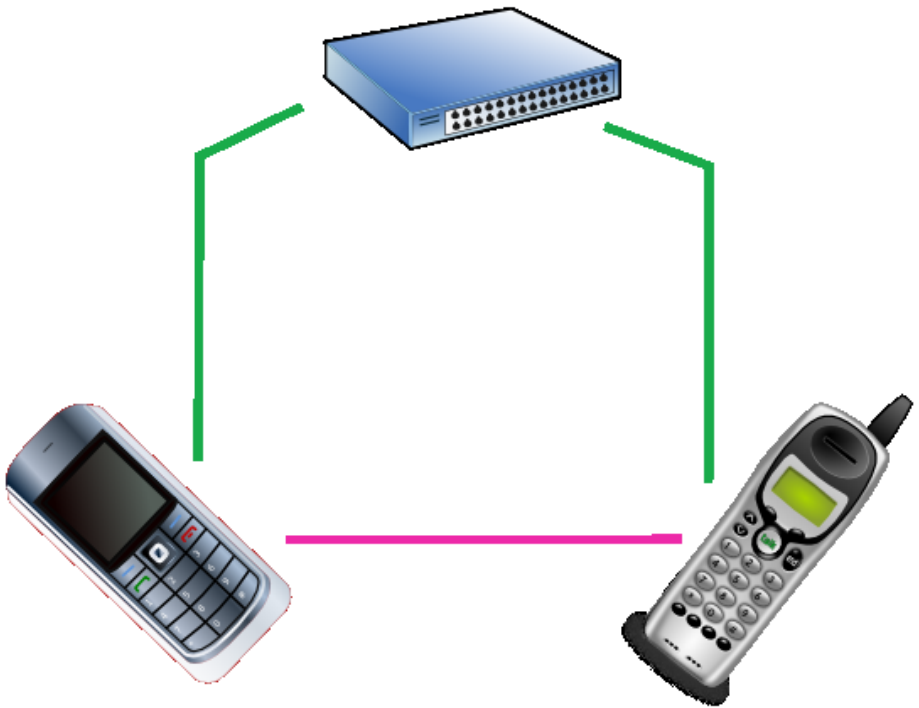
IPv6 telephony in an IPv4 world

OpenFortress*
digital signatures

running sip over ipv4

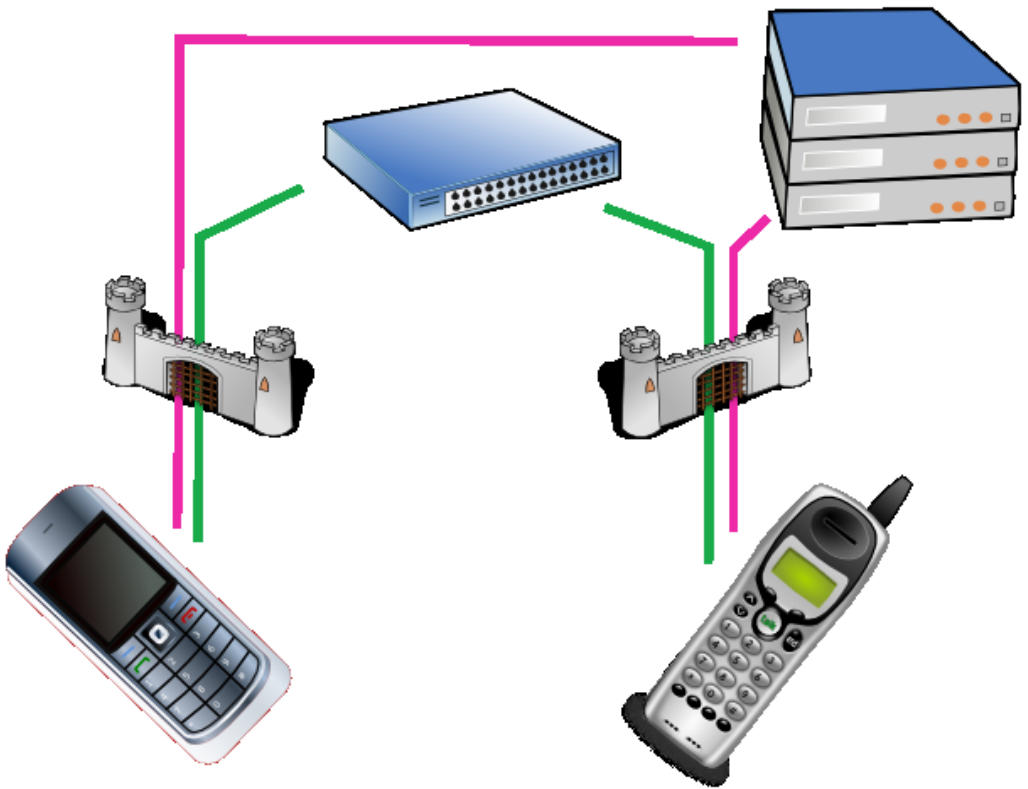


Rolling out SIP seems so easy. . .

running sip over ipv4

* SIP travels around to setup the call
* RTP connects media streams as directly as possible
* Why pay for telephony if media bandwidth is fixed-rate?

running sip over ipv4



. . . but it usually ends in hosting media sessions

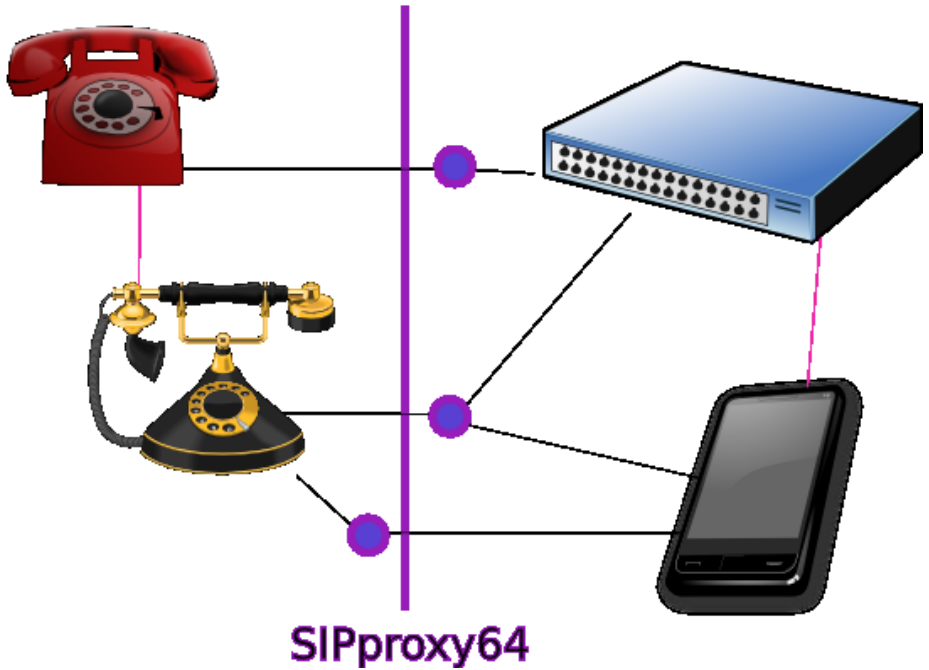
running sip over ipv4

* Firewalls are not transparent
* NAT makes SIP very difficult to get right
* Calls behind NAT cannot always be connected
* Direct phone calls over the Internet are not generally possible
* To provide certainty, an RTP proxy is needed
* This means carrying media traffic in your bandwidth
* Not likely to scale up to other forms of media

The (bold) solution is SIP over IPv6 only

* IPv6-only implies a need for transitioning measures

sipproxy64: bridging sipv4 and sipv6



* IPv4 phones with IPv6 representation
* Possibly forward default-routed traffic

OpenFortress*

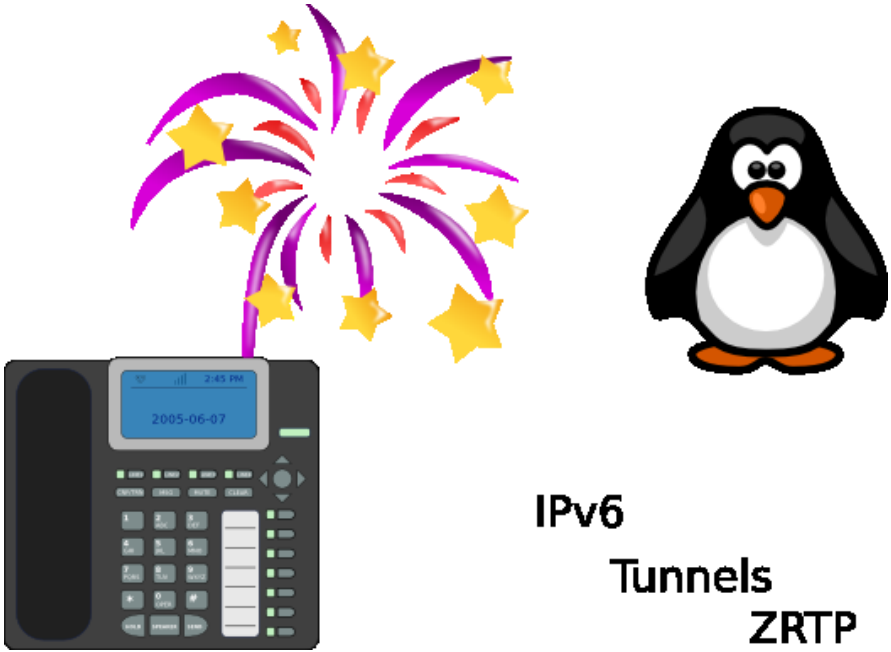
sipproxy64: bridging sipv4 and sipv6

* SIPproxy64 has an IPv4-side and an IPv6-side
* Phones or PBXs often support one address family
* SIPproxy64 makes such phones visible on the other side
* It will relay and translate SIP and RTP

* Size is about 32 kB (even before minimising)
* Only depends on `libosipparser2`

* SIPproxy64 is ideal in routers
* SIPproxy64 assumes local presence of IPv6
* SIPproxy64 takes away a *let's wait until...* motivation
* SIPproxy64 is entirely symmetrical in IPv4/IPv6

open source firmware for sip phones



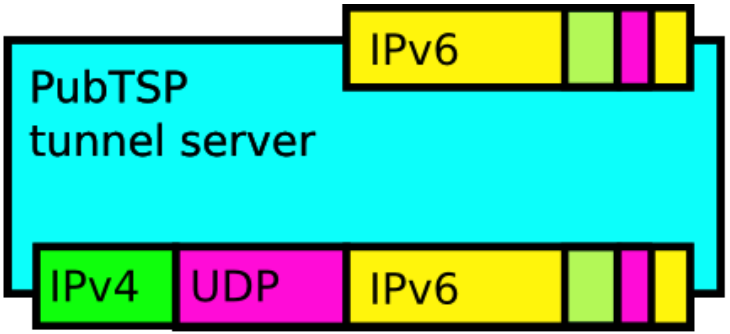
* working on open source firmware for SIP phones
* IPv4-only nets supported with tunneling

open source firmware for sip phones

* Telco's and phone makers wait for each other
* Lack of critical mass, nobody's moving
* We decided to break through it with open source firmware

* Seeking active manufacturer participation
* License (probabbly) will be GPLv3:
  → want manufacturers to share their porting work
  → want firmware upgrades open to user
  → want to see an active developer community

pubtsp: support for ipv6 apps on ipv4 nets



1. **Not anonymous**: NAT IPv4/UDP in IPv6
2. **No registration** needed
3. **Stateless** tunnel service
4. **Anycast** addressable service

protocol in search of routers

Embedded apps can be IPv6-only *if tunnels are available*

pubtsp: support for ipv6 apps on ipv4 nets

* Embedded devices are not likely to support two address families
* Due to lack of resources: time, vision, money, memory space
* IPv4 will be the safest choice for years to come

Embedded devices will probably stick to IPv4

* IPv6 is currently not likely to work everywhere
* Exception is when tunnels are suitable:
  → Not anonymous == no extra danger of abuse
  → No registration == no configuration
  → Stateless == easy to use, easy to serve
  → Anycast == can be found nearby, keep traffic local
* No current tunneling protocol seems to support this?

pubtsp: support for ipv6 apps on ipv4 nets

* PubTSP is a profile of RFC5572
* Low 64 address bits contain IPv4 address and UDP port:
  → Obtained during tunnel negotiation
  → tunnel→IPv6: egress check
  → tunnel←IPv6: derive IPv4/UDP values

* PubTSP server is a simple tunnel program
* Looking for LIR/BGP speakers for anycast address
* **Looking for routing parties to host tunnel servers**
* Suggesting ISPs terminate the traffic locally

references

http://devel.0cpm.org/sipproxy64/

http://public-tsp.org/

info@openfortress.nl

http://openfortress.nl

OpenFortress*

digital signatures