Overview

- Internet Protocol Version 6 (IPv6)
- IP Addressing: practical aspects

Gert Doering, gert@net.in.tum.de

IP(v4): Shortcomings

- IPv4 addresses have 32 bits only
  - not enough to have even 1 IP address per person globally
  - $\Rightarrow$ dynamic IPs, Address Translation (NAT), ...
- manual configuration
  - time consuming (in larger networks)
  - error prone (wrong addresses, duplicates, ...)
  - difficult for embedded appliances (print server, video recorder, fridge, ...)
- IPv4 header format
  - variable length header (option field)
  - very inefficient to parse if IP options present

The Solution: IP Next Generation = IPv6

- new layer 3 protocol, sits next to IPv4 in protocol stack
- runs on top of usual L2 protocols (Ethernet, PPP, . . .)
- is used by usual L4 protocols: TCP, UDP, ICMP
- key changes:
  - 128 bit address length (vs. 32 bit)
  - autoconfiguration
  - restructured / optimized layer 3 headers
  - IPSEC security layer (*)
  - mobile IP(v6) (*)
- but don't panic: *all basic principles stay the same*

IPv6: some first examples

- gert@mobile:/home/gert$ traceroute6 -n www.space.net
  traceroute to www.space.net (2001:608:0:8::136), 30 hops max
  1 2001:608:b:1:204:75ff:fe9d:79d4 3.055 ms 2.329 ms 0.6
  2 2001:608:0:11::119 24.648 ms 23.167 ms 23.02 ms
  3 2001:608:0:11::121 23.06 ms 22.839 ms 23.962 ms
  4 2001:608:0:8::136 24.115 ms 24.255 ms 24.578 ms

- gert@mobile:/home/gert$ telnet www.space.net 80
  Trying 2001:608:0:8::136...
  Connected to www.space.net.
  Escape character is '^]'.
  HEAD / HTTP/1.0

  HTTP/1.1 200 OK
  Date: Mon, 24 Nov 2003 15:51:00 GMT
  Server: Apache/2.0.47 (SpaceNet)
  ...

IPv6 vs. IPv4: packets on the wire

```
Ethernet II, Src: 00:d0:b7:a9:9f:77, Dst: 00:c0:f0:3b:15:fe
    Type: IP (0x0800)
Internet Protocol, Src Addr: 195.30.0.44, Dst Addr: 195.30.0.18
    Version: 4
    Header length: 20 bytes
    Protocol: TCP (0x06)
Transmission Control Protocol, Src Port: 4874, Dst Port: 80,
    Seq: 495047653, Ack: 71155954, Len: 17
Hypertext Transfer Protocol
    HEAD / HTTP/1.0\r\n


Ethernet II, Src: 00:d0:b7:a9:9f:77, Dst: 00:c0:f0:3b:15:fe
    Type: IPv6 (0x86dd)
Internet Protocol Version 6
    Version: 6
    Payload length: 49
    Next header: TCP (0x06)
    Source address: 2001:608::1000:44
    Destination address: 2001:608::1000:18
Transmission Control Protocol, Src Port: 4875, Dst Port: 80,
    Seq: 462650288, Ack: 2871965228, Len: 17
Hypertext Transfer Protocol
    HEAD / HTTP/1.0\r\n
```

IPv6: Benefits (1): Address length

- 32 bits in IPv4 $\Leftrightarrow$ 128 bits in IPv6
- 340282366920938463463374607431768211456 addresses
- restores end-to-end transparency
  - kludges like NAT or proxies are not needed anymore
  - new possibilities for applications (p2p, VoIP, UMTS, ...)
- static network assignments for every customer
  - dynamic addresses still possible (privacy reasons)
- flexibility in network design and planning
- room for growth

IPv6: new address format

- IPv4:
  - 32 bits, 4 x 8 bits, decimal notation, separated by '.'
  - examples: `203.178.141.194`, `195.30.0.2`, `10.0.0.1`
- IPv6:
  - 128 bits, 8 x 16 bits, hexadecimal notation, separated by ':'
  - leading zeroes can be left away (':0123:0001' = ':123:1')
  - exactly one series of zeroes can be reduced to '::'
  - examples:
    - `2001:200:0:8002:203:47ff:fea5:3085`
    - `2001:608::2`
    - `fe80::210:60ff:fe80:3a16`

IPv6: Address delegation: hierarchy

- Hierarchical structure stays mostly unchanged:
  - ICANN ⇒ RIPE ⇒ Provider ⇒ customers
- but much bigger networks, and *fixed size* assignments
  - providers receive /19. . . /32 network blocks
  - every customer network receives a /48 network block
  - every multiaccess network (LAN) uses a /64 network
  - inside LAN: always 64 bit host part = “interface ID”
- right now: only allocations from p=001 (`2xxx::` and `3xxx::`)

IPv6: Routing

- packet forwarding / routing table lookup: similar to IPv4
- same basic rule: “most specific wins”
  - `2001:608:b:1::/64`
  - `2001:608:b::/48`
  - `2001:608:0:1::/64`
  - `2001:608::/32`
- default route is: `0::0/0`
- routing protocols (BGP, OSPF, . . .) and routing table buildup follow the same principles as with IPv4

IPv6 routing: BGP packet snapshot

- TCP-Session over IPv4 or IPv6 transport (!)
- IPv6-prefixes transported in attribute MP_REACH_NLRI:

```
+ Frame 9 (200 bytes on wire, 200 bytes captured)
+ Ethernet II, Src: 00:b0:8e:91:38:1a, Dst: 00:00:0c:3d:dc:c9
- Internet Protocol Version 6
      Source address: 2001:608:0:10::115 (2001:608:0:10::115)
      Destination address: 2001:608:0:10::99 (2001:608:0:10::99)
      Next header: TCP (0x06)
+ Transmission Control Protocol, Src Port: 179, Dst Port: 39463, Len: 126
- Border Gateway Protocol
      UPDATE Message
        Total path attribute length: 65 bytes
        Path attributes
            ORIGIN: IGP (4 bytes)
            AS_PATH: empty (3 bytes)
            MULTI_EXIT_DISC: 0 (7 bytes)
            LOCAL_PREF: 100 (7 bytes)
            COMMUNITIES: 0:100 0:1000 5539:500 (15 bytes)
>           MP_REACH_NLRI (29 bytes)
>               Address family: IPv6 (2)
>               Subsequent address family identifier: Unicast (1)
>               Next hop network address (16 bytes)
>                   Next hop: 2001:608:0:10::115 (16)
>               Subnetwork points of attachment: 0
>               Network layer reachability information (5 bytes)
>                   2001:608::/32
```

IPv6 Benefits (2): Autoconfiguration

- concept of link-local addressing formalized:
  every link uses `fe80::/64` prefix for link-local stuff
  ⇒ hosts in isolated networks can automagically communicate
- if routers are present, they can announce global addresses (e.g. `2001:608:4:0::/64`) via Router Advertisement ICMP packets
- clients will use **all** available /64 prefixes on a given link (link-local + RAs) and compute the host part from their MAC address ⇒ machines usually have multiple IPv6 addresses
- algorithm for computing 64-bit host part from 48-bit (ethernet) MAC address is documented in EUI-64
- autoconfiguration with EUI-64 is one of the reasons for the assignment rule “every link gets a /64 network”

EUI-64 autoconfiguration example

- Notebook with MAC address `00:10:60:80:3A:16`
- link-local prefix `fe80::/64`
- router advertises RA prefix `2001:608:4:0::/64`
- Ethernet MAC is converted to host part of IPv6 address:
  `00:10:60:80:3A:16` ⇒ `::210:60ff:fe80:3a16`
  and appended to all (!) available prefixes
- resulting interface configuration:

```
eth0 Link encap:Ethernet  HWaddr 00:10:60:80:3A:16
     inet addr:193.149.48.163  Mask:255.255.255.224
     inet6 addr: 2001:608:4:0:210:60ff:fe80:3a16/64 Scope:Global
     inet6 addr: fe80::210:60ff:fe80:3a16/64 Scope:Link
```

- note: this can create privacy problems, see RFC3041

IPv6: Adress types frequently seen

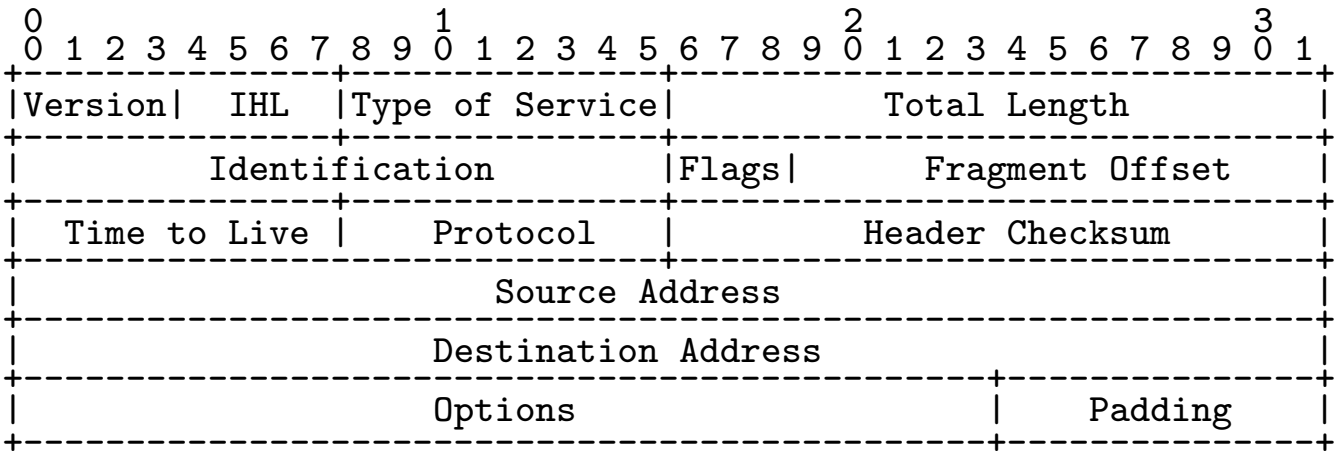
- "local" addresses
  - `fe80::/64` link-local addresses
  - `fec0::/48` site-local addresses (deprecated)
  - `fc00:`*random*`::/48` unique local addresses (ULA), RFC4193
- "global" addresses
  - `3ffe::` 6bone test network, finished 06/06/06
  - `2001::` early IPv6 production networks
  - `2002:`*IPv4*`::/48` 6to4 migration method
  - `2000:::/32` Teredo migration method (Vista!)
  - `ff0x::` global multicast address ranges

IPv6 Benefits (3): Header Format

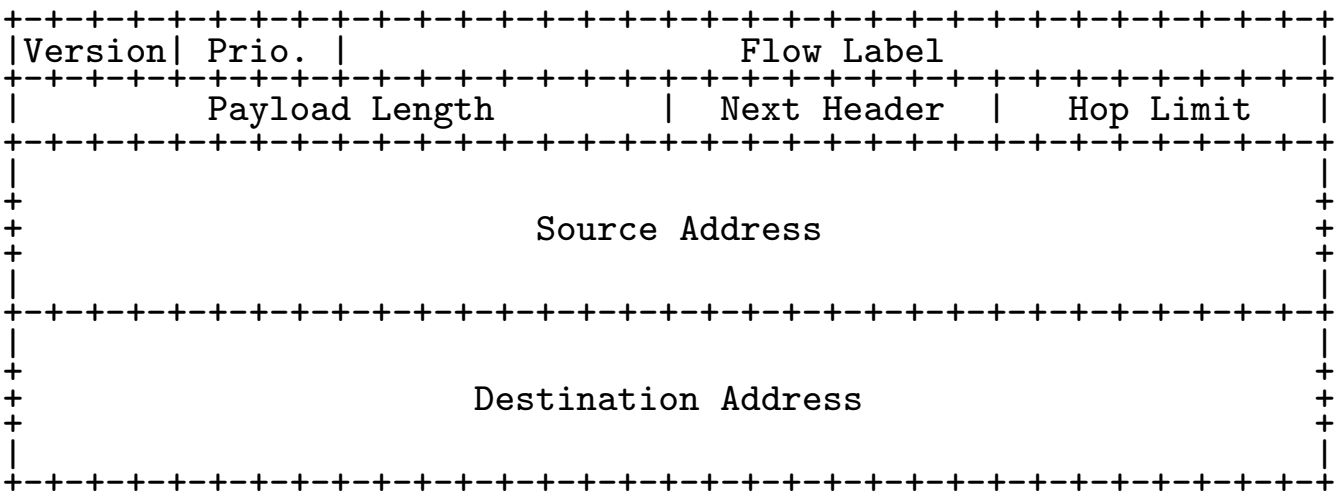
- headers fundamentally reorganized
- some seldomly-used stuff dropped
- option handling and "ip protocol" field collapsed into "IPv6 next header" field
- header checksum dropped: performance (no checksum update)
- router fragmentation dropped: performance / code simplicity
- fixed size (basic) IPv6 header: optimized for CAM hardware
- typical case: IPv6 header → next header = tcp/udp/icmp
- potential chaining of "next header" fields possible
- advanced example:
  IPv6 → fragmentation → encryption → tcp (→ payload)

IPv4 vs. IPv6 header: details

- IPv4: 24 bytes (or more)



- IPv6: 40 bytes



IPv6: open questions

- autoconfiguration and DNS / service discovery
  - one approach: stateless DHCP
- header chaining vs. “middleboxes”:
  - routers that do layer 4 accounting (netflow)
  - firewalls with layer 4 filtering (“tcp port 80”)
- end-to-end IPSEC vs. local policies “no e-donkey allowed!”
  - $\Rightarrow$ distributed security models
- global routing table (BGP), how to scale?
  - early visions of “hard and strict hierarchy” do not work

Migration towards IPv6

- how to introduce IPv6?
- "overnight" approach, switching from IPv4 to IPv6 world-wide on a certain flag day (as for IP in 1983) is not possible
- this creates two kinds of typical problems:
  - v4 host wanting to talk to v6 host
  - v6 networks that are only connected by v4 infrastructure
- $\Rightarrow$ a number of migration techniques have been developed
  - dual-stacked hosts (v4+v6 IP stack on same machine)
  - dual-stacked proxies / application-level gateways
  - tunneling IPv6 over IPv4
  - (and lots of other special-case variants)

IPv6: why is the migration so slow (2003 view)?

- operating system upgrades
- application changes (socket API, numeric display)
- all sorts of "data storage" (SQL dbs, Excel, ...) with IPs
- router vendors
- firewall vendors
- old hardware that cannot be upgraded
- internet providers that do not see any need for IPv6
- service providers, like "www.google.com"

IPv6: why is the migration so slow (2006 view)?

- operating systems and core networks are “done”
- for (nearly) every problem, an IPv6-capable solution exists
- “strategic” target from the EU Commission
- military very much interested, with good reasoning
  - “compatible” protocols for joint EU/NATO operations
  - “network centric warfare”
- 3G mobile network standards *require* IPv6 for IMS
- *still* nearly no end user demand
- users want *applications*, not *protocols*
- but: Windows Vista will ship with default-enabled IPv6 and *IPv6-only P2P application framework*, using Teredo IPv6

Dual-Stack: example output

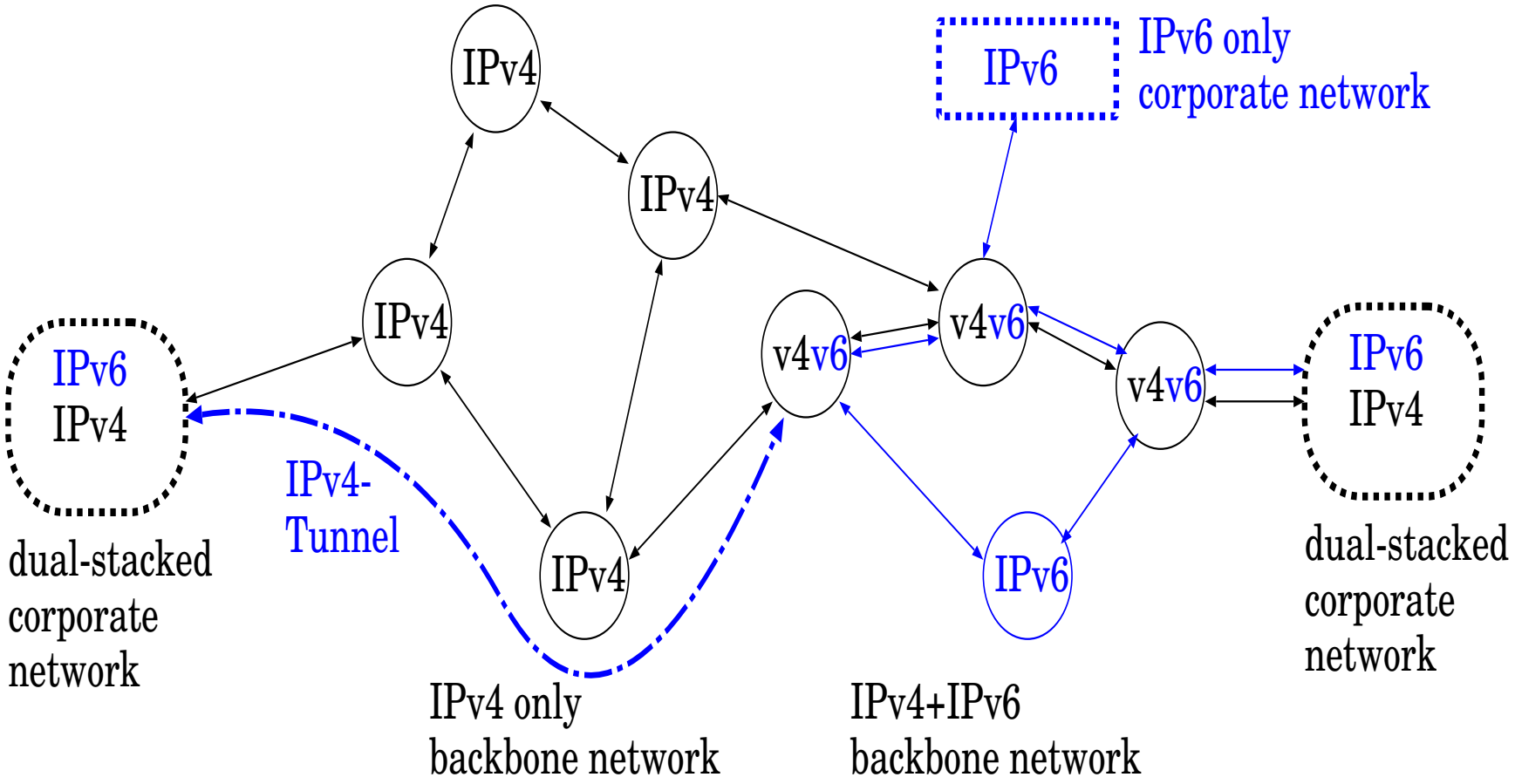
```
Ethernet II, Src: 00:d0:b7:a9:9f:77, Dst: 00:c0:f0:3b:15:fe
    Type: IP (0x0800)
Internet Protocol, Src Addr: 195.30.0.44, Dst Addr: 195.30.0.18
    Version: 4
    Header length: 20 bytes
    Protocol: TCP (0x06)
Transmission Control Protocol, Src Port: 4874, Dst Port: 80,
    Seq: 495047653, Ack: 71155954, Len: 17
Hypertext Transfer Protocol
    HEAD / HTTP/1.0\r\n


Ethernet II, Src: 00:d0:b7:a9:9f:77, Dst: 00:c0:f0:3b:15:fe
    Type: IPv6 (0x86dd)
Internet Protocol Version 6
    Version: 6
    Payload length: 49
    Next header: TCP (0x06)
    Source address: 2001:608::1000:44
    Destination address: 2001:608::1000:18
Transmission Control Protocol, Src Port: 4875, Dst Port: 80,
    Seq: 462650288, Ack: 2871965228, Len: 17
Hypertext Transfer Protocol
    HEAD / HTTP/1.0\r\n
```

Migration: IPv6-in-IPv4 Tunneling

- frequent problem: two IPv6-capable networks want to communicate, but there is some network / network equipment in between that cannot do IPv6
- putting up a direct leased line between those networks is expensive and won't scale
- solution: put up a *virtual* line between them
- When an IPv6 packet leaves network A, towards network B, it will be encapsulated into an IPv4 packet targeted at network B's border router.
- Network B's border router will recognize the IPv4 packet type, and decapsulate the embedded IPv6 packet. The packet is then delivered as normal IPv6 packet to the destination host.

transition period: example networks



IP Addressing: Basics

- if two machines want to communicate over IP, they need to address each other
- addressing is done via *unique* IP addresses
  - unique *network number* for each LAN (L3 network segment)
  - unique *host part* inside each network
- easy in the local LAN:
  - just pick a random network number, e.g. `1.0.0.0/8`
  - and give each machine a unique host ID, e.g. `1.0.0.1`, `1.2.2.2`
- easy in the local enterprise network:
  - pick some unique network numbers for each LAN, e.g. `130.0.0.0/16`, `140.0.0.0/16`, `150.0.0.0/16`

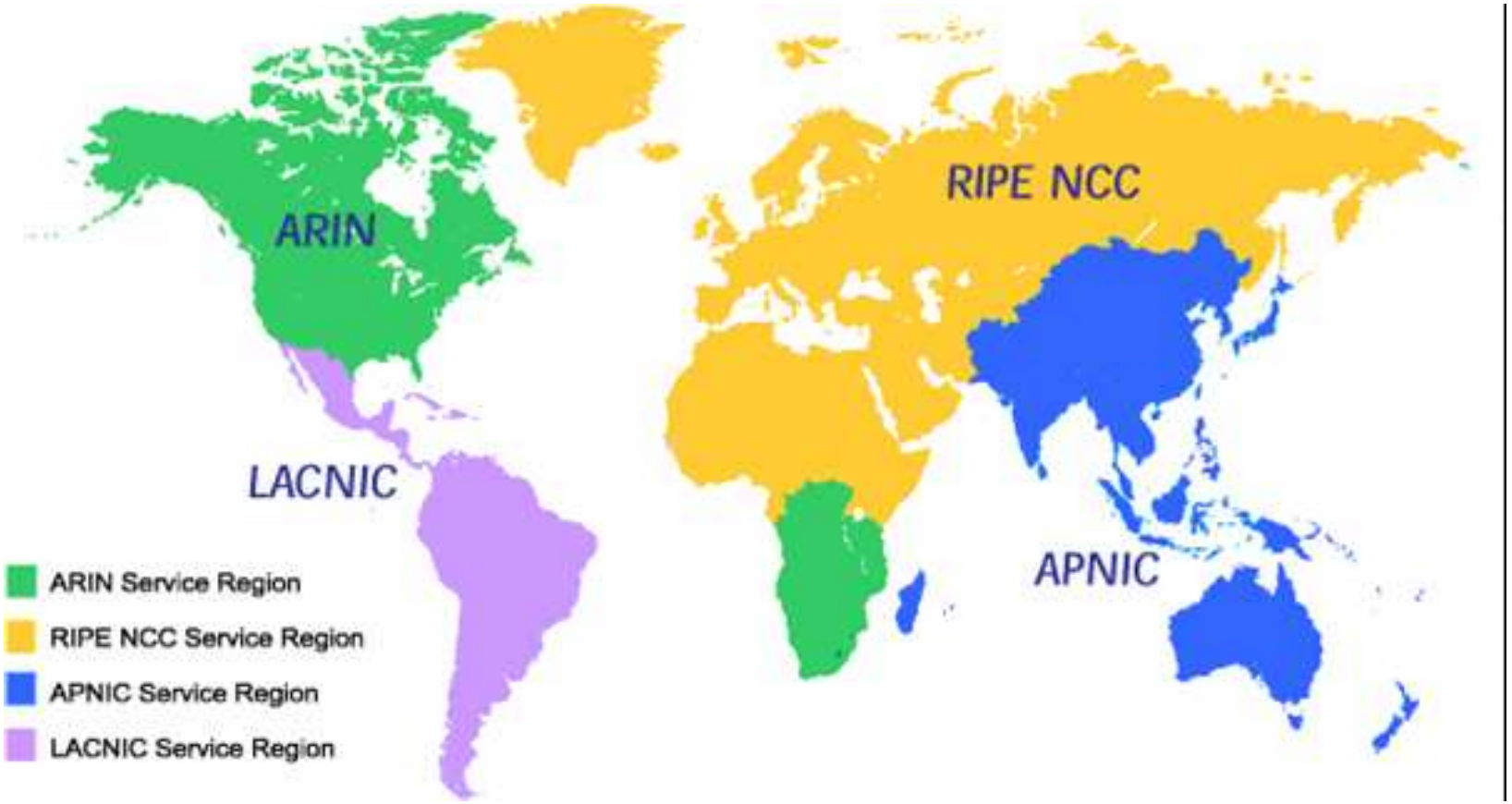
IP Addressing: is it so easy?

- basically, it is...
- ...but...
- this only works if a *central network management* makes sure that network numbers are not assigned twice
- the Internet has no central network management
- ...and now?
- easy approach does *NOT* work!

IP Addressing: hierarchical approach

- central management doesn't scale
- $\Rightarrow$ build a distribution tree
- Root: **ICANN/IANA**
  - hands out /8 network *blocks* to
- Regional Internet Registries (**RIR**)
  - RIPE (europe, m.east), ARIN (north america), LACNIC (latin america), APNIC (asia pacific), AfriNIC (africa)
  - hand out /14. . . /21 to
- Local Internet Registries (**LIR**) - mostly Internet Providers
  - hand out /19. . . /32 to
- End Users

RIR regions


ARIN
RIPE NCC
LACNIC
APNIC
ARIN Service Region
RIPE NCC Service Region
APNIC Service Region
LACNIC Service Region

IP Addressing hierarchy: example

- www.bayern3.de
  - ICANN → RIPE: `193.0.0.0/8`
  - RIPE → SpaceNet: `193.149.32.0/19`
  - SpaceNet → Bayerischer Rundfunk: `193.149.63.64/27`
  - BR → www.bayern3.de = 193.149.63.67
- www.nytimes.com
  - ICANN → ARIN: `199.0.0.0/8`
  - ARIN → Verio.Net: `199.236.0.0/14`
  - Verio → NY Times: `199.239.136.0/24`
  - NYT → www.nytimes.com = 199.239.136.245
- `whois -h whois.ripe.net 193.149.63.67`
  `whois -h whois.arin.net 199.239.136.245`

IP Addressing hierarchy: IPv6 example

- www.space.net
  - ICANN → RIPE: `2001:0600::/23`
  - RIPE → SpaceNet: `2001:0608::/32`
  - SpaceNet → own network: `2001:0608:0::/48`
  - SpaceNet → www.space.net = `2001:608:0:8::136`
- www.kame.net
  - ICANN → APNIC: `2001:0200::/23`
  - APNIC → WIDE project.: `2001:0200::/32`
  - WIDE → www.kame.net = 2001:200:0:8002:203:47ff:fea5:3085
- `whois -h whois.ripe.net 2001:608:0:8::136`
  `whois -h whois.apnic.net 2001:200:0:8002:203:47ff:fea5:3085`

IP Addressing: shortcuts

- for local (isolated) network, specific network numbers are reserved in **RFC1918** for private use
  - `10.0.0.0/8`
  - `172.16.0.0/16` – `172.31.0.0/16`
  - `192.168.0.0/24` – `192.168.255.0/24`
- for ad-hoc networks that have no connection to any other layer 3 network, **RFC3330** documents a /16 for link-local usage
  - `169.254.0.0/16`
  - machines can pick an address from that range if they are set up for automatic address configuration and DHCP fails
- for IPv6, ULA and fe80:: link-local addresses are used instead

References

- http://www.6bone.net/
- http://www.ietf.org/rfc.html
- http://www.icann.org/
- http://www.ripe.net/
- gert@net.in.tum.de