet Protocol, Src: 202.74.202.114 (202.74.202.114), Dst: 192.88.99.1 (192.88.99.1)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 124
    Identification: 0x0000 (0)
    Flags: 0x04 (Don't Fragment)
    Fragment offset: 0
    Time to live: 64
    Protocol: IPv6 (0x29)
    Header checksum: 0x8242 [correct]
    Source: 202.74.202.114 (202.74.202.114)
    Destination: 192.88.99.1 (192.88.99.1)
Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 64
    Next header: ICMPv6 (0x3a)
    Hop limit: 64
    Source address: 2002:ca4a:ca72::1 (2002:ca4a:ca72::1)
    Destination address: 2001:2f0:104:1:2e0:18ff:fea8:16f5 (2001:2f0:104:1:2e0:18ff:fea8:16f5)
Internet Control Message Protocol v6
```

(magical hex->dotted-decimal conversion)

```
2000::/3 via 2002:c058:6301::
2002::/16 via 6to4
```

Teredo

Automatic IPv6 tunnelling through IPv4 NAT

Teredo

- Relays need to go wherever there is non-Teredo IPv6
- More slides later if we get time, or find me later and I'll be happy to tell you about it

PPP for IPv6

- Same as PPP, but with LCP etc. for IPv6. IPv6CP
- Runs along side IPv4 PPP, in the same PPP session
- Only a few lines of config for basic functionality

Recommendations

- If you want to do IPv6, you should:
  - Get some transit
  - Deploy a 6to4 relay (so your IPv6 customers can reach 6to4 users fast)
  - Deploy a Teredo relay (so your IPv6 customers can reach Teredo users fast)
  - Peer at APE/WIX

Recommendations

- If you don't want to do IPv6 in your whole network
  - Deploy a 6to4 relay, with a tunnel for international transit, and a tunnel to the v6ix
    - Only needs to be one small router, or a linux/bsd box
    - Lets your users who choose to use 6to4 get semi-efficient transit

Filtered public networks

- If you have public address space on your end hosts (desktop machines, etc.) AND if you filter or NAT what those hosts can do
- 6to4 on Windows will break - it thinks it has a plain Internet connection, so thinks 6to4 will work
  - It has no way of testing this right now
- Because of this, IPv6 TCP connections will time out
  - Web sites who turn on IPv6 in parallel will lose visitors

Filtered public networks

- So:
  - If you want to block 6to4:
    - Reject IP protocol 41 at your border, and return ICMP admin prohibited or something
  - If you want users to be able to do 6to4:
    - Allow protocol 41
    - But, no firewalls let you filter the the IPv6-in-IPv4 stuff at the moment

Filtered public networks

- You’re better off blocking it now, but planning to do a real IPv6 deployment

Tui

Tui Features

- Lets networks connect to each other with IPv6
- Only needs IPv4 between providers
- Does not require a full mesh of statically configured tunnels
- Simple to implement in your network

Tui Features

- Lets networks connect to the WIX and APE IPv6 bits
- Only requires IPv4 to the exchange - if you can get to the NZ-only stuff on Citylink, it'll work

Tui Features

- Gives networks a 6to4 and Teredo relay

Tui requirements

- Some kind of IPv6 transit
  - Tui is the tunnel end point for an IPv6 tunnel from a provider; or
  - One of your routers is the tunnel end point for an IPv6 tunnel from a provider; or
  - You have native IPv6 transit
- IPv4
- One IPv4 address
- One ethernet port
- One (or more) iBGP session to your gear

Tui routing

- Tui advertises in to your network:
  - 2002::/16
  - 2001::/32
  - 192.88.99.0/24
- It will advertise whatever prefixes you give it, to other Tui users

Tui

- Useful for
  - Content providers with IPv6 content
  - IPv4 only ISPs
  - IPv6 capable ISPs
  - IPv6 enabled companies

Tui

- FreeBSD 6.something
- Soekris 4801 (also runs on 4501)
- Quagga version something.or.other
- Miredo 1.0.6 (I think)

Tui is Free

- InternetNZ are awesome, and bought 6 initial boxes
  - We're giving them away
    - Tentatively:
      - TelstraClear
      - Orcon
      - AKL University
      - Catalyst (for the .nz boxes I think)
      - ihug (or whatever they're called)
- I'm happy to give people the images to build their own

Tui

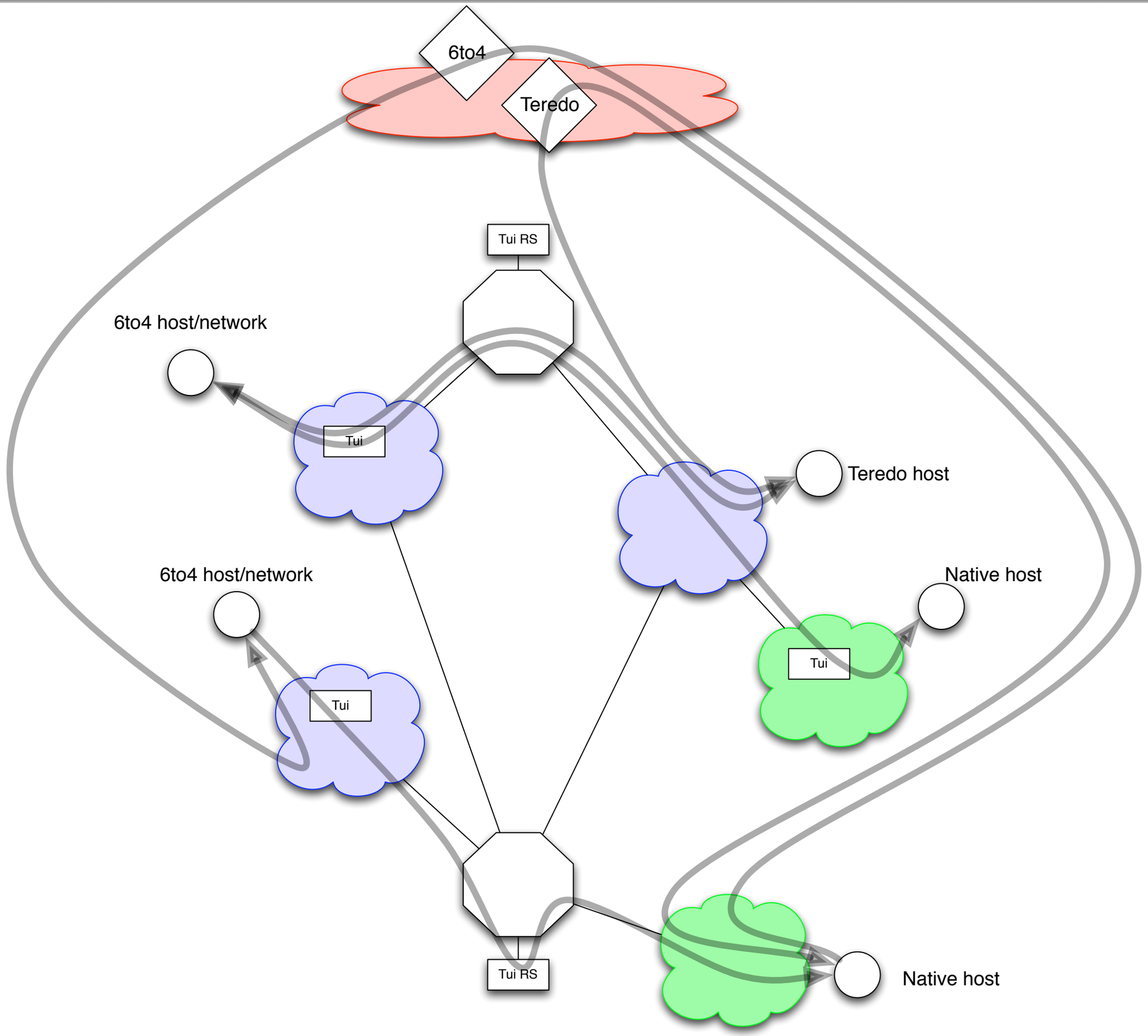
- Builds a ‘full mesh’ of 6to4 tunnels by using BGP
  - As each Tui advertises prefixes to the Tui route servers, they set the next-hop to be their IPv4-mapped 6to4 address

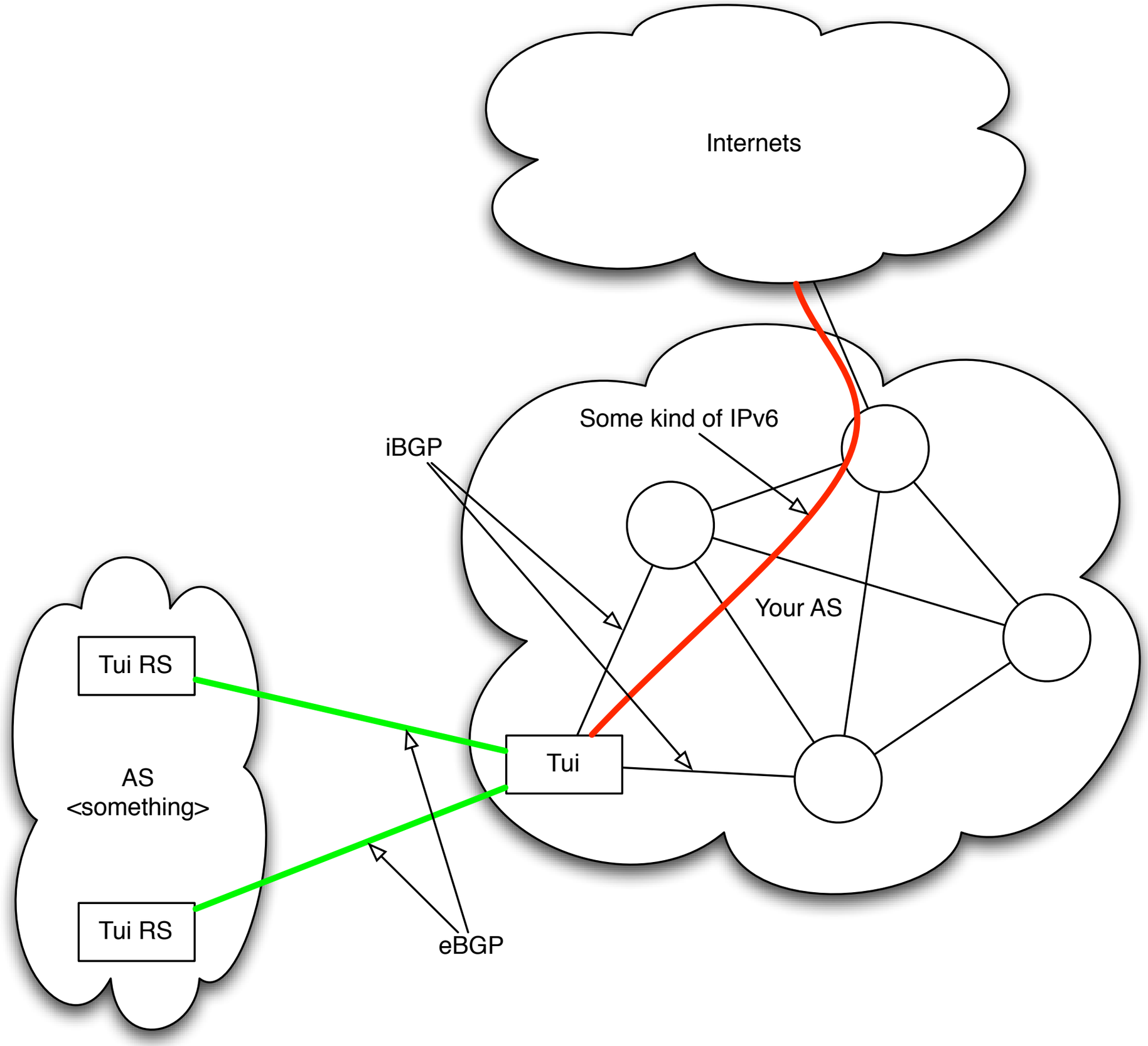
Tui

- If you have some aversion to Soekris/BSD/ Whatever routers, I can give you config for whatever vendor you use
  - Provided it's Cisco
  - Or provided you can give me a box to test config on
    - This is the part where vendors give me free stuff

Tui

- If the box dies, it stops advertising prefixes
- The Teredo and 6to4 relays stop being used, and new relays start being used, without customers noting
  - TCP sessions stay up, IPv6 addresses stay the same


6to4
Teredo
Tui RS
6to4 host/network
Tui
Teredo host
6to4 host/network
Native host
Tui
Tui
Tui RS
Native host


Internets
Some kind of IPv6
iBGP
Your AS
Tui RS
AS
<something>
Tui RS
Tui
eBGP