Homenet

multi-router, multi-ISP – fully automagic!

Gert Döring, SpaceNet AG, Munich

Swiss IPv6 Business Conference, June 17 2014, Zurich

Overview

- „nobody needs more than one router at home!“

- … but could it be done?
  - ~~DHCPv6-PD („old IETF approach“)~~
  - ~~CableLabs hipnet~~
  - homenet („the future of the home“)

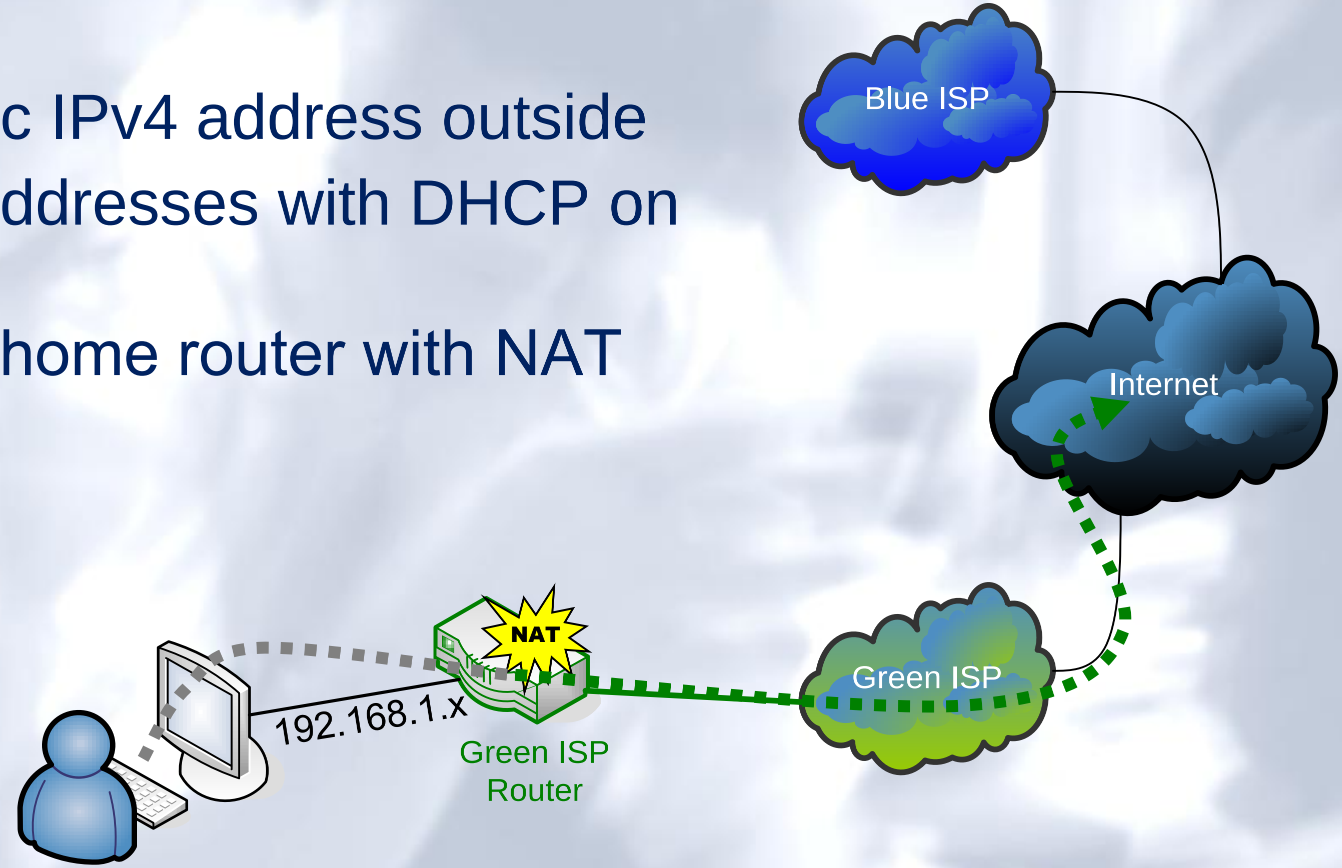
not presented today,
but included in slides
provided online

- „Proof of Concept“-setup with OpenWRT

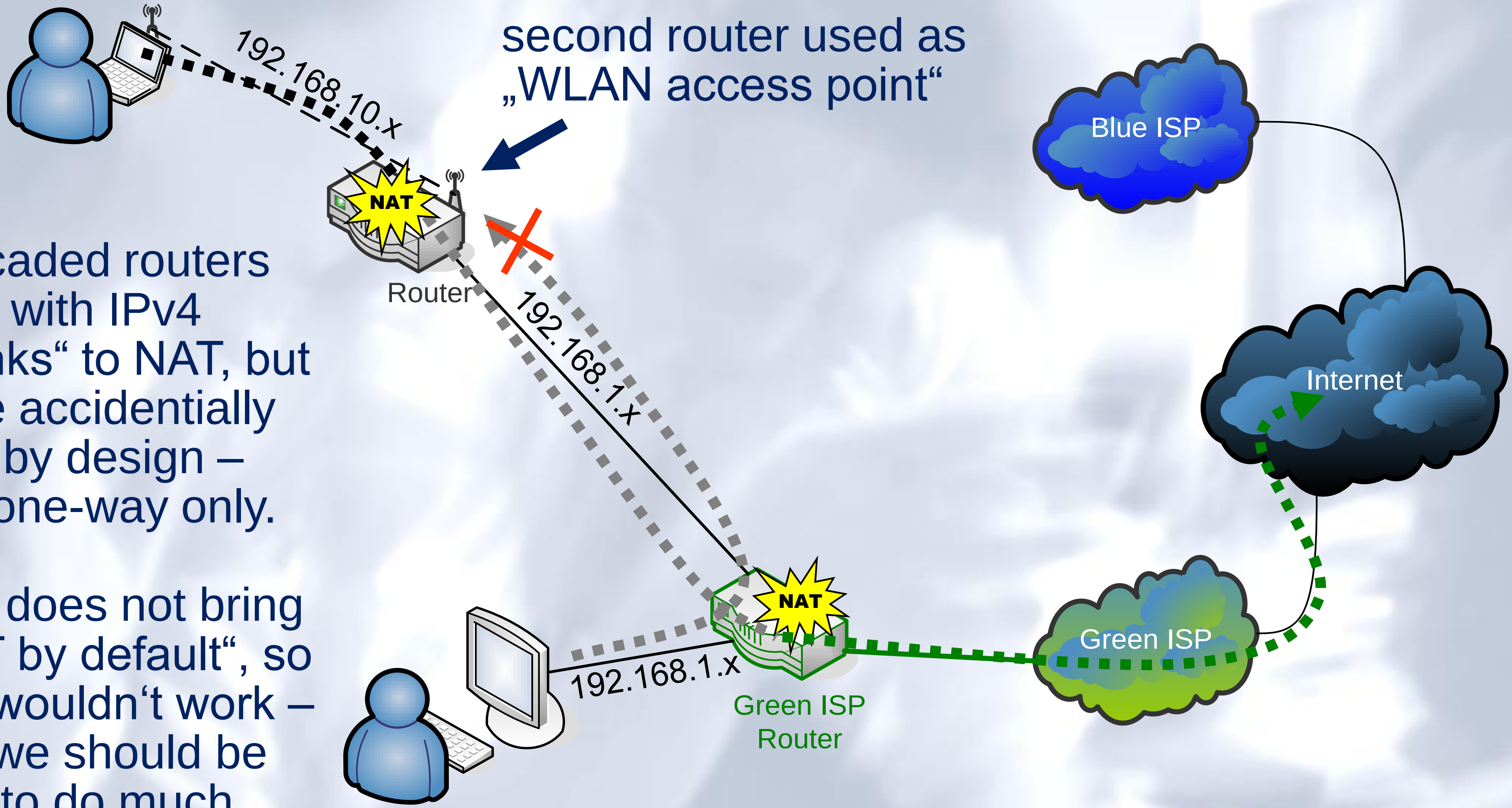
nobody needs more than one router at home!

a „typical“ network at home:

- end user ISP
- single dynamic IPv4 address outside
- private IPv4 addresses with DHCP on the inside
- „off the shelf“ home router with NAT



nobody needs more than one router at home?

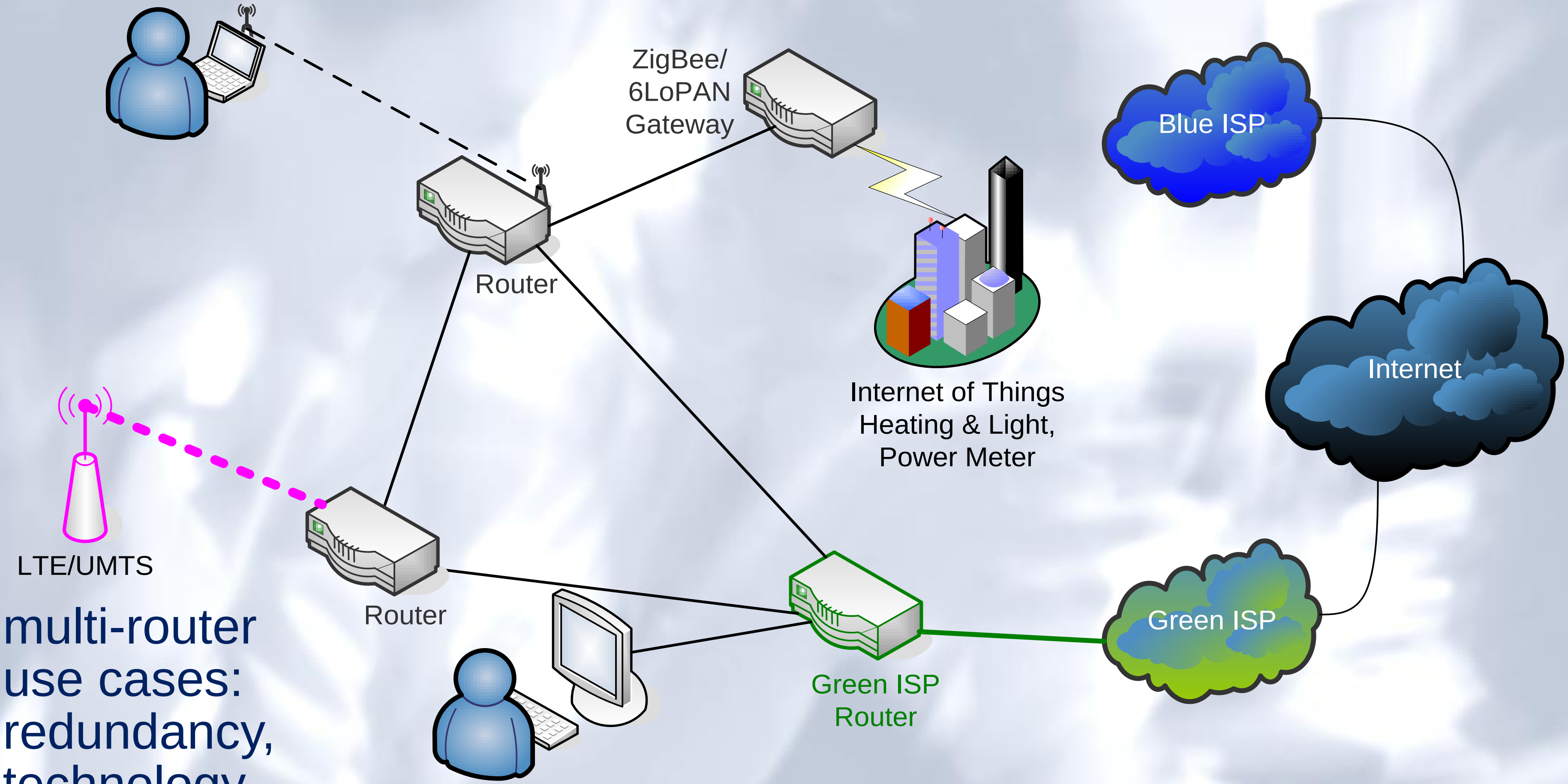


second router used as „WLAN access point“

Cascaded routers work with IPv4 „thanks“ to NAT, but more accidentially than by design – and one-way only.

IPv6 does not bring „NAT by default“, so that wouldn‘t work – and we should be able to do much better → new ideas?

more routers!



multi-router use cases: redundancy, technology bridging, ...

multi-router / multi-ISP: the problem spots

- support for non-trivial topologies
  - networks with „loops“ (e.g. by accident, or for redundancy)
  - networks with multiple routers connecting to different ISPs
  - networks with ISP routers on different *sides* of the home
- support for active/active multihoming to multiple ISPs
  - use cable ISP for bittorrent, DSL ISP for web surfing?
- proper support for naming across complex topology
  - humans want to access things by name, not by IP address
- efficient prefix distribution
  - multi-level DHCPv6-PD „burns“ too many subnets
- efficient signalling of new information
  - no elegant signalling mechanism in DHCPv6-PD or RA to notice if upstream router is turned off, etc.
- everything has to work *without manual configuration*!*

Homenet

- what is „Homenet“?
  - working group inside IETF, focussing on „how should networking in a typical ‚home‘ environment look like, in a few years from now?“
  - fully automatic (zero-conf for addresses and naming)
  - multihoming to multiple Internet providers
  - if needed, new „routing protocols“ will be created
  - focus on IPv6, support for IPv4 „as well as possible“

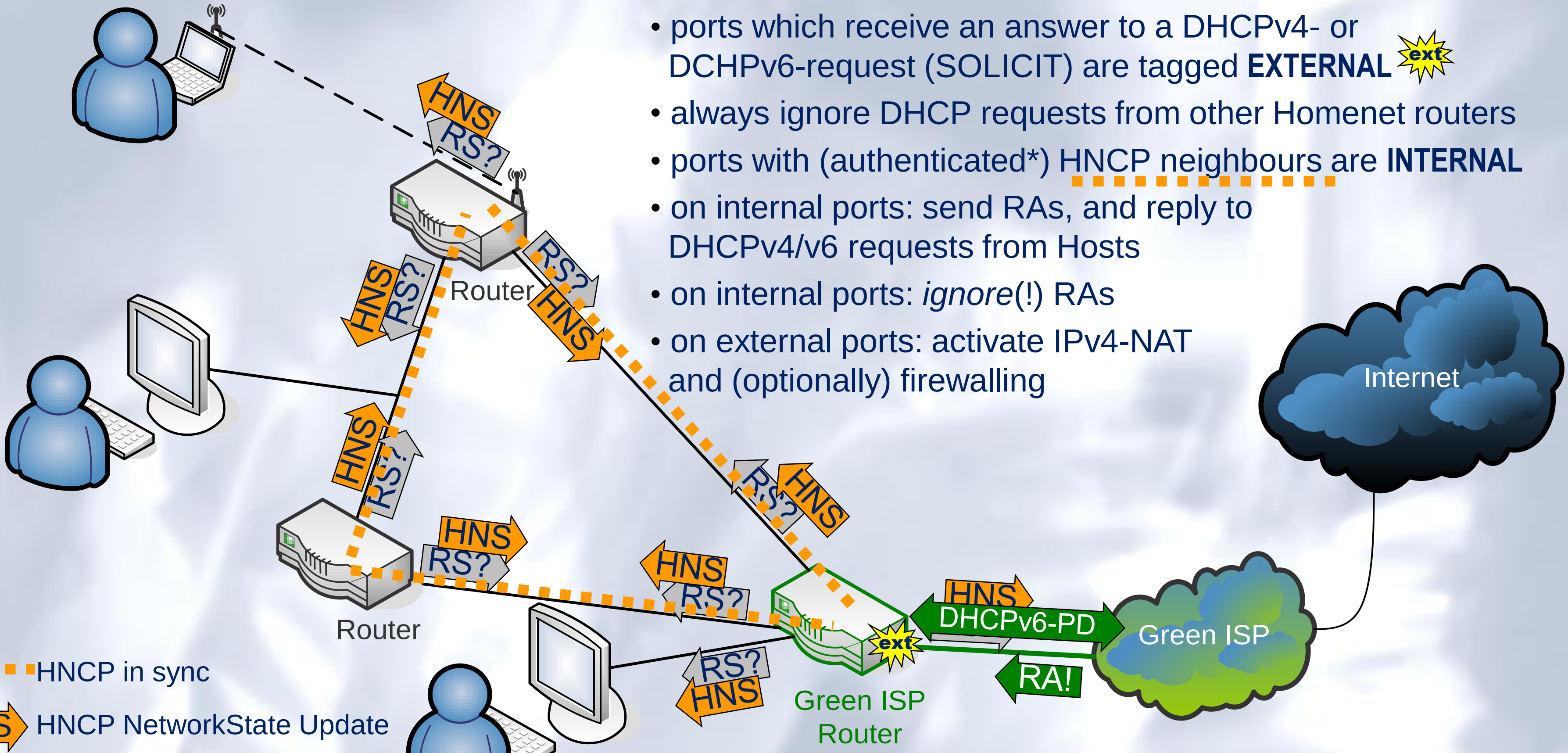
  - http://datatracker.ietf.org/wg/homenet/
  - http://datatracker.ietf.org/doc/draft-ietf-homenet-arch/

Homenet: The Protocols

- towards the Internet Service Provider (ISP):
  - RS/RA, DHCPv6-PD – „the usual stuff“
- in the Homenet:
  - HNCP as „information flooding“ protocol (earlier drafts used extentions to OSPFv3)
  - distributed algorithm for prefix assignment (/64)
  - hybrid mDNS + DNS for „naming“
  - source/destination address routing (SADR), either using HNCP, or external protocols (Babels, OSPFv3, …)
  - RA, DHCPv6 for signalling towards hosts – „as usual“

Homenet – single-homed

- Homenet routers are „directionless“ (no LAN or WAN ports)
- ports which receive an answer to a DHCPv4- or DCHPv6-request (SOLICIT) are tagged **EXTERNAL** ext
- always ignore DHCP requests from other Homenet routers
- ports with (authenticated*) HNCP neighbours are **INTERNAL**
- on internal ports: send RAs, and reply to DHCPv4/v6 requests from Hosts
- on internal ports: *ignore*(!) RAs
- on external ports: activate IPv4-NAT and (optionally) firewalling



▪ ▪ ▪ ▪ HNCP in sync

HNS HNCP NetworkState Update

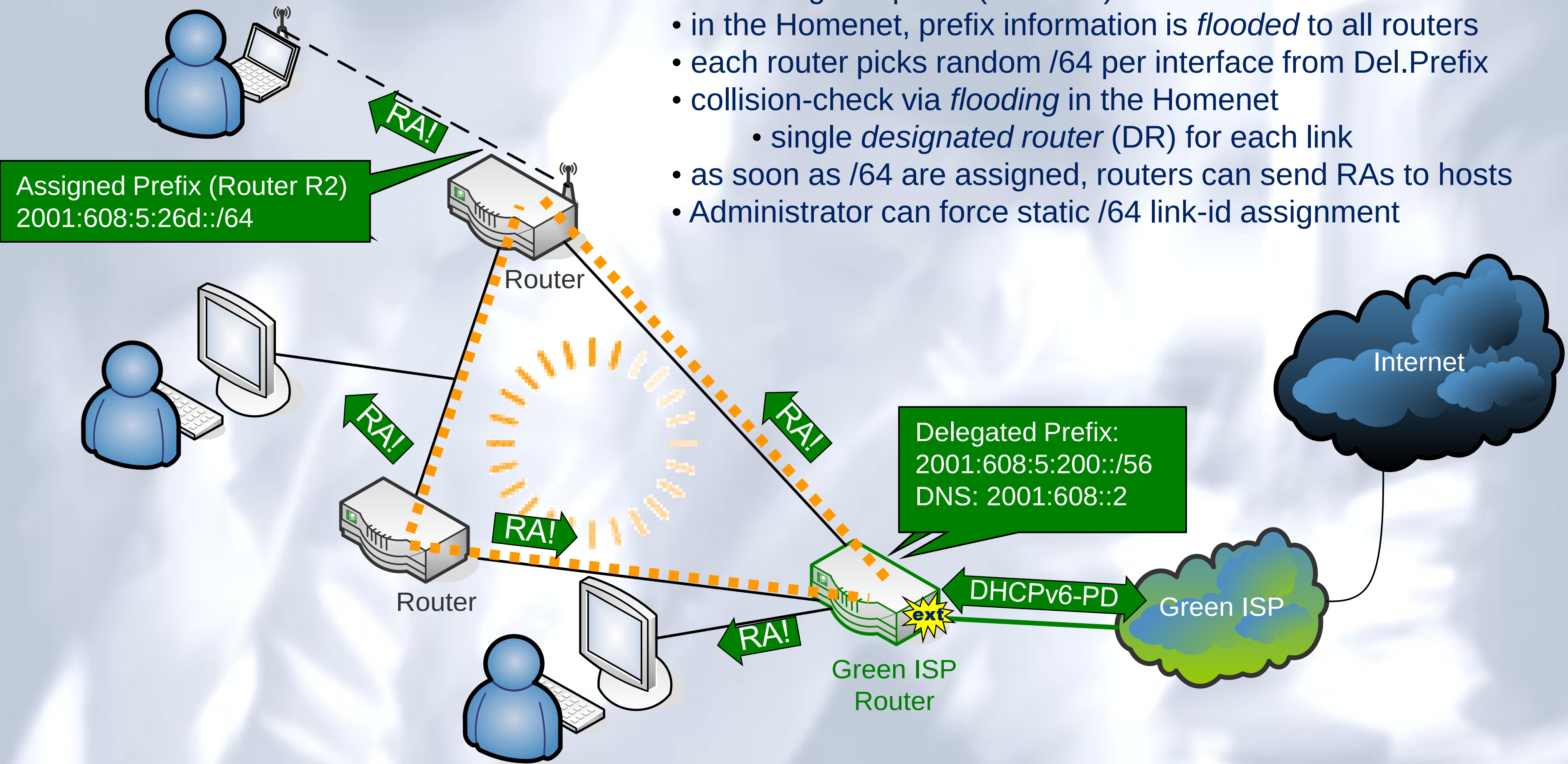
RS? router solicitation, DHCPv4- and DHCPv6-PD-SOLICIT

RA! router advertisement, DHCPv4 and/or DHCPv6 ADVERTISE

DHCPv6-PD still used on ISP uplink. Inside the Homenet, a more efficient mechanism is needed.

Homenet – single-homed

- ISP delegates prefix (as usual) via DHCPv6-PD
- in the Homenet, prefix information is *flooded* to all routers
- each router picks random /64 per interface from Del.Prefix
- collision-check via *flooding* in the Homenet
  - single *designated router* (DR) for each link
- as soon as /64 are assigned, routers can send RAs to hosts
- Administrator can force static /64 link-id assignment



Assigned Prefix (Router R2)
2001:608:5:26d::/64

Delegated Prefix:
2001:608:5:200::/56
DNS: 2001:608::2

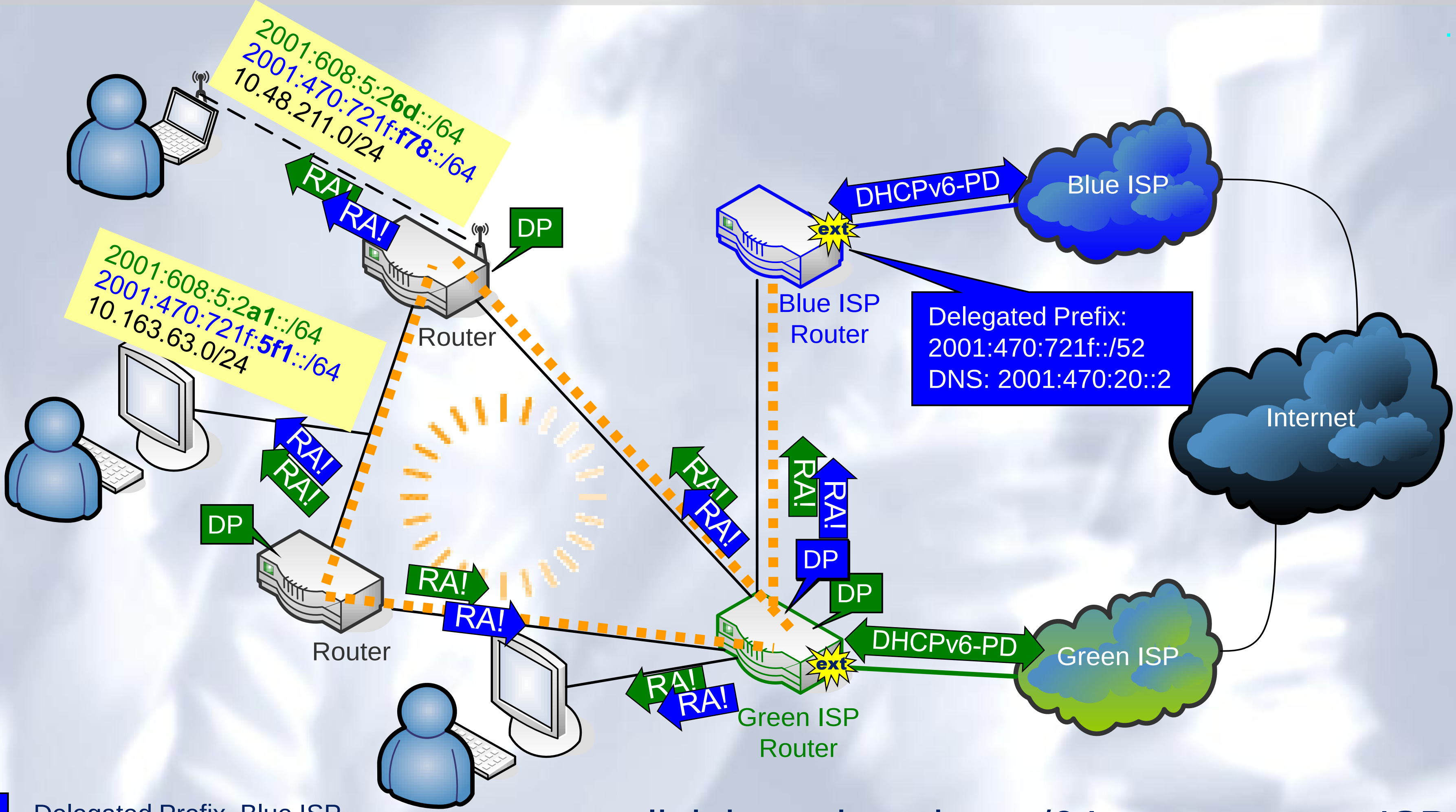
RA! router advertisement, DHCPv4
and/or DHCPv6 ADVERTISE

each router selects /64 from delegated prefixes
using *distributed algorithm*, no router hiearchy

Homenet – HNCP detailed

- Every router has a Router ID and a set of data in TLV form, building this routers's "NodeState" data set. For less computionally intensive comparison, a hash value of this NodeState data is built, H(R).
  - this NodeData is data like "which neighbour routers exist on which link" (intrinsic to HNCP) or "which prefixes have been assigned by upstream routers" ("external" data used by protocols on top of HNCP)
- In steady state (synchronized):
  - each router knows the NodeState for all other routers
  - each router periodically publishes a "NetworkState Update" message, which consists of a hash of (all individual NodeState hashes). This update is sent by link-local multicast on all links.
  - as every router has the same state, the NetworkState hash will be the same on every router, so comparison of "are we all in sync?" is very easy, just the hash needs to be compared
- If a router has new information (like: new prefix from upstream), it updates its NodeState data, increments its serial number, calculates a new hash H(R), and a new NetworkState hash. Then it sends the new H(H) value out
  - each neighbour will notice that the hash values do no longer match, and request a detailed list of individual router hashes + serial numbers (NetworkState Request) by link-local unicast message
  - by comparing hashes and serial numbers, the router will know *which* router has updated its data, and request an update for *this* router's data from the publishing neighbour (NodeData Request), again by link-local unicast message
  - then, this newly updated router will update its local copy of the announcing router's NodeData with all new information, build a new NetworkState hash, and publish the new hash in a NetworkState Update multicast message
  - the next neighbours "further down" in the network will now see the updated NetworkState hash, and come back querying for the updated NodeData.
- In other words: updated information is only flooded by means of a change in the NetworkState hash, and optionally NodeData hash, but the actual update is unicast-pulled by each neighbour who does not yet have it.
- The use of a NetworkState hash and serial numbers for each individual NodeData hash avoids the need to have a global "Network serial number“ and ACKed TLA updates etc. to get the network synchronized

Homenet – multi-homed: prefixes



2001:608:5:26d::/64
2001:470:721f:f78::/64
10.48.211.0/24

2001:608:5:2a1::/64
2001:470:721f:5f1::/64
10.163.63.0/24

Delegated Prefix:
2001:470:721f::/52
DNS: 2001:470:20::2

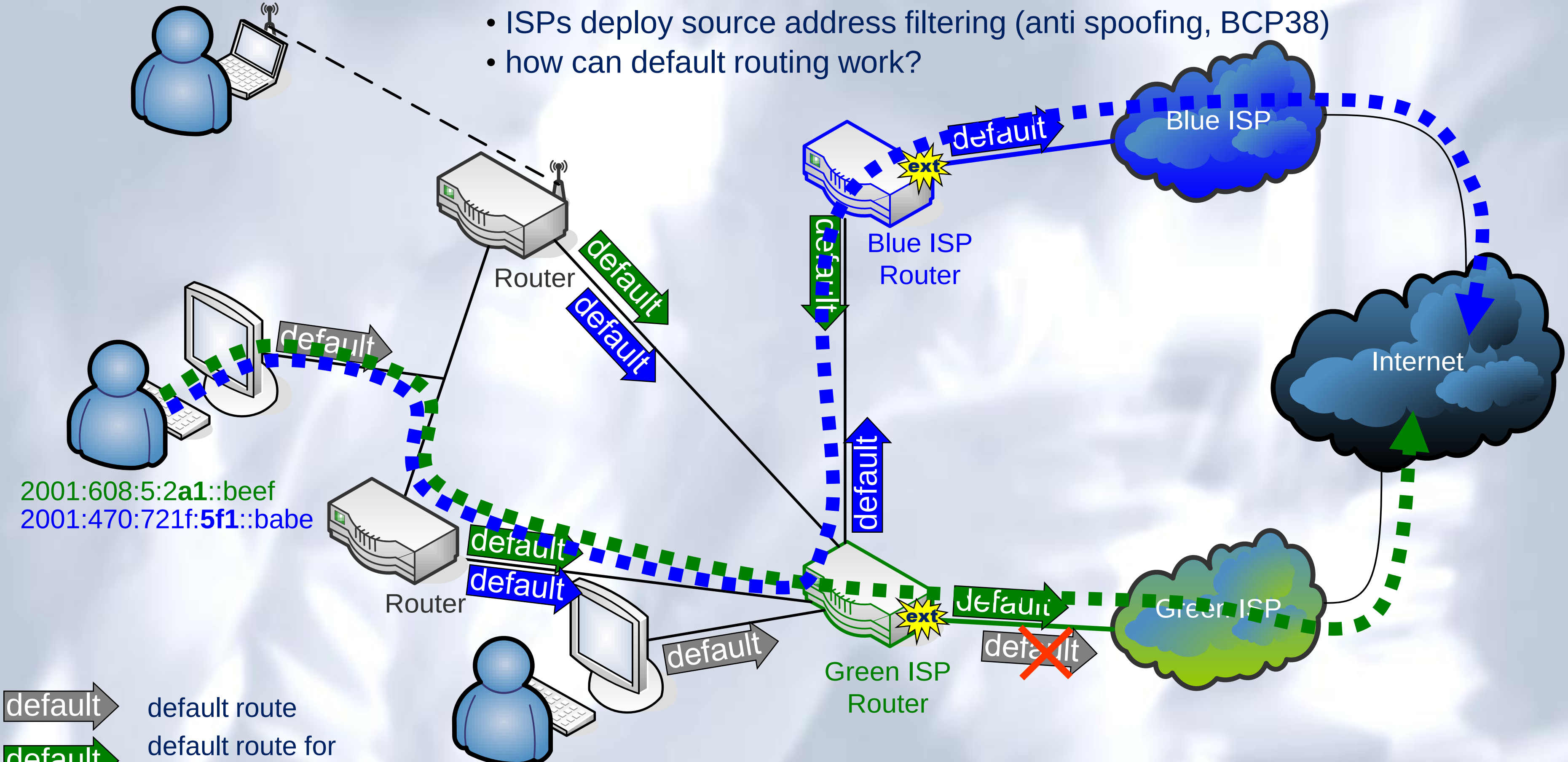
DP Delegated Prefix, Blue ISP

RA! router advertisement, DHCPv4 and/or DHCPv6 ADVERTISE, Green ISP

every link is assigned one /64 per upstream ISP (1 blue, 1 green), plus IPv4, and optionally ULA

Homenet – multi-homed: routing

- PCs have multiple global IPv6 addresses
- ISPs deploy source address filtering (anti spoofing, BCP38)
- how can default routing work?



default	default route
default	default route for green **source**-IPs
default	default route for blae **source**-IPs

Due to ISP filters, default routing in the Homenet *has* to take *source* addresses into account

Homenet – multihomed: consequences

- Homenet multihoming approach implies:
  - end systems (PCs etc) have multiple global IPv6 addresses
  - forwarding in routers has to take source address into account
  - for outgoing connections, the *end system* selects the ISP to use for *every new session* by picking the appropriate source address
  - ISP selection is controlled by application, not by router or ISP
  - session survivability achieved via shim6, sctp or mp-tcp
  - for incoming connections, destination address selects ISP to use (read: user controlled via DNS entry)

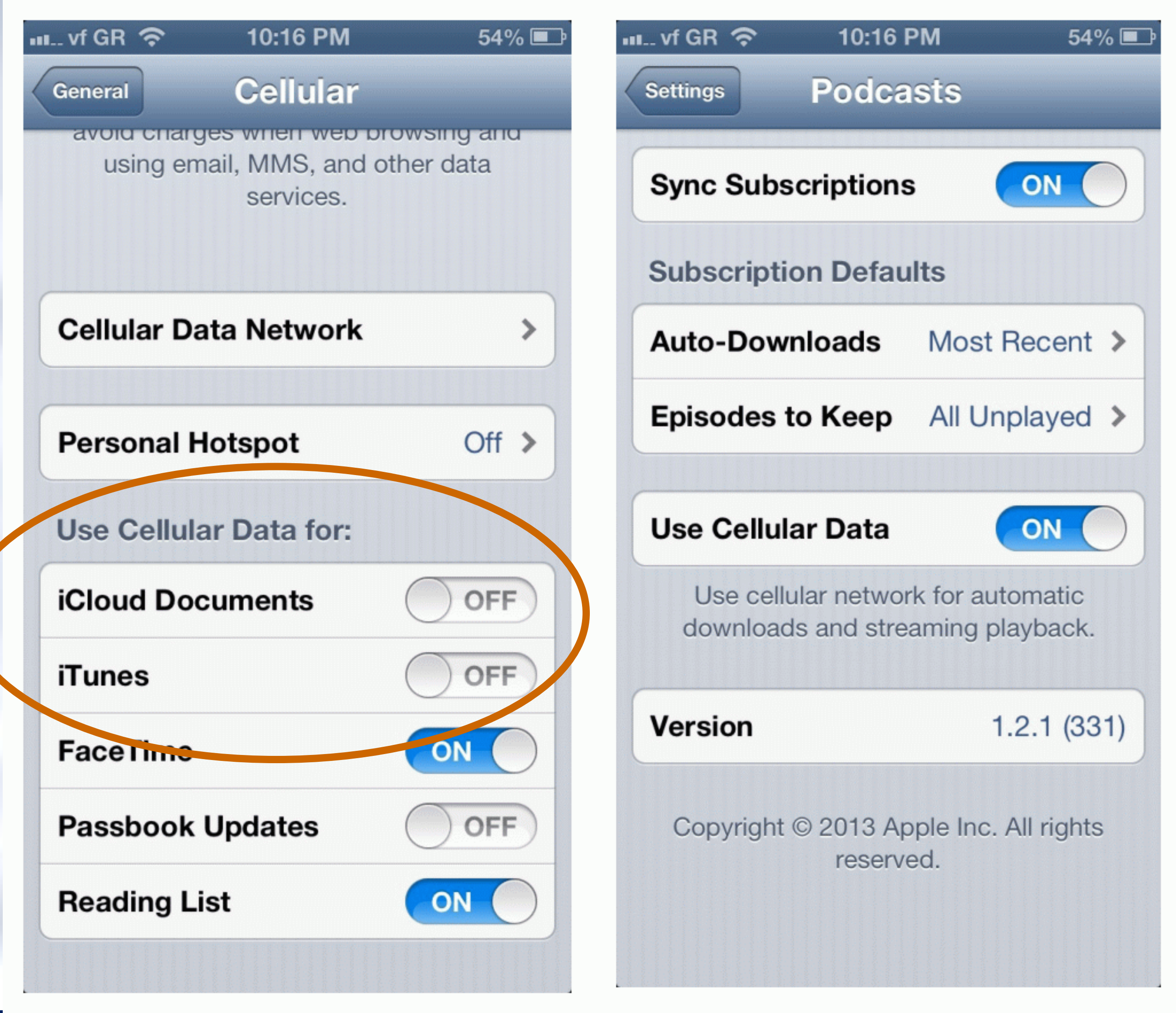
- „this is never going to work!!“
  - (side remark: this is intended *for home networks only!*)

Multi-Address Multihoming

won‘t work?

too complicated?

users will not understand?



vf GR 10:16 PM 54%

General

Cellular

avoid charges when web browsing and using email, MMS, and other data services.

Cellular Data Network ›

Personal Hotspot Off ›

Use Cellular Data for:

iCloud Documents	OFF
iTunes	OFF
FaceTime	ON
Passbook Updates	OFF
Reading List	ON

vf GR 10:16 PM 54%

Settings

Podcasts

Sync Subscriptions ON

Subscription Defaults

Auto-Downloads	Most Recent ›
Episodes to Keep	All Unplayed ›

Use Cellular Data ON

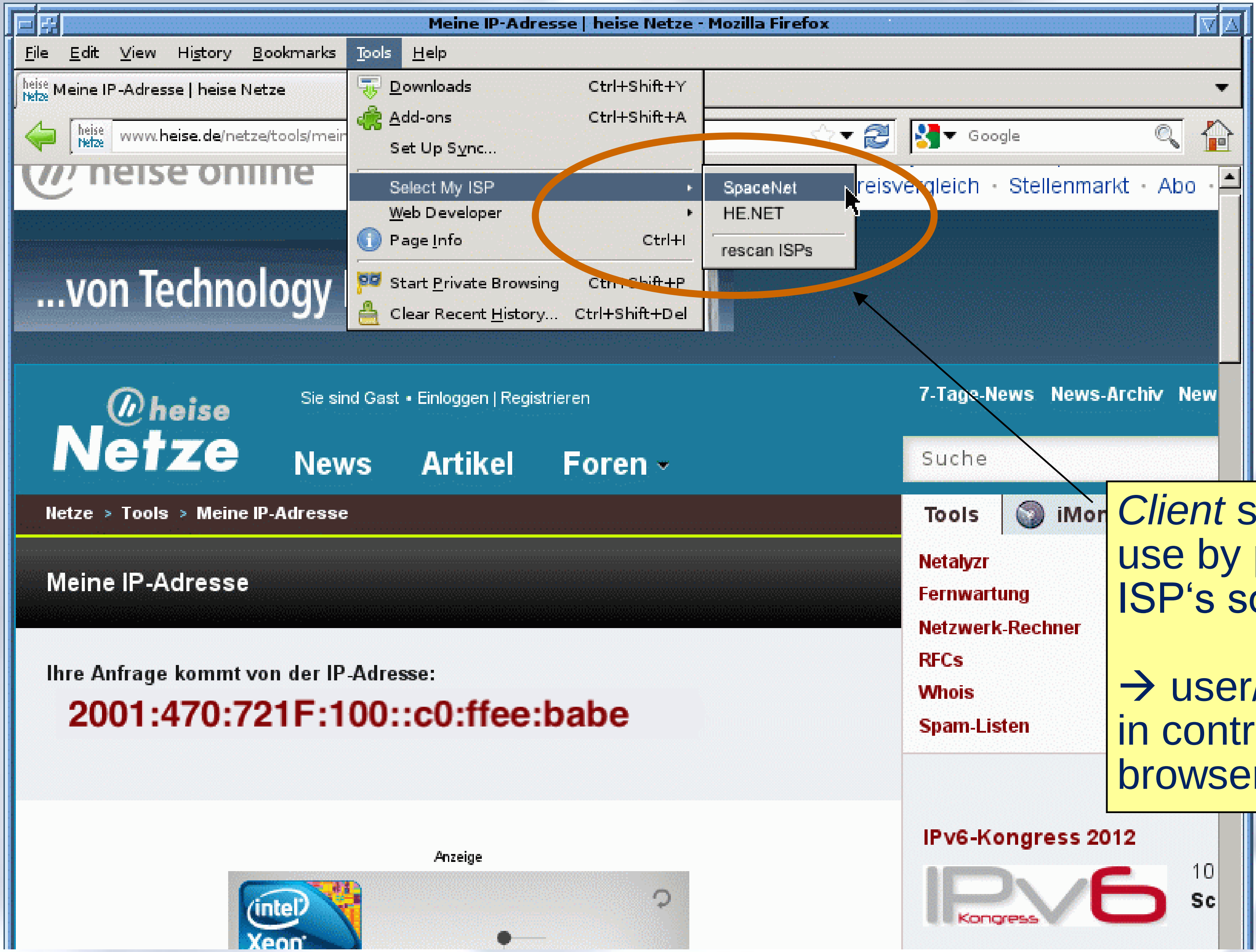
Use cellular network for automatic downloads and streaming playback.

Version 1.2.1 (331)

Copyright © 2013 Apple Inc. All rights reserved.

Image taken from Mark Townsley‘s homenet talk at RIPE 67 and used with permission

Multi-Address Multihoming



Meine IP-Adresse

Ihre Anfrage kommt von der IP-Adresse:

2001:470:721F:100::c0:ffee:babe

Client selects ISP to use by picking this ISP‘s source address

→ user/application is in control, e.g. using browser plugin

Multi-Address Multihoming: missing pieces

- for „using what works best“:
  - source address failover and/or source address probing and caching („happy eyeballs 2“)
    - ietf-mif-happy-eyeballs-extension
- for „application X shall use ISP Y (only)“:
  - some sort of system policy table to define *per-application preference* for ISP selection, *easy* to configure (like iOS)
  - a standard mechanism how to attach a human-readable „label“ to an IPv6 prefix („SpaceNet“, „Telekom“) – e.g. using DHCPv6, RA, or „well-defined lookups“
    - draft-lepape-6man-prefix-metadata, draft-korhonen-6man-prefix-properties, draft-bhandari-dhcp-class-based-prefix
  - direct API support in applications, that want to present the options to their users („SpaceNet“, „HE.Net“, „try both“)

Homenet – other issues addressed

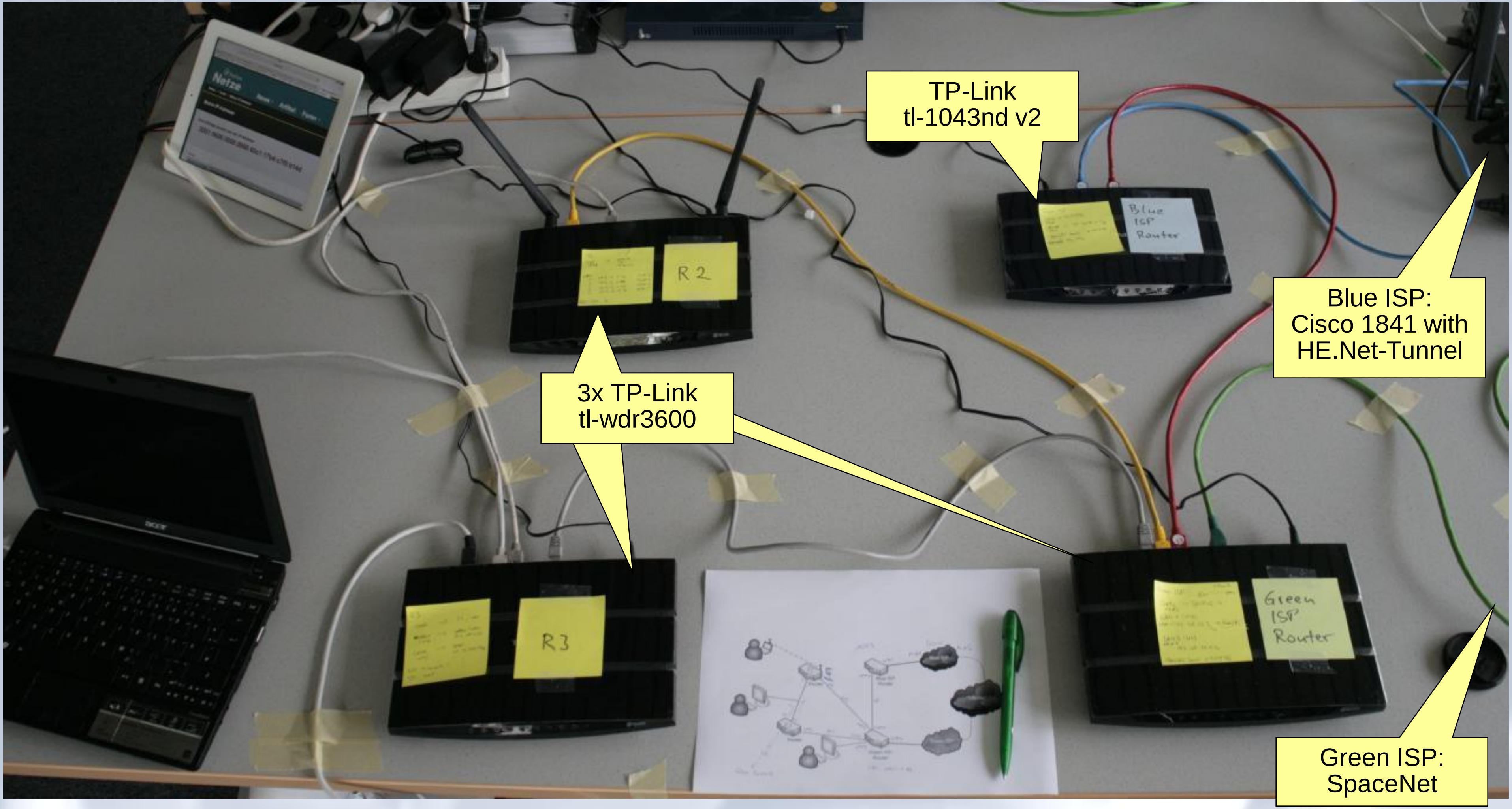
- Naming and Service Discovery
  - mDNS + DNS „hybrid“
  - every router learns mDNS announcement (and answers mDNS queries). Data is put in local DNS zones, which other routers learn about by HNCP.
  - local DNS recursor knows homenet DNS names and zones
  - draft-stenberg-homenet-dnssd-hybrid-proxy-zeroconf
  - draft-cheshire-mdnsext-hybrid
- Security model using „fully authenticated“ homenet routers *(optional!)*
  - registering new routers master router, e.g. helped by smartphone app
  - HNCP already incorporates strong cryptographic hashes and signatures
  - draft-behringer-homenet-trust-bootstrap
- Interaction with Hipnet or RFC7084 CPEs
  - Homenet routers offer DHCPv6-PD and DHCPv4 for downstream routers
  - Service-Discovery and „Homenet behind RFC7084“ are „tricky“
  - draft-winters-homenet-sper-interaction

Homenet – will it work in practice?

- first available implementation of homenet drafts based on OpenWRT
- Setup:
  - install OpenWRT „trunk“ (http://openwrt.org/)
  - # opkg update && opkg install hnet-full
  - # vi /etc/config/network

    ```
    config interface  hlan
         option interface  eth1
         option proto  hnet
    config interface  hwan
         option interface  eth0
         option proto  hnet
    ```
  - # /etc/init.d/network reload
- http://www.homewrt.org/doku.php

Homenet – lab testing



TP-Link
tl-1043nd v2

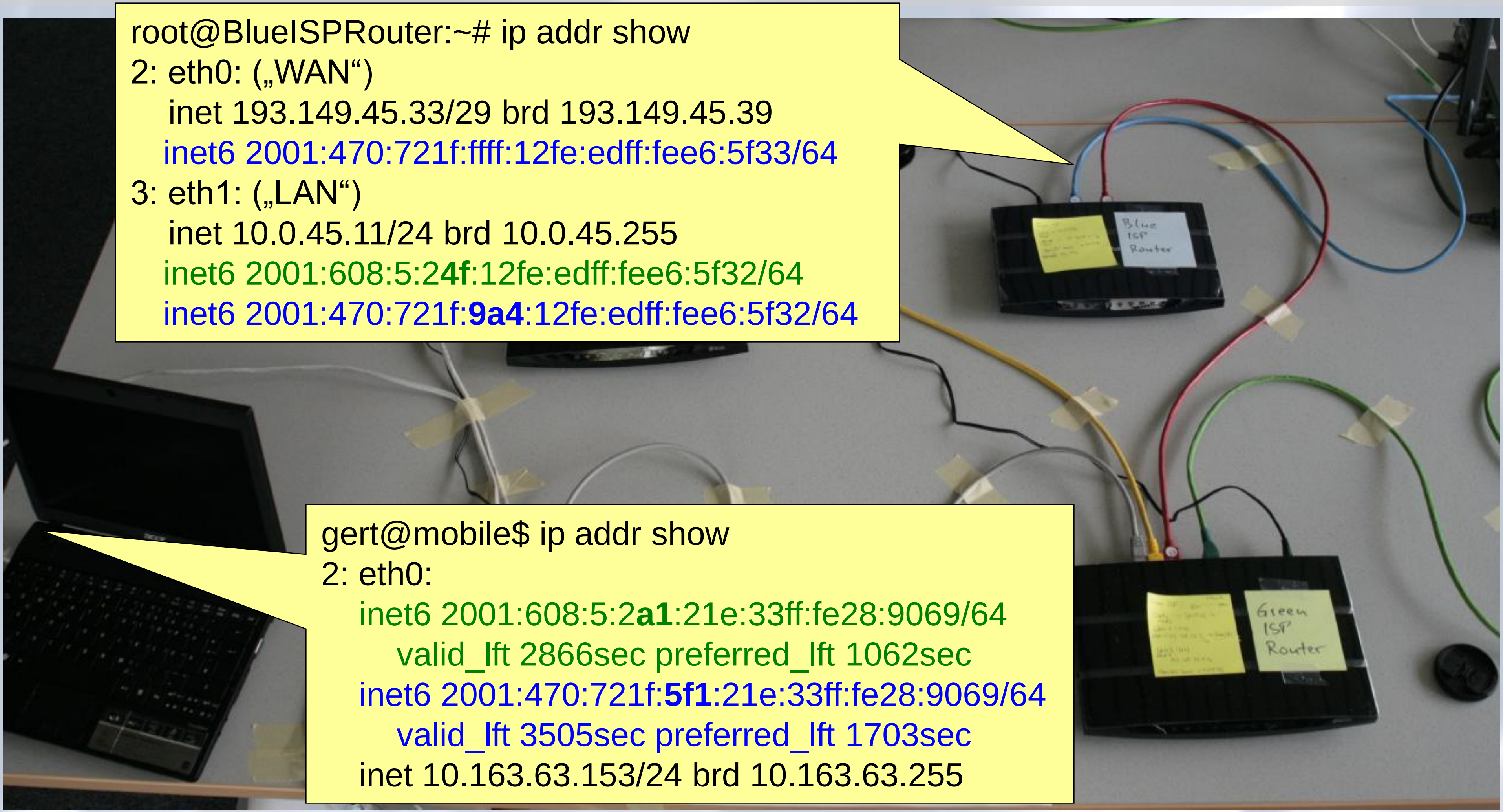
Blue ISP:
Cisco 1841 with
HE.Net-Tunnel

3x TP-Link
tl-wdr3600

Green ISP:
SpaceNet

test platform: OpenWRT „barrier breaker“ (trunk r40576), May 02, 2014

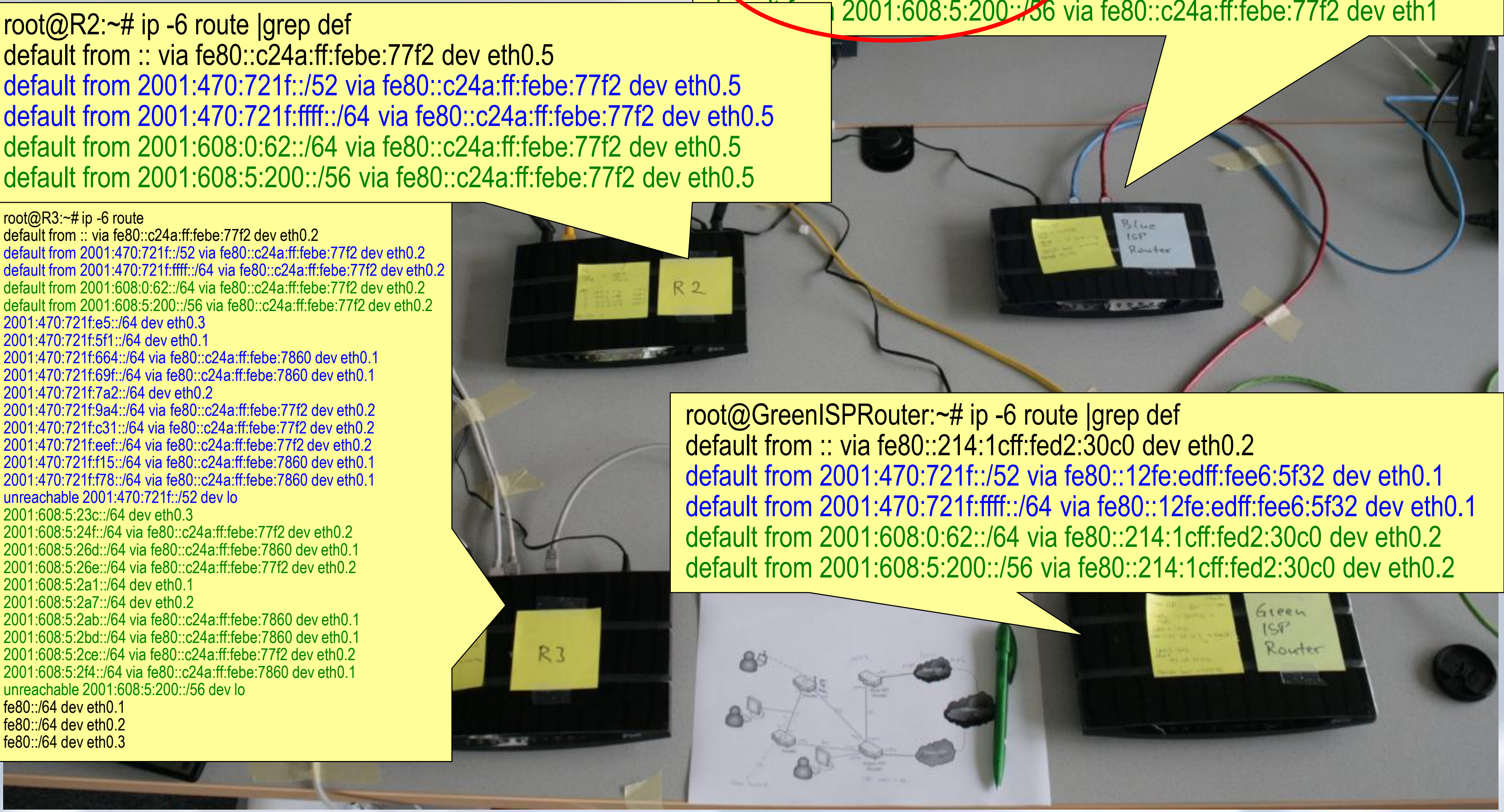
Homenet – lab testing



```
root@BlueISPRouter:~# ip addr show
2: eth0: („WAN“)
    inet 193.149.45.33/29 brd 193.149.45.39
   inet6 2001:470:721f:ffff:12fe:edff:fee6:5f33/64
3: eth1: („LAN“)
    inet 10.0.45.11/24 brd 10.0.45.255
   inet6 2001:608:5:24f:12fe:edff:fee6:5f32/64
   inet6 2001:470:721f:9a4:12fe:edff:fee6:5f32/64
```

```
gert@mobile$ ip addr show
2: eth0:
    inet6 2001:608:5:2a1:21e:33ff:fe28:9069/64
       valid_lft 2866sec preferred_lft 1062sec
    inet6 2001:470:721f:5f1:21e:33ff:fe28:9069/64
       valid_lft 3505sec preferred_lft 1703sec
    inet 10.163.63.153/24 brd 10.163.63.255
```

IP address distribution for blue and green ISP s IPv6 prefixes (and IPv4!) works



```
root@BlueISPRouter:~# ip -6 route |grep def
default from :: via fe80::21e:f7ff:fe38:afd5 dev eth0
default from 2001:470:721f::/52 via fe80::21e:f7ff:fe38:afd5 dev eth0
default from 2001:470:721f:ffff::/64 via fe80::21e:f7ff:fe38:afd5 dev eth0
 2001:608:5:200::/56 via fe80::c24a:ff:febe:77f2 dev eth1
```

```
root@R2:~# ip -6 route |grep def
default from :: via fe80::c24a:ff:febe:77f2 dev eth0.5
default from 2001:470:721f::/52 via fe80::c24a:ff:febe:77f2 dev eth0.5
default from 2001:470:721f:ffff::/64 via fe80::c24a:ff:febe:77f2 dev eth0.5
default from 2001:608:0:62::/64 via fe80::c24a:ff:febe:77f2 dev eth0.5
default from 2001:608:5:200::/56 via fe80::c24a:ff:febe:77f2 dev eth0.5
```

```
root@R3:~# ip -6 route
default from :: via fe80::c24a:ff:febe:77f2 dev eth0.2
default from 2001:470:721f::/52 via fe80::c24a:ff:febe:77f2 dev eth0.2
default from 2001:470:721f:ffff::/64 via fe80::c24a:ff:febe:77f2 dev eth0.2
default from 2001:608:0:62::/64 via fe80::c24a:ff:febe:77f2 dev eth0.2
default from 2001:608:5:200::/56 via fe80::c24a:ff:febe:77f2 dev eth0.2
2001:470:721f:e5::/64 dev eth0.3
2001:470:721f:5f1::/64 dev eth0.1
2001:470:721f:664::/64 via fe80::c24a:ff:febe:7860 dev eth0.1
2001:470:721f:69f::/64 via fe80::c24a:ff:febe:7860 dev eth0.1
2001:470:721f:7a2::/64 dev eth0.2
2001:470:721f:9a4::/64 via fe80::c24a:ff:febe:77f2 dev eth0.2
2001:470:721f:c31::/64 via fe80::c24a:ff:febe:77f2 dev eth0.2
2001:470:721f:eef::/64 via fe80::c24a:ff:febe:77f2 dev eth0.2
2001:470:721f:f15::/64 via fe80::c24a:ff:febe:7860 dev eth0.1
2001:470:721f:f78::/64 via fe80::c24a:ff:febe:7860 dev eth0.1
unreachable 2001:470:721f::/52 dev lo
2001:608:5:23c::/64 dev eth0.3
2001:608:5:24f::/64 via fe80::c24a:ff:febe:77f2 dev eth0.2
2001:608:5:26d::/64 via fe80::c24a:ff:febe:7860 dev eth0.1
2001:608:5:26e::/64 via fe80::c24a:ff:febe:77f2 dev eth0.2
2001:608:5:2a1::/64 dev eth0.1
2001:608:5:2a7::/64 dev eth0.2
2001:608:5:2ab::/64 via fe80::c24a:ff:febe:7860 dev eth0.1
2001:608:5:2bd::/64 via fe80::c24a:ff:febe:7860 dev eth0.1
2001:608:5:2ce::/64 via fe80::c24a:ff:febe:77f2 dev eth0.2
2001:608:5:2f4::/64 via fe80::c24a:ff:febe:7860 dev eth0.1
unreachable 2001:608:5:200::/56 dev lo
fe80::/64 dev eth0.1
fe80::/64 dev eth0.2
fe80::/64 dev eth0.3
```

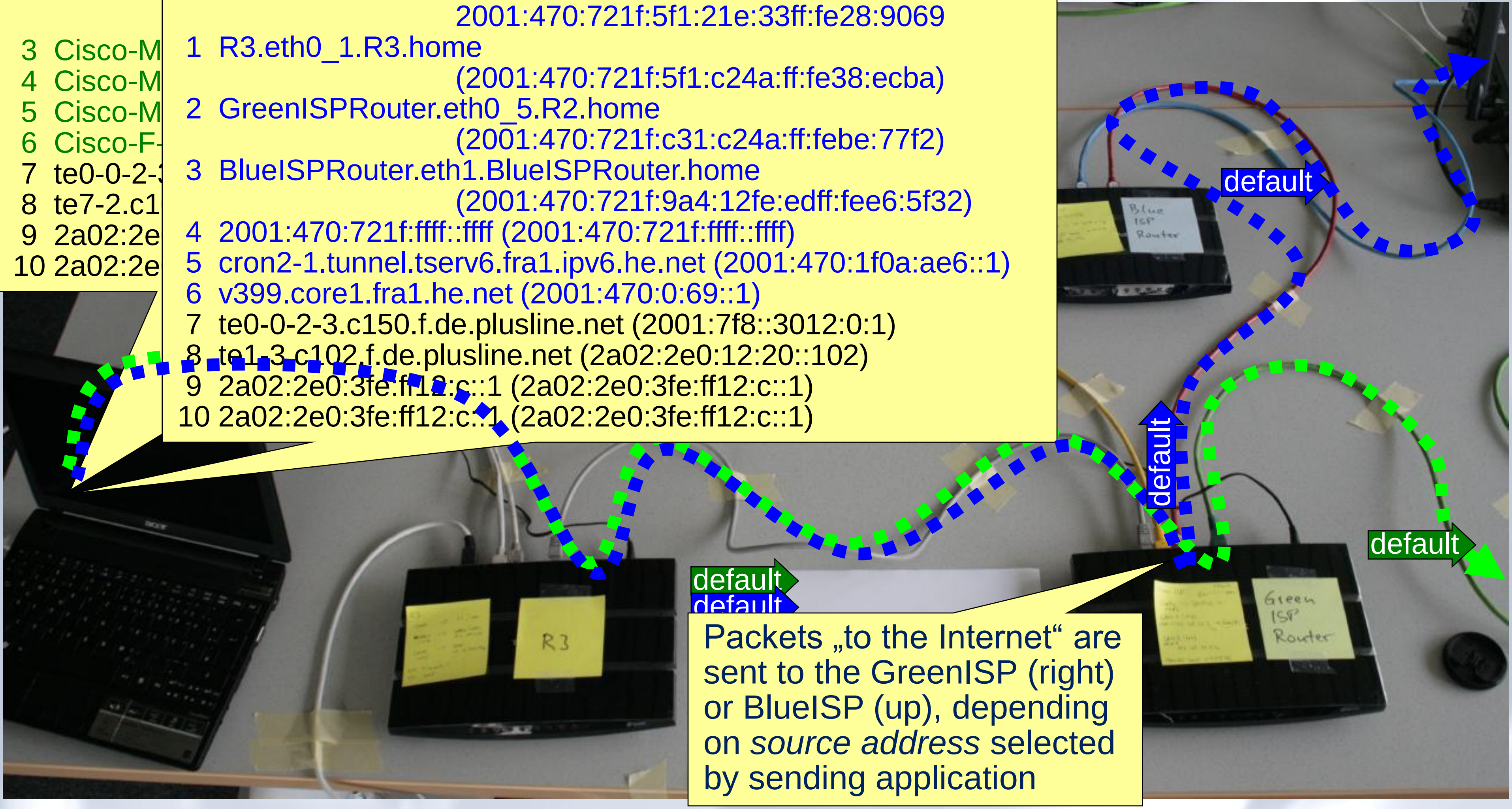
```
root@GreenISPRouter:~# ip -6 route |grep def
default from :: via fe80::214:1cff:fed2:30c0 dev eth0.2
default from 2001:470:721f::/52 via fe80::12fe:edff:fee6:5f32 dev eth0.1
default from 2001:470:721f:ffff::/64 via fe80::12fe:edff:fee6:5f32 dev eth0.1
default from 2001:608:0:62::/64 via fe80::214:1cff:fed2:30c0 dev eth0.2
default from 2001:608:5:200::/56 via fe80::214:1cff:fed2:30c0 dev eth0.2
```

Routing in the Homenet based on target *and source-address* of packets

et – lab testing

```
traceroute to www.heise.de from 2001:608:5:2a1:21e:33ff:fe28:9069
 1  R3.eth0_1.R3.home (2001:608:5:2a1:c24a:ff:fe38:ecba)
 2  GreenIS
 
 3  Cisco-M
 4  Cisco-M
 5  Cisco-M
 6  Cisco-F-
 7  te0-0-2-3
 8  te7-2.c1
 9  2a02:2e
10 2a02:2e
```

```
traceroute to www.heise.de from
                        2001:470:721f:5f1:21e:33ff:fe28:9069
 1  R3.eth0_1.R3.home
                        (2001:470:721f:5f1:c24a:ff:fe38:ecba)
 2  GreenISPRouter.eth0_5.R2.home
                        (2001:470:721f:c31:c24a:ff:febe:77f2)
 3  BlueISPRouter.eth1.BlueISPRouter.home
                        (2001:470:721f:9a4:12fe:edff:fee6:5f32)
 4  2001:470:721f:ffff::ffff (2001:470:721f:ffff::ffff)
 5  cron2-1.tunnel.tserv6.fra1.ipv6.he.net (2001:470:1f0a:ae6::1)
 6  v399.core1.fra1.he.net (2001:470:0:69::1)
 7  te0-0-2-3.c150.f.de.plusline.net (2001:7f8::3012:0:1)
 8  te1-3.c102.f.de.plusline.net (2a02:2e0:12:20::102)
 9  2a02:2e0:3fe:ff12:c::1 (2a02:2e0:3fe:ff12:c::1)
10 2a02:2e0:3fe:ff12:c::1 (2a02:2e0:3fe:ff12:c::1)
```



Packets „to the Internet“ are sent to the GreenISP (right) or BlueISP (up), depending on *source address* selected by sending application

Routing in the Homenet based on target *and source-address* of packets

closing words and summary

- „home networks with more than one router“ are here today already, and we‘ll see more of them in the future
- current approaches (RFC7084- and Hipnet-CPEs) are not addressing all issues properly
- while the Homenet architecture is quite new, and still evolving, the available code basis is very promising
- most important next step: acceptance and implementation by „traditional“ CPE vendors
- ... IPv6 „experts“ need to stop damaging new solutions that contain „routing protocols in the home“ by being all-negative about it
- ... then we‘ll succeed ☺

Referenzen

- Homenet:
  - http://datatracker.ietf.org/wg/homenet/
  - http://datatracker.ietf.org/doc/draft-ietf-homenet-arch/
    - draft-stenberg-homenet-hncp-00.txt
    - draft-pfister-homenet-prefix-assignment-00.txt
    - draft-stenberg-homenet-dnssd-hybrid-proxy-zeroconf-00.txt
    - draft-kline-homenet-default-perimeter-00.txt
    - draft-ietf-mif-happy-eyeballs-extension-04.txt
    - draft-v6ops-ipv6-multihoming-without-ipv6nat-06.txt → RFC7157
  - http://www.homewrt.org/ - Implementation (für OpenWRT)
- Hipnet:
  - http://www.cablelabs.com/the-future-of-home-networking-putting-the-hip-in-hipnet
  - http://tools.ietf.org/id/draft-grundemann-hipnet-00.txt
  - https://ripe66.ripe.net/presentations/115-HIPnet_RIPE66.pdf

Finis

- more questions? Feel free to send to

  gert@space.net

- SpaceNet AG…
  - Internet Service Provider since 1994
  - your partner for complex hosting requirements
  - operational experience with IPv6 since 1997
  - http://www.space.net (of course with IPv6!)