6bed4: IPv6-only appliances on any network

OpenFortress*
digital signatures

this is a bit odd. . .

Most talks today are about adding IPv6. . .

. . . this one is about removing IPv4!

Question:

Which are the last parts on your network that resent IPv6?

this is a bit odd. . .

Most talks today are about adding IPv6. . .

. . . this one is about removing IPv4

Question:

Which are the last parts on your network that resent IPv6?

Answer:

Embedded devices aka appliances!
printers, phones, netradios, satellite tuners, netstorage, . . .

embedded devices are different

We've come a long way in rolling out IPv6:

* Internet core routing is runs dual-stack

* Desktops and servers are dual-stack

* SOHO routers are slow to adapt

* Appliance manufacturers are not even *considering* IPv6

embedded devices are different

Embedded environments are different:

* Majority of customers runs IPv4, so why worry?

* Lack of memory resources

* Need to be cheap: No development funds

* Need to be quick: No time for dual-stack

embedded devices are different

Embedded environments are different:

* Majority of customers runs IPv4, so why worry?

* Lack of memory resources

* Need to be cheap: No development funds

* Need to be quick: No time for dual-stack

Embedded devices are holding back the transition to IPv6. . .

. . . but maybe we can persuade manufacturers to go IPv6-only

project background: 0cpm

SIP telephony was never designed for IPv4

Doing it IPv6-only solves *lots* of problems. . .

. . . and introduces *lots* of new ones!

project background: 0cpm

SIP telephony was never designed for IPv4

Doing it IPv6-only solves *lots* of problems. . .

. . . and introduces *lots* of new ones!

Support IPv4-only customers in a transitionary way

* **SIPproxy64** translates SIP over IPv4 ↔ SIP over IPv6

* **6bed4** is a tunnel designed for embedded devices

requirement: standard technology

* Open, exchangeable implementations

* Clarity of standards

* Expand upon existing work

requirement: simplicity

* Complex code may simply not fit in an embedded device

* A simple-to-use solution will be adopted easily

* Being able to drop NAT traversal is a big selling argument

requirement: any router

* Appliances cannot assume a co-operative router

* NAT comes in quite a few flavours
  → they all taste bad though

requirement: zero configuration

* Configuration is not end-user compatible

* User accounts cannot be rolled out en masse

* *10 easy steps* will be 12 too many to some

requirement: traceability

* Abusers of a network should be traceable

* This usually causes user accounts

* Publishing the IPv4 address, this could be skipped

desire: stateless tunnel service

* Straightforward downtime/reboot handling

* Straightforward traffic diversion

* Uplink/downlink traffic separation possible

desire: anycast addressable tunnel service

* A well-known service address could be anycasted over BGP4

* This can simply be preconfigured into appliances

* Straightforward to add/remove service nodes

surprise: none of the tunnels will work

Goal	6in4	6to4	Softwire	TSP	Teredo	AYIYA	6bed4
Standard	√	√	√	±	±	×	√
Simple	√	√	×	√	×	√	√
Any router	×	×	√	√	×	√	√
No config	×	√	×	√	√	×	√
Traceable	×	√	√	?	?	√	√
Stateless	√	√	×	×	×	×	√
Anycast	×	√	×	×	√	×	√

We will need another tunnel mechanism. . . **6bed4**

decision: run over udp

* The most stupid routers can (only) handle TCP, UDP, ICMP

* UDP will neatly pierce out through NAT

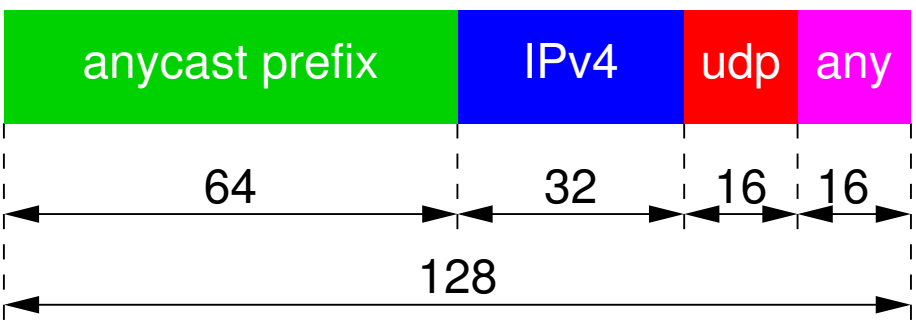
* Many tunnels have shown this to work

* Tunnel packets are: IPv4 — UDP — IPv6

decision: not anonymous

* For a tunnel, IPv4 is always assumed present
* Embed the public IPv4 address in the IPv6 address
* Traceability is ‘inherited’ from IPv4
* Also embed the ‘outside’ UDP port in the IPv6 address

IPv6–side address format:



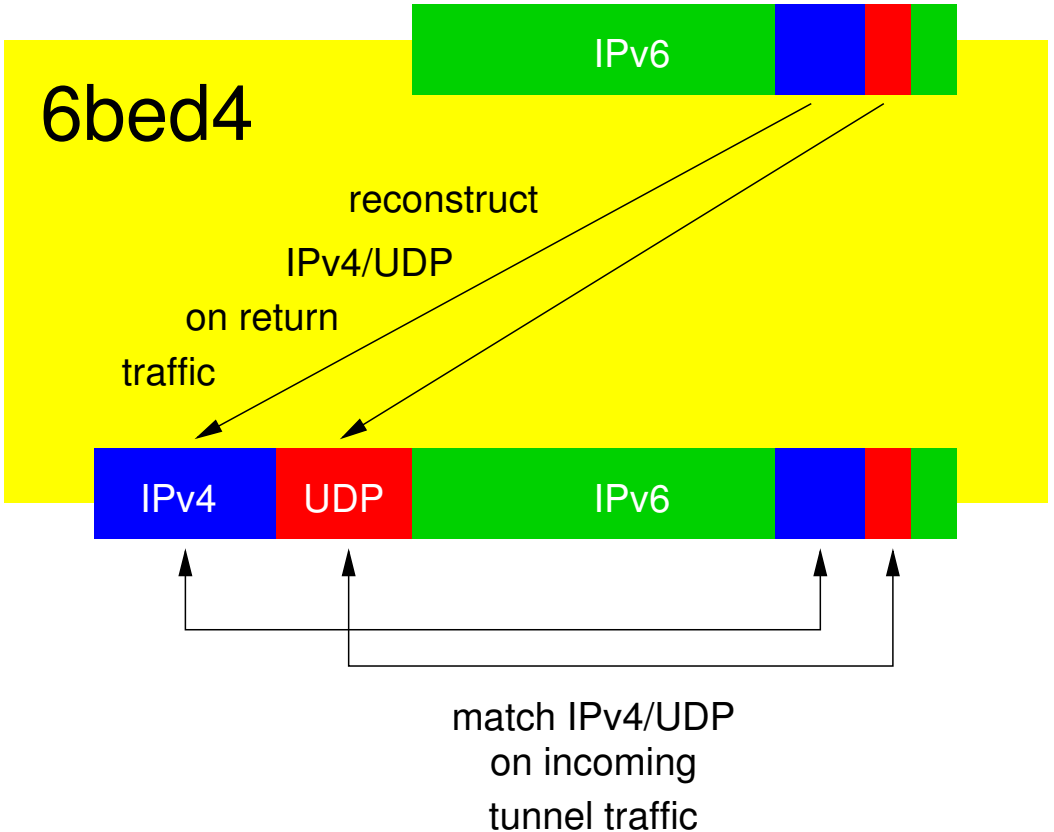
decision: no registration

* Traceability would be the only reason

* This means that 6bed4 is a public service

* Not necessarily central; it can be done en-route

decision: stateless service



decision: anycast addressable service

* Select an anycast IPv4 address for the service
* Perform 6bed4 translation en-route in gateway routers
* Announce 6bed4 locally through router protocols
* Announce 6bed4 globally through BGP4
* Setup redundant 6bed4 service without effort
* Withdraw the anycast address if maintenance is needed
* Benefit from BGP4's least-cost routing mechansims

decision: stateless autoconfiguration

* The IPv4 remote end is a well-known address and port

* The IPv4 local end can be determined locally

* Statelessness means the client can *assume* the tunnel

* Over the assumed tunnel, run stateless autoconfiguration

* Receive a /96 prefix, including ‘outside’ IPv4 and UDP port

* Note that /64 is dedicated to Ethernet, but not to autoconfig

practical issues

* OpenFortress programmed a working server, and demo-client code

* SURFnet will be the first node to run 6bed4

* OpenFortress implements SIP firmware (with partial NLnet funding)

* OpenFortress is preparing an RFC detailing 6bed4

practical issues

* OpenFortress programmed a working server, and demo-client code

* SURFnet will be the first node to run 6bed4

* OpenFortress implements SIP firmware (with partial NLnet funding)

* OpenFortress is preparing an RFC detailing 6bed4

We can use your help:

* Traffic juggler: always need more 6bed4 nodes

* Router manufacturers: en-route translation opportunities

* Students: support 6bed4 in μClinux

info@openfortress.nl

http://openfortress.nl

OpenFortress*

digital signatures