6bed4: IPv6-only appliances on any network

OpenFortress*
digital signatures

background and motivation

SIP telephony consists of unconnected islands:

* IPv4 $\Rightarrow$ NAT $\Rightarrow$ RTP proxy

* RTP proxy $\Rightarrow$ Media restrictions

* RTP Proxy $\Rightarrow$ Phone tapping point

* RTP Proxy $\Rightarrow$ no ENUM, no ITAD, no sip:bakker@orvelte.nep

We should use IPv6 (and only IPv6):

* Always direct media connections

* Need to have IPv6 on each end... backport to IPv4

backporting techniques

Support IPv4-only customers in a transitionary way

* **0cpm Firmerware** reprograms physical phones

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

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

Today's topic is 6bed4.

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

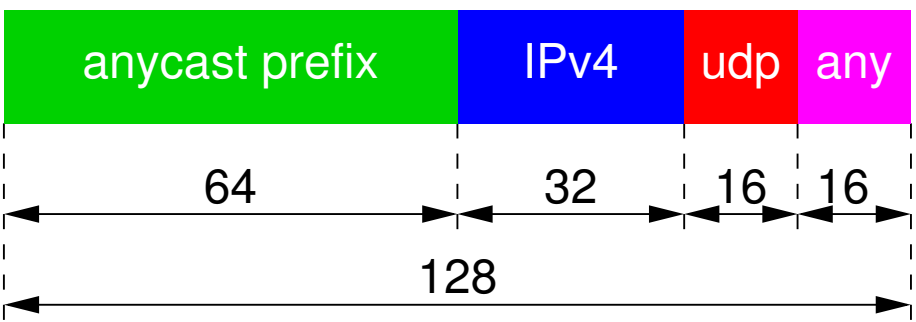
* Many tunnels have shown this to work

* 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:



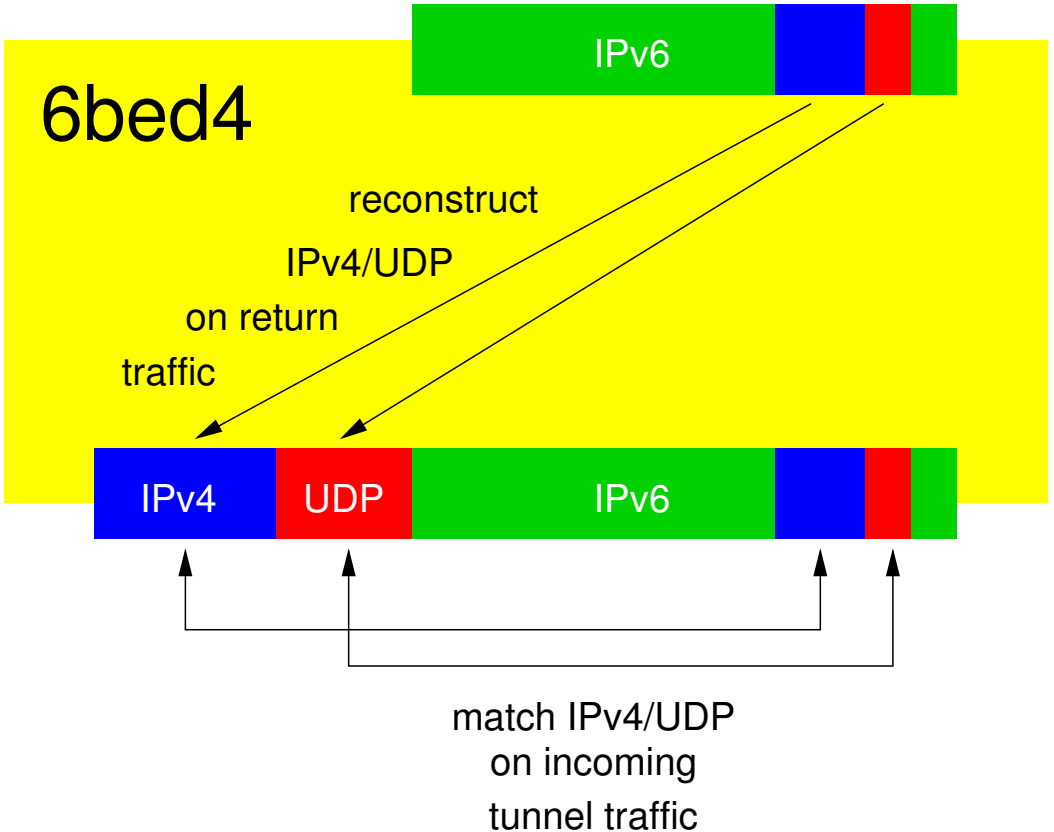
decision: no registration

* 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 /112 prefix, including ‘outside’ IPv4 and UDP port

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

practicalities and progress

* OpenFortress programmed a working server and demo-client

* OpenFortress' 0cpm Firmerware will be an actual client

* SURFnet hosts the first 6bed4 node: 145.100.190.242

* OpenFortress proposed `draft-vanrein-v6ops-6bed4-00`

draft feedback

* Why not tunnel *X*?

* What scope would be ‘embedded’?

draft feedback

* Why not tunnel *X*?

* What scope would be ‘embedded’?

* You forgot about local connections!
  → *Oops. . .*
  → Peer review clearly is a Good Thing™

train of thought

* Need to be aware of LAN peers; use multicast
* The protocol to use would simply be Neighbour Discovery
* IPv6 on the LAN may raise alarms → embed in 6bed4
* If the LAN fails, perhaps NAT hairpinnnig can work
* We could also try direct access to a remote peer
* Anything but symmetric NAT should accept direct 6bed4
* Effectively bypassing the public 6bed4 server

another approach to nat traversal

* Have a working IPv6 connection *everywhere*

* Start from this ability, and reduce to optimise

* Make *no assumptions* about the NAT model

* Direct connections *will* be found when possible

upcoming changes in draft v01

Embedded → Revive peer-to-peer using IPv6

Incorporating direct 6bed4 traffic:

* Public 6bed4 nodes function as *fallback service*

* Try the IPv4 address and UDP port directly

* Underlying IPv4 routing more dynamically

* Confusion? Remote peer might see changed IPv4/UDP

* Security? Redirections might be dangerous

* Keepalive? Asymmetric paths could close holes in NAT

conclusions and references

* SIP over IPv6 holds many advantages. . .
  . . . and you can switch to it *today*

* Information about this tunnel:
  http://devel.0cpm.org/6bed4/

* Phone firmware using 6bed4 as its IPv6 fallback:
  http://devel.0cpm.org/firmerware/

* Wrap IPv6 around your SIP/IPv4 solution:
  http://devel.0cpm.org/sipproxy64/

* Find news on Diaspora: `#0cpm`

info@openfortress.nl

http://openfortress.nl

OpenFortress*

digital signatures