IPv6 Transition and Integration with IPv4

Tim Chown
School of Electronics and Computer Science
University of Southampton (UK)

tjc@ecs.soton.ac.uk

Agenda

- Approaches to deploying IPv6
  - Standalone (IPv6-only) or alongside IPv4
  - Phased deployment plans
- Considerations for IPv4 and IPv6 coexistence
- Approaches to coexistence
  - 1: Dual-Stack
  - 2: Tunnelling
  - 3: Translation
- Specific examples
  - 6to4 and Tunnel broker, ISATAP
  - NAT-PT, Application Layer Gateways (ALGs)

IPv6 only?

- When deploying IPv6, you have two choices:
  - Deploy IPv6-only networking
  - Deploy IPv6 alongside IPv4
- Currently IPv6 is not mature in certain commercial application and management products, though support on host OS and router platforms is very good
  - This will influence current decisions towards dual-stack (use IPv6 where available, else IPv4)

Deploy IPv6 standalone

- Typically IPv6 will be deployed today dual-stack
- But one option is to deploy an IPv6-only network
- This introduces specific requirements:
  - All components (network, OS, apps) must be IPv6-capable
  - Likely to need to talk to legacy IPv4-only systems
    - Need a way to 'translate' between the protocols at some layer
  - Likely to want to communicate with remote IPv6 network 'islands' that may only be connected through existing IPv4 networks
    - Need a way to send IPv6 packets over/through an intermediate IPv4-only network
  - IPv6-only deployments are rare today, but will come

Deploy IPv6 alongside IPv4

- Existing network runs IPv4
- (Incrementally) introduce IPv6 to the same network, deploying IPv6 in parallel to IPv4
  - Known as ‘dual-stack’ operation
  - Hosts and routers are able to talk using either protocol
- Choice of protocol is application-specific
  - DNS returns IPv4 and IPv6 addresses for a given hostname
  - As an example, MS Internet Explorer by default prefers IPv6 connectivity, but can fall back to IPv4 (after a timeout)
  - Thus need to be confident IPv6 connectivity is good, else the application may perform worse than in an IPv4-only network

Phased IPv6 deployment

- Each site or network will need to form its own plan for IPv6 deployment
- Need to consider various factors, e.g:
  - – Technical
    - do we need upgrades? Applications?
  - – Policy
    - How do we handle (manage and monitor) IPv6 traffic?
  - – Education
    - Are our support people trained to operate IPv6?
- Then schedule the process

Phase 1: advanced planning

- Phase 1 includes:
  - Add IPv6 capability requirements to future tenders
    - Ensure you have capability to deploy
  - Obtain IPv6 address space from your ISP/NREN
    - Typically a /48 size prefix
  - Arrange IPv6 training
  - Encourage in-house experiments by systems staff
    - e.g. using Tunnel Broker services
  - Review IPv6 security issues
    - IPv6 is often enabled by default - your users may be using IPv6 without your knowledge…

Phase 2: Testbed/Trials

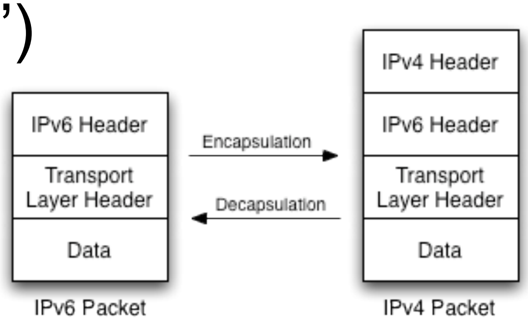
- Phase 2 includes:
  - Deploy IPv6 capable router, with cautious ACLs applied
  - Establish connectivity (probably a tunnel) to your ISP
  - Set up an internal link with host(s), on a /64
    - Can be isolated from regular IPv4 network (e.g. a dual-stack DMZ running IPv4 and IPv6 together)
  - Enable IPv6 on the host systems, add DNS entries if appropriate
- And in parallel
  - Survey systems and applications for IPv6 capabilities
  - Formulate an IPv6 site addressing plan
  - Document IPv6 policies (e.g. address assignment methods)

Phase 3: Production rollout

- Prudent to enable IPv6 on the wire first, then services
- Phase 3 includes:
  - Plan initial deployment areas, e.g. your existing IPv4 DMZ or WLAN may be good first steps
  - Enable external IPv6 connectivity and ACLs/filters
  - Enable IPv6 routing ‘on the wire’ on selected internal links
  - Deploy IPv6 support in management/monitoring tools
- Then enable the services and advertise via DNS:
  - Enable IPv6 in selected services (e.g. web, SMTP)
  - Add IPv6 addresses to DNS, enable IPv6 DNS transport
- Remember IPv6 security:
  - e.g. include IPv6 transport in all penetration tests

Various transition approaches

- 1: Dual Stack
  - Servers/devices speaking both protocols
- 2: Tunnels ("connecting the IPv6 islands")
  - IPv6 encapsulated over IPv4 links
  - IPv6 packet is payload of IPv4 packet
  - Requires "open" holes in firewalls
    - Packets whose Protocol field is '41'
- 3: Translation methods ("IPv4-only to IPv6-only")
  - Rewriting IP header information
  - Rewriting TCP headers
  - Application layer gateways (ALGs)



1: Dual-stack

- Support both protocols on selected links (and nodes)
- Requires support in:
  - Host platforms
  - Router platforms
  - Applications and services
    - e.g. web, DNS, SMTP
- Adds considerations for
  - Security in all components
  - New policies dependent on IPv6-specific features
- Can run global IPv6 alongside NAT-ed IPv4

Dual-stack issues

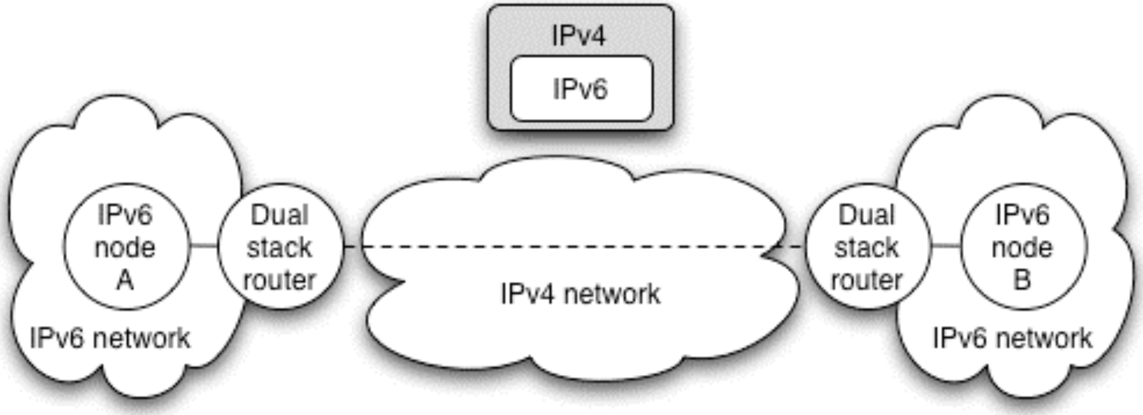
- Application must choose which IP protocol to use
  - DNS returns IPv4 (A record) and IPv6 addresses (AAAA record)
  - e.g. MS Internet Explorer prefers IPv6
  - Don’t advertise AAAA record for a host unless you have good IPv6 connectivity (for all services on host)
- Enabling IPv6 should not adversely impact IPv4 performance
  - Consider whether IPv6 tunnels use router CPU for example
- Security should be no worse
  - Hosts listen on both protocols; secure both

Aside: IPv4 mapped addresses

- An IPv6 address used to represent an IPv4 address
- A socket API may receive an IPv4 connection as an IPv6 address, known as an IPv4-mapped address
  - Format is ::ffff:<ipv4-address>
  - e.g. ::ffff:152.78.64.1
- NB: This is one socket for both address families
- Should not be seen ‘on the wire’, i.e. not as source or destination address
- May appear in log files, depending on how the application handles a connection
- Typically seen in dual-stack deployments

2: Tunnelling

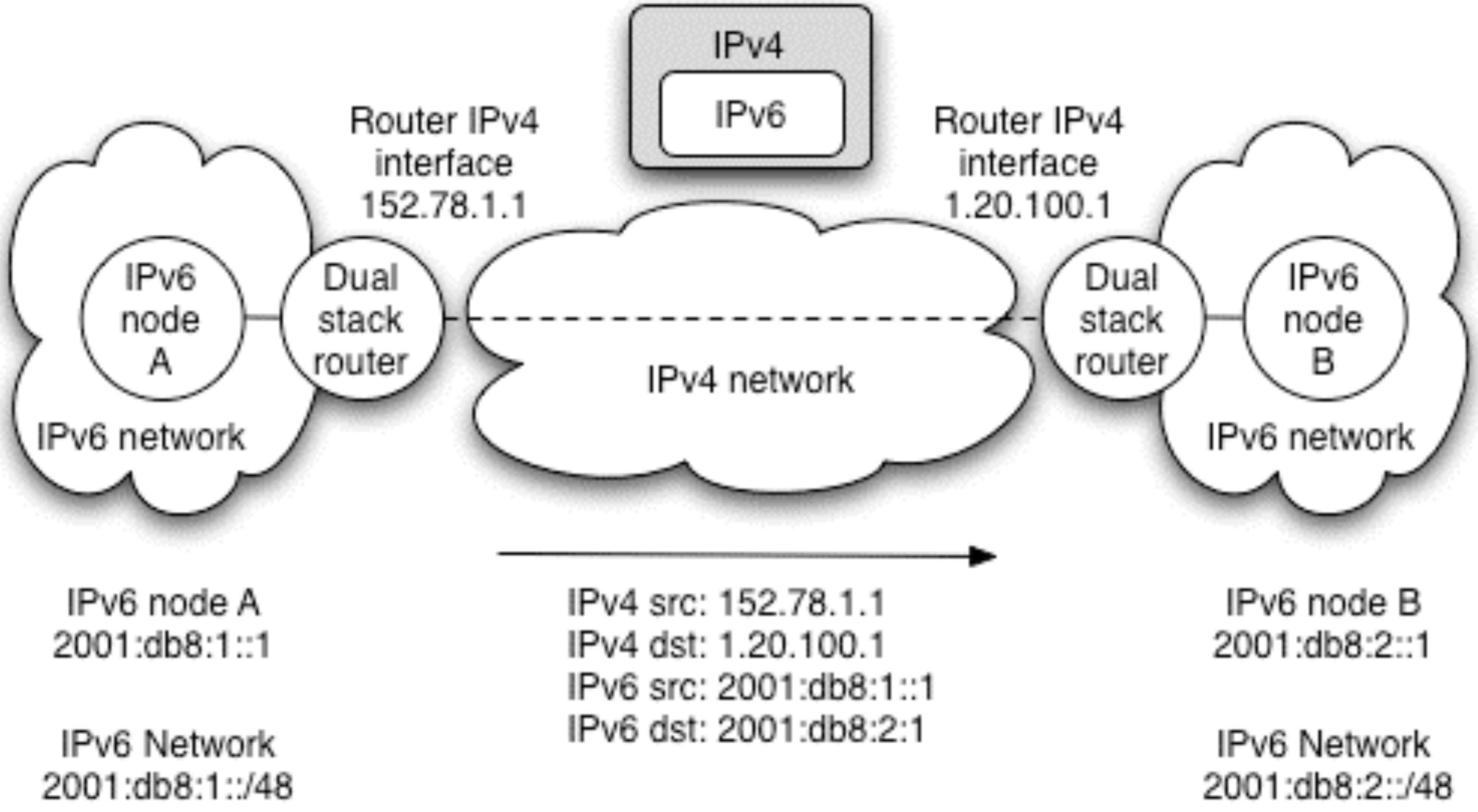
- Initially IPv6 in IPv4, (much) later IPv4 in IPv6
- So, IPv6 packets are encapsulated in IPv4 packets
  - IPv6 packet is payload of IPv4 packet
- Usually used between edge routers to connect IPv6 'islands'
  - Edge router talks IPv6 to internal systems
  - Encapsulates IPv6 in IPv4 towards remote tunnel endpoint



Packet delivery over the tunnel

- IPv6 node A sends packet to IPv6 node B
  - Routed internally to edge router A
- Edge router A sees destination network B is reachable over tunnel interface
  - Encapsulates IPv6 packet in IPv4 packet(s)
  - Sends resulting IPv4 packet(s) to edge router B
  - Delivered over existing IPv4 Internet infrastructure
- Edge router B decapsulates IPv6 packet from payload of received IPv4 packet
  - Packet routed internally in network B to node B
  - Node B receives the IPv6 packet

Tunnel addressing view



IPv6 node A
2001:db8:1::1

IPv6 Network
2001:db8:1::/48

IPv4 src: 152.78.1.1
IPv4 dst: 1.20.100.1
IPv6 src: 2001:db8:1::1
IPv6 dst: 2001:db8:2:1

IPv6 node B
2001:db8:2::1

IPv6 Network
2001:db8:2::/48

Fragmentation

- IPv6 requires that packet fragmentation only occurs at end systems, not on intermediate routers
  - Use Path Maximum Transmission Unit (PMTU) Discovery to choose the MTU
  - Achieved using special ICMP messages
  - Minimum MTU is 1280 bytes in IPv6
- When tunnelling IPv6 in IPv4, the IPv4 packets may be fragmented
  - Depends on the IPv4 packet size
  - Additional IPv6 headers (e.g. Authentication Header) will affect this

Tunnel solution considerations

- These include:
  - Security
  - Manual or automatic setup
  - Ease of management
  - Handling dynamic IPv4 addresses
  - Support for hosts or sites to be connected
  - Scalability: 10, 100, or 10,000 served tunnels?
  - Support for NAT traversal
  - Tunnel service discovery
  - Support for special services (e.g. multicast)
  - Tunnel concentration/bandwidth usage issues
- We'll come back to these later…

Manual or automatic?

- Can create tunnels manually or automatically
- Manual tunnels
  - Requires manual configuration, at both ends
    - Usually just one command/config line in the router at each end
    - Agreement on addresses to use for interfaces
  - Good from a management perspective: you know who your tunnels are created with
- Automatic tunnelling
  - Tunnels created on demand without manual intervention
  - Includes 6to4 (RFC3056)
    - Quite popular in SOHO deployments
  - Also: ISATAP and Teredo

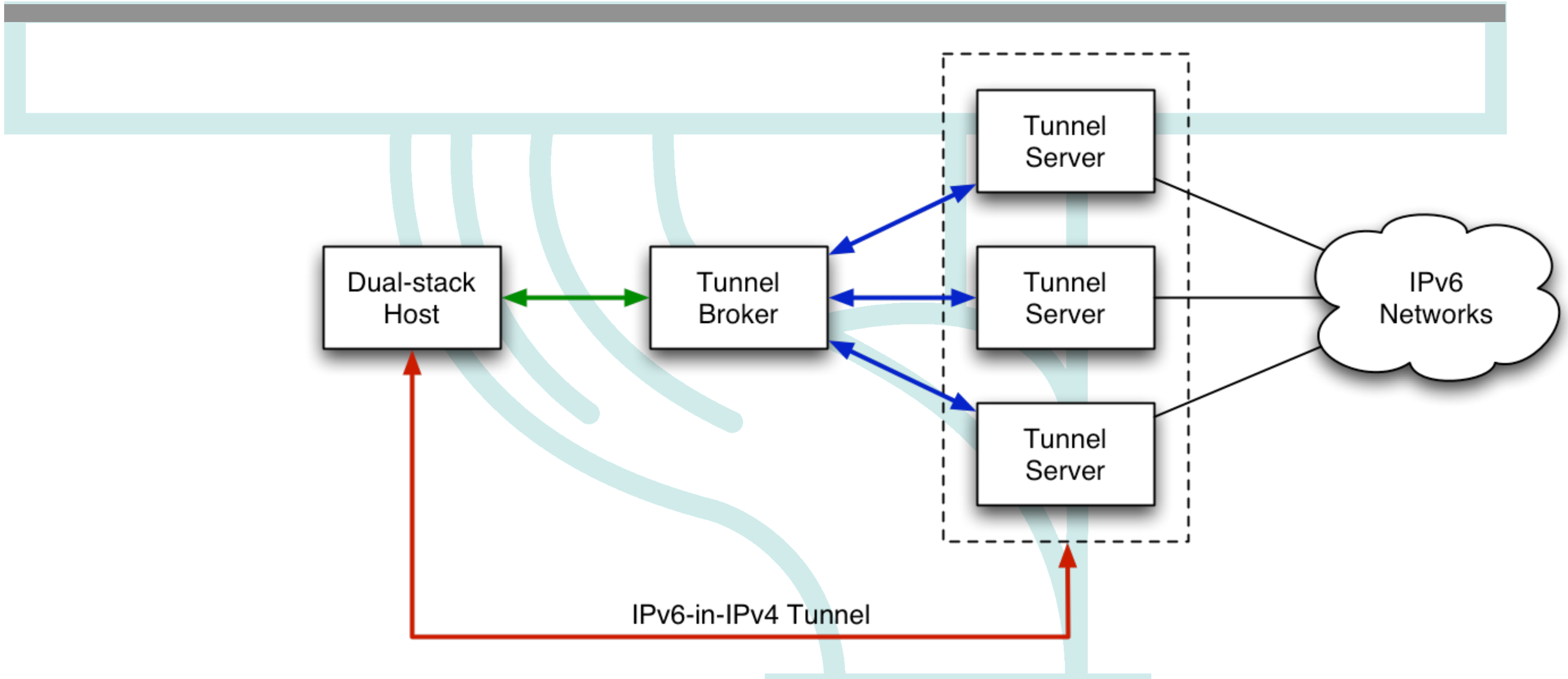
Manually configured tunnels

- Very easy to setup and configure
- Good management potential
  - ISP configures all tunnels, so is in control of its deployment
  - This is the current approach used by many NRENs (including UKERNA and Renater) to connect academic sites/users over IPv6 where native IPv6 connectivity is not available
- Usually used router-to-router or host-to-router
  - Desirable to allow end user to register (and subsequently authenticate) to request a tunnel
  - The IPv6 Tunnel Broker (RFC3053) offers such a system, usually for host-to-router connectivity, but sometimes for router-to-router.

Tunnel broker

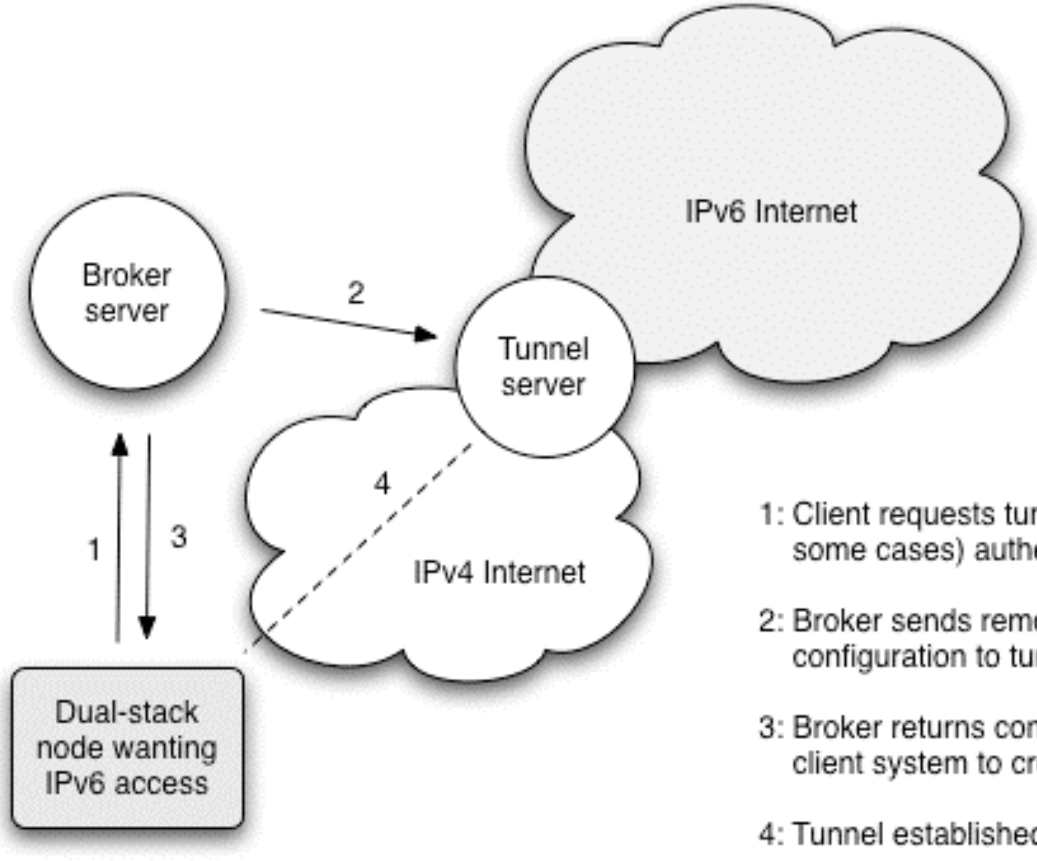
- Very popular in IPv6 user community
- Most well-known broker is www.freenet6.net
  - Hosted in Canada by Hexago
- General mode of operation is:
  - User/client registers with the broker system
  - A tunnel is requested from a certain IPv4 address
  - The broker sets up its end of the requested tunnel on its tunnel server
  - The broker communicates the tunnel settings to the user, for client-side configuration
- Can traverse a NAT, e.g. if UDP tunnelling used

Broker: systems view



1. User connects to Tunnel Broker web interface requesting tunnel
2. TB returns script to create tunnel to the Tunnel Server, and informs TS of new client
3. Client executes script, and gains access to IPv6 networks via the TS

Broker: Logical view



1: Client requests tunnel and (in some cases) authenticates

2: Broker sends remote endpoint configuration to tunnel server

3: Broker returns configuration to client system to create tunnel

4: Tunnel established

Broker issues

- Broker’s key advantage is its manageability
  - ISP can track usage levels
- A few downsides:
  - If broker is topologically remote, round trip times for data may suffer
    - e.g. using freenet6 in Canada to reach UK sites
  - Not well-suited if IPv4 address is dynamic
    - Common problem in home DSL networks
  - Client tool required to operate through a NAT
  - If using a remote tunnel broker, your own ISP may not perceive a demand for IPv6

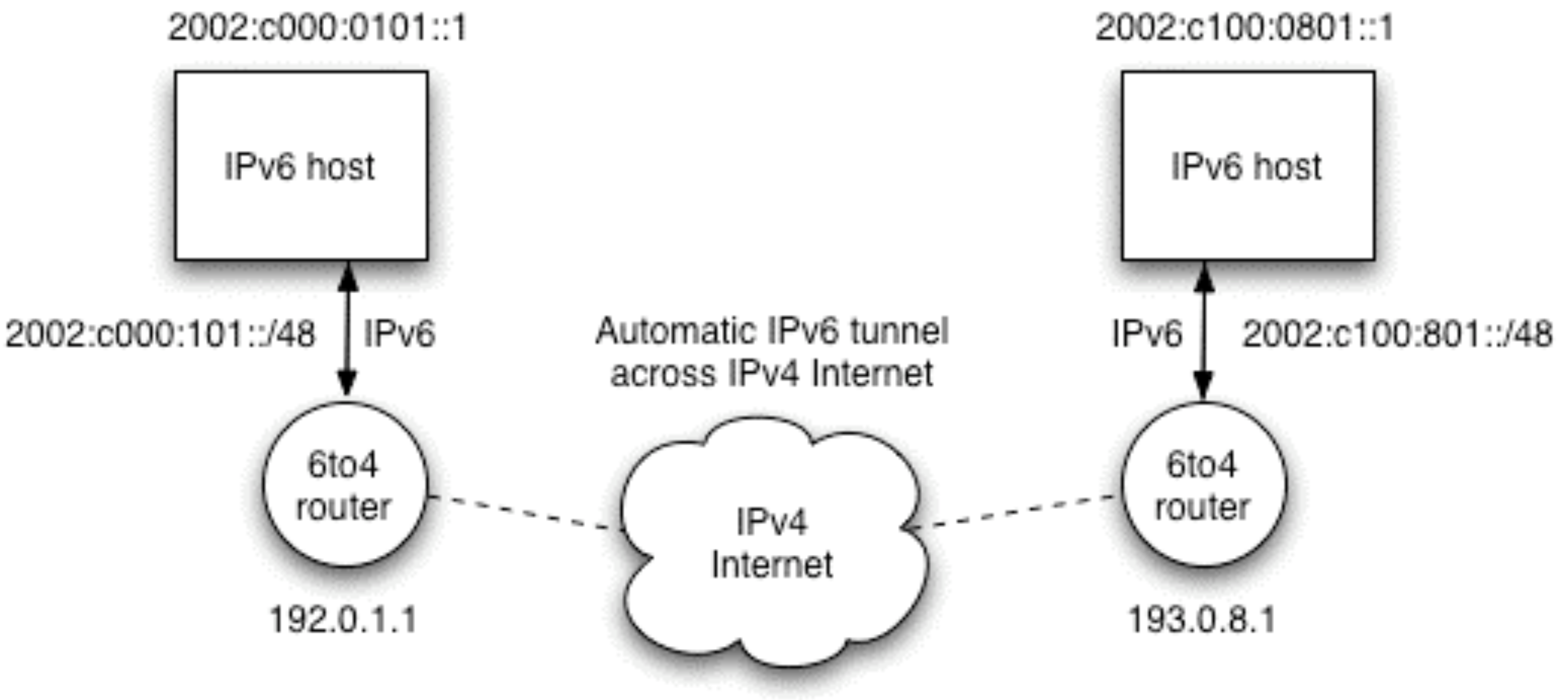
Automatic tunnelling

- Goal is to avoid requiring support staff effort to setup and maintain tunnels
- Set up required tunnels on demand
- Make deployment and usage simple(r) for the end user
- Most common automatic method is 6to4 (RFC3056)
  - Generally used router-to-router
  - Well supported in commercial routing platforms
- Other methods include ISATAP (RFC4214) and Teredo
  - We don’t cover Teredo (RFC4380) in this slideset; it is a NAT-traversing IPv6 connectivity method used by Microsoft in XP/Vista.

6to4

- In its basic configuration, 6to4 is used to connect two IPv6 islands across an IPv4 network
- Uses special ‘trick’ for the 2002::/16 IPv6 prefix that is reserved for 6to4 use
  - Next 32 bits of the prefix are the 32 bits of the IPv4 address of the 6to4 router
  - For example, a 6to4 router on 192.0.1.1 would use an IPv6 prefix of 2002:c000:0101::/48 for its site network
- When a 6to4 router sees a packet with destination prefix 2002::/16, it knows to tunnel the packet in IPv4 towards the IPv4 address indicated in the next 32 bits

6to4 basic overview



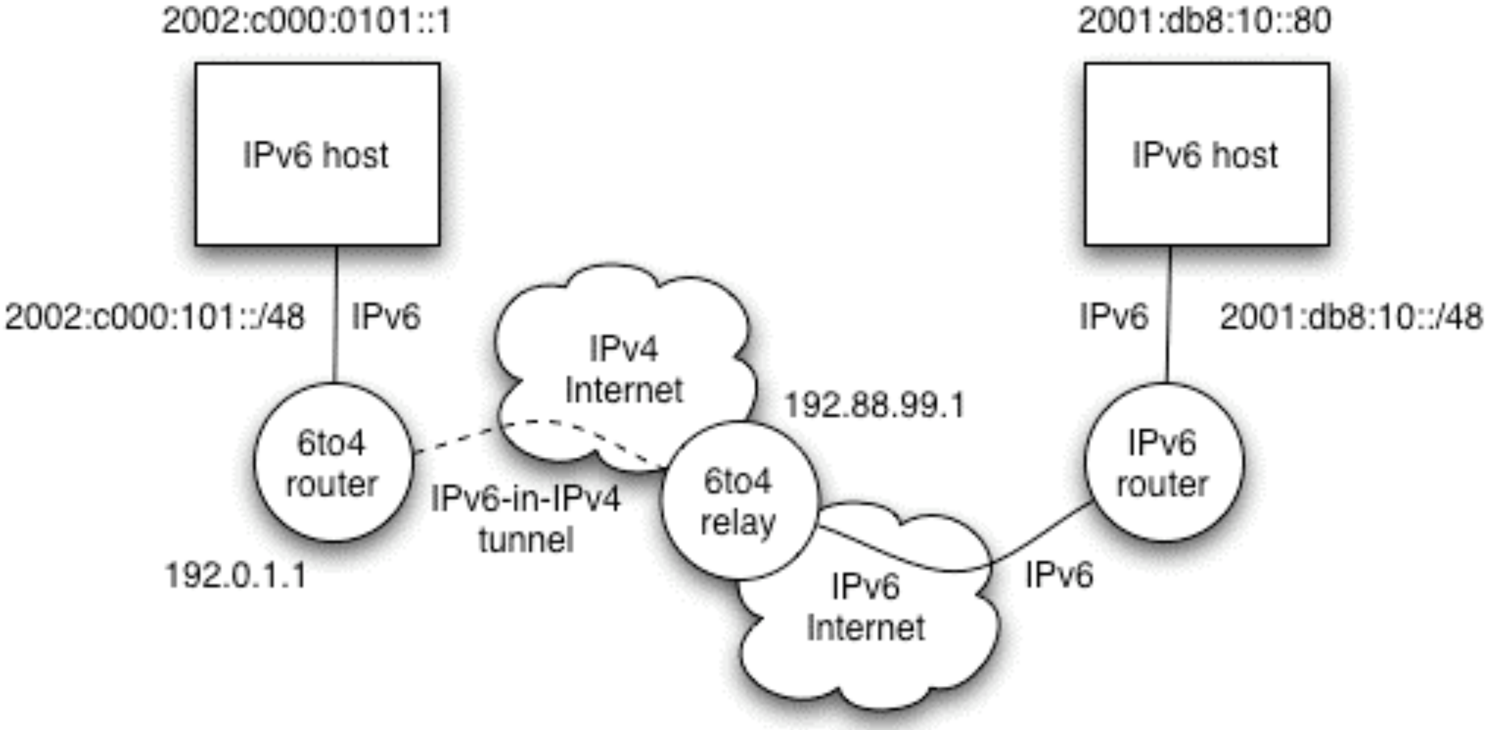
6to4 features

- Simple to deploy and use
  - Fully automatic; no administrator effort per tunnel
  - Tunnelled packets automatically route efficiently to the destination network (following best IPv4 path)
- But there's an important capability missing:
  - How does a node on a 6to4 site communicate with an IPv6 node on a regular, 'real' IPv6 site?
    - Without requiring all IPv6 sites to support 6to4
- And 6to4 relays can be abused (DoS attacks)
  - See RFC3964 for appropriate checks to deploy

6to4 relay

- A 6to4 relay has a 6to4 interface and a ‘real’ IPv6 interface
- Two cases to consider:
  - IPv6 packets sent from a 6to4 site to a destination address outside 2002::/16 are tunnelled using 6to4 to the relay, are decapsulated, and then forwarded on the relay’s ‘real’ IPv6 interface to the destination site
    - The 6to4 relay is advertised on a well-known IPv4 anycast address 192.88.99.1.
  - IPv6 packets sent from a ‘real’ IPv6 site towards an address using the 2002::/16 prefix (a 6to4 site) are routed to the 6to4 relay and then tunnelled using 6to4 to the destination 6to4 site
    - The relay advertises 2002::/16 to connected IPv6 neighbours

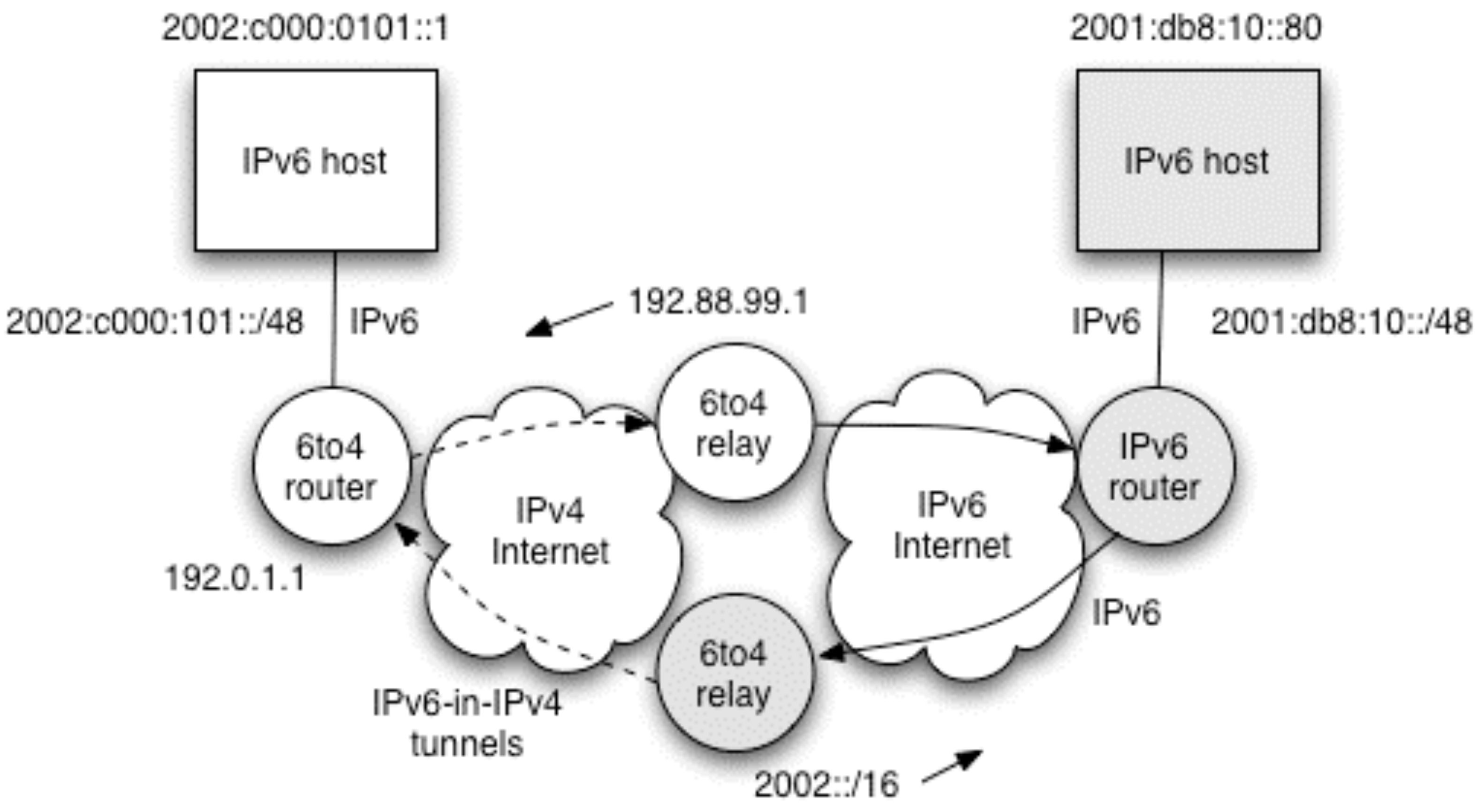
6to4 with relay



6to4 issues

- In principle 6to4 is attractive
  - But there are operational concerns
- Problem 1: possible relay abuse
  - Relay could be used for a DoS attack
  - Tunnelled IPv6 traffic addresses may be spoofed
- Problem 2: asymmetric model/reliability
  - The 6to4 site may use a different 6to4 relay to the ‘real’ IPv6 site
  - One of the sites may not see a 6to4 relay at all, if ISPs choose to only deploy relays for their own customers, and thus filter routing information
- But for 6to4 relay to 6to4 relay operation, it’s good

Asymmetric 6to4



Looking back at considerations

- Earlier we asked how do 6to4 and the tunnel broker fare for:
  - Security
  - Manual or automatic setup
  - Ease of management
  - Handling dynamic IPv4 addresses
  - Support for hosts or sites to be connected
  - Scalability: 10, 100, or 10,000 served tunnels?
  - Support for NAT traversal
  - Tunnel service discovery
  - Support for special services (e.g. multicast)
  - Tunnel concentration/bandwidth usage issues

Comparison for discussion

Feature	6to4	Tunnel broker
Security	Potential for abuse	Supports authentication
Setup	Automatic	Manual
Ease of management	Poor (automatic)	Good
Dynamic IPv4 addresses	Poor	Poor
Host or site tunnels	Primarily site	Primarily host
Scalability	Very good	Good
NAT traversal	Tricky	Yes, with TSP
Tunnel service discovery	Automatic	Manual configuration
Special service support	Variable	Variable
Bandwidth concentration	Only at 6to4 relay	At tunnel server

ISATAP

- Intra-Site Automatic Tunnel Addressing Protocol (RFC4214)
  - Automatic tunneling
  - Designed for use *within* a site
  - Used where dual-stack nodes are sparsely deployed in the site (very early deployment phase)
- Host-to-host or host-to-router automatic tunnels
  - Uses a specific EUI-64 host address format
  - Format can be recognised and acted upon by ISATAP-aware nodes and routers

ISATAP addresses

- The EUI-64 is formed by
  - A reserved IANA prefix (00-00-5e)
  - A fixed 8-bit hex value (fe)
  - The 42-bit IPv4 address of the node
  - Toggling the globally unique (u) bit
- For example, 152.78.64.1 would have an EUI-64 host address for IPv6 of:
  - 0200:5efe:984e:4001

ISATAP tunneling

- Relies on the OS supporting ISATAP
- Use one ISATAP router per site, usually advertised under FQDN ‘isatap.domain’
  - Virtual IPv6 link over the IPv4 network
  - Know the IPv4 tunnel end-point address from last 32-bits of the IPv6 ISATAP address
  - Get network prefix via ND from router
- Not widely deployed
- Better to deploy proper dual-stack
  - Allows better managed control of deployment

3: Translation

- When an IPv4-only system needs to communicate with an IPv6-only system translation is required
- Can be done at various layers
- Network layer
  - Rewrite IP headers
- Transport layer
  - Use a TCP relay
- Application layer
  - Use an application layer gateway (ALG)
- Ideally avoid translation
  - Use IPv4 to speak to IPv4 systems and IPv6 for IPv6 systems

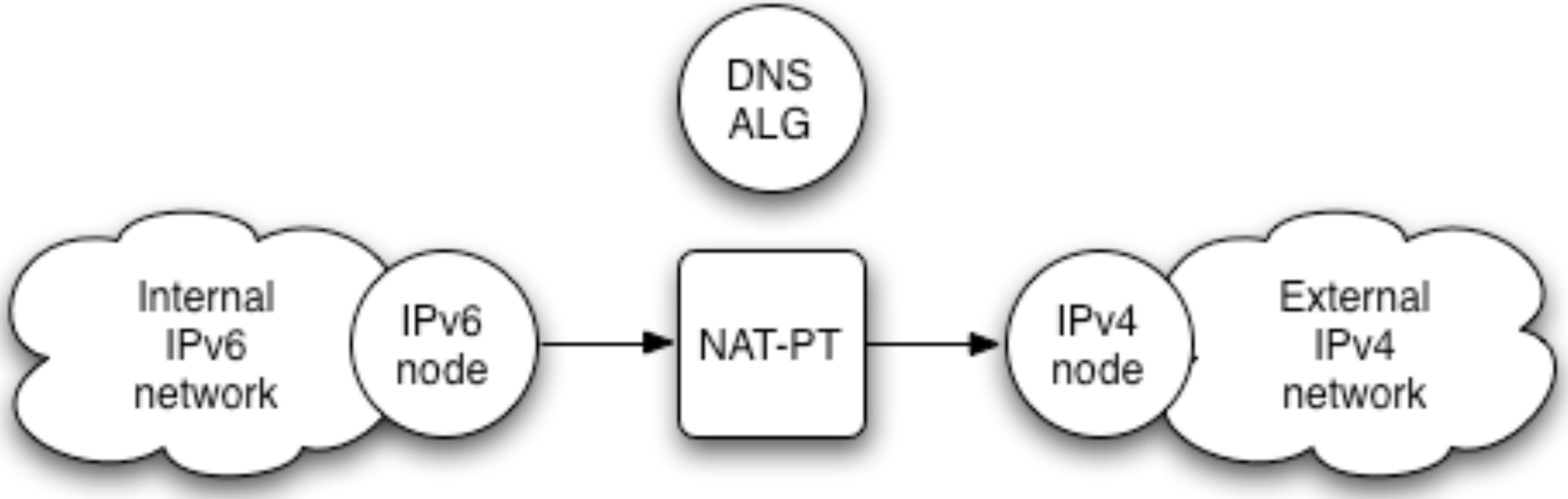
Translation scenarios

- Generally when deploying IPv6-only network elements and you need them to communicate with IPv4-only systems
  - Legacy applications that cannot be ported to support IPv6
    - Or perhaps source code not available
  - Legacy IPv4-only operating systems
    - For example Windows 98
  - Legacy IPv4-only hardware
    - Printers

Network layer: NAT-PT

- Network Address Translation - Protocol Translation
  - Defined in RFC2766
  - Like IPv4 NAT, but with protocol translation
- Uses Stateless IP/ICMP Translation (SIIT)
  - Defined in RFC2765
  - SIIT defines algorithms to translate between the IPv4 and IPv6 header fields, where it’s possible
- NAT-PT extends SIIT with IPv4 address pools
  - IPv4-to-IPv6 and IPv6-to-IPv4 supported

NAT-PT topology



Src: IPv6 address
Dst: <IPv6-prefix>:<IPv4-address>

NAT-PT and DNS

- Internal network IPv6 only
- DNS ALG watches for IPv6 (AAAA) DNS queries, and translates to IPv4 (A) queries
- When IPv4 DNS response comes back, DNS ALG maps the result to an IPv6 address
  - <IPv6-prefix>:<IPv4 address>
  - A special NAT-PT IPv6 prefix is taken from the IPv6 network's address space
- Querying host now uses an IPv6 destination that NAT-PT maps to real IPv4 destination

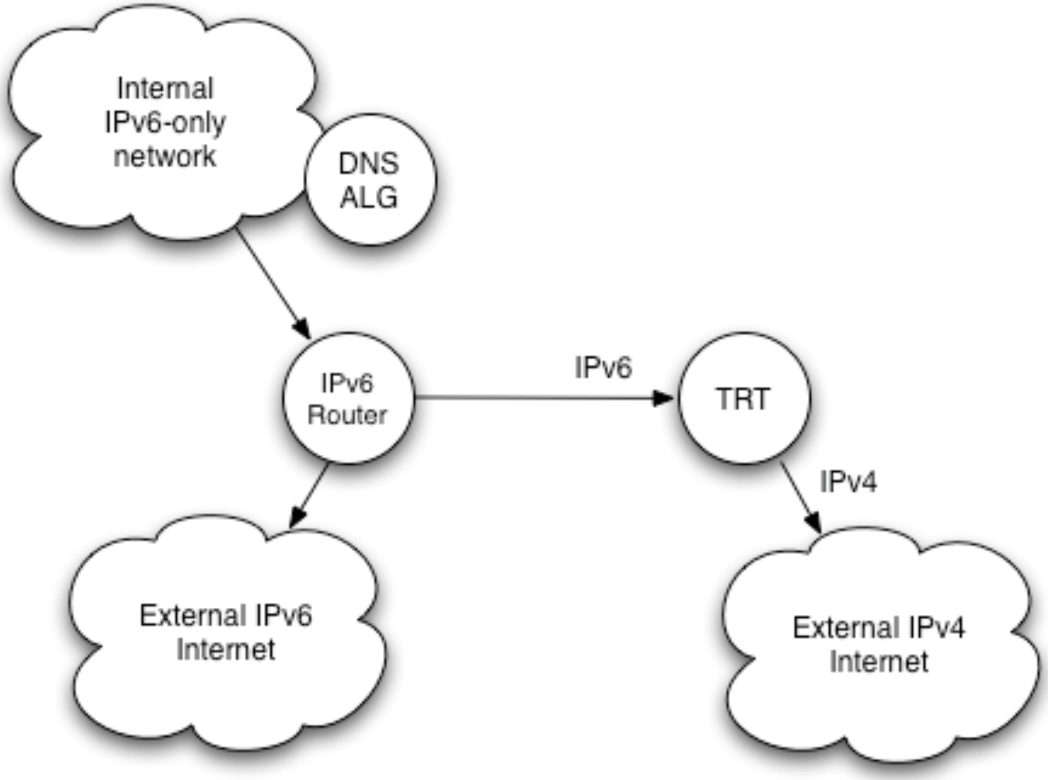
NAT-PT downsides

- Has all shortcomings of IPv4 NAT, and more
  - Needs state to be held in the NAT-PT device
  - Needs to handle IP addresses information embedded in packet payload (e.g. FTP)
  - DNS considerations are complex
- Can use from IPv4 network into IPv6 network
  - If enough IPv4 global addresses available to advertise special NAT-PT prefix addresses externally
- It's considered a 'last resort' mechanism
  - NAT-PT is being deprecated within the IETF

Transport layer: TRT

- Transport Relay Translator (TRT)
  - Designed for use in IPv6-only networks wishing to connect to external IPv4-only systems
  - TRT has internal IPv6 and external IPv4 interfaces
- External IPv6 connections work as usual
- Trick is handling connections to IPv4 nets
  - Relies on use of a DNS proxy
  - Internal IPv6 host looks up destination IP address
  - If an IPv6 address, traffic is sent to IPv6 Internet
  - If an IPv4 address, traffic is routed to the TRT

TRT topology



DNS proxy address mapping

- If internal IPv6 host is trying to reach an IPv4-only system, the DNS proxy (ALG) returns a special IPv6 destination
  - First 64 bits assigned to be unique locally
  - Next 32 bits all zero
  - Last 32 bits are the real IPv4 destination
    - <IPv6-prefix>:0:0:<IPv4 address>
- <ipv6-prefix> is routed internally to the TRT
  - Which terminates the TCP/IPv6 connection
  - And opens connection to the real IPv4 destination

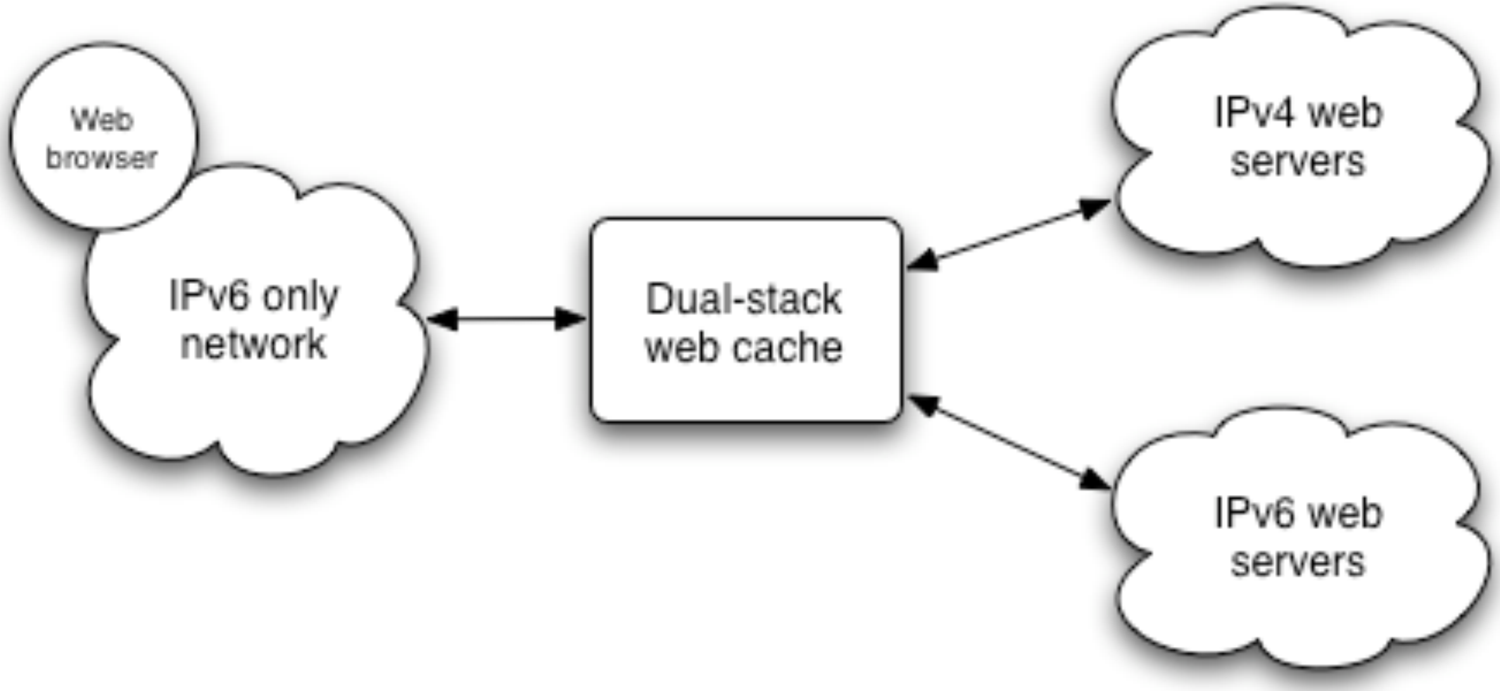
TRT pros and cons

- Pros
  - Transparent to hosts/applications
  - Scalable - can use multiple TRTs, with one internal /64 prefix used per TRT device
  - TRT can work with one global IPv4 address
- Cons
  - Like NAT, problems with embedded IP addresses in payload (e.g. FTP)
  - No simple way to allow connections initiated inbound from external IPv4 to internal IPv6 hosts

Application: ALGs

- NAT-PT and TRT are somewhat complex
- Luckily, application layer gateways (ALGs) offer a simpler alternative
- Many applications support ALGs already
  - Web cache
  - SMTP gateway
  - DNS resolver
  - SIP proxy
  - etc
- We can leverage this in a simple way

ALG topology



ALG pros and cons

- Pros
  - Simple to deploy
  - ALGs already commonly in use, e.g.
    - Web cache to reduce bandwidth usage
    - SMTP relay to channel mail through one server
  - Avoids complexity of NAT-PT or TRT
- Cons
  - Requires client configuration to use ALG
  - Only works for specific ALG-supported applications - not suited for peer-to-peer apps

But what’s the best method?

- We have a “toolbox” of IPv6 transition methods
- Some suited to certain scenarios
- IPv4 hosts will be around for a long time, with transition ongoing for many years (10-20+ years)
- Usage depends on scenario
  - A university may run dual-stack internally, and use a manual tunnel to their NREN until a native connection is available
  - A home user with IPv6 enabled on their laptop may use a tunnel broker to gain IPv6 connectivity to their home
  - Alternatively, a SOHO environment may be suited to 6to4
    - Especially where a static IPv4 address is available
- There is no single ‘best’ solution

Finally: perspectives

- Potentially deployed by a (campus) site:
  - Manual tunnels
  - Dual-stack networking
  - ALGs
  - 6to4 router (for small, typically SOHO, sites)
  - NAT-PT (for IPv6-only subnets without ALG capability)
- Potentially offered/supported by an NREN (ISP):
  - Tunnel broker server
  - Manual tunnels
  - 6to4 relay

Conclusions

- There is a large set of IPv6 transition tools available
  - No single ‘best’ solution
  - Transition plan is likely to be site-specific
- Current ‘best practice’ is dual-stack deployment
  - Natural path via procurement cycles
  - Allows experience in IPv6 operation to be gained early
- IPv6-only networks can be deployed
  - But very limited in number to date, and missing some apps
- Ultimate driver is IPv4 address space availability
  - But also need IPv4 addresses for a smooth transition